Automatic Dynamic Texture Segmentation Using Local Descriptors and Optical Flow

Jie Chen, *Member, IEEE*, Guoying Zhao, *Senior Member, IEEE*, Mikko Salo, Esa Rahtu, and Matti Pietikäinen, *Fellow, IEEE*

***Abstract*— A dynamic texture (DT) is an extension of the texture to the temporal domain. How to segment a DT is a challenging problem. In this paper, we address the problem of segmenting a DT into disjoint regions. A DT might be different from its spatial mode (i.e., appearance) and/or temporal mode (i.e., motion field). To this end, we develop a framework based on the appearance and motion modes. For the appearance mode, we use a new local spatial texture descriptor to describe the spatial mode of the DT; for the motion mode, we use the optical flow and the local temporal texture descriptor to represent the temporal variations of the DT. In addition, for the optical flow, we use the histogram of oriented optical flow (HOOF) to organize them. To compute the distance between two HOOFs, we develop a simple effective and efficient distance measure based on Weber's law. Furthermore, we also address the problem of threshold selection by proposing a method for determining thresholds for the segmentation method by an offline supervised statistical learning. The experimental results show that our method provides very good segmentation results compared to the state-of-the-art methods in segmenting regions that differ in their dynamics.**

***Index Terms*— Dynamic texture segmentation, local descriptor, optical flow, Weber's law.**

I. Introduction

DYNAMIC textures or temporal textures are textures with motion [1], [2], [3]. Dynamic textures could be loosely described as visual processes, which consist of a group of particles with random motion [4]. As shown in Fig. 1, the particles can be macroscopic (e.g., foliage flying in the wind), microscopic (e.g., fire plume and smoke), or even moving objects (e.g., a flock of birds). Potential applications of DT analysis include remote monitoring and various types of surveillance in challenging environments, such as monitoring forest fires to prevent natural disasters, traffic monitoring, homeland security applications, and animal behavior for scientific studies, video synthesis, motion segmentation, and video classification.

Segmentation is one of the basic problems in computer vision [5], [6], [7]. Meanwhile, DT segmentation is very challenging compared with the static case because of their unknown spatiotemporal extension, the different moving particles, and stochastic nature of the motion fields. DT segmentation is to separate the different groups of particles showing different random motion.

Manuscript received September 14, 2011; revised June 15, 2012; accepted June 17, 2012. Date of publication July 24, 2012; date of current version December 20, 2012. This work was supported in part by the Academy of Finland and Infotech Oulu. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Joseph P. Havlicek.

J. Chen, G. Zhao, E. Rahtu, and M. Pietikäinen are with the Department of Computer Science and Engineering, Center for Machine Vision Research, University of Oulu, Oulu FI-90014, Finland (e-mail: jiechen@ee.oulu.fi; gyzhao@ee.oulu.fi; erahtu@ee.oulu.fi; mkp@ee.oulu.fi).

M. Salo is with the Department of Mathematics and Statistics, University of Jyväskylä, Jyväskylä 40014, Finland (e-mail: mikko.j.salo@jyu.fi).

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

Digital Object Identifier 10.1109/TIP.2012.2210234

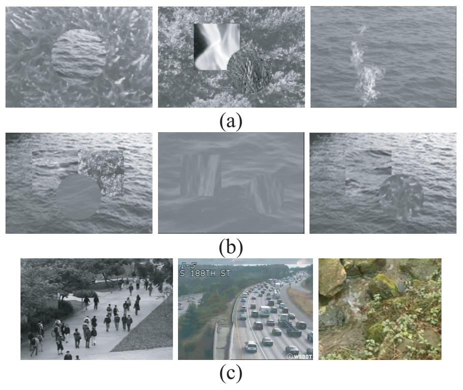


Fig. 1. Illustration of DTs. (a) DTs are different from their spatial mode (i.e., appearance) but similar to their temporal mode (i.e., motion). (b) DTs are different from their temporal mode but similar to their spatial mode. (c) Temporal and spatial modes that are similar to DTs are cluttered.

In this paper, we propose a new method based on both appearance and motion information for the segmentation of dynamic textures. For the appearance of DT, we use local spatial texture descriptors to describe the spatial mode of DT; for the motion of DT, we use the optical flow and local temporal texture descriptors to represent the movement of objects, and employ the Histogram of Oriented Optical Flow (HOOF) approach to organize the optical flow of a region. To compute the distance between two HOOFs, we develop a new distance measure based on Weber's Law, which is simple and efficient. The motivation, as shown in Fig. 1, to employ both the appearance and motion modes for the DT segmentation is that DTs might be different from their spatial mode (i.e., appearance) and/or temporal mode (i.e., motion field). Combining the spatial and temporal modes, we exploit the fuse of discriminant features of both the appearance and motion for the robust segmentation of cluttered DTs.

For the spatial mode of DT, we use the simple but effective local texture descriptor, i.e., local binary pattern (LBP) [8] and Weber local descriptor (WLD) [11]. LBP is robust to monotonic gray-scale changes caused, e.g., by illumination variations. It is widely used, e.g., in texture analysis [10], [8]. WLD is also simple and effective for texture classification [11]. We just use one of its components, i.e., differential

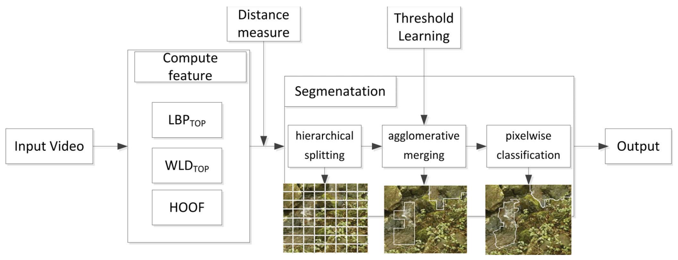


Fig. 2. Flow chart of the framework.

excitation. However, we still call it WLD for consistency. Here, we use both LBP and WLD because they are complementary to each other. Specifically, LBP describes the orientations of the edges in images well but loses the intensity information [8]. On the contrary, WLD preserves the intensity information well (we only use differential excitation here) and also detects the edges elegantly but loses the orientations of edges [11]. In addition, we generalize the spatial mode of WLD to a spatiotemporal mode as Zhao and Pietikäinen [12], in which they generalized LBP as a spatiotemporal descriptor, i.e., LBP in three orthogonal planes or LBP-TOP. Likewise, we call the spatiotemporal mode of WLD as WLD_{TOP}, and the combined LBP and WLD in three orthogonal planes as $(\text{LBP/WLD})_{\text{TOP}}$ [9].

For the temporal mode of DT, we use both optical flow and local temporal texture descriptors to capture the motion, and use HOOF feature to describe the optical flow motivated by [13]. For the distance computation between two HOOFs, we develop a new distance measure based on Weber's Law [14], which is also computationally simple and effective. For local temporal texture descriptors, we use the temporal component of $(\text{LBP/WLD})_{\text{TOP}}$, which describes the temporal variations.

Furthermore, we address the problem of threshold selection by proposing a method to determine thresholds by an offline supervised statistical learning. The learned threshold is then manually adjusted according to our observations. The adjusted threshold values work robustly and perform well and consistently on the videos in our experiments. The achieved threshold values can be used as a good start point to be further appropriately adjusted for the new dynamic texture videos although they might not work for very different videos as tested in our experiments.

1) Framework: We illustrate the framework of our method, as shown in Fig. 2. Given an input video, we firstly compute its feature, i.e., $(\text{LBP/WLD})_{\text{TOP}}$ and HOOF. Using these features, we perform the segmentation by hierarchical splitting, agglomerative merging and pixelwise classification.

Note that in this paper, we will use two terms to represent a part of an image: patch and region. A patch is used to represent the smallest grid split by splitting process. A region is employed to represent a patch or a larger part merged by several patches. The rest of this paper is organized as follows: In Section II, we present related work. In Section III, we describe features for segmentation, discuss how to use them for DT segmentation and propose a new distance measure for HOOF. In Section IV, we show the process of DT segmentation and describe how to compute the thresholds for the framework. In Section V, we present some results.

II. Related Work

In this section, we review the methods of dynamic texture segmentation. Two types of statistical models can be applied to DT segmentation: generative models and discriminative models. A generative model is a model for randomly generating observable data, typically given some hidden parameters [15], [16]. For example, Doretto et al. [17] modeled the spatiotemporal dynamics by Gauss–Markov models, and inferred the model parameters. Vidal and Ravichandran [18] proposed to model each moving dynamic texture with a time varying linear dynamical system (LDS) plus a 2-D translational motion model. Cooper et al. [19] represented a single temporal texture by a low dimensional linear model and segmented DT using generalized principal component analysis (GPCA). Chan and Vasconcelos [4] presented a statistical model for an ensemble of video sequences. They derived an expectation-maximization algorithm for learning the parameters of the model. After that, they proposed the layered dynamic texture (LDT) to represent a video as a collection of stochastic layers of different appearance and dynamics [20], and then proposed a variational approximation for the LDT that enables efficient learning of the model [21]. Rahman and Murshed [22] detected the presence of multiple dynamic textures in an image sequence. Milko et al. [23] modeled the dynamics of each pixel as an auto regressive process perturbed with Gaussian noise. Ghoreyshi and Vidal [24] modeled the spatial statistics of a dynamic texture with a set of second order Ising descriptors whose temporal evolution is governed by an autoregressive exogenous model.

Discriminative methods enable the construction of flexible decision boundaries, resulting in classification performances often superior to those obtained by purely probabilistic or generative models [15]. For example, Vidal and Singaraju [25] introduced the multibody brightness constancy constraint, a polynomial equation relating motion models, image derivatives and pixel coordinates to segment multiple 2-D motion models. Fazekas et al. [26] developed the optical flow method to capture the intrinsic dynamics of dynamic textures. After that, Chetverikov et al. [27] developed this method and applied the singular value decomposition to a temporal data window in a video to detect targets in DT via the residual of the largest singular value.

In addition, there are some researches using the similar methods as ours but for the different task. For example, Stein and Hebert [28] combined the appearance and motion by low-level detection and mid-level reasoning for the occlusion boundary detection. Derpanis and Wildes employed the spatiotemporal oriented energy measurements for the spatiotemporal grouping [29].

For our method, we learn the thresholds by an offline supervised statistical learning. We then segment DT by an automatic and discriminative way. We do not need any initialization or a priori information about the DTs in the test set during segmentation. This paper is an extension of our previous work [9]. In this paper, we extend WLD to a 3D version (i.e., WLD$_{\text{TOP}}$ feature), and combine (LBP/WLD)$_{\text{TOP}}$ and HOOF to describe the dynamic textures. We also develop a new distance measure (i.e., weighted Weber distance) for HOOF. In addition, we also provide more extensive evaluations on our method.

III. Features and Distances for Segmentation

In this section, we first discuss the features and then present how to use them for the description of DTs. After that we describe the proposed Weber distance.

