Unsupervised Dynamic Texture Segmentation Using Appearance and Motion

Jie Chen, Guoying Zhao, Mikko Salo, Esa Rahtu, and Matti Pietikäinen
Machine Vision Group, Department of Computer Science and Engineering,
P. O. Box 4500 FI-90014 University of Oulu, Finland
E-mail:{jiechen, gyzhao, erahtu, mkp}@ee.oulu.fi, msa@rni.helsinki.fi

Abstract Dynamic texture (DT) is an extension of texture to the temporal domain. How to segment DTs is a challenging problem. In this paper, we address the problem of segmenting DT into disjoint volumes in an unsupervised way. DTs might be different from their 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 DT; for the motion mode, we use the optical flow and the local temporal texture descriptor to represent the temporal variations of 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 Law. Each volume is characterized by its appearance and motion modes. Furthermore, we also address the important problem of threshold selection by proposing a method for determining thresholds for the segmentation method by statistical learning. Experimental results show that our method provides very good segmentation results compared to the state-of-the-art methods in segmenting volumes that differ in their dynamics.

1. Introduction

Dynamic textures or temporal textures are textures with motion [10, 16, 37]. 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 or petals flying in the wind), microscopic (e.g. fire plume, sea-waves, smoke, shower and whirlwind), or even moving objects (e.g. a flock of birds, a human crowd and a traffic jam). Potential applications of DT analysis include remote monitoring and various type 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 [1, 26, 30]. 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 despite the practical significance of DT. DT segmentation is to separate the different group of particles showing different random motion. In general, existing approaches of DT segmentation can be generally categorized into supervised and unsupervised methods. For supervised segmentation, a priori information about the textures present is needed. In contrast, unsupervised segmentation does not need a priori information. This makes it a very challenging research problem. To make it easier, most of the recent methods need an initialization. Examples of recent approaches are methods based on mixtures of dynamic texture model [4], mixture of linear models [13], multi-phase level sets [14], Gauss-Markov models and level sets [17], Ising descriptors [21], and optical flow [38].

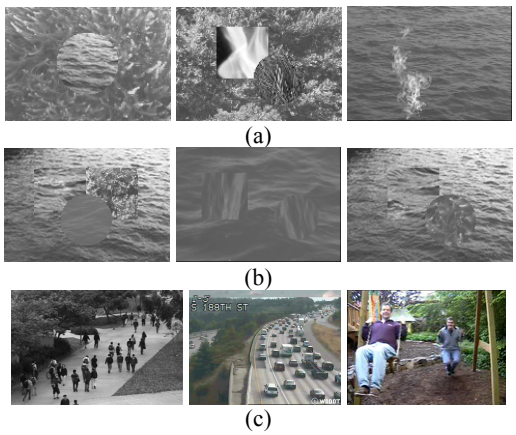

Fig. 1. Illustration of DTs; (a) DTs are different from their spatial mode (i.e., appearance) but show the similar temporal mode (i.e., motion), (b) DTs are different from their temporal mode but show the similar spatial mode, and (c) the similar temporal/spatial mode of DTs are cluttered.

In this paper, we propose an unsupervised 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 volume. 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 to fuse discriminant features of both the appearance and motion for the robust segmentation of cluttering DTs.

For the spatial mode of DT, we use the simple but effective local texture descriptor, i.e., local binary pattern (LBP) [28] and Weber local descriptor (WLD) [9]. LBP is robust to monotonic gray-scale changes caused, e.g., by

illumination variations. It is widely used, e.g., in texture analysis [27, 28]. WLD is also simple and effective for texture classification [8]. We just use one of its components, i.e., differential excitation. However, we still call it WLD for consistency. Motivated by Ojala and Pietikäinen [27], who used the LBP and contrast for the unsupervised static texture segmentation and obtained good performance, we combine LBP and WLD for the unsupervised DT segmentation. 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 [28]. 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 [8]. Similarly, combining histogram of gradient (HOG) and LBP, Wang et al. obtained good performance in human detection [40].

In this paper, we generalize the spatial mode of WLD to a spatiotemporal mode as Zhao and Pietikäinen [41], in which they generalized the LBP as a spatiotemporal descriptor, i.e., LBP in three orthogonal planes or LBP-TOP, which has a promising ability to recognize DTs. Likewise, we call the spatiotemporal mode of WLD as WLD_{TOP}, and the combined LBP and WLD in three orthogonal planes as $(LBP/WLD)_{TOP}$ [9]. It is a theoretically and computationally simple approach to model DT.

For the temporal mode of DT, we use both optical flow and local temporal texture descriptors to capture the motion of DT, and use the non-Euclidean HOOF feature to describe the optical flow motivated by [7]. For the computation of distance between two HOOFs, we develop a distance measure based on Weber's Law [22], which is also computationally simple and effective. For local temporal texture descriptors, we use the temporal component of $(LBP/WLD)_{TOP}$, which describes the temporal variations of DT.

Furthermore, we address the important problem of threshold selection by proposing a method for determining thresholds by statistical learning. As proposed in [27], the segmentation method needs two thresholds, one for the splitting process and the other for the merging process (see Fig. 2). Especially, the threshold for merging varies with regard to different textures. It degrades the performance of segmentation. To this end, we propose a method for determining thresholds by statistical learning, which is robust for different textures.

1.1 Framework

We illustrate the framework of our method as shown in Fig.2. Given an input video, we firstly compute the feature of the video, i.e., the spatial-temporal feature $(LBP/WLD)_{TOP}$ and the temporal feature 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 a video: cube and volume. A cube is used to represent the smallest grid split by splitting process. A volume is employed to represent a cube or a larger part merged by several cubes.

The rest of this paper is organized as follows: In Section 2, we present the related work. In Section 3, we describe the features for segmentation and discuss how to use them for DT segmentation. In Section 4, we show the process of DT segmentation; propose a distance measure for HOOF and describe how to learn the thresholds for the framework. In Section 5, we present some experimental results.

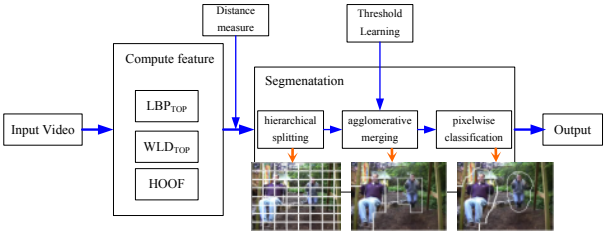

Fig. 2. A flow chart of the framework.

2. Related work

In this section, we review the methods of dynamic texture segmentation.

Two types of statistical models can be applied to dynamic texture segmentation: generative models and discriminant models. A generative model is a model for randomly generating observable data, typically given some hidden parameters [20, 33]. For example, Doretto et al. [17] modeled the spatiotemporal dynamics by Gauss-Markov models, and inferred the model parameters. Vidal and Ravichandran[38] 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. [13] 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 [5], and then proposed a variational approximation for the LDT that enables efficient learning of the model [6]. Rahman and Murshed [31] detected the presence of multiple dynamic textures in an image sequence. Milko et al. [25] modeled the dynamics of each pixel as an auto regressive process perturbed with Gaussian noise.

Discriminative methods enable the construction of flexible decision boundaries, resulting in classification performances often superior to those obtained by purely probabilistic or generative models [20]. For example, Vidal and Singaraju [39] introduced the multibody brightness constancy constraint, a polynomial equation relating motion models, image derivatives and pixel coordinates to segment multiple 2-D motion models. Ghoreyshi and Vidal [21] 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. Fazekas et al. [18] developed the optical flow method to capture the intrinsic dynamics of dynamic textures. After that, Chetverikov et al. [11] developed this method and applied the singular

value decomposition to a temporal data window in a video to detect targets in dynamic texture via the residual of the largest singular value.

