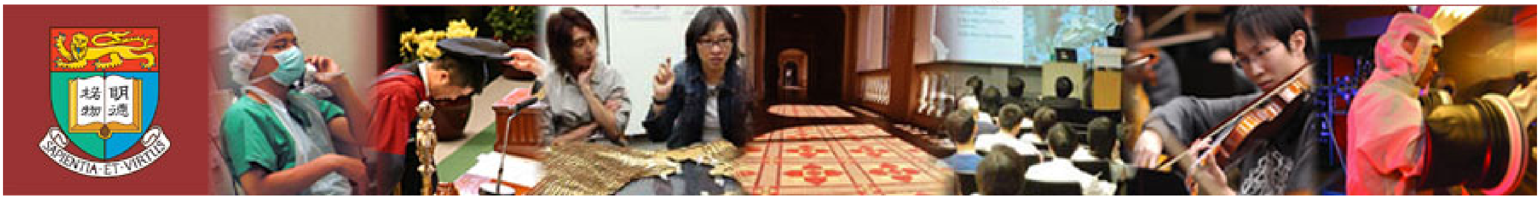


SIAM J. IMAGING SCIENCES
Vol. 6, No. 4, pp. 2484–2520

# Motion Textures: Modeling, Classification, and Segmentation Using Mixed-State Markov Random Fields*

Tomás Crivelli[†], Bruno Cernuschi-Frias[‡], Patrick Bouthemy[§], and Jian-Feng Yao[¶]

**Abstract.** A motion texture is an instantaneous motion map extracted from a dynamic texture. We observe that such motion maps exhibit values of two types: a discrete component at zero (absence of motion) and continuous motion values. We thus develop a mixed-state Markov random field model to represent motion textures. The core of our approach is to show that motion information is powerful enough to classify and segment dynamic textures if it is properly modeled regarding its specific nature and the local interactions involved. A parsimonious set of 11 parameters constitutes the descriptive feature of a motion texture. The motivation of the proposed formulation runs toward the analysis of dynamic video contents, and we tackle two related problems. First, we present a method for recognition and classification of motion textures, by means of the Kullback–Leibler distance between mixed-state statistical models. Second, we define a two-frame motion texture maximum a posteriori (MAP)-based segmentation method applicable to motion textures with deforming boundaries. We also investigate a new issue, the space-time dynamic texture segmentation, by combining the spatial segmentation and the recognition methods. Numerous experimental results are reported for those three problems which demonstrate the efficiency and accuracy of the proposed two-frame approach.

**Key words.** dynamic textures, random fields, motion analysis, segmentation, classification, mixed-state models

**AMS subject classifications.** 62M40, 62H35, 68T45

**DOI.** 10.1137/120872048

## 1. Introduction.

In the context of visual motion analysis, *motion textures* designate video contents similar to those named *temporal* or *dynamic textures* [28, 50, 63], first introduced by Nelson and Polana [50]. The term "textured motion" is also employed in [67], and "space-time texture" is used in [27]. Different from *actions* or *activities* (walking, climbing, playing) and *events* (open a door, answer the phone), *temporal textures* show some type of homogeneity, both in space and time. Mostly, they refer to dynamic video contents displayed by natural scene elements such as flowing rivers, wavy water, falling snow, rising bubbles, spurting fountains, expanding smoke, blowing foliage or grass, and swaying flame. Illustrative samples are displayed in Figure 1. They also encompass any dynamic visual information that, from the observer's point of view, can be classified as a texture with motion. For example, consider a walking person. His/her *activity* can be analyzed as attached to an articulated motion; however, a walking crowd in a far view may show a repetitive motion pattern more adequate

*Received by the editors April 2, 2012; accepted for publication (in revised form) August 1, 2013; published electronically December 5, 2013.
http://www.siam.org/journals/siims/6-4/87204.html

[†]University of Buenos Aires, Argentina, and Inria, Centre Rennes - Bretagne Atlantique, 35042 Rennes Cedex, France (tomas.crivelli@gmail.com).

[‡]CONICET, University of Buenos Aires, Buenos Aires 1063, Argentina (bcf@ieee.org).

[§]Inria, Centre Rennes - Bretagne Atlantique, 35042 Rennes Cedex, France (patrick.bouthemy@inria.fr).

[¶]Department of Statistics and Actuarial Science, The University of Hong Kong, Hong Kong (jeffyao@hku.hk).

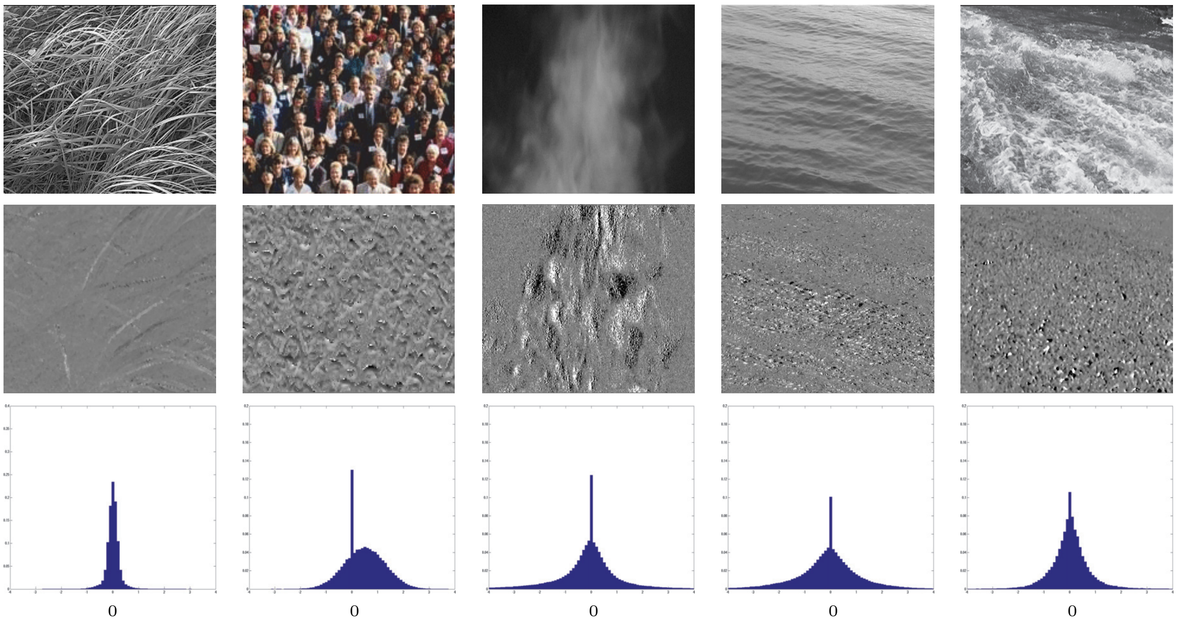


**Figure 1.** *Top row: Sample images from dynamic textures of different kinds (grass, crowd, steam, water, and river). Middle row: Scalar motion map based on normal flow computation and obtained using two consecutive frames of the sequence, which we call a motion texture. Here we mapped the motion measurements to the range of gray* $[0, 255]$ *where* $128$ *corresponds to null motion. Bottom row: Motion histograms from a motion texture. Motion values display two components: A discrete value at zero and a continuous distribution for the rest.*

to be considered a temporal texture. Similar arguments can be raised for traffic views or views of animal flocks.