A. Features

1) Local Texture Descriptor: To describe DT, we use a local spatial-temporal texture descriptor. Specifically, as shown in Fig. 3, (a) is a sequence of frames (or images) of a DT; (b) denotes the three orthogonal planes or slices *XY*, *XT* and *YT*, where *XY* is the appearance (or a frame); *XT* shows the visual impression of a row changing in temporal space; and *YT* describes the motion of a column in temporal space; (c) illustrates the vertex coordinates of the three orthogonal planes for the feature computation of LBP/WLD of one pixel; (d) shows how to compute LBP and WLD for each pixel of these three planes; (e) shows how to compute sub-histograms from three slices which are denoted as $H_{\lambda,\pi}(\lambda =$ LBP, WLD and $\pi = XY, XT, YT)$.

LBP$_{\text{TOP}}$ is a spatiotemporal descriptor [12], which computes LBP feature in three orthogonal planes, as shown in Fig. 3(c). For each plane, it is computed as follows:

$$LBP_p = \sum_{p=0}^{P-1} s(g_p - g_c)2^p, \quad \text{where}$$
$$s(A) = \begin{cases} 1, & if\ A \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

the gray value g_c corresponds to the gray value of the center pixel; g_p correspond to the gray values of P neighboring pixels on a rectangle of length $2L_x+1$ and $2L_y+1$ [$L_x > 0$, $L_y > 0$, e.g., $L_x = L_y = 1$ in Fig. 3(d)]. If we concatenate these three sub-histograms $H_{LBP,\pi}(\pi = XY, XT, YT)$ into a histogram, we get an LBP$_{\text{TOP}}$ feature histogram.

The WLD feature is the differential excitation of WLD descriptor of [11]. It is computed as follows:

$$WLD(I_c) = \text{sigmoid}(x) = \frac{1-e^{-x}}{1+e^{-x}}, \quad \text{where}$$
$$x = \sum_{i=0}^{P-1}\left(\frac{I_i - I_c}{I_c + C_0}\right). \tag{2}$$

Here, I_c denotes the center pixel, I_i ($i = 0, 1, \ldots, P-1$) are the neighbors, and P is the number of neighbors (e.g., $P = 8$), and where C_0 is a given constant to avoid the case that I_c is equal to zero. In our case, we set $C_0 = 5$.

Likewise, we also compute the WLD feature in the three orthogonal planes, which are denoted as WLD$_\pi$ ($\pi = XY$, *XT* and *YT*). These WLD features are regrouped in three sub-histograms $H_{WLD,\pi}(\pi = XY, XT, YT)$. Due to WLD$_\pi$, here it refers to the features in three orthogonal planes, we call it

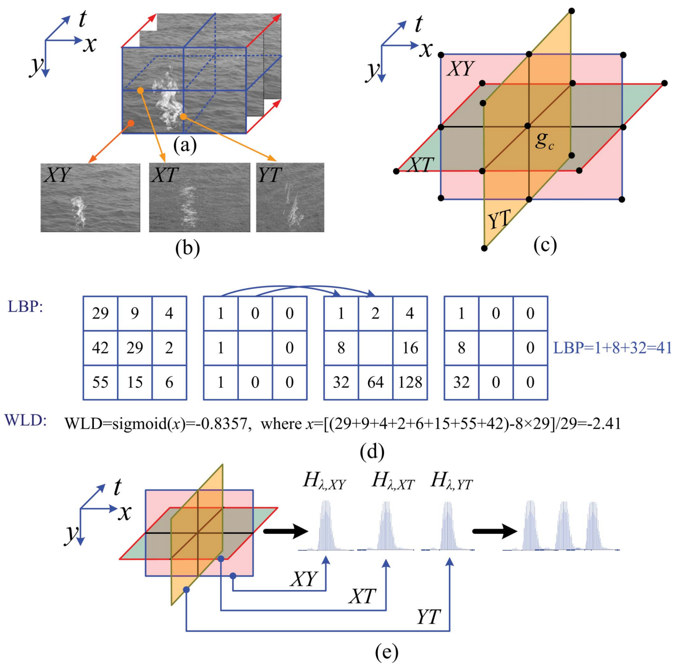


Fig. 3. Computation of (LBP/WLD)TOP for a DT. (a) Sequence of a DT. (b) Three orthogonal planes of the given DT. (c) Vertex coordinates of the three orthogonal planes. (d) Computation of LBP and WLD of a pixel. (e) Computation of subhistograms for (LBP/WLD)$_{\text{TOP}}$, where $\lambda =$ LBP, WLD.

WLD$_{\text{TOP}}$. We denote (LBP/WLD)$_{\text{TOP}}$ as features of LBP and WLD in these three planes. The (LBP/WLD)$_{\text{TOP}}$ feature can be denoted as a vector form

$$\varphi = \{H_{\lambda,\pi} | \lambda = LBP \text{ or WLD, and } \pi = XY,\ XT \text{ or } YT\}. \tag{3}$$

Motivated by [12], the (LBP/WLD)$_{\text{TOP}}$ is computed over an ellipsoid, and the number of neighboring points in *XY*, *XT* and *YT* planes can also be different, which are marked as R_X, R_Y and R_T, P_{XY}, P_{XT} and P_{YT}, the corresponding LBP/WLD feature is denoted as $(LBP/WLD)^{\text{TOP}}_{P_{XY},P_{XT},P_{YT},R_X,R_Y,R_T}$. Here, R_x, R_y, and R_t are the radii in axes X, Y and T, and P_{XY}, P_{XT} and P_{YT} are the numbers of neighboring points in the *XY*, *XT* and *YT* planes.

In our implementation, we use the "uniform patterns" as in [8] to shorten the length of the feature vector LBP. Thus, we have 59 bins for $H_{LBP,\pi}$ when the number of neighboring points is 8. On the other hand, we use 16 bins experimentally for $H_{WLD,\pi}$. See Ref. [10] for a detailed description of the mapping from the continuous WLD space to the discrete bin index. In addition, we also experimentally set $R_X = 1$, $R_Y = 1$ and $R_T = 3$, $P_{XY} = 8$, $P_{XT} = 8$ and $P_{YT} = 8$. Thus, (LBP/WLD)$_{\text{TOP}}$ is denoted as $(LBP/WLD)^{\text{TOP}}_{8,8,8,1,1,3}$. Note that we set $R_T = 3$, which means (LBP/WLD)$_{\text{TOP}}$ represents the appearance and motion features of neighboring $T = 7$ frames.

2) HOOF: To compute the optical flow (OF) for each pixel, we use the method proposed in [30]. Motivated by [31], we organize the OF of a region, as shown in Fig. 4. I.e., the OF magnitudes are accumulated directly into orientation histograms. We call the descriptor as HOOF following the idea in [13].

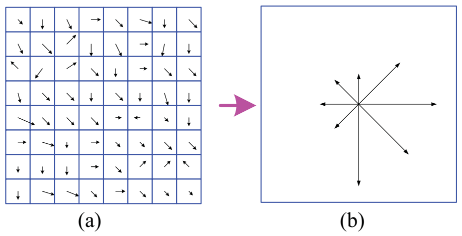


Fig. 4. (a) Local descriptor is created by first computing OF magnitude and orientation at each frame sample point in a region. (b) These OF magnitudes are accumulated into orientation histograms summarizing the contents over the region.

Considering that some pixels in regions of DTs are still and their OF values are close to zero, we use an extra bin in HOOF to build the statistic for these pixels. Intuitively, using an extra bin in HOOF guarantees that the similarity between two still regions is one. To filter the noise in the OF field of a frame, we use a threshold for the OF value of a pixel and set (0.1, 0.1) for the two directions of a pixel. In other words, if the OF values of a pixel in one direction is smaller than 0.1, we set the OF values of this pixel in this direction as zero. It then accumulates to the extra bin of HOOF.

Given two HOOFs (i.e., H_1, H_2) of regions R_1 and R_2, $H_i (i = 1, 2)$ are normalized *as* $H_i = H_i/A_i$, where A_i is the area of R_i. We use this normalization because two regions for merging might not have the same area.

B. Feature Fusion

The appearance of DT is described by a local spatial texture descriptor $\varphi_s = \{H_{LBP,XY}, H_{WLD,XY}\}$. The motion of DT is described by HOOF and local temporal texture descriptor $\varphi_t = \{H_{\lambda,\pi} | \lambda = LBP \text{ or } WLD, \text{ and } \pi = XT \text{ or } YT\}$. Here, we use both HOOF and the temporal texture descriptor φ_t, because they are complementary to each other: HOOF describes the motion field of particles (macroscopic or microscopic), while φ_t describes the temporal variations of the texture appearance. We call the combined feature as the motion of DT for simplicity.

The resulting histograms of (LBP/WLD)$_{\text{TOP}}$ and HOOF are used to describe a local region of DT. Thus, the similarity measurement between two regions is computed as:

$$\Pi(\varphi_s, \varphi_t, HOOF) = \omega_S \cdot \Pi_S + \omega_T \cdot \Pi_T \tag{4}$$

where ω_S and ω_T denote the weights for the spatial and temporal modes, respectively; Π_S and Π_T are the spatial and temporal similarities. These two weights (i.e., ω_S and ω_T) are used to balance the effects of spatial and temporal features. In our implementation, we set $\omega_S = \omega_T = 0.5$ because we suppose that the appearance and motion of DT play the same important roles. In addition, how to compute Π_S and Π_T of two regions will be presented in Section III-C.

C. Distance Measures

In this part, we present the distance measures for LBP/WLD and HOOF. For the LBP/WLD, we use the histogram intersection; For the HOOF, we develop a new distance measure based on the Weber's Law.

1) Histogram Intersection for LBP/WLD: In this subsection, we describe the similarity measure for (LBP/WLD)$_{\text{TOP}}$. To compute the similarity between two given histograms H_1 and H_2, we use the histogram intersection $\Xi(H_1,H_2)$ as a similarity measurement of two normalized histograms, i.e.,

$$\Xi(H_1, H_2) = \sum_{i=1}^{L} \min(H_{1,i}, H_{2,i}) \tag{5}$$

where L is the number of bins in a histogram.

For (LBP/WLD)$_{\text{TOP}}$, the similarity between any two regions consists of six components $\Gamma = \{\Pi_{LBP,XY}, \Pi_{LBP,XT}, \Pi_{LBP,YT}, \Pi_{WLD,XY}, \Pi_{WLD,XT}, \Pi_{WLD,YT}\}^T$, where $\Pi_{\lambda,\pi}$ (λ = LBP or *WLD* and π = *XY*, *XT*, *YT*) are the histogram similarities of (LBP/WLD)$_{\text{TOP}}$ in three orthogonal planes (i.e., *XY*, *XT*, *YT*).