In addition, Stein and Hebert [36] 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 [15].

For our method, it is an unsupervised and discriminative method. It does not need any initialization or priori information about the dynamic texture in the test set.

3 Features for segmentation

In this section, we first discuss the features used for DT and then present how to use them for the description of the appearance and motion of DTs.

3.1 Features

3.1.1 Local texture descriptor

To describe dynamic texture, 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) of DT; *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 $H_{\lambda,\pi}$ (λ=*LBP*, *WLD* and π=*XY*, *XT*, *YT*).

LBP_{TOP} is a spatiotemporal descriptor [41], which compute the LBP feature in three orthogonal planes as shown in Fig. 3 (c). For each plane, it is computed as follows:

$$LBP_{P,R} = \sum_{p=0}^{P} s(g_p - g_c)2^p, \qquad (1)$$

where $s(A) = \begin{cases} 1, & if\ A\ is\ ture \\ 0, & otherwise \end{cases}$; the gray value g_c corresponds to the gray value of the center pixel of the local neighborhood; g_p correspond to the gray values of P equally spaced pixels on a square of length R (R>0). If we concatenate these three sub-histograms $H_{\lambda,\pi}$ (λ=*LBP*, and π=*XY*, *XT*, *YT*) into a single histogram, we get an LBP_{TOP} feature histogram.

The WLD feature is the differential excitation of the WLD descriptor of [8]. It is computed as follows:

$$WLD(I_c) = sigmoid(x), \text{ where } x = \sum_{i=0}^{p-1}\left(\frac{I_i - I_c}{I_c}\right). \qquad (2)$$

Here, I_c denotes the center pixel, I_i (i=0, 1, …, p-1) are the neighbors, and p is the number of neighbors (e.g., p=8 as shown in Fig.3 (d)).

Note that we use the sigmoid function to compute WLD instead of the arctangent function employed in [8], since both of these two functions work comparably but the sigmoid function is widely used [2]. In addition, we use WLD instead of contrast used by [27] because the extensive experiments in [8] demonstrate the discriminability of WLD.

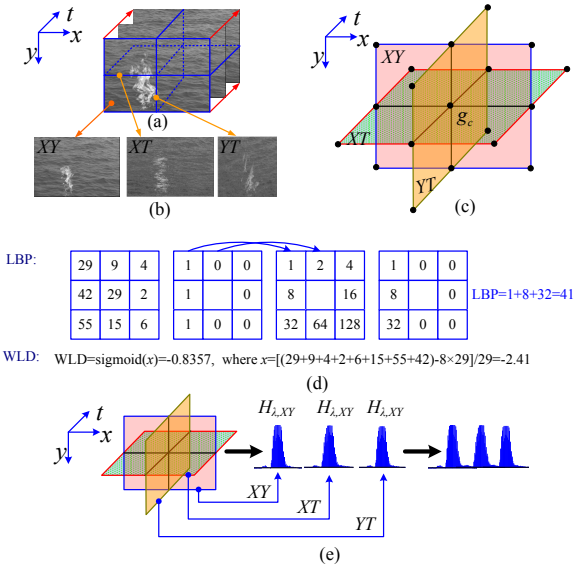

Fig. 3. Computation of $(LBP/WLD)_{TOP}$ for a DT: (a) a sequence of a DT; (b) three orthogonal planes of the given DT; (c) the vertex coordinates of the three orthogonal planes; (d) computation of LBP and WLD of a pixel; (e) computation of sub-histograms for $(LBP/WLD)_{TOP}$, where λ=*LBP*, *WLD*.

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

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

$$\boldsymbol{\varphi} = \{H_{\lambda,\pi} \mid \lambda = LBP \text{ or } WLD, \text{ and } \pi = XY, XT \text{ or } YT\}. \qquad (3)$$

Intuitively, $\boldsymbol{\varphi}$ is composed of six sub-histograms $H_{\lambda,\pi}$ of LBP and WLD in the three orthogonal planes.

Motivated by [41], the radii of $(LBP/WLD)_{TOP}$ in axes *X*, *Y* and *T*, 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 $(LBP/WLD)^{TOP}_{P_{XY},P_{XT},P_{YT},R_X,R_Y,R_T}$.

We denote l_π the size of the components of $\boldsymbol{\varphi}$ (*i.e.*, $H_{\lambda,\pi}$). In our implementation, we also use the "uniform patterns" as in [28] to shorten the length of the feature vector LBP. Here, a pattern is considered uniform if it contains at most two bitwise transitions from 0 to 1 or vice versa when the bit pattern is considered circularly (*e.g.*, 11110011.). When using the uniform patterns, all non-uniform LBP patterns are collected in a single bin during histogram computation. Thus, we have l_{LBP}=59 for $H_{LBP,\pi}$ when the number of neighboring points is 8. Meanwhile, l_{WLD} is the number of bins for $H_{WLD,\pi}$. We choose to use l_{WLD}=16 bins experientially. See Ref. [27] for a detailed description of the mapping from the continuous WLD space to the discrete bin index. In addition, we also experientially set R_X=1, R_Y=1 and R_T=3, P_{XY}=8, P_{XT}=8 and P_{YT}=8. Thus,

$(\text{LBP/WLD})_{\text{TOP}}$ is denoted as $(LBP/WLD)^{TOP}_{8,8,8,1,1,3}$. Note that we set R_T=3, which means $(\text{LBP/WLD})_{\text{TOP}}$ represent the appearance and motion features of neighboring T=7 frames. Likewise, each cube or volume consists of T=7 frames.

3.1.2 HOOF

To compute the optical flow (OF) for each pixel, we use the method proposed in [3]. Motivated by [23], we then organize the optical flow of a volume 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 [7].

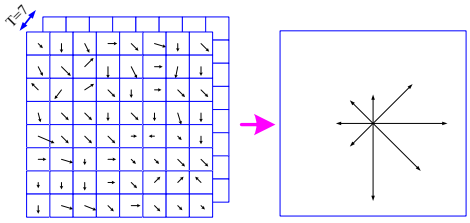

Fig. 4: A local descriptor is created by first computing the OF magnitude and orientation at each frame sample point in a volume, as shown on the left. These OF magnitudes are then accumulated into orientation histograms summarizing the contents over the volume, as shown on the right, with the length of each arrow corresponding to the sum of the OF magnitudes near that direction within the volume. This figure shows an 8×8×7 descriptor array, whereas the experiments in this report use 16×16 descriptors computed from a 16×16×7 sample array.

Considering that some pixels in volumes of dynamic textures are still and their OF values are close to zero, we use an extra bin in HOOF to build the statistic for these pixels. To filter the noise in the OF field of a frame, we use a threshold for OF value of a pixel and set (0.1, 0.1) for the two directions of a pixel. Intuitively, using an extra bin in HOOF guarantees that the similarity between two still volumes is one.

3.2 Feature fusion

The appearance of DT is described by a local spatial texture descriptor $\boldsymbol{\varphi_s}$={ $H_{\lambda,\pi}$ |λ=*LBP* or *WLD*, and π= *XY* }, i.e., the *XY* plane of $(\text{LBP/WLD})_{\text{TOP}}$. The motion of DT is described by HOOF and local temporal texture descriptor $\boldsymbol{\varphi_t}$={$H_{\lambda,\pi}$ |λ=*LBP* or *WLD*, and π=*XT* or *YT*}, i.e., the *XT* and *YT* planes of the $(\text{LBP/WLD})_{\text{TOP}}$.