By definition, a motion texture is an instantaneous motion map obtained from a dynamic texture (Figure 1). This term, introduced in [5], makes reference not to a new type of dynamic phenomena but more specifically to the type of information that is processed and modeled in these classes of video sequences. When analyzing a complex scene, the three types of dynamic visual information (actions, events, and temporal textures) may be present. However, their dissimilar nature leads us to consider substantially different approaches in tasks such as detection, segmentation, classification, and tracking.

From motion detection to optical flow estimation [1, 2, 7, 19, 49], efforts have been devoted to extracting reliable and representative motion quantities from a sequence of images. In recent years, there has been increasing interest in indexing, recognition, classification, and retrieval of long sequences of video data for dynamic content analysis. In this context, motion information has been effectively used as a key feature in dynamic content characterization or action recognition in videos [16, 32, 33, 39, 40, 53, 54, 56].

The goal of this work is to build a unified framework for modeling, categorizing, recognizing, and segmenting textured motion patterns. Additionally, we aim at specifying a compact and efficient modeling from visual motion information only. Indeed, relying on motion information makes the model independent of illumination conditions and irrelevant appearance features. It enables capturing the intrinsic properties of these textures. To deal with reliable and easy-to-compute motion maps, we exploit *(scalar) normal flows*.

We experimentally observe that these motion maps exhibit values appertaining to two different types : a *discrete* component at zero, accounting for absence of motion, and *continuous* motion values. These two types of values are tightly interwoven in the normal flow maps forming a spatial configuration similar to intensity textures (Figure 1). Analogous spatial properties such as texture orientation, isotropy, repetitive local patterns, and statistical interaction are present and are characteristic of the dynamic content of the scene [5]. Moreover, discrete and continuous values are not independent, nor do they constitute two different processes. It is a single (motion) observation process that depicts what we call *mixed-state values* [5, 53].

Therefore, the designed model must acknowledge this particular nature and the related local spatial interactions. It must capture the statistical spatial properties of the observed apparent motion, effectively integrating discrete and continuous values and their mutual interactions. Markov random fields (MRFs) are known as being a powerful statistical representation of general textured patterns, either for modeling intensity or, as in our case, for modeling the spatial distribution of mixed-state motion observations extracted from dynamic textures. We have thus developed a *mixed-state MRF (MS-MRF) model* to represent motion textures. To each site of the motion map is attached a random variable that can take either the discrete null value or motion values distributed according to a continuous density from a family of exponential distributions. This framework will then be exploited for classification and segmentation of dynamic textures as well. The two equivalent representations of an MRF, i.e., local conditional densities and Gibbs distributions, are defined for this mixed-state model. Local conditional densities allow us to estimate the parameters of the field that characterize the motion texture, while the Gibbs energy formulation enables us to tackle the segmentation and the classification problems.

The remainder of this paper is organized as follows. In section 2, we highlight the new contributions presented in this paper. In section 3, we review previous work on dynamic texture characterization and applications to segmentation and classification. We also comment on some discrete-continuous models related to the mixed-state framework. In section 4, the proposed motion measurements are specified, leading to the definition of motion textures and the formulation of their statistical properties. Sections 5, 6, and 7 are devoted to the design of MS-MRFs and the construction of the motion texture model. In sections 8 and 9, the model is used for the classification and the segmentation of motion textures, respectively. A noteworthy feature is that this involves only 11 parameters and requires only two frames. Finally, in section 10, we report experimental results on classification and segmentation. We also include comparative evaluations and investigate several properties of our modeling approach. Section 11 contains our concluding remarks.

## 2. New contributions.

In this section, we put forward the main contributions of the work described in this paper, with respect to our past publications and the state of the art.

We defined a first mixed-state motion texture model in [5], where the theoretical concept of a unified discrete-continuous framework (mixed-state auto-models) was introduced. However, as a first attempt to model motion textures, it resulted in a simple and limited representation which allowed us to conduct simulations and a preliminary analysis of the texture properties expressed by the model. Here, we describe significant extensions of this basic model, ending up in an effective MS-MRF representation of motion textures exploitable for segmentation and classification. The main differences and additions are listed below:

- The mixed-state model is now able to capture more properties of the motion texture such as a deeper spatial correlation between motion values. An extended neighborhood is considered for specifying local conditional characteristics leading to a 11-parameter model where nonzero mean Gaussian distributions are involved.
- The partition function calculation is explicitly tackled and solved.
- A simple but powerful similarity measure between mixed-state models is derived based on the Kullback–Leibler (KL) divergence, allowing motion texture classification and recognition.

Our preliminary work on motion texture segmentation was described in the conference papers [22, 23]. In [22] we used a simulated annealing algorithm for minimizing the energy function and a less accurate estimation of the partition function. The main modifications and extensions are the following:

- More experiments have been conducted on both synthetic and real examples to assess the performance of the method using a graph-cut-based minimization.
- A comparison to a simple MRF motion model is reported in order to show the necessity of the mixed-state approach.
- We have combined the spatial segmentation stage with the recognition of the motion texture classes to address the spatiotemporal segmentation of motion textures in videos, which is the first attempt of this kind to our knowledge.

Our conference paper [25] was devoted to motion texture classification. Several important additions have been made:

- We have further evaluated our method on the UCLA dataset [60] and its variations.
- We provide an analysis of the temporal stability of the two-frame model estimation.
- We exploit the two-frame approach for tackling the problem of motion texture change detection.

Finally, we developed a quite different approach in [24], where we defined a purely causal model, a temporal mixed-state Markov chain, for tracking motion textures along the video sequence.