The spatial similarity between two regions is computed as follows:

$$\Pi_S = \omega_{XY} \cdot (\omega_{\text{LBP}} \cdot \Pi_{LBP,XY} + \omega_{\text{WLD}} \cdot \Pi_{WLD,XY}) \tag{6}$$

and the temporal similarity with regards to the texture mode between two regions is computed as:

$$\begin{aligned}\Pi_{T,\text{ texure}} &= \omega_{XT}(\omega_{\text{LBP}} \cdot \Pi_{LBP,XT} + \omega_{\text{WLD}} \cdot \Pi_{WLD,XT}) \\ &\quad + \omega_{YT} \cdot (\omega_{\text{LBP}} \cdot \Pi_{LBP,YT} + \omega_{\text{WLD}} \cdot \Pi_{WLD,YT})\end{aligned} \tag{7}$$

where ω_{LBP} and ω_{WLD} are two weights to balance the effects of the LBP and WLD features, and $\omega_\pi (\pi = XY, XT, YT)$ are the weights for each plane.

In our implementation, we experimentally set $\omega_{\text{LBP}} = \omega_{\text{WLD}} = 0.5$, and $\omega_{XY} = 0.625$, $\omega_{XT} = 0.25$, $\omega_{YT} = 0.125$. Here, we let $\omega_{\text{LBP}} = \omega_{\text{WLD}}$ because both descriptors have achieved a similar performance in texture classification [11]; For the value of $\omega_\pi (\pi = XY, XT, YT)$, we use the same values as shown in [12]. Meanwhile, the value of ω_{XY} is larger than that of ω_{XT} and ω_{YT}. One possible reason is that the discriminability of LBP and WLD over texture appearance is more powerful than the discriminability of LBP and WLD over texture motion.

2) Weber Distance for HOOF: Given two HOOFs, it is difficult to compute the distance between them due to the following reasons: (1) HOOF is non-Euclidean [13]; (2) HOOF is non-linear (i.e., different bins showing different cumulative occurrences and so different weights); (3) HOOF during segmentation can not be normalized to sum to one due to the importance of the different cumulative occurrences of each bin; (4) distance measure should not only work for those regions whose OF values are large (corresponding to the motion regions), but also work for those regions whose OF values are small or zeros (corresponding to still regions); (5) noise resulted from the computation of OF due to the brightness variations and algorithm failure for those non-texture regions [30].

The constraints (1), (2) and (3) result in difficulties that many current distance measures are not able to handle. For example: Euclidean distance fails to the constraints (1); Normalized histogram intersection fails to the constraints (2) and (3) although it works properly for the histogram of LBP/WLD. Specifically, for example, given two patches P_1 and P_2, suppose that the HOOF of P_1 is $H_1 = \{a_0, a_1, \ldots, a_{L-1}\}$, where a_i is a bin of this histogram, and L is the dimensionality of this histogram, and suppose that the HOOF ofP_2 is $H_2 = k \cdot \{a_0, a_1, \ldots, a_{L-1}\}$, and k is a real number. However, the similarity between H_2 and H_1 is one although the parameter k takes any value if we use the normalized histogram intersection. Likewise, Chi square fails to the constraints (2) and (3). Furthermore, the constraint (4) brings out extra difficulties for the design of a new distance measure and the constraint (5) would degrade the efficiency of a new distance measure.

In addition, HOOF, in our case, cannot be normalized to sum to one due to the importance of the different cumulative occurrences of each bin during the dynamic texture segmentation. An alternative method is to use the level set scheme, as shown in [26]. However, the level set scheme usually needs an initialization and the performance of segmentation depends on the initialization (see more details in Section V-D).

a) Weber distance: The Weber's Law denotes that the ratio (k) of the increment threshold (ΔI) to the background intensity (I) is a constant [14], i.e., $\Delta I/I = k$. Similarly, Given two HOOFs of regions R_1 and R_2, $H_1 = \{a_0, \ldots, a_{L-1}\}$, $H_2 = \{b_0, \ldots, b_{L-1}\}$, Weber distance is computed as follows. We firstly compute the ratios between each bin of two HOOFs: $k_i = \max\left\{\frac{|a_i-b_i|}{C+b_i}, \frac{|a_i-b_i|}{C+a_i}\right\}$, where C is a given constant to avoid the case that b_i or a_i is equal to zero. In our case, we set $C = 1$. We then use sigmoid function to normalize the ratio: $x_i = \text{sigmoid}(k_i) = \frac{1-e^{-k_i}}{1+e^{-k_i}}$, and so $x_i \in [0, 1)$. The distance between two HOOFs is computed as

$$D_{\text{HOOF}} = \frac{1}{L}\sum\nolimits_{i=0}^{L-1} x_i. \tag{8}$$

The value D_{HOOF} can be used as the distance measure between H_1 and H_2. For example, if H_1 and H_2 are the same, we have $D_{\text{HOOF}} = 0$, i.e., there is no difference between H_1 and H_2. If $H_2 = k \times H_1$, we have:

$$D_{\text{HOOF}} = \frac{1}{L}\sum_{i=0}^{L-1} \text{sigmoid}\left(|k-1|\, a_i \times \max\left\{\frac{1}{C+ka_i}, \frac{1}{C+a_i}\right\}\right)$$

and hence, $D_{\text{HOOF}} \neq 0$. In other words, it works well for the case that the H_2 and H_1 are proportional in each dimension.

b) Weighted Weber distance: In our experiments, we found the different bins of HOOF play different roles to compute the similarities between two patches. One idea is to weight the different bins according to their importance to describe the motion of a patch. Specifically, given two HOOFs of patches P_1 and P_2, $H_1 = \{a_0, \ldots, a_{L-1}\}$, $H_2 = \{b_0, \ldots, b_{L-1}\}$, the weight of a bin is computed as

$$\omega_i = \frac{a_i + b_i}{\sum\limits_{i}^{L-1}(a_i + b_i)}.$$

Combining this formula with Eq. (8), we have the weighted Weber distance between two HOOFs:

$$D_{\text{HOOF}} = \frac{1}{L}\sum\nolimits_{i=0}^{L-1}(\omega_i x_i). \tag{9}$$

In addition, we found that the weighted Weber distance works comparably on LBP/WLD to normalized histogram intersection although the weighted Weber distance is developed for the HOOF.

c) Weighted Weber distance and one extra bin for zero optical flow values: We use one bin to build the statistics of the number of pixels whose OF values are equal to zero. Specifically, let H_i, H_j be the two HOOFs of two patches P_i, P_j, and $\Pi_{\text{HOOF}}(P_i, P_j)$ be the motion similarity between the two patches P_i, P_j. Thus, we have:

$$\Pi_{\text{HOOF}}(P_i, P_j) = \frac{A_0}{A} + \frac{A_1}{A}\cdot\left[1 - D_{\text{HOOF}}(H_i, H_j)\right] \tag{10}$$

as shown in Fig. 5, where $A_0 = \min(A_{i,0}, A_{j,0})$, and $A_{i,0}$, $A_{j,0}$ are the numbers of the pixels (i.e., areas) in the two patches P_i, P_j whose OF values are equal to zero; $A_1 = \min(A_{i,1}, A_{j,1})$, and $A_{i,1}$, $A_{j,1}$ are the numbers of the pixels (i.e., areas) in the two patches P_i, P_j whose OF values are not equal to zero; and A is the area of P_i, P_j. Note that P_i and P_j have the same area (i.e., unit area).

Furthermore, combining the similarities of texture mode [i.e., $\Pi_{T,texture}$ computed by Eq. (7)] and the motion mode of two regions [i.e., Π_{HOOF} by Eq. (10)], we have the temporal similarity (variations in *XT* and *YT* planes of DT) as follows:

$$\Pi_T = \Pi_{T,texture} + \Pi_{\text{HOOF}}. \tag{11}$$

In addition, plugging Eqs. (6) and (11) into Eq. (4), we have the similarity between two regions as follows:

$$\begin{aligned}&\Pi(\varphi_s, \varphi_t, HOOF)\\ &= \omega_S\cdot[\omega_{XY}\cdot(\omega_{\text{LBP}}\cdot\Pi_{LBP,XY} + \omega_{\text{WLD}}\cdot\Pi_{WLD,XY})] + \omega_T\\ &\quad\cdot\{\omega_{XT}\cdot(\omega_{\text{LBP}}\cdot\Pi_{LBP,XT} + \omega_{\text{WLD}}\cdot\Pi_{WLD,XT}) + \omega_{YT}\cdot(\omega_{\text{LBP}}\\ &\quad\cdot\Pi_{LBP,YT} + \omega_{\text{WLD}}\cdot\Pi_{WLD,YT}) + \frac{A_0}{A} + \frac{A_1}{A}\cdot(1 - D_{\text{HOOF}})\}.\end{aligned} \tag{12}$$

In the following part of this paper, we use the weighted Weber distance to replace the term weighted Weber distance plus one extra bin for short.

IV. Segmentation Framework

In this section, we firstly describe the framework for the segmentation of DT. We then present how to choose the thresholds for this framework.

A. Segmentation Steps

As shown in Fig. 2, the segmentation method consists of three phases: hierarchical splitting, agglomerative merging and pixelwise classification, which follows the ideas proposed by [10]. We illustrate the framework of our method, as shown in Fig. 2. Given an input video, we firstly compute its features. Using these features, we perform the segmentation by hierarchical splitting, agglomerative merging and pixelwise classification. The details of this segmentation algorithm are shown in Algorithm 1.