Here, we use both HOOF and the temporal texture descriptor $\boldsymbol{\varphi_t}$, because they are complementary to each other: HOOF describes the motion field of particles (macroscopic or microscopic), while $\boldsymbol{\varphi_t}$ describes the temporal variations of the texture appearance. We call the combined feature as the motion of DT for simplicity.

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

$$\Pi = \omega_S \times \Pi_S + \omega_T \times \Pi_T, \quad (4)$$

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 of volumes. 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 ω_S=ω_T=0.5 because we suppose that the appearance and motion of DT play the same important roles during the description and segmentation of DT. In addition, how to compute the Π_S and Π_T of two volumes will be presented in Section 4.

4 Framework

In this section, we firstly describe the framework for the segmentation of DT. We then present the distance measure and the thresholds for this framework.

4.1 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 [27]. We would briefly discuss these steps. For more details, please refer to [9]

For the splitting step, we recursively split the input video into square volumes of varying size. The decision whether a volume is split into four sub-volumes is based on a uniformity test. However, a necessary prerequisite for the following merging to be successful is that the individual volumes 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 volume, we perform splitting.

After the input frame has been split into cubes of roughly uniform texture, we merge those similar adjacent volumes/cubes until a stopping criterion is satisfied. For each merging, we use the following criterion—the two adjacent volumes should satisfy the two conditions: (1) Either Π_{LBP}, Π_{WLD}, or Π_{HOOF} between these two volumes is larger than the thresholds (i.e., Y_{LBP}, Y_{WLD}, or Y_{HOOF}); The merger importance (*MI*) is minimal in those volume pairs which satisfy the first condition. Here, *MI* is defined as: $MI=f(p)\times(1-\Pi)$, where Π is the appearance and motion similarity between two volumes. The function $f(p)$ is the percentage of pixels in the smaller one of the two volumes. It is computed as: $f(p)=N_b/N_f$, where N_b is the number of pixels in the smaller one of the two volumes, and N_f is the number of pixels in the current frame. The stopping rule for merging is that Π_{LBP}, Π_{WLD} and Π_{HOOF} between any two volumes are smaller than the thresholds (i.e., Y_{LBP}, Y_{WLD}, and Y_{HOOF}, see Section 4.3), respectively.

Finally, we perform a simple pixelwise classification to improve the localization of the boundaries. To this end, we switch into a DT classification mode by using the appearance and motion feature (i.e., $(\text{LBP/WLD})_{\text{TOP}}$ histogram and HOOF) of the video segments as the DT models. For each boundary pixel (*i.e.*, at least one of its 4-connected neighbors with a different label), we compute its appearance and motion features over its circular neighbor $B(r)$, where r is the radius of the circular neighbor. We then compute the similarity between the neighbor histograms and the models of those volumes (which are 4-connected to the pixel in query). We re-label the pixel if the label of the nearest model votes a different label from the current label of the pixel. In our implementation, we set r=11 as Ojala and Pietikäinen [27].

4.2 Distance measures

In this part, we present the distance measure 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.

4.2.1 Histogram intersection for LBP/WLD

In this sub-section, we describe the similarity measure of histogram for the feature $(\text{LBP/WLD})_{\text{TOP}}$.

To compute the similarity between two given histograms H_1 and H_2, we use the histogram intersection $\Pi(H_1, H_2)$ as a similarity measurement of two normalized histograms $\Pi(H_1, H_2) = \sum_{i=1}^{L} \min(H_{1,i}, H_{2,i})$, where L is the number of bins in a histogram.

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

The spatial similarity between two volumes is computed as

$$\Pi_S=\omega_{XY}\times(\omega_{LBP}\times\Pi_{LBP,\,XY}+\omega_{WLD}\times\Pi_{WLD,\,XY}), \quad (5)$$

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

$$\begin{aligned}\Pi_{\text{T,texure}}=&\omega_{XT}\times(\omega_{LBP}\times\Pi_{LBP,\,XT}+\omega_{WLD}\times\Pi_{WLD,\,XT})\\&+\omega_{YT}\times(\omega_{LBP}\times\Pi_{LBP,\,YT}+\omega_{WLD}\times\Pi_{WLD,\,YT}),\end{aligned} \quad (6)$$

where ω_{LBP} and ω_{WLD} are two weights to balance the effects of the LBP and WLD features, and ω_π (π=*XY*, *XT*, *YT*) are the weights for each plane since we find that these three planes do not play the same roles for the representation of DT.

In our implementation, we experientially set ω_{LBP}=ω_{WLD}=0.5, and ω_{XY}=0.625, ω_{XT}=0.25, ω_{YT}=0.125. Here, we let ω_{LBP}=ω_{WLD} because both descriptors have achieved a similar performance in texture classification [8]; For the value of ω_π (π=*XY*, *XT*, *YT*), we use the same values as shown in [41], which are also obtained statistically by dynamic texture recognition over a given training set.

4.2.2 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-Euclidian [7]; (2) HOOF is non-linear (i.e., different bins showing different frequency and so different weights); (3) HOOF during segmentation can not be normalized due to the importance of the different frequency of each bin; (4) distance measure should not only work for those volumes whose OF values are large (corresponding to the motion volumes) but also work for those volumes whose OF values are small or zeros (corresponding to the still volumes); (5) noise resulted from the computation of optical flow due to the brightness variations and algorithm failure for those non-texture volumes [3].

The constraints (1), (2) and (3) result in difficulties that many current distance measures are not able to handle. For example: principle component analysis (PCA) fails to the constraints (1), (2) because PCA usually works well for Euclidian and linear space; 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 cubes 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 of P_2 is $H_2=k\times\{a_0, a_1, \ldots, a_{L-1}\}$, and k is a real number. In other words, the H_2 and H_1 are proportional in each dimension. 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 is not a probability mass function because it can not be normalized to sum up to one. Hence, the space of HOOF is not a Riemannian manifold and we can not use the Binet-Cauchy kernels proposed in [7]. An alternative method is to use the level set scheme as shown in [18]. However, the level set scheme usually needs an initialization and the performance of segmentation depends on the initialization (see more details in Section 5.4).

Weber distance

Given two HOOFs of cubes R_1 and R_2, $H_1=\{a_0, a_1, \ldots, a_{L-1}\}$, $H_2=\{b_0, b_1, \ldots, b_{L-1}\}$, we first normalize H_i(i=1, 2), *i.e.*, $H_i = H_i/A_i$, where A_i is the volume of R_i. We use this normalization because two cubes for merging might not have the same volume. Using this normalization over the volume of a cube, we only pay attention to the HOOF over a unit volume.

The Weber's Law denotes that the ratio (k) of the increment threshold (ΔI) to the background intensity (I) is a constant [22], i.e., $\Delta I/I=k$. Similarly, the Weber distance for HOOFs 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 takes zero. In our case, we set C=1. Here, we take the maximum of the ratios for the following intuitive idea: if the values of b_i and a_i are close, the distance between them is small; if the difference between the values of b_i and a_i are significant, the distance between them should be significant.

We then use sigmoid function to normalize the ratio: x_i=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

$$D_{HOOF} = \frac{1}{L}\sum_{i=0}^{L-1} x_i, \quad (7)$$

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_{HOOF}=0, i.e., there is no difference between H_1 and H_2. If $H_2=k\times H_1$, we have:

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

and hence, $D_{HOOF}\neq 0$. In other words, it works well for the

case that the H_2 and H_1 are proportional in each dimension, in comparison to the normalized histogram intersection which fails in this case.