The most important features or properties of our method which distinguish it from the state-of-the-art methods and the main contributions of this work can be stated as follows:

- We have defined a *parsimonious* compact model for temporal textures. It involves 11 parameters only, relies on normal flows, and accounts for local spatial interactions. The inherent mixed-state nature of the motion information is explicitly formalized. Normal flows are reliable enough in many situations and easy to compute compared to optical flows. Normal flows are more *intrinsic* to the dynamic texture nature than intensity values used in the linear dynamical system (LDS) based methods [12, 28, 56] while still conveying partial information on the texture appearance, since they are linked to the spatial intensity gradients.
- This is an *instantaneous* model in the sense that it grasps only the spatial structure of the motion texture, knowing that the temporal information is brought by the computed measurements. Then, our motion texture model requires only *two frames* to be prop-

erly estimated, which imparts several unique advantages. Without any changes in the method, we can segment motion textures with either fixed boundaries, or deformable boundaries, or moving spatial support. We can deal with temporally stationary and nonstationary motion textures as well.

- The *same* modeling framework can be exploited for classification (or recognition) and for segmentation of multiple dynamic textures.
- *Space-time* segmentation of temporal textures can be straightforwardly addressed by combining the spatial segmentation and the classification methods.

**3. Related work.** Whereas two-dimensional (2D) spatial textures have been vastly analyzed in the computer vision literature [26, 31, 44], temporal or dynamic textures have received increasing interest in recent years. A first distinction between different approaches lies in the type of image features utilized from the image sequence. Doretto et al. [28] proposed the use of autoregressive moving average (ARMA) models directly on image intensities for dynamic texture synthesis. In [71] an improvement is proposed based on a control theory approach. Other intensity-based dynamic texture models and their applications can be found in [12, 13, 30, 58, 63]. Recently a wavelet-based representation has been proposed in [41] as a descriptor for dynamic textures.

Alternatively, there has been increasing interest in the modeling of motion features extracted from dynamic textures in contrast to pixel-based intensity representations [5, 22, 32, 54]. Particularly, normal flow is an efficient and natural way of characterizing the local spatiotemporal dynamics of a temporal texture [22, 25, 32, 33, 51]. A survey on dynamic texture characterization can be found in [17]. The combination of appearance and motion modes, the descriptor used for the latter being histograms of oriented optical flow (HOOFs), has been recently explored in [15]. The generative model developed in [67] for textured motions encompasses three components: a photometric model based on Gabor, LoG, and Fourier bases, a geometric one called “moton,” and a dynamic one involving Markov chains. It is mainly dedicated to handling natural scenes formed by a number of particle and wave elements.

Much effort in dynamic texture analysis has been devoted to the recognition and classification of these types of image sequences. Methods based on motion features give high recognition and classification rates for dynamic textures depicting natural scenes [16, 22, 47, 51]. They are based on computing motion statistics across the image sequence and using them as class descriptors for the classification task. A different approach to dynamic texture recognition was formerly proposed in [60], where a dissimilarity measure between LDSs estimated from the image intensity sequences is utilized. It has been further investigated in [12] and more recently in [56, 57], where the bag-of-words framework is applied to LDSs. An efficient clustering method is defined in [9]. Local efficient texture features have been employed as well, such as the spatiotemporal orientation analysis designed in [27], to represent and recognize spacetime textures or the local binary patterns (LBPs) in [59, 66, 72]. Finally, the extension of the concept of dynamic textures to more complex scenes requires considering more elaborated approaches to specific applications such as crowd analysis [48, 66, 69], dynamic texture synthesis [46], and facial expression recognition [72].

The segmentation of dynamic textures amounts to determining and locating regions in the image that correspond to a dynamic texture class against the scene background, or to separating different dynamic texture classes in cases of multiple dynamic textures. Mostly, existing segmentation methods of dynamic textures are based on linear intensity models, as proposed originally by Doretto et al. [28]. In [29], a level-set variational approach is formulated for minimizing a cost functional with respect to (w.r.t.) geometric discrepancy measures between models and w.r.t. the boundary of the different regions. Vidal and Ravichandran [65] have proposed modeling each dynamic texture as an LDS plus a 2D translational motion component. This solves the problem of moving regions (or nonstatic camera) under rigid motion. Segmentation is achieved by a generalized principal component analysis (PCA) method for subspace separation. In [18], a mixture of linear models is used coupled with the generalized PCA technique, but the segmentation results are rather fragmented. Categorization and segmentation are jointly addressed in [57]. Chan and Vasconcelos [12] and [13] have chosen to simultaneously model several dynamic textures, which naturally leads to a segmentation approach. In the former, a mixture generative model of multiple linear dynamic textures (LDTs) is defined, while in the latter an explicit spatial model of layer distribution is introduced, which improves the results on segmentation. A different approach is followed in [15], where a split-and-merge strategy is adopted associated with a pixelwise classification based on the Weber distance between HOOFs. Detection of flames is addressed in [64] using Markov models.

One of the limitations of the linear dynamical intensity model is the need of processing a whole group of successive frames of the sequence in order to estimate the dynamic texture models. This in practice leads to a restrictive assumption of temporal homogeneity of the video content over the considered time interval. It also means that the region occupied by a certain dynamic texture cannot vary considerably in time and that a spatial segmentation is computed within a given set of frames. This is convenient for a dynamic texture with static boundaries over time, but this approach cannot appropriately handle a deformable support of the dynamic texture (e.g., a fire flame, a water leakage, or an explosion). Eventually, small spatiotemporal volumes could be used to track boundaries and their variations as shown in [12], but as they point out, the temporal extent should be large enough to capture the distinguishing characteristics of the dynamic texture. An extension to [13], called temporally switching LDT (TS-LDT), is described in [14] to handle layer shapes changing over time. The segmentation method described in [36] relies on a variational level-set framework. The local spatial properties of the dynamic textures are specified with Ising descriptors and their temporal evolution by an autoregressive exogenous (ARX) model. First, it is applied to dynamic textures with fixed boundaries. Then, an extension based on an iterated process is proposed to handle dynamic textures with moving boundaries and still requires several successive frames. We will see how our mixed-state spatial model of motion textures (requiring only two frames) is able to capture these complex dynamic variations in a simpler frame-by-frame basis.

As for other uses of mixed-state type distributions in computer vision, it is worth mentioning previous works on fuzzy pixel classifications such as [61] and [62], where a class of fuzzy Markov models is introduced, defining two hard classification states $x_i = 0$ or $x_i = 1$ that have a positive probability, while all the soft classification states, i.e., $x_i \in (0, 1)$, follow a continuous distribution with an ad hoc density function. These models are restricted to classification problems with a fixed state space $[0, 1]$. Finally, in [8, 21] the concept of a mixed-state variable is extended to discrete states that take symbolic or abstract labels instead of a specific numeric value as considered here. This enables us to solve simultaneous decision-estimation problems, such as motion detection and background reconstruction [21], in a unified way.