Algorithm 1 Segmentation Framework

```
1: Input: A video consisting of several dynamic textures
2: Output: The boundary of each dynamic texture in the input video.
3: Step 1: Hierarchical splitting: Recursively split each input frame into square regions of varying size
    1) Given a region $\Omega_i$, and its four sub-regions $\Omega_{i,k}$ ($k = 0, 1, 2, 3$), compute the features of (LBP/WLD)$_{\text{TOP}}$ and HOOF of $\Omega_{i,k}$.
    2) Compute the distance $\boldsymbol{d}= \{ d_{\lambda,\pi}, d_{\text{HOOF}} | d_{\lambda,\pi} = 1\text{-}\Pi_{\lambda,\pi}, d_{\text{HOOF}} = 1\text{-}\Pi_{\text{HOOF}}, \lambda=$ LBP or WLD and $\pi = XY, XT$, or $YT\}^T$ between any two sub-regions of the region $\Omega_i$, where $\Pi_{\lambda,XY}$ is computed as in Eq. (5), and $\Pi_{\lambda,XT}$ or $\Pi_{\lambda,YT}$ is also computed as in Eq. (5), and $\Pi_{\text{HOOF}}$ is computed as in Eq. (10). Get a distance matrix $D$ (*i.e.*, $D = \{\boldsymbol{d}_0, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_5\}$) between these four sub-region $\Omega_{i,k}$.
    3) Compute the measures $R_j = D_{j-Max}/D_{j-Min}$, where $D_{j-Max}$ and $D_{j-Min}$ are the largest and smallest entries of each row of the distance matrix $D$.
    4) Split the region $\Omega_i$ into four sub-regions $\Omega_{i,k}$ if $R_j$ is greater than a threshold $X$:
           $R_j > X (j = 0, 1, \ldots, 5)$.
    The splitting step is stopped when none of the regions in one frame satisfy the criterion of $R_j > X$ or until a predetermined minimum patch size $S_{min}$ is reached.
4: Step 2: Agglomerative merging: Merge those similar adjacent regions/patches until a stopping criterion is satisfied
    1) Given any frame $F$ of a video and its regions $\Omega_k$ by splitting, compute the similarities $\Pi_{LBP,XY}$, $\Pi_{WLD,XY}$, and $\Pi_{\text{HOOF}}$ between any two neighboring regions ($\Omega_s$,$\Omega_t$).
    2) Construct the region pair set $\Psi = \{(\Omega_{s1},\Omega_{t1}), \ldots, (\Omega_{sN},\Omega_{tN})\}$, where each pair ($\Omega_{sr}$,$\Omega_{tr}$) satisfy the criterion: Either $\Pi_{LBP,XY}$, $\Pi_{WLD,XY}$, or $\Pi_{\text{HOOF}}$ between these two regions is larger than the threshold (i.e., $Y_{\text{LBP}}$, $Y_{\text{WLD}}$, or $Y_{\text{HOOF}}$, see Section IV-B for more details). Here $N$ is the number of the region pairs.
    3) Compute the merger importance ($MI$) of these region pairs in $\Psi$.
    4) Select the region pair ($\Omega_{sR}$,$\Omega_{tR}$) from $\Psi$ whose $MI$ is minimal and then merge them into one region $\Omega_R$.
        a) Update the features of the merged region $\Omega_R$ by accumulating the histograms of (LBP/WLD)$_{\text{TOP}}$ and HOOFs of two merged regions ($\Omega_{sR}$,$\Omega_{tR}$).
        b) Compute the similarities $\Pi_{LBP,XY}$, $\Pi_{WLD,XY}$, and $\Pi_{\text{HOOF}}$ between the merged region $\Omega_R$ and its neighboring regions.
        c) Update the region pair set $\Psi$.
    The merging step is stopped when there is no region pair in the set $\Psi$, i.e., $\Psi=\varphi$.
5: Step 3: Pixelwise classification: Improve the localization of the boundaries
    1) For each dynamic texture, get the boundary pixel set $\Phi$;
    2) Compute the subset of $\Phi_s = \{x_i\}$, for each $x_i \in \Phi_s$, at least one of its 4-connected neighbors has a different label.
    3) For each$x_i$, we do:
        a) Compute the appearance and motion features (i.e., (LBP/WLD)$_{\text{TOP}}$ and HOOF) over the circular neighbor $B(r)$ of $x_i$, where $r$ is the radius of the circular neighbor.
        b) Compute the similarities between the appearance and motion features of $x_i$ and the models of those neighboring regions (which are 4-connected to the pixel in query). The similarities computation is as mentioned in Eq. (4).
        c) Re-label the pixel $x_i$ if the label of the nearest model votes a different label from the current label of the pixel.
    4) In the next scan over the frame, we only check the neighborhoods of those pixels $x_i$, which were relabeled in previous sweep.
    5) The process of pixelwise classification continues until no pixels are relabeled (i.e., $\Phi_s = \varphi$) or the maximum number of sweeps (we set it as 11) is reached.
```

1) Splitting: We recursively split each input frame into square regions of varying size (see Fig. 2 and Algorithm 1). Here, a necessary prerequisite for the following merging to be successful is that the individual regions are uniform in texture. To this task, we use the following criterion to perform the splitting: if the appearance or motion votes for splitting of the current region, we perform splitting. This splitting procedure is repeated recursively on each sub-region until none of the regions in one frame satisfy the criterion of $R_j > X(X = 1.1)$ or a predetermined minimum patch size S_{min} is reached. We set $S_{min} = 16$ as [10].

2) Merging: After the input frame has been split into patches of roughly uniform texture, we merge those similar adjacent regions/patches until a stopping criterion is satisfied. The merger importance (MI) in Algorithm 1 is minimal in those region pairs which satisfy the condition i.e., the similarity between these two regions is larger than the threshold. Here, MI is defined as $MI= f(p) \times (1 - \Pi)$, where Π is the appearance and motion similarity between two regions computed by Eq. (4). The function $f(p)$ is the percentage of pixels in the smaller one of the two regions. It is computed as:$f(p)=N_b/N_f$, where N_b is the number of pixels in the

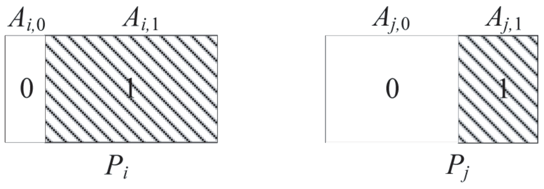


Fig. 5. Illustration of the area computation for OF equal to zero.

smaller one of the two regions, and N_f is the number of pixels in the current frame.

Before moving to the next merger, we compute the feature of each region by accumulating the histograms of (LBP/WLD)$_{\text{TOP}}$ and HOOFs of two merged regions. We then compute the similarities between the new region and its all adjacent regions using Eq. (4). The stopping rule for merging is that Π_{LBP}, Π_{WLD} and Π_{HOOF} between any two regions are smaller than the thresholds (i.e., Y_{LBP}, Y_{WLD}, and Y_{HOOF}), respectively.

3) Pixelwise Classification: We perform a simple pixelwise classification to improve the localization of the boundaries (see Fig. 2 and Algorithm 1). To this end, we switch into a DT classification mode by using the appearance and motion feature (i.e., (LBP/WLD)$_{\text{TOP}}$ histogram and HOOF) of the frame segments as the DT models. In our implementation, we set $r = 11$. How to set its value follows the same rule as S_{min}, as mentioned in Section IV-A-1).

Note that the splitting and merging here are performed in *xy* dimension and frame by frame. During the splitting/merging steps, both the spatial and temporal features of the dynamic textures are employed. Specifically, the appearance of DT is described by a local spatial texture descriptor $\phi_s = \{H_{LBP,XY}, H_{WLD,XY}\}$. The temporal features of the dynamic textures (i.e., the motion of DT) during segmentation is described by HOOF and local temporal texture descriptor, i.e., the *XT* and *YT* planes of the (LBP/WLD)$_{\text{TOP}}$, as discussed Section III-B.

B. Thresholds for LBP/WLD and HOOF

We address the problem of threshold selection by proposing a method for determining thresholds for the dynamic segmentation method by an offline supervised statistical learning.

1) Two Observations: Running a segmentation tool on a DT, one can have two observations as follows: (1) if two neighboring patches belong to the same texture category (e.g., patches 1 and 2 in Fig. 6), we should merge them into one region. (2) If the dominant parts of the two neighboring patches belong to the same texture (e.g., patches 3, 4 and 5 in Fig. 6), we should also merge them into one region. In real word, especially for a cluttered frame/image or patches close to the boundary of the two dynamic textures, neighboring patches might not contain completely the same texture due to the splitting method. Specifically, following the idea in [10], we split each frame of DT into patches of roughly uniform texture. However, the neighboring patches might not contain completely the same texture (e.g., patches 3, 4 and 5 in Fig. 6). One of them might contain several kinds of textures (e.g., Patches 4 and 5 contains both the *road* and *grass*), especially when it locates the boundary of two texture regions. However, the dominant

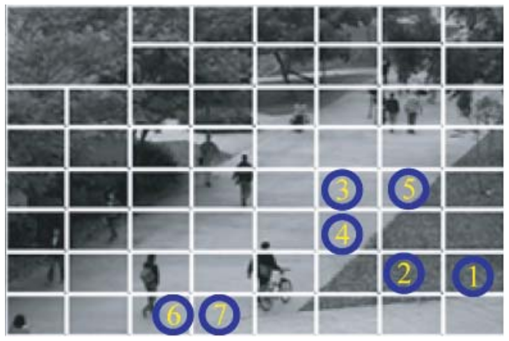


Fig. 6. Illustration of two observations during DT segmentation.

texture of these three patches might still be the same (e.g., the dominant texture of Patches 3, 4 and 5 is *road*). In this case, merging these neighboring patches sounds reasonable.

In our case, we use the first observation to determine the threshold by learning and use the second one to adjust the threshold for a better segmentation performance.

2) Threshold for LBP/WLD: In the following sub-section, we learn a threshold Y to determine when merging step is stopped. The intuitive idea is to learn the similarity difference between the intra-class and the inter-class when we use the descriptors LBP and WLD to represent features of textures. To construct a training dataset, given a dataset $D = \{D^0, D^1, \ldots D^{c-1}\}$, having N images and C classes. Here, each class D^i has N^i samples and $D^i = \left\{x_0^i, x_1^i, \ldots x_{N_i-1}^i\right\}$. To learn the threshold Y, we need to compute the within- and between-class similarities. As the feature, we use a local descriptor, e.g., LBP/WLD histogram. The similarity between any two images is computed by histogram intersection.

Let S_b and S_w denote the between- and within-class similarities of the images in the training set D. We have $S_w = \left\{\left\{s_{j,l}^i\right\}_i, i = 0, 1, \ldots c-1\right\}$, where $s_{j,l}^i = f(x_j^i, x_l^i)$, $x_j^i, x_l^i \in D^i$, j, $l = 0, 1, \ldots, N_i - 1$, $j \neq l$ and $f(.)$ is the function to compute the similarity between two images using histogram intersection. Intuitively, the set S_w is composed of c subsets. Each subset is a group of similarities of any two different images from the same class D^i. Thus, the cardinality of S_w is computed as

$$|S_w| = \frac{1}{2}\sum_{i=0}^{c-1}[N_i \times (N_i - 1)].$$

Likewise, S_b is computed as

$$S_b = \left\{\left\{s_{j,l}^{i,k}\right\}, i, k = 0, 1, c-1, \text{ and } i \neq k\right\}$$

where $s_{j,l}^{i,k} = f(x_j^i, x_l^k)$, $x_j^i \in D^i$, $x_l^k \in D^k$, $i \neq k$, and $j = 0, 1, \ldots, N_i - 1$; $l = 0, 1, \ldots, N_k - 1$. Intuitively, the set S_b is also composed of c subsets. Each subset is a group of similarities of two images but from different classes. Thus, we have

$$|S_b| = \frac{1}{2}\sum_{i=0}^{c-1}\left(N_i \times \sum_{k=0, k\neq i}^{c-1} N_k\right).$$

After obtaining the two sets S_b and S_w, we compute the distributions of these similarities in these two sets. In our

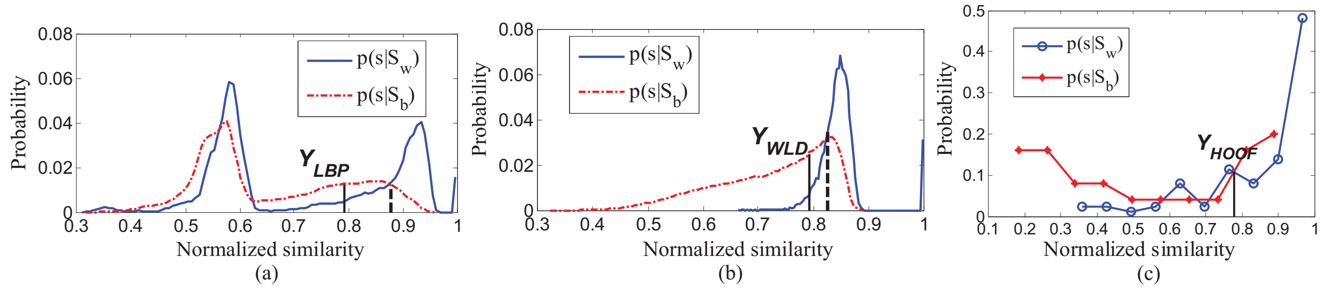


Fig. 7. Learning thresholds on Brodatz texture dataset for (a) LBP and for (b) WLD, and on the synthesized database for (c) HOOF.

case, shown in Fig. 7(a) and (b), we use histograms to describe similarity distributions. Here, x-axis denotes the normalized similarity; y-axis denotes the similarity probability i.e., $P(s|S_w)$ and $P(s|S_b)$, which are approximated by the normalized frequencies (the normalized factors use the cardinalities of S_b and S_w computed above).