Weighted Weber distance

In our experiments, we found the different bins of HOOF play different roles to compute the similarities between two cubes. One reasonable idea is to weight the different bins according to their importance to describe the motion of a cube. Specifically, given two HOOFs of cubes P_1 and P_2, $H_1=\{a_0, a_1, \ldots, a_{L-1}\}$, $H_2=\{b_0, b_1, \ldots, b_{L-1}\}$, the weight of a bin is computed as $\omega_i = \frac{a_i + b_i}{\sum_{i}^{L-1}(a_i + b_i)}$, Combining this formula with Eq. (7), we have the weighted Weber distance between two HOOFs:

$$D_{HOOF} = \frac{1}{L}\sum_{i=0}^{L-1}(\omega_i x_i). \quad (8)$$

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

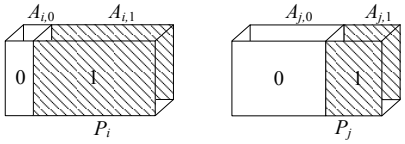

Fig. 5. Illustration of the volume computation for optical flow equal to zero.

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 close to zero. Spcifically, let H_i, H_j be the two HOOFs of two cubes P_i, P_j, and $\Pi_{HOOF}(P_i, P_j)$ be the motion similarity between the two cubes P_i, P_j. Thus, we have:

$$\Pi_{HOOF}(P_i, P_j) = \frac{A_0}{A}\times 1 + \frac{A_1}{A}\times\left[1 - D_{HOOF}(H_i, H_j)\right], \quad (9)$$

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

Furthermore, combining the similarities of texture mode (i.e., $\Pi_{T,\,texture}$ computed by Eq. (6)) and the motion mode of two volumes (i.e., Π_{HOOF} by Eq. (9)), we have the temporal similarity (variations in XT and YT planes of DT) as:follows:

$$\Pi_T = \Pi_{T,\,texture} + \Pi_{HOOF}. \quad (10)$$

In addition, plugging Eqs. (5) and (10) into Eq. (4), we have the similarity between two volumes.

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

4.3 Threshold for LBP/WLD and HOOF

We address the important problem of threshold selection by proposing a method for determining thresholds for the segmentation method by statistical learning. We will briefly present the thresholds for LBP/WLD and then discuss about the threshold for HOOF. For more details about the thresholds for LBP/WLD, please refer to [9]

4.3.1. Threshold for LBP/WLD

In the following sub-section, we attempt to learn a threshold Y to determine when merging step is stopped. Given a training dataset $D=\{D^0, D^1, \ldots D^{c-1}\}$, having N images and C classes. 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, …, 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 $|S_w| = \frac{1}{2}\sum_{i=0}^{c-1}\left[N_i \times (N_i - 1)\right]$.

Likewise, S_b is computed as $S_b = \left\{\left\{s_{j,l}^{i,k}\right\}, i,k = 0,1,\ldots c-1, \text{and } i \neq k\right\}$, 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, …, N_i-1; l=0,1, …, 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 case, shown in Fig. 6(a) and (b), we use histograms to describe similarity distributions. Here, x-axis denotes the normalized similarity (e.g., computed by the normalized LBP/WLD histogram intersection); y-axis denotes the similarity probability (i.e., $P(s|S_w)$ and $P(s|S_b)$, where $P(s|S_w) + P(s|S_b)$=1), 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, considering the noise in each frame of a video, we can adjust the threshold for obtaining better performance. 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), and the two thresholds used only for the spatial mode of DT, i.e., the XY plane of $(\mathrm{LBP/WLD})_{\mathrm{TOP}}$.

4.3. 2. Threshold for HOOF

Given a training dataset $D=\{D^0, D^1,\ldots D^{N-1}\}$, it is composed of N dynamic textures, each consisting of N_i frames. We first divide every T neighboring frames into cubes evenly (in our case, T=7 and these cubes have the same size as those cubes after the splitting process during segmentation, e.g., 16×16.). We then label the relation between neighboring cubes 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 cubes show a similar motion mode and so should be merged into one volume, we label ω_{ij}=1, or 0 otherwise. Subsequently, we compute the HOOFs for all these cube pairs and the similarities for each cube pair.

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

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 histogram to describe similarity distributions (refer to Fig. 6(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.

5. Experiments

In this section, we first introduce how to learn the thresholds for the appearance and motion features during merging. 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.

5.1 Threshold learning

As shown in Section 4.3, we present how to learn the thresholds for the appearance and motion features. In this subsection, we will describe how to use the training sets to set the values for these thresholds.

Thresholds for LBP/WLD

As discussed in Section 4.3, we compute the two thresholds (i.e., Y_{LBP}, Y_{WLD}) by texture classification on a given dataset. In our case, we use the well-known Brodatz texture database [29]. The images are 256×256 pixels in size, and they have 256 gray levels. It comprises 2048 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 texture dataset are shown in Fig. 6 (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, by which the feature WLD has 96.4% confidence to merge the similar cubes as shown in Fig.6 (b) (a similar confidence with 2σ of normal distribution, i.e., 95.5% confidence). 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 cubes belongs to the same texture, we should also merge them into one volume. However, in this case, the similarity between these two cubes would be smaller than those two cubes which show the same texture (i.e., appearance). To merge these two cubes whose dominant textures are the same, we reduce the values of Y_{LBP}, Y_{WLD} as shown in Fig. 6.

Threshold for HOOF

As discussed in Section 4.3, 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 7 neighboring frames from each clip for training (i.e., 700 frames totally), and these clips are not used as test clips in the following experiments. We first split the selected clips evenly (16×16×7 for each cube). We then label the neighbor cube 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 histograms shown in Fig. 6 (c).

In our case, we set Y_{HOOF}=0.79 for simplicity, which is equal to Y_{LBP} and Y_{WLD}. In addition, from Fig. 6 (c), one can find that $P(s|S_w)$ fluctuates obviously between [0.55, 0.9]. It is resulted from the boundary cubes of two different motion segments. Because at some time it is difficult to label ω_{ij} for the boundary cube pair (P_i, P_j), which result in the tail of $P(s|S_w)$ (its value is smaller than 0.55) and the rise of $P(s|S_b)$ in the right side (its value is larger than 0.8). Furthermore, the OF noise brought from the computation method also promote the fluctuation of $P(s|S_w)$ in the interval [0.55, 0.9] and its tail, and the rise of $P(s|S_b)$ in the right side.

As shown in Fig. 7, we show the variations of the similarities of two merged cubes during the merging step on the DT. In Fig. 7 (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 Fig. 7 (e), one can find that $\Pi_{WLD,XY}$ is larger than the given threshold Y_{WLD} before the merging stop except the 27th, 40th, 43th, 46th, 52th, 53th and 60th iterations. However, $\Pi_{LBP,XY}$ is larger than the given threshold Y_{LBP} at 27th, 46th and 60th iterations (missed iterations by $\Pi_{WLD,XY}$) although it varies during merging process. It shows that WLD and LBP are well complementary. In addition, Π_{HOOF} is larger than the Y_{HOOF} at 40th, 43th, 52th, and 53th iterations (missed iterations by both $\Pi_{WLD,XY}$ and $\Pi_{LBP,XY}$); Likewise, for the missed iterations by Π_{HOOF} (42th, 49th, 50th, 51th, 57th, 58th, 59th and 61th iterations), either $\Pi_{WLD,XY}$ or $\Pi_{LBP,XY}$ (or both of them) is larger than the learned threshold. It demonstrates 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. Thus, it is experimentally reasonable to

combine the appearance and motion of DT in our

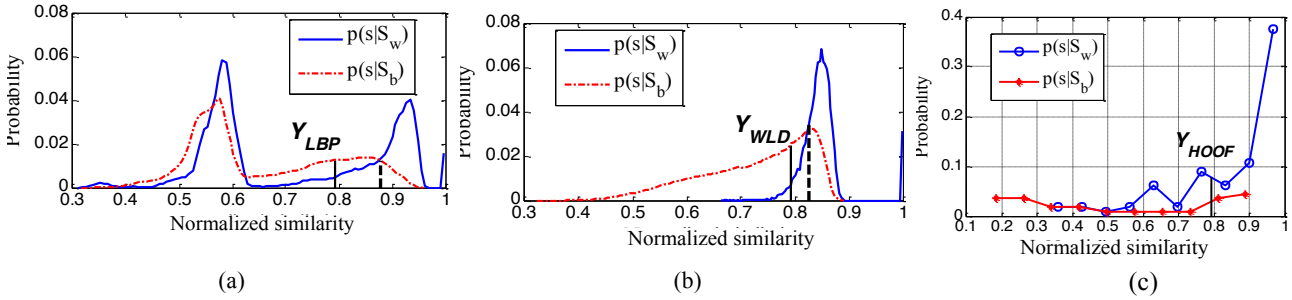

Fig. 6. Learning thresholds on Brodatz texture dataset for (a) LBP and (b) WLD; and on the synthesized database for HOOF.