## 4. Motion textures.

Let $I_i(t)$ be a scalar function that represents the image intensity at image point $i \in S$ for time $t$, where $S$ denotes the image grid containing $N$ points. A motion texture is extracted by computing scalar motion measurements between two consecutive images, say, $I(t-1)$ and $I(t)$, for some given instant. The definition can be extended to any motion measure on the image sequence. This accounts for vectorial observations as well.

Briefly, our approach consists in the following: once the motion measurement is obtained, a sequence of intensity images is substituted for a sequence of *motion maps or fields*. These fields are in essence a function of the bidimensional location, and the problem is reduced to modeling a spatial distribution of motion values.

### 4.1. Local motion measurements.

Obtaining reliable and, at the same time, easily computable motion information from image sequences is essential in our formulation, which is intended for the analysis of large amounts of data while avoiding the problem of explosion of the model dimension. Here, we follow the approach described in [22, 24]. We emphasize that the objective is not motion estimation by itself, but dynamic content analysis.

The optical flow constraint [38] is a condition over intensity images from which velocity fields can be effectively estimated. Locally, it gives valuable information about the spatiotemporal structure of the scene. However, the *aperture problem* allows us to measure only the component of the velocity of an image point in the direction of the spatial intensity gradient, i.e., *normal flow*, defined as

$$\mathbf{V}_i^{\perp}(t) = -\frac{\frac{\partial I_i(t)}{\partial t}}{\parallel \boldsymbol{\nabla} I_i(t) \parallel} \frac{\boldsymbol{\nabla} I_i(t)}{\parallel \boldsymbol{\nabla} I_i(t) \parallel}, \tag{4.1}$$

where $\nabla I_i(t)$ is the spatial intensity gradient at location $i$. We first compute a weighted vectorial average of normal flow over a small local window $W$ centered at pixel $i$:

$$\tilde{\mathbf{V}}_i^{\perp}(t) = \frac{\sum\limits_{j \in W} \mathbf{V}_j^{\perp}(t) \parallel \boldsymbol{\nabla} I_j(t) \parallel^2}{\max(\sum\limits_{j \in W} \parallel \boldsymbol{\nabla} I_j(t) \parallel^2, \eta^2)}, \tag{4.2}$$

where $\eta^2$ is a constant related to noise. This average results in a denoised local estimation of normal flow. The projection of this quantity over the intensity gradient direction gives rise to the following scalar motion observation:

$$v_i(t) = \tilde{\mathbf{V}}_i^{\perp}(t) \cdot \frac{\boldsymbol{\nabla} I_i(t)}{\parallel \boldsymbol{\nabla} I_i(t) \parallel}, \tag{4.3}$$

with $v_i(t) \in (-\infty, +\infty)$. As said before, (4.3) gives a good compromise between quality of estimation and simplicity of calculation.

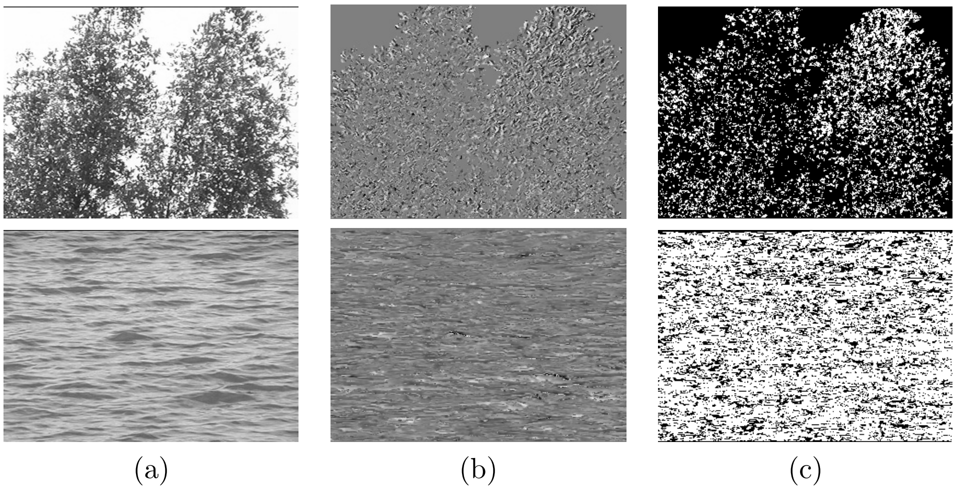


**Figure 2.** (a) *Sample images from motion textures.* (b) *The scalar motion values are spatially distributed, forming a textured pattern. Here we mapped the motion measurements to the range of gray* $[0, 255]$ *where* 128 *corresponds to null motion.* (c) *The binary motion–no-motion map also is distributed following a textured pattern. White represents a motion value different from zero.*

**4.2. Statistical properties of motion measurements.** At this stage, we emphasize a fundamental property of the scalar motion field (4.3). Let us observe Figure 1 (last row). The plotted histograms reflect the statistical distributions of motion values for different motion textures (assuming at that stage independent and identically distributed random variables at each image location). Note that the motion measurements depict two distinguishable elements: a predominant discrete component at the null value $v_i = 0$, and a continuous distribution for the rest of the motion values. This is a typical characteristic of the motion measurements extracted from motion textures.

This observation could naively lead us to model only the motion histograms to characterize a motion texture. However, the significant null value component appears repeatedly in the motion maps, producing a textured binary (motion/no-motion) pattern as spatial layout (Figure 2c). Analogously, continuous motion values are spatially correlated (Figure 2b). As such, discrete and continuous values are not independently distributed in space, indeed displaying a mixed-state texture pattern. In other words, we have to model physical data that display mixed-state values and local interactions. In this context, the discrete null motion value has a specific place in the sample space and, consequently, has to be modeled accordingly.

We call these types of fields *mixed-state random fields* as the corresponding random variables take their values in a mixed discrete-continuous space [5].

**5. Mixed-state random variables.** The key observation made in the previous section about the statistical properties of motion measurements requires an adequate representation of the associated random variables. Consider the case of a random variable that is 0 with probability $\rho$ or is distributed following a continuous density with probability $1-\rho$. We proceed directly to define a probability measure for a mixed-state random variable, resorting to the theory of measure and integration [5, 8]. We can then construct a mixed-state probability density defined as

$$p(x) = \rho \mathbf{1}_0(x) + \rho^* \mathbf{1}_0^*(x) p^c(x), \tag{5.1}$$

with $\rho \in [0,1]$, $\rho^* = 1 - \rho$, and where we define the characteristic functions as

$$\mathbf{1}_0(x) = \begin{cases} 1 & \text{if } x = 0, \\ 0 & \text{if } x \neq 0, \end{cases} \qquad \mathbf{1}_0^*(x) = 1 - \mathbf{1}_0(x), \tag{5.2}$$

and $p^c(x)$ is a continuous probability density function. The density $p(x)$ in (5.1) is given w.r.t. a reference measure $m(dx) = m_0(dx) + \lambda(dx)$, where $m_0(dx)$ is a counting measure for the value 0 and $\lambda(dx)$ is the usual Lebesgue measure, i.e., the length of the interval in the real line. Interpret this equation as follows: the density function $p(x)$ assigns a probability mass $\rho$ to the discrete value (here, zero) and acts as a continuous density function $p^c(x)$ for the nonzero continuous values.