Ideally, we can use the point according to the minimum error rate (i.e., the dash line) as the threshold. However, we adjust the threshold for obtaining better performance according to the Observation 2 in Section IV-B-1). In addition, we learn two thresholds (i.e., Y_{LBP} and Y_{WLD}) since we use two kinds of texture features (i.e., LBP and WLD).

3) Threshold for HOOF: In this sub-section, we learn a threshold Y for HOOF to determine when merging step is stopped. The intuitive idea is to learn the similarity difference between the patch motions from the same dynamic textures and that of between the patch motions from the different dynamic textures. To construct a training dataset to learn the threshold Y, for a given dataset $D = \{D^0, D^1, \ldots D^{N-1}\}$, which is composed of N dynamic texture videos, each consisting of N_i frames, we first divide each frame into patches evenly (16×16 pixels). We then manually label the relation between neighboring patches P_i and P_j, e.g, $\{(P_i, P_j, \omega_{ij})\}$, where $\omega_{ij} \in \{0, 1\}$. In other words, if the two patches show a similar motion mode (and also similar appearance) which should be merged into one region, we label ω_{ij} =1, or 0 otherwise. Subsequently, we compute the HOOFs for all these patches and the similarities for each patch pair.

To learn the threshold Y_{HOOF}, we need to compute the within- and between-class similarities. Let S_b and S_w denote the between- and within-class similarities of the patch pairs in the set D. We have $S_w = \{s_{ij}\}$, where $\omega_{ij} = 1$, and $s_{ij} = f(P_i, P_j)$, which is computed by weighted Weber distance. Likewise, S_b is computed as $S_b = \{s_{ij}\}$, where $\omega_{ij} = 0$ and $s_{ij} = f(P_i, P_j)$.

After obtaining the two sets S_b and S_w, we compute the distributions of these similarities in these two sets. In our case, we also use the histogram to describe similarity distributions (refer to Fig. 7(c)). Likewise, x-axis denotes the normalized similarity; y-axis denotes the similarity probability (i.e., $P(s|S_w)$ and $P(s|S_b)$), which are approximated by the normalized frequencies (the normalized factors are the cardinalities of S_b and S_w, respectively). Similar to Y_{LBP} and Y_{WLD}, Y_{HOOF} also does not take the value according to the minimum error rate and its value is adjusted for obtaining better performance.

V. Experiments

In this section, we first introduce how to learn the thresholds. After that, we discuss the function of each component in our framework for the dynamic texture segmentation. Finally, we compare our method with the state-of-the-art methods.

A. Threshold Computing

1) Thresholds for LBP/WLD: As discussed in Section IV, we compute the two thresholds (i.e., Y_{LBP}, Y_{WLD}) by texture classification on a given dataset. In our case, we use the Brodatz texture database [32]. The images are 256×256 pixels in size, and they have 256 gray levels. It comprises 2,048 samples, with 64 samples in each of 32 texture categories.

For the two features, LBP and WLD, the distributions of S_b and S_w on the Brodatz dataset are shown in Fig. 7(a) and (b), where the dash lines correspond to the points of minimum error. However, we use those points where the solid lines locate, and set $Y_{\mathrm{LBP}} = Y_{\mathrm{WLD}} = 0.79$, as shown in Fig. 7(b). Here, both the values of Y_{LBP} and Y_{WLD} are smaller than the points according to the minimum error considering the noise in the video. For example, if the appearance of the dominant parts of the two neighboring patches belongs to the same texture (e.g., patches 3, 4 and 5 in Fig. 6), we should also merge them into one region. However, in this case, the similarity between these two patches would be smaller than those two patches which show the same texture (i.e., appearance). To merge these two patches whose dominant textures are the same, we reduce the values of Y_{LBP}, Y_{WLD} according to the Observation 2 in Section IV-B-1).

2) Threshold for HOOF: As discussed in Section IV-B, we compute the threshold (i.e., Y_{HOOF}) on a training set of dynamic texture dataset. The training set is from the synthesized database [4]. The database includes 300 clips, three groups of 100 videos. Each group consists of $K = \{2,3,4\}$ motion segments. Each clip has 60 frames and each frame size is 160×110. We extract 100 clips randomly and two neighboring frames from each clip for training, and these clips are not used as test clips in the following experiments. We first split the selected clips evenly (16×16 for each patch). We then label the neighboring patch as $\{(P_i, P_j, \omega_{ij})\}$. Subsequently, we compute the within- and between-class similarities S_b and S_w and describe similarity distributions by the histograms shown in Fig. 7(c). In our case, we set $Y_{\mathrm{HOOF}} = 0.79$ for simplicity, which is equal to Y_{LBP} and Y_{WLD}.

In Fig. 8, we show the variations of the similarities of two merged patches in one frame during the merging step on a DT.

In Fig. 8(e), we conclude the similarities (i.e., $\Pi_{LBP,XY}$, $\Pi_{WLD,XY}$, and Π_{HOOF}) over the iterations of merging, and mark a block in shadow to denote the similarity of a feature smaller than the learned threshold (we call it *missed* for short), and mark a block in white to denote the similarity of a feature larger than the learned threshold. From this figure, one can find that the appearance of DT (using the feature of WLD and LBP) and the motion of DT (using the feature of HOOF) are also well complementary. In addition, for the variations of $\Pi_{LBP,XY}$ during merging process, one possible reason is the cluttering background of this sequence.

From Fig. 8(d), one can find that MI ratio (MIR), employed by [10], is smaller than 1.5 even when the merging step stops. This shows that using MIR fails for this DT.

B. Comparison of Distance Measure

As shown in Fig. 9, we test the following six measures: Weber distance, weighted Weber distance, weighted Weber distance with one extra bin for still pixels, normalized histogram intersection, Chi-square distance and Kullback–Leibler (KL) divergence measure. Given two normalized histograms a and b, the Chi-square distance is defined as $\chi^2(a,b) = \sum_i \frac{(a_i-b_i)^2}{a_i+b_i}$. The KL measure is defined as $D_{KL}(a,b) = [D_{KL}(a||b) + \mathrm{D}_{KL}(b||a)]/2$ and $D_{KL}(a||b) = \sum_i a_i \log(a_i/b_i)$. The dataset is the training set used in Section V-A for the threshold learning of Y_{HOOF}.

For the normalized histogram intersection and Chi-square distance, from Fig. 9(d) and (e), we can find that the two curves $P(s|S_w)$ and $P(s|S_b)$ vary irregularly. As shown in Fig. 9(a), Weber distance increases the probabilities $P(s|S_w)$ of larger similarities. In Fig. 9(b), the weighted Weber distance decreases the probabilities $P(s|S_b)$ of larger similarities and increases the probabilities $P(s|S_b)$ of small similarities. In Fig. 9(c), the weighted Weber distance with one extra bin for still pixels significantly increases the probabilities $P(s|S_w)$ of larger similarities and, in contrast, significantly decreases the probabilities $P(s|S_w)$ of small similarities. In addition, compared to histogram intersection and Chi-square measure, KL measure increases the probabilities of larger similarities for both $P(s|S_w)$ and $P(s|S_b)$, and also decreases the probabilities of small similarities for both $P(s|S_w)$ and $P(s|S_b)$. However, these two curves for $P(s|S_w)$ and $P(s|S_b)$ are too close, which shows KL measure is not suitable for HOOF during DT segmentation.

C. Qualitative Evaluation

We compare the performance of $(LBP/C)_{\mathrm{TOP}}$, $(LBP/WLD)_{\mathrm{TOP}}$, HOOF and the proposed framework over the public videos. Some resulting example frames are shown in Fig. 10. Specifically, for the first column, we combine LBP and contrast for DT segmentation. Here, the contrast (C) is a simple contrast measure of a local neighbor (e.g., 3×3, the same as LBP), which is the difference between the average gray-level of those pixels whose values are larger than the center pixel and those whose values are smaller than the center pixel [10]. For the contrast, we compute its histogram as WLD and use the same similarity function as WLD. For the second column, we combine LBP and WLD instead of contrast. In addition, both $(LBP/C)_{\mathrm{TOP}}$ and $(LBP/WLD)_{\mathrm{TOP}}$ employ the normalized histogram intersection as a distance measure. For the fourth column, we use only HOOF and weighed Weber distance. For the last column we use $(LBP/WLD)_{\mathrm{TOP}}$ and HOOF to represent DT; and we employ normalized histogram intersection for $(LBP/WLD)_{\mathrm{TOP}}$ and weighed Weber distance for HOOF.

Fig. 10(a) is a frame from *vidf1_33_000.y*. One can find that both $(LBP/WLD)_{\mathrm{TOP}}$ and $(LBP/WLD)_{\mathrm{TOP}}$ + HOOF preserve correctly the small patches which are different from the neighbors. However, $(LBP/C)_{\mathrm{TOP}}$ fails in segmenting the two passengers in the *road*. HOOF performs poor because the OF values in the motion region of this video are small.

From Fig. 10(b), one can find that $(LBP/WLD)_{\mathrm{TOP}}$ + HOOF works best, which segments the three synthesized regions successfully. In comparison, $(LBP/WLD)_{\mathrm{TOP}}$ segments two of them because the top-left region has the same texture appearance (water flow but in different motion field) as the background. In addition, $(LBP/C)_{\mathrm{TOP}}$ segments only one of the regions for this sequence. The HOOF performs poor again.