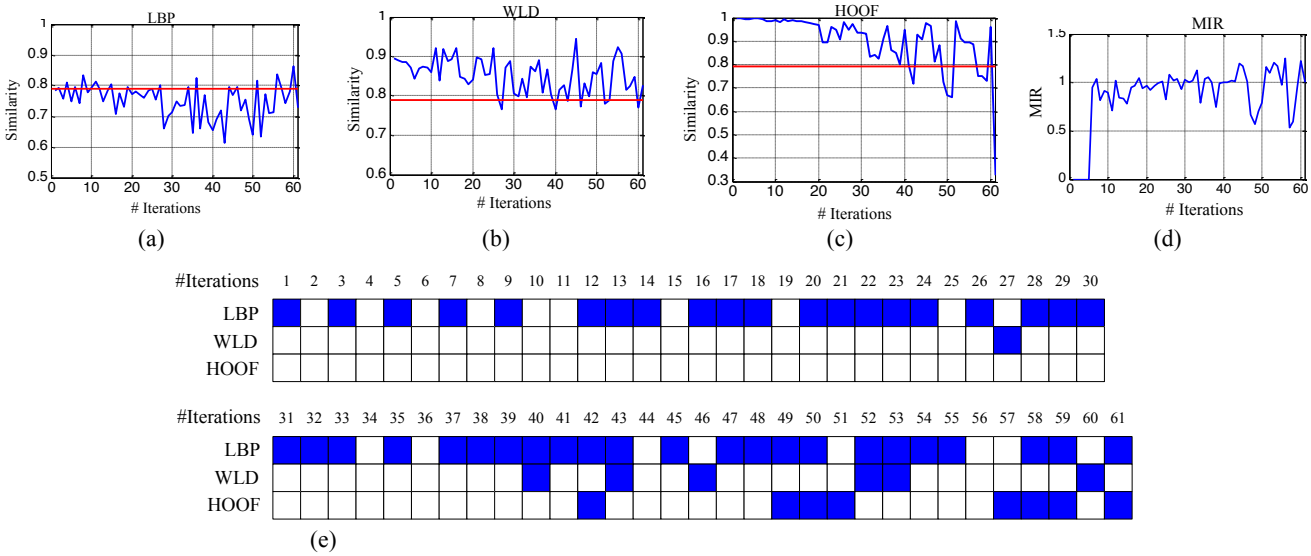

Fig. 7. Illustration of the variations of the similarities of two merged cubes and MIR during the merging step on the DT shown in Fig. 2; (a), (b), (c) and (d) are the variations of the similarities between two merged cubes for the features of LBP and WLD in XY plane of DT, HOOF (i.e., $\Pi_{LBP,XY}$, $\Pi_{WLD,XY}$, and Π_{HOOF}), and MIR, respectively; (e) shows in which iterations the corresponding similarities (i.e., $\Pi_{LBP,XY}$, $\Pi_{WLD,XY}$, and Π_{HOOF}) are smaller than the learned thresholds (we call them *missed* for short), and the block in shadow denotes a feature is missed and the block in white denote a feature works well (the similarity is larger than the threshold).

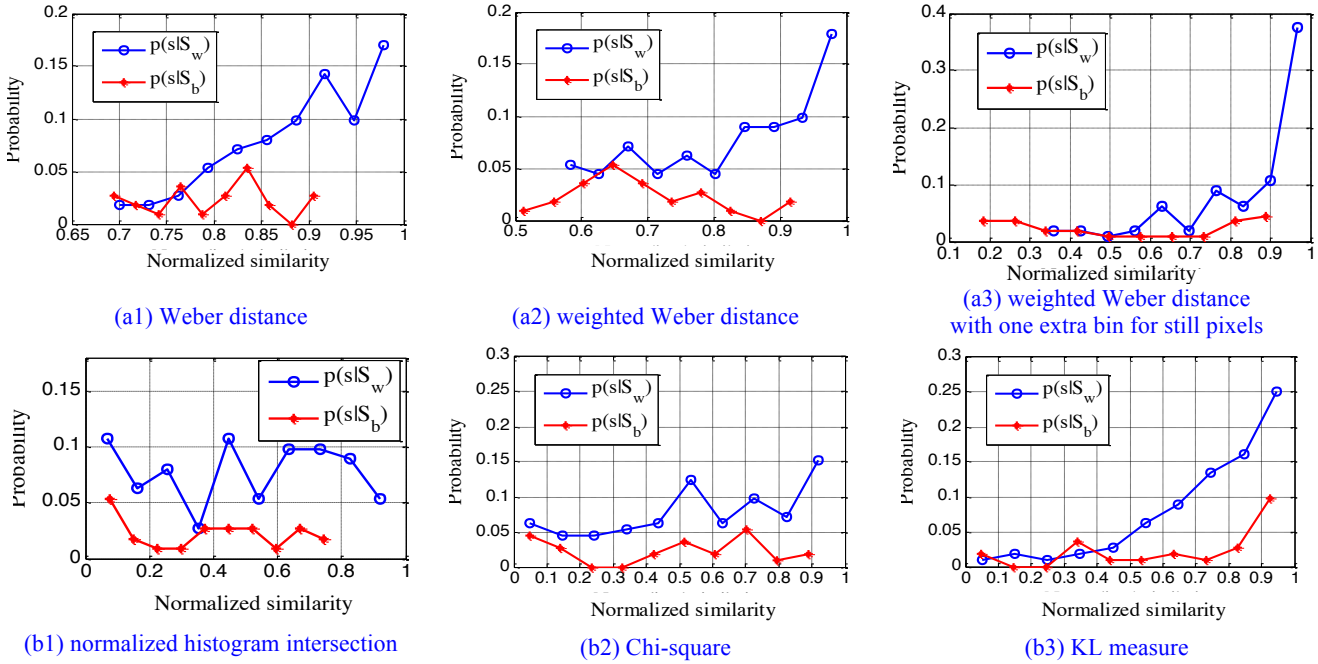

Fig. 8. Comparison of different measure for the computation of the HOOF similarity

framework. In addition, for the variations of $\Pi_{LBP,\ XY}$ during merging process, one possible reason is the cluttering background of this sequence.

Furthermore, from Fig. 7 (d), one can find that MI ratio

(MIR), employed by [27], is smaller than 1.5 even when the merging step stops. It shows that using MIR fails for this DT (if MIR works for a DT, it should grow suddenly and be larger than a threshold, e.g., being equal to 2 in [27]. However, as shown in Fig. 7 (d), the MIR varies steadily without a sudden growth.).