The reader with a background on measure theory and probability will note that this also enables us to generalize the case of a specific real value considered as a discrete value (i.e., $x = 0$) to a generic discrete symbolic value or abstract label that may lie on an arbitrary label set (as investigated in [21] for motion detection by background subtraction). A full formulation of mixed-state random variables and distributions following a measure theoretic approach can be found in [5, 8, 21, 37].

## 6. Mixed-state Markov models.

As the Hammersley–Clifford theorem states, MRFs with an everywhere positive density function are equivalent to Gibbs distributions. The joint pdf of the random variables that compose the field has the form $p(\mathbf{X}) = \exp[-Q(\mathbf{X})]/Z$, where $Q(\mathbf{X})$ is an energy function and $Z$ is called the partition function or normalizing factor of the distribution. The power of these models was primarily demonstrated in [3] with the introduction of the so-called *auto-models* and their numerous applications.

Let $S$ be a lattice of $N$ points or image locations such that $\mathbf{X} = \{x_i\}_{i \in S}$. Define $\mathbf{X}_A$ as the subset of random variables restricted to $A \subset S$, i.e., $\mathbf{X}_A = \{x_i\}_{i \in A}$. Then the Markovian property yields $p(x_i \mid \mathbf{X}_{S \setminus \{i\}}) = p(x_i \mid \mathbf{X}_{\mathcal{N}_i})$, where $\mathcal{N}_i \in S$ is a set of sites called the neighborhood of location $i$.

The Markovian property is as well expressed in the global form of the process. The energy $Q(\mathbf{X})$ can be expressed as a sum of potential functions, $Q(\mathbf{X}) = \sum_{\mathcal{C} \subset S} V_{\mathcal{C}}(\mathbf{X}_{\mathcal{C}})$, where the summation runs over those subsets $\mathcal{C}$ of $S$ such that $V_{\mathcal{C}} \neq 0$, called *cliques* [3].

For an MS-MRF model, we have the following local conditional mixed-state densities:

$$p(x_i \mid \mathbf{X}_{\mathcal{N}_i}) = \rho(\mathbf{X}_{\mathcal{N}_i}) \mathbf{1}_0(x_i) + \rho^*(\mathbf{X}_{\mathcal{N}_i}) \mathbf{1}_0^*(x_i) p^c(x_i \mid \mathbf{X}_{\mathcal{N}_i}), \tag{6.1}$$

where $\rho(\mathbf{X}_{\mathcal{N}_i}) = P(x_i = 0 \mid \mathbf{X}_{\mathcal{N}_i})$ is now a function of the values taken by the neighbors. In this context, the arising questions are as follows: can we arbitrarily choose conditional pdfs in (6.1), and, which is the general form for the joint distribution of the field that responds to such a formulation? For a certain family of conditional distributions, the answer is known, and we give in what follows a useful result to be applied to MS-MRFs.

As stated in [3], when the conditional probability densities that define the local characteristics of an MRF belong to a one-parameter exponential family, and assuming that the corresponding global Gibbs energy depends only on cliques that contain no more than two sites, i.e., *auto-models*, the expression for the parameter is known as an affine function of a sufficient statistic of the neighbors. In the case of $d$-parameter auto-models, this result has been extended through the following proposition [37].

Theorem 1. *Assume a second-order MRF, where the local conditional characteristics belong to the d-parameter exponential family, i.e.,* $\log p\,(x_i \mid \mathbf{X}_{\mathcal{N}_i}) = \boldsymbol{\Theta}_i^T(\mathbf{X}_{\mathcal{N}_i})\mathbf{S}_i(x_i) + C_i(x_i) + D_i(\mathbf{X}_{\mathcal{N}_i})$*, with* $\mathbf{S}_i(x_i) \in \mathbb{R}^d$*,* $\boldsymbol{\Theta}_i(\mathbf{X}_{\mathcal{N}_i}) \in \mathbb{R}^d$*, and* $C_i(x_i)$ *and* $D_i(\mathbf{X}_{\mathcal{N}_i}) \in \mathbb{R}$*. Then, the conditional densities are restricted to the form given by*

$$\boldsymbol{\Theta}_i(\mathbf{X}_{\mathcal{N}_i}) = \boldsymbol{\alpha}_i + \sum_{j\in\mathcal{N}_i} \boldsymbol{\beta}_{ij}\mathbf{S}_j(x_j), \tag{6.2}$$

*with* $\boldsymbol{\beta}_{ij} \in \mathbb{R}^{d\times d}$ *and* $\boldsymbol{\alpha}_i = [\alpha_1 \dots \alpha_d]^T \in \mathbb{R}^d$*; and the energy potential functions take the form*

$$\begin{aligned} V_i(x_i) &= -\boldsymbol{\alpha}_i^T \cdot \mathbf{S}_i(x_i) - C_i(x_i), \\ V_{ij}(x_i, x_j) &= -\mathbf{S}_i(x_i)^T \boldsymbol{\beta}_{ij}\mathbf{S}_j(x_j). \end{aligned} \tag{6.3}$$

For a complete proof see [37]. Consequently, $\boldsymbol{\Theta}_i(\mathbf{X}_{\mathcal{N}_i})$ is a function of the neighbors of a particular location $i$, where the conditional dependence between sites cannot be arbitrary under the mentioned hypotheses, having a particular shape as seen in (6.2). Note that the matrices $\boldsymbol{\beta}_{ij}$ define the pairwise interaction between neighboring points. To ensure the symmetry condition, $V_{ij}(x_i, x_j) = V_{ji}(x_j, x_i)$, we have $\boldsymbol{\beta}_{ij} = \boldsymbol{\beta}_{ji}^T$.

In the next section, we design a mixed-state auto-model for motion texture modeling exploiting these results. Extensions of the new model, with respect to the basic one presented in [5], enable us to capture more properties of the analyzed motion textures. First, we use an extended neighborhood for local conditional characteristics. Second, we consider a nonzero mean Gaussian distribution for the continuous part which allows us to express a stronger spatial correlation between continuous motion values. Consequently, we can now handle real dynamic content analysis issues such as segmentation and recognition of motion textures.