Furthermore, we also test (LBP/WLD)$_{\mathrm{TOP}}$ but using weighted Weber distance instead of normalized histogram intersection. The experimental results of (LBP/WLD)$_{\mathrm{TOP}}$ using weighted Weber distance are also shown in the third column of Fig. 10 (i.e., $(LBP/WLD)_{\mathrm{TOP}}$-*Weber*). We can find that $(LBP/WLD)_{\mathrm{TOP}}$-*Weber* works comparably to (LBP/WLD)$_{\mathrm{TOP}}$ (using the normalized histogram intersection).

The regular horizontal and vertical boundaries around the frame border are due to the radius of the circular neighbor being equal to 11, which stops the pixel classification after the merging step (see Section IV-A-3).

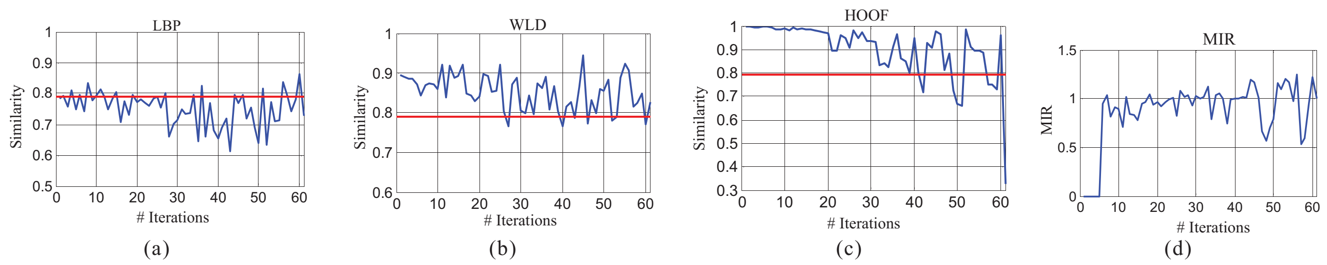
D. Level Set Scheme

We test the optical flow plus level set for the segmentation of DT. For the level set scheme, we use the method proposed by Li et al. [33]. Different from Li et al. [33] who used the gradient flow, we use the optical flow as input for the level set scheme instead.

In Fig. 11, we show the effect of different initializations for the level set method. The level set method performs very well (e.g., Fig. 11(a)) if the initialization is good. However, the performance of segmentation degrades when the initialization becomes biased (e.g., Fig. 11(b) and (c)).

E. Comparison With Existing Methods

In Fig. 12, we present some results performed on various types of sequences, and a comparison with existing methods. Specifically, *DytexMixIC* and *DytexMixCS* are methods proposed by [4]. *Ising* uses the level-sets method and Ising models [24]. *GPCA* is the generalized principle component analysis used for DT segmentation [18]. *NormCuts* is based on normalized cuts and “motion profile” representation proposed in [36] and [34]. *OpFlow* represents each pixel as a feature-vector containing the average optical flow over a 5×5 window and clusters the feature-vectors using the mean-shift algorithm [37]. These existing methods



#Iterations	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
LBP	■		■		■		■		■			■	■	■		■	■	■		■	■	■	■	■		■		■	■	■
WLD																											■			
HOOF																														

#Iterations	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61
LBP	■	■	■		■		■	■	■	■	■	■	■		■		■	■	■	■		■	■	■	■			■	■		■
WLD										■			■			■						■	■							■	
HOOF												■							■	■	■						■	■	■		■

(e)

Fig. 8. Illustration of the variations in similarities of two merged patches in one frame and MIR during the merging step on a DT. (a)–(d) Variations of the similarities between two merged patches in one frame for the features of LBP and WLD in XY plane of DT, HOOF (i.e., $\Pi_{\mathrm{LBP},XY}$, $\Pi_{\mathrm{WLD},XY}$, and Π_{HOOF}), and MIR, respectively. (e) Iterations in which the corresponding similarities (i.e., $\Pi_{\mathrm{LBP},XY}$, $\Pi_{\mathrm{WLD},XY}$, and Π_{HOOF}) are smaller than the learned thresholds. The shaded block denotes a missed feature, and the white block denotes a feature works well (the similarity is larger than the threshold).

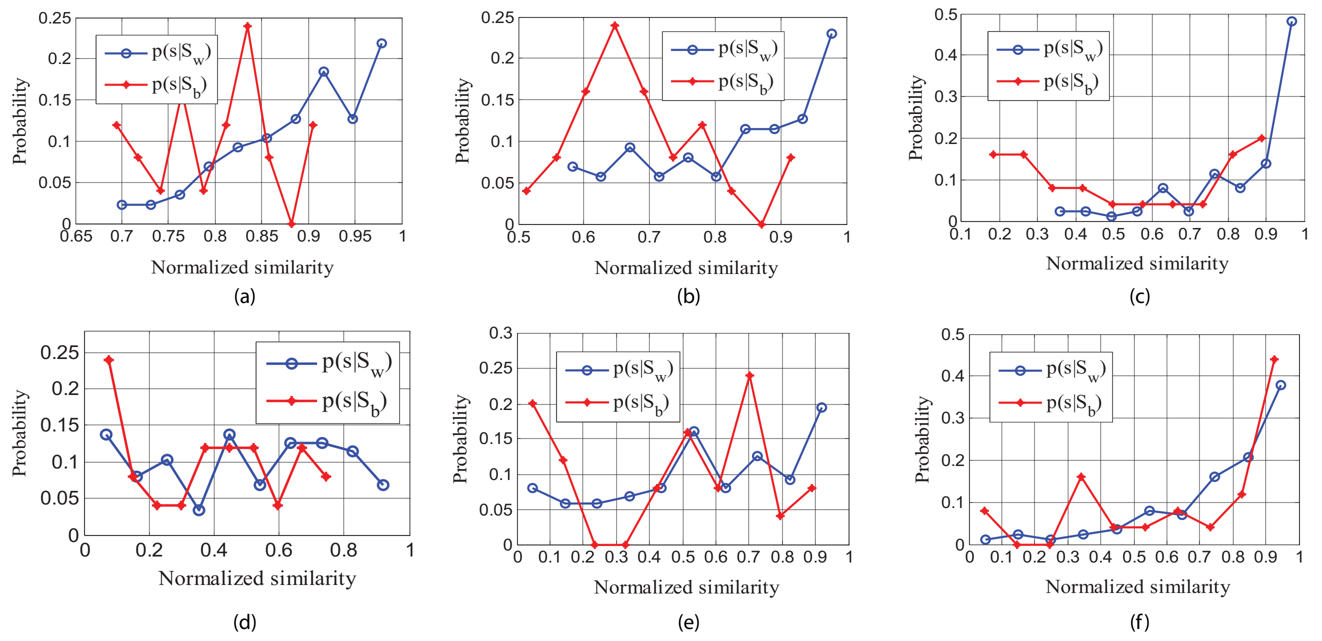


Fig. 9. Comparison of different measures for the computation of the HOOF similarity. (a) Weber distance. (b) Weighted Weber distance. (c) Weighted Weber distance with one extra bin for still pixels. (d) Normalized histogram intersection. (e) Chi-square. (f) KL measure.

are implemented by Chan and Vasconcelos [4], and the results shown here are extracted from their website: http://www.svcl.ucsd.edu/projects/motiondytex/demo2.htm.

In Fig. 12, we compare our method with existing methods on more sequences. Specifically, for the sequence *vidf1_33_006.y* (the first row), our framework works comparably with *DyTexMixCS*, *NormCuts* and better than *OpFlow*. One reason for the poor performance of *OpFlow* is resulted from optical flow noise, caused by the computed methods and the brightness variations of the sequence. Although we also use HOOF to represent the motion of this sequence, we conclude that the weighted Weber distance overcomes the noise significantly.

For the sequence *texture_004.y* (the second row), our framework segments the three synthesized regions successfully. *GPCA* [18], which also uses optical flow to describe the motion of this sequence, fails. One reason for the failure of GPCA over this sequence is also resulted from the noise among the optical flow values. For the sequence *ocean-fire-small* (the third row), *DyTexMixCS* and our framework work comparably and better than *Ising*. For the sequence *ocean-steam-small* (i.e., the fourth row), all the methods work well

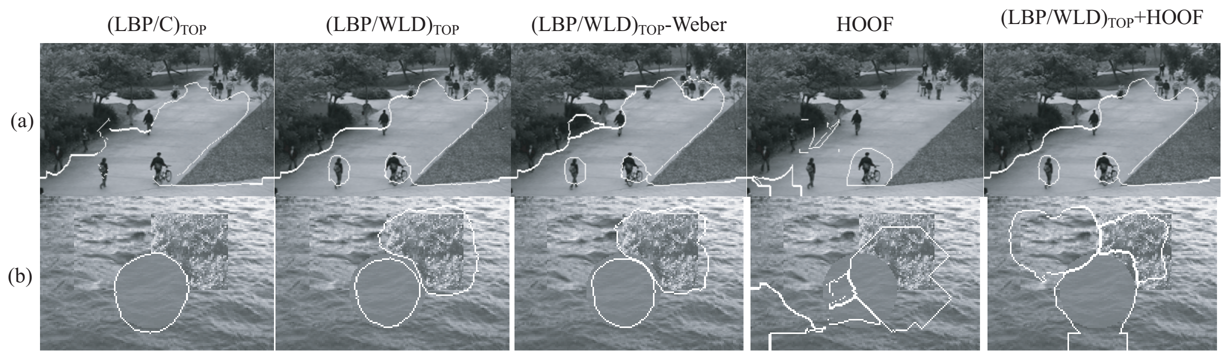


Fig. 10. Illustration of DT segmentation of (a) *vidf1_33_000.y* [4] and (b) *texture_004.y* [4].

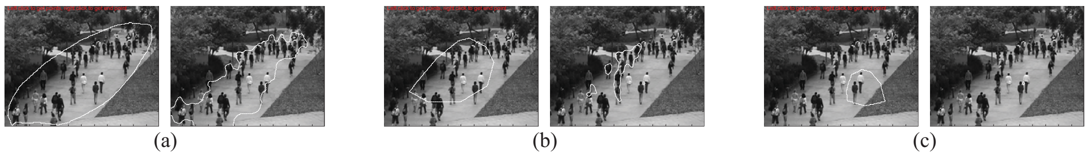


Fig. 11. Experimental results on the *vidf1_33_006.y* sequence [4] using optical flow and level set scheme but using different initializations. (a) Good initialization. (b) and (c) Biased initialization.