In addition, although the values of these three thresholds (i.e., Y_{LBP}, Y_{WLD} and Y_{HOOF}) depend on the distribution of the training set (here we learn them using the Brodatz texture dataset and the synthesized database [4], we found that the learned thresholds work well for different types of testing dynamic textures.

5.2 Comparison of distance measure

We compare the different similarity measures by experiments. As shown in Fig. 8, we test the following four 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 $D_{KL}(a,b)=[D_{KL}(a||b)+D_{KL}(b||a)]/2$ and $D_{KL}(a\,||\,b)=\sum_i a_i \log(a_i/b_i)$. Here we compare the Chi-square and KL measure because they are similar to the proposed Weber distance. The dataset is the training set used in Section 5.1 for the threshold learning of Y_{HOOF}.

For the measure of normalized histogram intersection and Chi-square distance, from Fig. 8 (b1) and (b2), we can find that the two curves $P(s|S_w)$ and $P(s|S_b)$ vary irregularly. As shown in Fig. 8 (a1), Weber distance increases the probabilities $P(s|S_w)$ of larger similarities. It shows that the similar cubes represented by HOOF obtain larger similarities using Weber distance. In Fig. 8 (a2), 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. It shows that the dissimilar cubes by HOOF obtain smaller similarities. It helps to merge those similar cubes together and improve the performance of the dynamic texture segmentation. In Fig. 8 (a3), 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 decrease the probabilities $P(s|S_w)$ of small similarities. It further helps to improve the performance of segmentation. In addition, although KL measure increases the probabilities $P(s|S_w)$ of larger similarities and decrease the probabilities $P(s|S_w)$ of small similarities compared to histogram intersection and Chi-square measure, it also increases the probabilities $P(s|S_b)$ of larger similarities and the probabilities $P(s|S_w)$ of larger similarities of KL measure is significantly smaller than that of the weighted Weber distance in Fig. 8 (a3).

In conclusion, the normalized histogram intersection and Chi-square measure is not suitable for HOOF. The Weber distance works better for HOOF, and weighting plus using one more additional bin for still pixel are important, for the HOOFs of similar DTs, to increase the probabilities $P(s|S_w)$ and decrease the probabilities $P(s|S_b)$. It helps to improve the performance of dynamic texture segmentation.

5.3 Performance evaluation of components of the proposed framework

In Fig. 9, we compare the performance of $(LBP/C)_{TOP}$, $(LBP/WLD)_{TOP}$, HOOF and the proposed framework. Specifically, for the first column, we combine LBP and contrast for DT segmentation. For the second column, we combine LBP and WLD instead of contrast. In addition, both $(LBP/C)_{TOP}$ and $(LBP/WLD)_{TOP}$ 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 $(LBP/WLD)_{TOP}$ and HOOF to represent DT; and we employ normalized histogram intersection for $(LBP/WLD)_{TOP}$ and weighed Weber distance for HOOF.

Fig. 9 (a) is a frame from *vidf1_33_000.y*. One can find that both $(LBP/WLD)_{TOP}$ and $(LBP/WLD)_{TOP}$+ *HOOF* preserve correctly the small cubes which are different from the neighbors. However, $(LBP/C)_{TOP}$ fails in segmenting the two passengers in the *road*. *HOOF* performs poor because the OF values in the motion volume of this video are small.

From Fig. 9 (b) one can find that $(LBP/WLD)_{TOP}$+ *HOOF* works best, which segments the three synthesized volumes successfully. In comparison, $(LBP/WLD)_{TOP}$ segments two of them because the top-left volume has the same appearance of texture (water flow but in different motion field) with the background, which makes the local texture descriptors LBP/WLD invalid. In addition, $(LBP/C)_{TOP}$ segment only one of the volumes for this sequence because the similar appearance of textures of the other two synthesized volumes to the background impended the performance of $(LBP/C)_{TOP}$. *HOOF* performs poor again because the OF values of the different motion volumes are similar to each other although they have the different appearance of texture.

Fig. 9 (c) shows a sequence in which two humans are swinging. From this figure, one can find that $(LBP/WLD)_{TOP}$+*HOOF* successfully segments the two swinging humans. $(LBP/WLD)_{TOP}$ fails to segment the human in the right side because he is far away from the camera, and the motion is weak and the variations of textures are small. Although $(LBP/C)_{TOP}$ segments the two humans successfully, the threshold for merging was tuned to obtain this result. Furthermore, there is one more but false segmented volume in the bottom-right side of this frame after segmentation because of the cluttering background. In contrast, the thresholds for $(LBP/WLD)_{TOP}$ +*HOOF* are fixed once they are learned using the method and training set. It demonstrates that $(LBP/WLD)_{TOP}$+ *HOOF* works much more robustly than $(LBP/WLD)_{TOP}$ and $(LBP/C)_{TOP}$. In addition, *HOOF* segments the front human successfully although the clustering background but fails to segment the back human because the OF values of this human are small.

Furthermore, we also test $(LBP/WLD)_{TOP}$ but using weighted Weber distance instead of normalized histogram intersection. The experimental results of $(LBP/WLD)_{TOP}$ using weighted Weber distance are also shown in the third column of Fig. 9 (i.e., $(LBP/WLD)_{TOP}$-*Weber*). From Fig. 9, we can find that $(LBP/WLD)_{TOP}$-*Weber* works comparably to $(LBP/WLD)_{TOP}$ (the method shown in the second

column, which using the normalized histogram

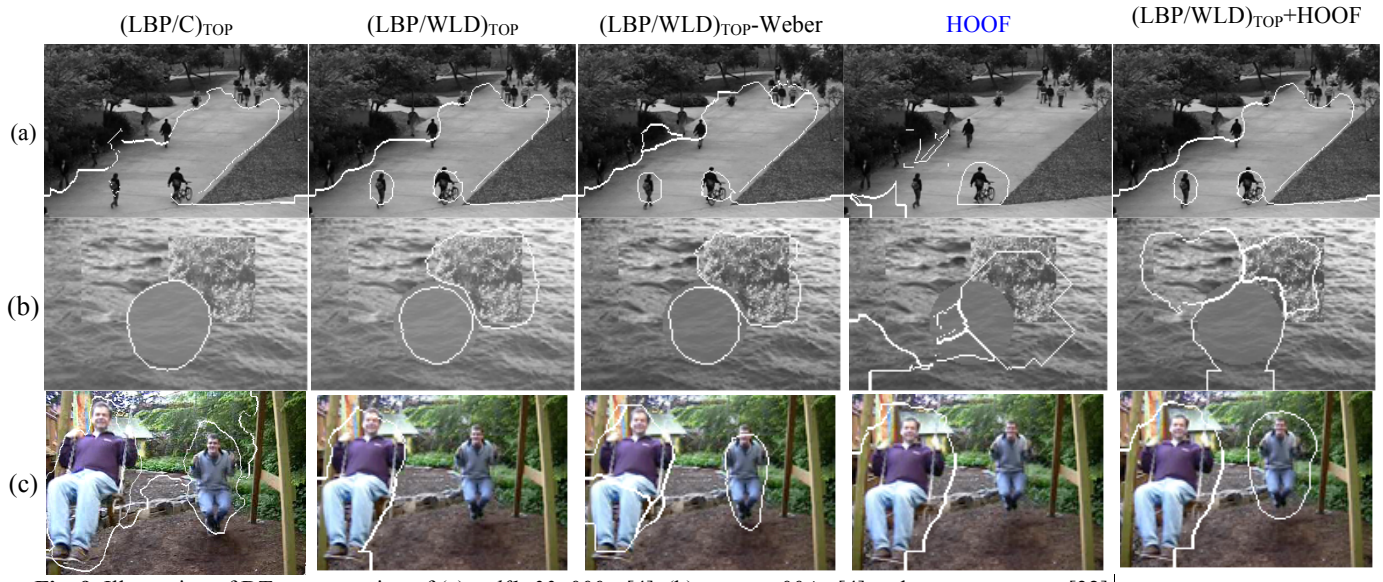

Fig. 9. Illustration of DT segmentation of (a) *vidf1_33_000.y* [4], (b) *texture_004.y* [4] and *swing* sequence [32].