## 7. A motion texture model.

As already said, our approach of modeling the instantaneous motion maps associated to dynamic textures amounts to introducing a spatial field of mixed-state values. In general terms, the proposed conditional models could be defined by a different set of parameters for each location of the image (see (6.2)). This would give rise to a motion texture model with a number of parameters proportional to the image size. However, such a high-dimensional representation is not required as motion textures usually exhibit stationarity properties. It is also unfeasible in practice and does not constitute a compact description of motion textures. Moreover, an increasing number of frames would be necessary for the estimation process. This is against a formulation oriented to efficient content segmentation and classification.

Henceforth, we will assume that the extracted motion fields can be considered as a realization of a homogeneous spatial model in the case of single motion textures. If several dynamic textures are present, it then corresponds to a realization of piecewise homogeneous spatial models. Indeed, the visual information attached to a dynamic texture is mostly displayed from spatially homogeneous motion regions and, moreover, mostly associated to statistically homogeneous textured intensity patterns. Nevertheless, note that no temporal stationarity hypothesis is needed in our theoretical framework, in contrast to other approaches [12, 13, 28]. This will be further validated in the reported experiments. To summarize, our model is based on three main assumptions:

1. The mixed-state Gibbs energy is composed of at most second-order potentials. Pairwise interaction is a good trade-off between simplicity and representativeness.
2. The continuous part of the conditional mixed-state densities is chosen to be a Gaussian distribution. On one side, this is coherent with the observed motion histograms (Figure 1). Moreover, it permits us to capture fundamental properties of the motion textures not considered in previous mixed-state approaches [5] along with a high discriminative power through a tractable and parsimonious representation. Nevertheless, the proposed framework could involve other choices for the continuous distributions as well.
3. We define $\mathcal{N}_i = \{i_E, i_W, i_N, i_S, i_{NW}, i_{SE}, i_{NE}, i_{SW}\}$ as the set of the eight-nearest neighbors for location $i$, where, for example, $i_E$ is the East neighbor of $i$ in the image grid, $i_{NW}$ the North-West neighbor, etc. This permits the model to better capture the orientation of the motion textures as both discrete and continuous scalar motion values usually show an anisotropic behavior.

### 7.1. Gaussian mixed-state model.

For the case of a Gaussian mixed-state conditional density we write

$$p(x_i \mid \mathbf{X}_{\mathcal{N}_i}) = \rho_i \mathbf{1}_0(x_i) + (1-\rho_i)\mathbf{1}_0^*(x_i)\frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{(x_i-m_i)^2}{2\sigma_i^2}}, \tag{7.1}$$

where $\rho_i \equiv \rho(\mathbf{X}_{\mathcal{N}_i})$, $m_i \equiv m(\mathbf{X}_{\mathcal{N}_i})$, and $\sigma_i \equiv \sigma(\mathbf{X}_{\mathcal{N}_i})$ for simplicity of notation. In what follows we apply Theorem 1. Equation (7.1) can be written in an exponential form, yielding

$$\log p(x_i \mid \mathbf{X}_{\mathcal{N}_i}) = \mathbf{\Theta}_i^T(\mathbf{X}_{\mathcal{N}_i})\mathbf{S}_i(x_i) + C_i(x_i) + D_i(\mathbf{X}_{\mathcal{N}_i}), \tag{7.2}$$

with

$$\begin{aligned}
\mathbf{\Theta}_i^T(\mathbf{X}_{\mathcal{N}_i}) &= [\theta_{1,i}\ ,\ \theta_{2,i}\ ,\ \theta_{3,i}] = \left[-\frac{m_i}{2\sigma_i^2} - \log\sigma_i\sqrt{2\pi} + \log\frac{1-\rho_i}{\rho_i}\ ,\ \frac{1}{2\sigma_i^2}\ ,\ \frac{m_i}{2\sigma_i^2}\right],\\
\mathbf{S}_i^T(x_i) &= \left[\mathbf{1}_0^*(x_i)\ ,\ -x_i^2\ ,\ x_i\right],\\
C_i(x_i) &= 0,\\
D_i(\mathbf{X}_{\mathcal{N}_i}) &= \log\rho_i. \qquad (7.3)
\end{aligned}$$

The Gaussian mixed-state density results in a 3-parameter exponential family where the parameterization of the conditional distribution in terms of $\mathbf{\Theta}_i$ allows us to express the dependence of a point on its neighbors through (6.2). Moreover, the parameters of the original parameterization, $\rho_i, m_i, \sigma_i$, are also functions of the neighborhood and can be obtained easily from the first line of (7.3), resulting in

$$\rho_i = \frac{(\sigma_i\sqrt{2\pi})^{-1}}{(\sigma_i\sqrt{2\pi})^{-1} + e^{\theta_{1,i}+\frac{m_i^2}{2\sigma_i^2}}}, \qquad \sigma_i^2 = \frac{1}{2\theta_{2,i}}, \qquad m_i = \frac{\theta_{3,i}}{2\theta_{2,i}}. \tag{7.4}$$

**7.2. Defining the set of parameters.** From (6.3) and (6.4), we see how to obtain the potential functions from the conditional density expansion, and consequently the expression for the Gibbs energy,

$$Q(\mathbf{X}) = -\left\{ \sum_i \boldsymbol{\alpha}_i^T \cdot \mathbf{S}_i(x_i) + C_i(x_i) + \sum_{i,j} \mathbf{S}_i(x_i)^T \boldsymbol{\beta}_{ij} \mathbf{S}_j(x_j) \right\}. \tag{7.5}$$

The full model is defined by the matrices $\boldsymbol{\beta}_{ij}$ and $\boldsymbol{\alpha}_i$:

$$\boldsymbol{\beta}_{ij} = \begin{pmatrix} d_{ij} & e_{ij} & f_{ij} \\ e'_{ij} & g_{ij} & q_{ij} \\ f'_{ij} & q'_{ij} & h_{ij} \end{pmatrix}, \quad \boldsymbol{\alpha}_i = \begin{bmatrix} a_i & b_i & c_i \end{bmatrix}^T. \tag{7.6}$$

One may rely on some principled assumptions that lead to reducing the order of the model. First, it is desirable that the conditional mean of the continuous values for a site depend linearly on the neighbors in order to effectively obtain a Gaussian texture for the continuous values. This is a fundamental extension w.r.t. the fixed null mean proposed in [5]: it enables us to extract the main properties of the field for recognition and segmentation purposes, while also keeping a reduced number of parameters to be estimated. To achieve this, we first set $q'_{ij} = f'_{ij} = e'_{ij} = g_{ij} = q_{ij} = 0$. Moreover, from the symmetry of the MRF potentials one necessarily has $e_{ij} = f_{ij} = 0$. Setting $m(\mathbf{X}_{\mathcal{N}_i}) \neq 0$ not only affects the continuous part but also enforces interaction between continuous and discrete states through $\rho_i$ in (7.4). The parameters are then

$$\boldsymbol{\beta}_{ij} = \begin{pmatrix} d_{ij} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & h_{ij} \end{pmatrix}, \quad \boldsymbol{\alpha} = \begin{bmatrix} a & b & c \end{bmatrix}^T. \tag{7.7}$$