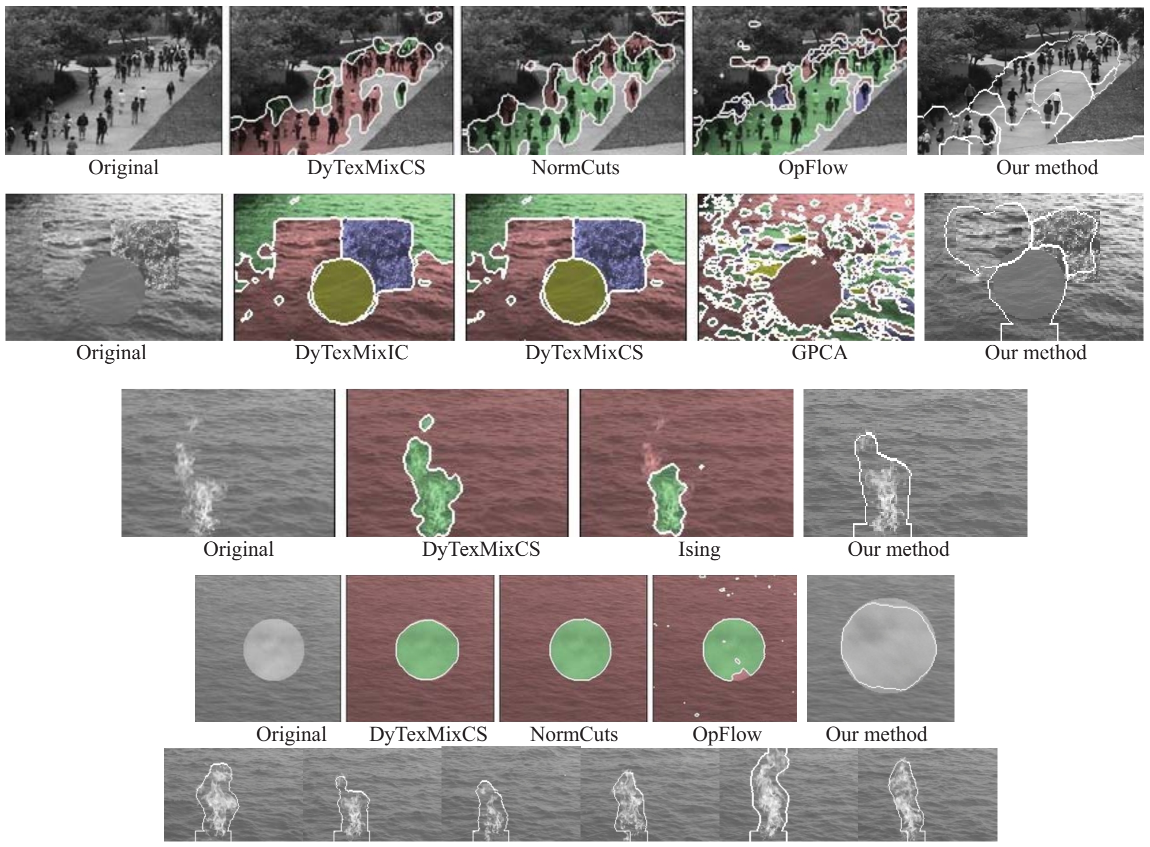


Fig. 12. Some experimental results compared with existing methods. The first row shows sequence *vidf1_33_006.y*, and the second row shows sequence *texture_004.y* [4]. The third row shows sequence *ocean-fire-small*, and the fourth row shows sequence *ocean-steam-small* [17]. The last row shows the experimental results of our method on the sequences of *ocean-fire-small* on, from left to right, frames 4, 12, 27, 47, 78, and 105.

except *OpFlow*. In the last row of Fig. 12, we show more experimental results of our method on the sequences of *ocean-fire-small*. From this figure, one can find that our method segments this sequence quite accurately.

In Fig. 13, we present some results performed on the sequences in the DynTex dataset [35], and a comparison with the methods proposed in [26]. The sequences are comprised of dynamic textures in a natural context (flowing water, smoke

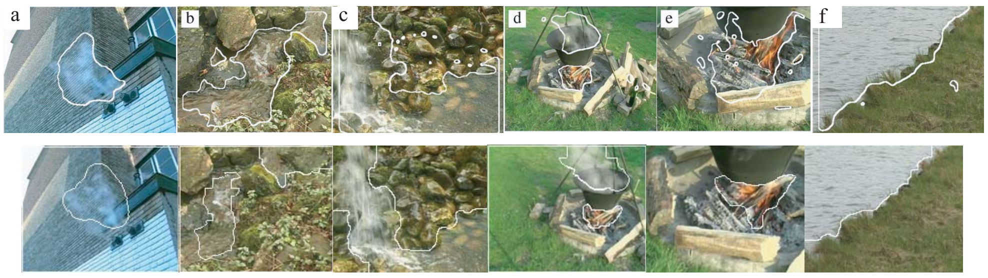


Fig. 13. Some experimental results compared with existing method. These samples are taken from the DynTex database [35]. (a) Sequence 648ea10, which depicts steam (or smoke) coming from a ventilation system. (b) Sequence 6483c10, which depicts a narrow creek winding between larger pebbles. (c) Sequence 6481i10, which depicts water falling over round pebbles and gathering into a pool. (d) Sequence 73v195t, which depicts fire, smoke, and steam (while cooking goulash). (e) Sequence 73v1985, which depicts fire and smoke [close-up of (d)]. (f) Sequence 647bc10 , which depicts a ship, canal, and shoreline in Amsterdam. The sequence in (a) was a shot with a stationary camera, and the sequences in (b)–(f) were shot with a moving camera. The upper row shows the results of [26], and the bottom row shows the results of our method.

and fire), shot with a moving or stationary camera. Here, the results of [26] are taken from the paper directly. From this figure, one can find that our method performs the dynamic texture segmentation quite competitively in comparison to the method of [26].

From Fig. 13, one can find that our method is robust to the camera motion in relation to static/dynamic texture. Specifically, for the static texture, the camera motion does not change the texture appearance and the feature of *XY* plane in $(\text{LBP/WLD})_{\text{TOP}}$ can model the texture appearance well, as shown in [11]. In addition, the motion of still texture can be well modeled by HOOF because all the pixels have the same motion. For the dynamic texture, the camera motion does not change the texture appearance, thus the feature of *XY* plane in $(\text{LBP/WLD})_{\text{TOP}}$ works as well. On the other hand, camera motion changes the positions of the dynamic textures in the following frames, which would change the values of (LBP/WLD) in the *XT* and *YT* planes. The pixel intensity variations of the following frames, resulted from the camera motion, can be regarded as noise. As shown in [8], [11], LBP is robust to the monotonic gray-scale changes, and WLD is robust to scale and translation variations of gray level. Furthermore, HOOF plus Weber distance is robust to the motion of the camera in relation to dynamic texture. Specifically, although the HOOF is also changed due to the camera motion, the proposed Weber distance improves the robustness of our method because the difference between each bin of two HOOFs is

$$k_i = \max\left\{\frac{|a_i - b_i|}{C + b_i}, \frac{|a_i - b_i|}{C + a_i}\right\} \tag{13}$$

which moves the additive motion resulted from the camera motion.

In addition, we also perform qualitative evaluation on the synthetic sequences [4]. An example frame is shown in the second row of Fig. 12. The database also provides the initial contours to the segmentation algorithms.

In Table 1, we compare the performance our method with the existing algorithms. Here, the two baseline segmentations are also included 1) "Baseline Random," which randomly assigns pixels and 2) "Baseline Init," which is the initial segmentation (that is, initial contour) provided to the algorithms.

TABLE I
BEST AVERAGE RAND INDEX FOR EACH SEGMENTATION ALGORITHM ON THE SYNTHETIC DATABASE

Algorithm	K = 2	K = 3	K = 4
LDT [20]	0.944(05)	0.894(12)	0.916(20)
DytexMixIC [4]	0.915 (17)	0.853 (15)	0.868 (15)
DytexMixCS [4]	0.915 (20)	0.825 (10)	0.835 (15)
Ising [15]	0.879 (05)	N/A	N/A
GPCA [34]	0.548 (02)	0.554 (17)	0.549 (10)
Baseline Rand.	0.607	0.523	0.501
Baseline Init	0.600	0.684	0.704
Our Method	0.924	0.884	0.855

(The number in the parenthesis is *n*, i.e., the dimension of state space).

We only use 200 videos of this dataset for testing because we use 100 sequences for threshold learning of Y_{HOOF}.

In this table, the results of LDT, DTM (DytexMixIC and DytexMixCS), Ising and GPCA are taken from [4] and [20] directly. These methods were run for several values of the state-space dimension (i.e., n) and the average rand index was computed for each K. The performance in this table shows the performance obtained by each algorithm with the best n. From the table, one can find that our method obtains competitive performance although we do not use the initial contours provided by this database.

F. Parameters and Distance Measure Evaluation

There are several parameters involved in our method, i.e., three weights, i.e., $\omega = \{\omega_{XY}, \omega_{XT}, \omega_{YT}\}$ and three thresholds $\boldsymbol{Y} = \{Y_{\text{LBP}}, Y_{\text{WLD}}, Y_{\text{HOOF}}\}$. Meanwhile, $\omega = \{\omega_{XY}, \omega_{XT}, \omega_{YT}\} = \{0.625, 0.25, 0.125\}$, is taken directly from [12], which are obtained statistically by dynamic texture recognition. $\boldsymbol{Y} = \{Y_{\text{LBP}}, Y_{\text{WLD}}, Y_{\text{HOOF}}\} = \{0.79, 0.79, 0.79\}$ are learned by an offline supervised statistical learning.

We perform the experiments on the subset $K = 2$ of the synthetic sequences [4] to observe the sensitivity of our method. Here, for the values of $\omega = \{\omega_{XY}, \omega_{XT}, \omega_{YT}\}$, we use the same setups as in [12]. We show these selected groups

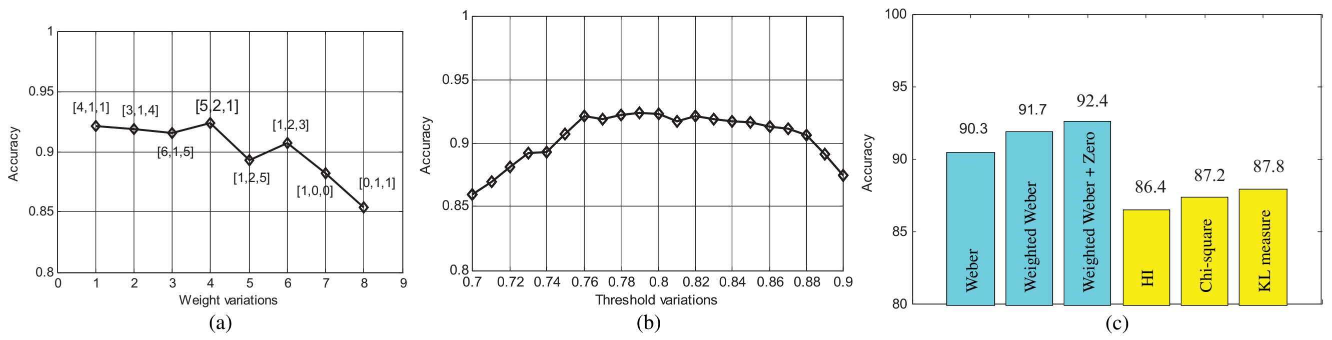


Fig. 14. Parameter evaluation on the subset $K = 2$ of the synthetic sequences [4]. (a) and (b) Performance variations according to the weight and threshold variations. (c) Performance comparison of different distance measures.

of weights in Fig. 14 (a), where each group of weights is the normalized values of $[x_1, x_2, x_3]$, i.e., $x_i/(x_1 + x_2 + x_3)$. The experimental results are shown in Fig. 14(a). From this figure, one can find that the weights [5, 2, 1] performs the best. In addition, the points [1, 0, 0] and [0, 1, 1] correspond to the cases of only using *XY* plane (pure texture), or only *XT*, *YT* planes (pure temporal).