Fig. 10. Experimental results on the *swing* sequence [32] using optical flow and level set scheme but using different initializations.

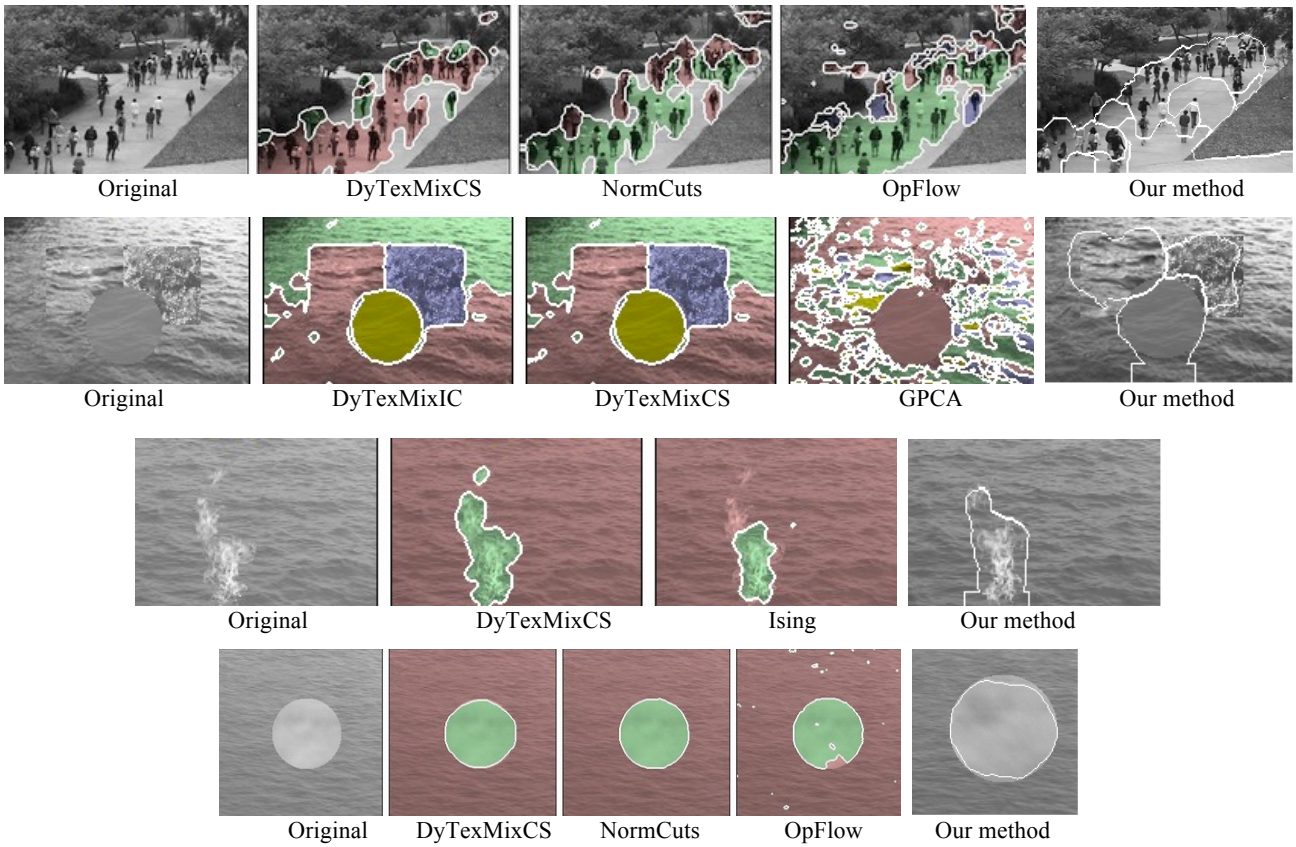

Fig. 11. Some experimental results comparing with existing methods: the first and second sequences are *vidf1_33_006.y* and *texture_004.y*, which are from [4]; the third and fourth sequences are *ocean-fire-small* and *ocean-steam-small*, which are from [17].

intersection) although the weighted Weber distance is developed for HOOF.

Note that for the thresholds of LBP and WLD (i.e., Y_{LBP} and Y_{WLD}) for $(LBP/WLD)_{TOP}$-*Weber*, we use the same thresholds learned in Section 5.1. The reason is that, after performing the learning procedure shown in Section 5.1

and using weighted Weber distance, we find that the difference between the values of Y_{LBP} and Y_{WLD} using Weber distance and those of using normalized histogram intersection is small. In the following experiments, we use normalized histogram intersection for $(LBP/WLD)_{TOP}$ because the performance of $(LBP/WLD)_{TOP}$ using either the normalized histogram intersection or weighted Weber distance is comparable. In addition, the regular boundary between two motion segments around the border of the frames is due to the radius of the circular neighbor is equal to 11, which stop the pixel classification after the merging step (see Section 4.1).

5.4 Level set scheme

We test the optical flow plus level set scheme for the segmentation of dynamic textures. For the level set scheme, we use the method proposed by Li et al. [24], who proposed a computationally efficient and variational level set method and achieved good performance for the segmentation of medical image. Different from Li et al. [24] who used the gradient flow, we use the optical flow as input for the level set scheme instead.

In Fig. 10 we show the effect of different initializations for the level set method. The level set method performs very well (e.g., Fig. 10 (a)) if the initialization is good. However, the performance of segmentation degrades when the initialization becomes biased (e.g., Fig. 10 (b) and (c)).

In comparison, our proposed framework, using the weighted Weber distance to compute the difference between the motions of different volumes represented by HOOFs, does not need any initialization but works well as shown in Fig. 9.

5.5 Comparison with existing methods

In Fig. 11 we present some experimental results performed on various types of sequences, and the comparison with existing methods. Specifically, *DytexMixIC* and *DytexMixCS* are methods proposed by [4]. The former one need a manual specification of a rough initial segmentation contour and the latter one is based on the component splitting strategy. *Ising* uses the level-sets method and Ising models [21]. *GPCA* is the generalized principle component analysis and used for DT segmentation in [38]. *NormCuts* is based on normalized cuts and the "motion profile" representation proposed in [34] and [35]. *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 [12]. These existing methods 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. 11, we compare our method with existing methods on additional sequences. Specifically, for the sequence *vidf1_33_006.y* (the first row of this figure), 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 optical flow noise significantly.

For the sequence *texture_004.y* (the second row), our framework segments the three synthesized volumes successfully without any small separated volumes. *GPCA*, which also uses optical flow to describe the motion of this sequence, fails. One reason is also resulted from optical flow noise.

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 except *OpFlow*.

In addition, we also perform the quantitatively evaluation on the synthetic sequences [8]. An example is shown in the second row of Fig. 11. The database also provides the initial contours to the segmentation algorithms.

Table 1 The best average rand index for each segmentation algorithm on the synthetic database
(The number in the parenthesis is the dimension of state space)

Algorithm	K=2	K=3	K=4
DytexMixIC[4]	0.915 (17)	0.853 (15)	**0.868 (15)**
DytexMixCS[4]	0.915 (20)	0.825 (10)	0.835 (15)
Ising[20]	0.879 (05)	N/A	N/A
GPCA[35]	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

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. We only use 200 videos of this dataset for testing because we use 100 sequences for threshold learning of Y_{HOOF}. Other methods in Table 1 use all the 300 videos. From this table, one can find that our method obtain competitive performance although we do not use the initial contours provided by this database.