Note that the homogeneity of the field leads us to set $\boldsymbol{\alpha}_i = \boldsymbol{\alpha}$ for the first-order potentials. With this choice and from (6.2) and (7.4), we obtain

$$m_i = \frac{c}{2b} + \sum_{j \in \mathcal{N}_i} \frac{h_{i,j}}{2b} x_j \quad \text{and} \quad \sigma_i = \frac{1}{2b}. \tag{7.8}$$

We refer the reader to [8] for a thorough theoretic discussion about the shape of the potentials for an MS-MRF.

A necessary condition in order to define a homogeneous and stationary spatial process is that the parameters related to symmetric neighbors (E-W, N-S, NW-SE, NE-SW) must be the same. This also implies the symmetry of the parameters for the second-order potentials. Thus, for the eight-point neighborhood, we have four interacting directions: vertical (V), horizontal (H), diagonal (D), and antidiagonal (AD). Then, $\boldsymbol{\beta}_{ij} = \boldsymbol{\beta}_{ji} = \boldsymbol{\beta}_k$ with $k \in \{H, V, D, AD\}$. Finally, a homogeneous Gaussian mixed-state model is defined by the 11 parameters

$$\boldsymbol{\phi} = \{a, b, c, d_H, h_H, d_V, h_V, d_D, h_D, d_{AD}, h_{AD}\}.$$

Another aspect related to spatial interaction is considered in the definition of the model. The type of motion textures that we want to study exhibit local motion smoothness, mostly associated to *cooperative* schemes. Then, this condition is explicitly imposed in the model, resulting in a constraint on the parameters. Formally, in a mixed-state cooperative model, the conditional mean of the continuous component for a site has to be an increasing function of its neighbors. See [37] for further comments. Following (7.8), this implies that $h_{ij} \geq 0$. Finally, we can write the full expression of the global Gibbs energy:

$$Q(\mathbf{X}) = -\sum_{i} a\mathbf{1}_0^*(x_i) - bx_i^2 + cx_i - \sum_{\substack{\langle i,j\rangle \\ j\in\mathcal{N}_i}} h_{ij}x_ix_j + d_{ij}\mathbf{1}_0^*(x_i)\mathbf{1}_0^*(x_j). \tag{7.9}$$

Note that we have obtained a compact and parsimonious representation of a motion texture as expressed by both the conditional mixed-state densities (defined by (7.8) and (7.4)) and the joint Gibbs distribution (7.9). Through only 11 parameters and a single model we are able to characterize: the orientation of the field, the spatial correlation between continuous values ((7.8) and quadratic terms in (7.9)), the probability (spatial density) of no-motion values, the spatial correlation between discrete values (discrete terms in (7.9)), and the correlation between discrete and continuous values (see the form of $\rho_i$ in (7.4)).

We finally need to check that the Gibbs density defined by the energy function in (7.9) is integrable. A sufficient and necessary condition for the proposed homogeneous cooperative MS-MRF is $b > \sum_j \frac{h_{ij}}{2}$. See [22] for a proof.

### 7.3. Parameter estimation.

We adopt the pseudolikelihood maximization criterion [3]. Therefore, we search the set of parameters $\hat{\boldsymbol{\phi}}$ that maximizes the function $L(\boldsymbol{\phi}) = \sum_{i\in S} \log p(x_i \mid \mathbf{X}_{\mathcal{N}_i}, \boldsymbol{\phi})$. We use a gradient descent technique for the optimization as the derivatives of $L$ w.r.t $\boldsymbol{\phi}$ are known in closed form. For this issue, having a complete representation of the model by means of the conditional densities in order to apply the pseudolikelihood method is crucial. Indeed, estimating the parameters from the joint Gibbs distribution would require the calculation of the partition function, which is intractable.

## 8. Recognition of motion textures.

Recognition or classification of motion textures necessitates defining a similarity measure between models. In this context, the KL divergence is a well-known distance (more precisely, a pseudodistance) between statistical models [20]. We now explain how to obtain an expression for the case of mixed-state models that will allow us to classify motion textures.

### 8.1. A similarity measure between mixed-state models.

The KL divergence from a density $p_1(\mathbf{X})$ to $p_2(\mathbf{X})$ [20] is given by

$$KL(p_1\|p_2) = \int_\Omega p_1(\mathbf{X}) \log \frac{p_1(\mathbf{X})}{p_2(\mathbf{X})} dm(\mathbf{X}). \tag{8.1}$$

As a distance, one considers the symmetrized KL divergence:

$$d_{KL}(p_1, p_2) = \frac{1}{2}\left[KL(p_1(\mathbf{X})\|p_2(\mathbf{X})) + KL(p_2(\mathbf{X})\|p_1(\mathbf{X}))\right]. \tag{8.2}$$

Now, if $p_1(\mathbf{X})$ and $p_2(\mathbf{X})$ are MRFs, then $\log \frac{p_1(\mathbf{X})}{p_2(\mathbf{X})} = \Delta Q(\mathbf{X}) + \log \frac{Z_2}{Z_1}$, where $\Delta Q(\mathbf{X}) = Q_2(\mathbf{X}) - Q_1(\mathbf{X})$, and

$$d_{KL}(p_1(\mathbf{X}), p_2(\mathbf{X})) = \frac{1}{2}\left(E_{p_1}\left[\Delta Q(\mathbf{X})\right] - E_{p_2}\left[\Delta Q(\mathbf{X})\right]\right), \tag{8.3}$$

where $E_{p_k}[\cdot]$ designates the expectation w.r.t. density $p_k$. We observe from this general expression that we do not need to know the partition functions of the Gibbs distributions, which enormously simplifies the handling of this equation. Now, let $p_1(\mathbf{X})$ and $p_2(\mathbf{X})$ be two Gaussian MS-MRFs. Then,

$$E_{p_k}\left[\Delta Q(\mathbf{X})\right] = \sum_i \Delta\boldsymbol{\alpha} E_{p_k}\left[\mathbf{S}(x_i)\right] + \sum_{\langle i,j\rangle} E_{p_k}\left[\mathbf{S}(x_i)\Delta\boldsymbol{\beta}_{ij}\mathbf{S}(x_j)\right], \tag{8.4}$$

where $\Delta\boldsymbol{\alpha} = \boldsymbol{\alpha}^{(2)} - \boldsymbol{\alpha}^{(1)}$ and $\Delta\boldsymbol{\beta}_{ij} = \boldsymbol{\beta}_{ij}^{(2)} - \boldsymbol{\beta}_{ij}^{(1)}$. As we consider a spatial homogeneous model, the expectations in (8.4) are equal for each site of the motion field. They are computed by generating synthetic fields of small size (typically 128 × 128) using a Gibbs sampler [34] from which we can estimate the involved expectations and finally calculate the divergence.