In Fig. 14(b), we show the accuracy variations according to the threshold variations. Here, we have the three thresholds taken the same value for simplicity's sake. From this figure, one can find the performance does not change significantly in the interval [0.76, 0.87]. In addition, the point that the thresholds take 0.79 works comparable to the neighboring values in the interval [0.76, 0.87].

In addition, we test different distance measures discussed in Section III-C and the results are shown in Fig. 14(c). Here, *Weber* denotes Weber distance; *Weighted Weber* denotes Weighted Weber distance; *Weighted Weber +Zero* denotes Weighted Weber distance with one extra bin for still pixels; *HI* denotes normalized histogram intersection; *Chi-square* denotes Chi-square distance and so for *KL measure*. The feature is $(\text{LBP/WLD})_{\text{TOP}}$+HOOF and these distance measures are used for HOOF. From this figure, one can find that *Weighted Weber +Zero* works better than other distance measures.

VI. Conclusion

We proposed a new framework for dynamic texture segmentation based on spatiotemporal features. For the spatial mode, we employed a new texture feature to characterize each region of a frame of DT, i.e., the histograms of LBP and WLD features in the *XY* plane of DT. For the temporal mode, we use the optical flow and the histograms of LBP and WLD features in *XT* and *YT* planes of DT to describe its motion field. We also addressed the problem of choosing thresholds for the segmentation framework. In addition, we developed a weighted Weber distance measure, which is computationally simple. Experimental results and comparison with existing methods show that our method is effective for DT segmentation. Furthermore, our method performs well on sequences with cluttered background.

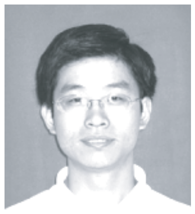
Acknowledgment

The authors would like to thank R. Péteri, G. Doretto, and A. Chan for sharing the datasets that were used in this paper.

References

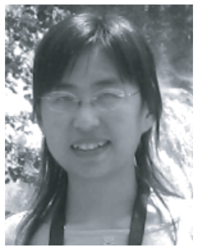
[1] D. Chetverikov and R. Péteri, "A brief survey of dynamic texture description and recognition," in *Proc. 4th Int. Conf. Comput. Recognit. Syst.*, 2005, pp. 17–26.
[2] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *Int. J. Comput. Vis.*, vol. 51, no. 2, pp. 91–109, 2003.
[3] M. Szummer and R. W. Picard, "Temporal texture modeling," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 1996, pp. 823–826.
[4] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 909–926, May 2008.
[5] T. Amiaz, S. Fazekas, D. Chetverikov, and N. Kiryati, "Detecting regions of dynamic texture," in *Proc. Conf. Scale Space Variat. Methods Comput. Vis.*, 2007, pp. 848–859.
[6] D. W. Murray and B. F. Buxton, "Scene segmentation from visual motion using global optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 2, pp. 220–228, Mar. 1987.
[7] R. Polana and R. Nelson, "Temporal texture and activity recognition," in *Motion-Based Recognition*. Norwell, MA: Kluwer, 1997.
[8] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[9] J. Chen, G. Zhao, and M. Pietikäinen, "An improved local descriptor and threshold learning for unsupervised dynamic texture segmentation," in *Proc. 12th IEEE Int. Conf. Comput. Vis. Workshop*, Oct. 2009, pp. 460–467.
[10] T. Ojala and M. Pietikäinen, "Unsupervised texture segmentation using feature distributions," *Pattern Recognit.*, vol. 32, no. 3, pp. 477–486, 1999.
[11] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, and W. Gao, "WLD: A robust local image descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1705–1720, Sep. 2010.
[12] G. Zhao and M. Pietikäinen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 6, pp. 915–928, Jun. 2007.
[13] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and Binet–Cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1932–1939.
[14] A. K. Jain, *Fundamentals of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[15] M. Fritz, B. Leibe, B. Caputo, and B. Schiele, "Integrating representative and discriminant models for object category detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2005, pp. 1363–1370.
[16] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 623–656, Jul. & Oct. 1948.
[17] G. Doretto, D. Cremers, P. Favaro, and S. Soatto, "Dynamic texture segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 1236–1242.
[18] R. Vidal and A. Ravichandran, "Optical flow estimation & segmentation of multiple moving dynamic textures," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 516–521.
[19] L. Cooper, J. Liu, and K. Huang, "Spatial segmentation of temporal texture using mixture linear models," in *Proc. Int. Conf. Dynamical Vis.*, 2005, pp. 142–150.

[20] A. B. Chan and N. Vasconcelos, "Layered dynamic textures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1862–1879, Oct. 2009.
[21] A. B. Chan and N. Vasconcelos, "Variational layered dynamic textures," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1062–1069.
[22] A. Rahman and M. Murshed, "Detection of multiple dynamic textures using feature space mapping," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 5, pp. 766–771, May 2009.
[23] S. Milko, E. Samset, and T. Kadir, "Segmentation of the liver in ultrasound: A dynamic texture approach," *Int. J. Comput. Assist. Radiol. Surgery*, vol. 3, nos. 1–2, pp. 143–150, 2008.
[24] A. Ghoreyshi and R. Vidal, "Segmenting dynamic textures with ising descriptors, ARX models and level sets," in *Proc. Eur. Conf. Comput. Vis. Dynamical Vis. Workshop*, 2006, pp. 127–141.
[25] R. Vidal and D. Singaraju, "A closed form solution to direct motion segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 510–515.
[26] S. Fazekas T. Amiaz, D. Chetverikov, and N. Kiryati, "Dynamic texture detection based on motion analysis," *Int. J. Comput. Vis.*, vol. 82, no. 1, pp. 48–63, 2009.
[27] D. Chetverikov, S. Fazekas, and M. Haindl, "Dynamic texture as foreground and background," *Mach. Vis. Appl.*, vol. 22, no. 5, pp. 741–750, 2011.
[28] A. Stein and M. Hebert, "Occlusion boundaries from motion: Low-level detection and mid-level reasoning," *Int. J. Comput. Vis.*, vol. 82, no. 3, pp. 325–357, 2009.
[29] K. Derpanis and R. Wildes, "Early spatiotemporal grouping with a distributed oriented energy representation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 232–239.
[30] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. Eur. Conf. Comput. Vis.*, 2004, pp. 25–36.
[31] J. Kim and K. Grauman, "Observe locally, infer globally: A space-time MRF for detecting abnormal activities with incremental updates," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 2921–2928.
[32] T. Ojala, K. Valkealahti, E. Oja, and M. Pietikäinen, "Texture discrimination with multidimensional distributions of signed gray level differences," *Pattern Recognit.*, vol. 34, no. 3, pp. 727–739, 2001.
[33] C. Li, C. Xu, C. Gui, and M. D. Fox, "Level set evolution without re-initialization: A new variational formulation," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 430–436.
[34] J. Shi and J. Malik, "Motion segmentation and tracking using normalized cuts," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jan. 1998, pp. 1154–1160.
[35] R. Péteri, M. Huskies, and S. Fazekas. (2006). *DynTex: A Comprehensive Database of Dynamic Textures* [Online]. Available: http://www.cwi.nl/projects/dyntex
[36] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[37] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.



Jie Chen (M'04) received the M.S. and Ph.D. degrees from the Harbin Institute of Technology, Harbin, China, in 2002 and 2007, respectively.

He has been a Senior Researcher with the Center for Machine Vision Research, University of Oulu, Oulu, Finland, since September 2007. He has authored or co-authored more than 30 papers in journals and conferences. His current research interests include pattern recognition, computer vision, machine learning, dynamic textures, and watermarking.



Guoying Zhao (SM'12) received the Ph.D. degree in computer science from the Chinese Academy of Sciences, Beijing, China, in 2005.

She was a Senior Researcher with the Center for Machine Vision Research, University of Oulu, Finland, from 2005 to 2010, where she has been an Adjunct Professor since 2010. She has authored or co-authored more than 70 papers in journals and conferences, and has served as a reviewer for many journals and conferences. She has lectured tutorials at the International Conference on Pattern Recognition in 2006 and the International Conference on Computer Vision (ICCV) in 2009. She has authored or edited three books and two special issues for respected journals. Her current research interests include gait analysis, dynamic-texture recognition, facial-expression recognition, human motion analysis, and person identification.

Dr. Zhao was a Co-Chair of the International Workshop on Machine Learning for Vision-based Motion Analysis (MLVMA) at ECCV2008, ICCV2009, and CVPR2011. She is an Editorial Board Member of the *International Journal of Applied Pattern Recognition*.

Mikko Salo is a Professor with the Department of Mathematics and Statistics, University of Jyväskylä, Jyväskylä, Finland. His work is in mathematical analysis and its applications. His current research interests include inverse problems for partial differential equations and applications in medical imaging, inverse problems in differential geometry and applications in seismic imaging, and geometric invariants in pattern recognition and applications in computer vision.

He is the Managing Editor of *Inverse Problems and Imaging*.

Esa Rahtu received the M.S. and Doctor of Science in Technology degrees (Hons.) from the University of Oulu, Oulu, Finland.

He was a Post-Doctoral Researcher with the Academy of Finland, Helsinki, Finland, and he is currently a Senior Researcher with the Center for Machine Vision Research, University of Oulu. His current research interests include feature extraction, geometric invariants, object detection and recognition, and medical imaging.

Dr. Rahtu was a recipient of the Ph.D. Thesis Prize from the Pattern Recognition Society of Finland.

Matti Pietikäinen (SM'95–F'12) received the Dr.Sc.Techn. degree from the University of Oulu, Oulu, Finland, in 1982.

He is a Professor of information engineering, the Scientific Director of Infotech Oulu, and the Director of the Center for Machine Vision Research, University of Oulu. He has made pioneering contributions to local binary pattern methodology, texture-based image and video analysis, and facial-image analysis. He has authored or co-authored more than 260 refereed scientific papers, and authored the book *Computer Vision Using Local Binary Patterns* (Springer, 2011).

Dr. Pietikäinen was an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and the Pattern Recognition Journal, and is currently as an Associate Editor of the *Image and Vision Computing Journal*. He is a fellow of the International Association for Pattern Recognition.