6. Conclusion

We proposed a new framework for unsupervised dynamic texture segmentation based on spatiotemporal features. For the spatial mode, we employed a new texture feature to characterize each volume 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 learning thresholds for the segmentation framework. In addition, we developed a weighted Weber distance measure, which is computationally simple compared to methods such as kernel methods and works better than the widely used normalized histogram intersection. Experimental results and comparison with existing methods show that our method is effective for DT segmentation, and is also computationally simple compared to methods such as those using mixtures of dynamic texture model or level sets. Furthermore, our method performs fairly well on a sequence with cluttered background.

Furthermore, the selection of suitable threshold values for different types of algorithms is a general problem in computer vision and pattern recognition. It would be of interest to see how the proposed approach, i.e. determining the thresholds by learning, could be extended to other tasks in these fields.

Although the proposed method works well for the segmentation of well-textured dynamic textures, its performance will degrade for dynamic textures with less texture content (e.g., smooth surface of an object, peaceful water surface). It is because both LBP and WLD are local texture descriptors. Furthermore, the less-textured surface of a DT will also impend the optical flow scheme. To overcome this problem, one future work is to combine our framework (a discriminant model) with a generative model to describe the DT. How to combine discriminant models and generative models is also an interesting research problem in object recognition field [20].

Another future work is how to use the segmentation results of previous frames to help the segmentation of the next frame. By this method, we can improve the efficiency of the segmentation of DT.

Acknowledgements

We would like to thank Dr. Gianfranco Doretto, Arno Schödl, Antoni B. Chan, and the Radiology Department of Digital Imaging Unit in Geneva University Hospital for sharing their datasets used in this study. This work was supported by the Academy of Finland.

Reference

[1] T. Amiaz, S. Fazekas, D. Chetverikov, and N. Kiryati, Detecting Regions of Dynamic Texture, *Conf. of Scale Space and Variational Methods in Computer Vision*, 2007

[2] J. Anderson, *An Introduction to Neural Networks*. The MIT Press, Cambridge, MA. 1995.

[3] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, High Accuracy Optical Flow Estimation Based on a Theory for Warping, *ECCV*, 2004

[4] A.B. Chan and N. Vasconcelos, Modeling, Clustering, and Segmenting Video with Mixtures of Dynamic Texture, *IEEE TPAMI*, 2008

[5] A.B. Chan and N. Vasconcelos, Layered dynamic textures, *IEEE TPAMI*, 2009

[6] A.B.Chan and N. Vasconcelos, Variational layered dynamic textures,*CVPR*, 2009.

[7] 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, *CVPR* 2009

[8] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, W. Gao. WLD: A Robust Local Image Descriptor, *IEEE TPAMI*, 2010.

[9] J. Chen, G. Zhao and M. Pietikäinen, An improved local descriptor and threshold learning for unsupervised dynamic texture segmentation, *Proc. 2nd IEEE International Workshop on Machine Learning for Vision-based Motion Analysis in ICCV*, 2009.

[10] D. Chetverikov and R. Péteri, A Brief Survey of Dynamic Texture Description and Recognition, *4th Int. Conf. on Computer Recognition Systems*, 2005

[11] D. Chetverikov, S. Fazekas and M. Haindl. Dynamic texture as foreground and background, *Machine vision and Applications*, 2010.

[12] D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, *IEEE TPAMI*, 2002.

[13] L. Cooper, J. Liu and K. Huang, Spatial Segmentation of Temporal Texture Using Mixture Linear Models. *The Dynamical Vision Workshop in ICCV*. 2005

[14] D. Cremers and S. Soatto, Motion Competition: A Variational Approach to Piecewise Parametric Motion Segmentation. *IJCV*, 2004

[15] K. Derpanis and R. Wildes, Early spatiotemporal grouping with a distributed oriented energy representation. CVPR 2009

[16] G. Doretto, A. Chiuso, Y. N. Wu and S. Soatto, Dynamic texture, *ICCV* 2001

[17] G. Doretto, A. Chiuso, Y. N. Wu and S. Soatto, Dynamic Texture Segmentation, *ICCV*, 2003

[18] S. Fazekas T. Amiaz, D. Chetverikov and N. Kiryati, Dynamic Texture Detection Based on Motion Analysis, *IJCV*, 2009

[19] R. J. Ferrari, H. Zhang, and C.R. Kube, Real-time detection of steam in video images, *Pattern Recognition* 2007.

[20] M. Fritz, B. Leibe, B. Caputo, and B. Schiele, Integrating Representative and Discriminant Models for Object Category Detection, *ICCV*, 2005

[21] A. Ghoreyshi and R. Vidal, Segmenting Dynamic Textures with Ising Descriptors, ARX Models and Level Sets, *Proc.ECCV Dynamical Vision Workshop*, 2006.

[22] A.K. Jain, *Fundamentals of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[23] J. Kim and K. Grauman, Observe locally, infer globally: A space-time MRF for detecting abnormal activities with incremental updates. CVPR 2009.

[24] C. Li, C. Xu, C. Gui, and M. D. Fox. Level Set Evolution without Re-initialization: A New Variational Formulation, *CVPR*2005.

[25] S. Milko, E. Samset and T. Kadir, Segmentation of the liver in ultrasound: a dynamic texture approach, *International Journal of Computer Assisted Radiology and Surgery*, 2008

[26] D. W. Murray and B. F. Buxton, Scene Segmentation from Visual Motion Using Global Optimization. *IEEE TPAMI*, 1987

[27] T. Ojala and M. Pietikäinen, Unsupervised Texture Segmentation Using Feature Distributions, *Pattern Recognition,* 1999

[28] T. Ojala, M. Pietikäinen and T. Mäenpää. Multiresolution gray scale and rotation invariant texture analysis with local binary patterns, *IEEE TPAMI* 2002.

[29] T. Ojala, K. Valkealahti, E. Oja and M. Pietikäinen. Texture Discrimination with Multidimensional Distributions of Signed Gray Level Differences. *Pattern Recognition*, 2001.

[30] R. Polana and R. Nelson, Temporal Texture and Activity Recognition, *In Motion-based Recognition*, 1997

[31] A. Rahman and M. Murshed, Detection of Multiple Dynamic Textures Using Feature Space Mapping, *IEEE TCSVT*, 2009

[32] A. Schödl and I. Essa, Controlled Animation of Video Sprites, *SIGGRAPH*, 2002

[33] C.E. Shannon, A Mathematical Theory of Communication, *Bell System Technical Journal*, 1948

[34] J. Shi and J. Malik, Normalized Cuts and Image Segmentation, *IEEE TPAMI*, 2000.

[35] J. Shi and J. Malik, Motion Segmentation and Tracking Using Normalized Cuts, *ICCV*, 1999.

[36] A. Stein and M. Hebert, Occlusion Boundaries from Motion: Low-Level Detection and Mid-Level Reasoning. *IJCV* 2009.

[37] M. Szummer and R. W. Picard, Temporal Texture Modeling, *ICIP*, 1996.

[38] R. Vidal and A. Ravichandran, Optical Flow Estimation and Segmentation of Multiple Moving Dynamic Textures, *CVPR*, 2005

[39] R.Vidal and D.Singaraju, A Closed Form Solution to Direct Motion Segmentation, *CVPR,* 2005.

[40] X. Wang, T. Han and S. Yan, An HOG-LBP Human

Detector with Partial Occlusion Handling, *ICCV,* 2009
[41] G. Zhao and M. Pietikäinen, Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions, *IEEE TPAMI*, 2007