## 9. Motion texture segmentation.

The motion texture segmentation problem is equivalent to assigning a label to each point in the image grid, indicating that it belongs to a certain motion texture class. In our method, the representation of a motion texture with a relatively small set of parameters permits a parsimonious characterization of the different parts of a scene consisting of more than one dynamic texture. Moreover, the lack of a temporal homogeneity assumption allows us to overcome some of the limitations of the existing dynamic texture segmentation methods. Here, we follow a Bayesian approach for determining in an optimal way the distribution of the motion texture labels with the motion map as input data.

Thus, we search for a label realization $\mathbf{l} = \{l_i\}$, where $l_i \in \{0, 1, \ldots, c-1\}$ is the motion texture class label value at site $i$ that maximizes $p(\mathbf{l} \mid \mathbf{X}) \propto p(\mathbf{X} \mid \mathbf{l})p(\mathbf{l})$, where $\mathbf{X}$ represents the motion map including up to $c$ motion textures. This corresponds to a maximum a posteriori (MAP) estimation of the label field $\mathbf{l}$.

In the proposed method we do not assume conditional independence, given the label field, within a motion texture but only between different motion texture classes. We introduce the following notation. We call $Q_k$ the energy function corresponding to the texture class $k$ with parameters $\boldsymbol{\phi}_k$. We define $\mathbf{X}^{(k)} = \{x_i : l_i = k\}$ as the vector of motion random variables that belong to texture $k$. $\mathbf{X}^{(k)}$ is a subset of $\mathbf{X}$. $Z_k(\mathbf{l})$ is the corresponding partition function and depends on the distribution of the $k$th texture on the lattice corresponding to the label field $\mathbf{l}$.

If we suppose that the $c$ different motion textures come from independent dynamic phenomena, given the label field, we can write

$$p(\mathbf{X} \mid \mathbf{l}) = \prod_{k=0}^{c-1} p(\mathbf{X}^{(k)}) = \prod_{k=0}^{c-1} \frac{e^{-Q_k(\mathbf{X}^{(k)})}}{Z_k(\mathbf{l})}. \tag{9.1}$$

This approach allows us to account for conditional dependence. The only approximation is that we do not formally account for interactions between different motion textures along the boundaries of each dynamic texture.

For the a priori information on the segmentation label field, $p(\mathbf{l})$, we introduce another eight nearest-neighbor MRF that behaves as a regularization term for the labeling process. We have $p(\mathbf{l}) \propto \exp[Q_S(\mathbf{l})]$ with

$$Q_S(\mathbf{l}) = \sum_{\langle i,j \rangle} \gamma \psi(l_i, l_j), \tag{9.2}$$

where $\psi(l_i, l_j) = 1$ if $l_i = l_j$ and zero otherwise. With $\gamma > 0$, $p(\mathbf{l})$ penalizes the differences of labeling between adjacent neighbors, smoothing the segmentation output. The complete formulation can be stated as the minimization of

$$E(\mathbf{l}) = \sum_{k=0}^{c-1} Q_k(\mathbf{X}^{(k)}) + \sum_{k=0}^{c-1} \log(Z_k(\mathbf{l})) - Q_S(\mathbf{l}). \tag{9.3}$$

### 9.1. Initialization.

We do not assume that the motion texture parameters for each class are known. Then, it is necessary to correctly estimate the intervening mixed-state motion texture models. As a simplification, we will assume that the number of classes is known.

As an initialization of the label field, we divide the motion map into nonoverlapping square blocks $B_m \in S$ of a fixed size, and for each block the set of 11 motion-texture model parameters is estimated. Then, we apply a clustering technique to obtain a first classification of blocks. As a simplification, we calculate the symmetrized KL distance, $KL(p_1(x), p_2(x)) = \frac{1}{2}(KL(p_1(x)\|p_2(x)) + KL(p_2(x)\|p_1(x)))$, between the marginal distribution for single points $p(x_i)$ (which is assumed to be the same for every site within a block). This simplified distribution is taken as a mixed-state Gaussian density, i.e., $p(x) = \rho \mathbf{1}_0(x) + (1-\rho)\mathcal{N}(\mu, \sigma)$, for which the three parameters $(\rho, \mu, \sigma)$ are easily obtained for each block. The KL divergence from $p_1(x)$ to $p_2(x)$ is given by

$$\begin{aligned} KL(p_1(x)\|p_2(x)) = \rho_1 \log\left[\frac{\rho_1}{\rho_2}\right] &+ (1-\rho_1)\left[\log\left[\frac{\sigma_2(1-\rho_1)}{\sigma_1(1-\rho_2)}\right]\right. \\ &\left.+ \frac{1}{2}\left[\frac{\sigma_1^2}{\sigma_2^2} + \frac{(\mu_2-\mu_1)^2}{\sigma_2^2} - 1\right]\right]. \end{aligned} \tag{9.4}$$

A partition-around-medoids (PAM) clustering algorithm is used [70]. Similar to the $k$-means method, it allows operating over a dissimilarity matrix between samples, which we obtain for the set of blocks from (9.4). At this stage, we discard blocks that are likely to be unreliable representatives of a motion texture, including those which have mixtures of classes. To this end, we compute the diameter $D$ of each cluster as the maximum distance between any two of its elements, and we keep only as valid blocks those that have a distance to the medoid lower than a given fraction of $D$, typically 0.3. Note that during this process the number of classes could be estimated on-line as well.

Once we have a first segmentation of the field by this clustering step, we obtain a set of accurate motion-texture model parameters for each class by estimating the 11 parameters for each final cluster using the valid blocks.

**9.2. Handling the partition function.** Equation (9.3) involves the calculation of the partition function for each class. It is a fundamental matter not to neglect, as it normalizes the global distribution, allowing one to correctly make a MAP decision between different classes. It should be noted that avoiding the partition function assumes conditional independence. Moreover, it depends on the label realization $\mathbf{l}$; that is, its value changes according to how the labels are assigned.

We propose the following approach for appropriately handling the partition functions. For a general Gibbs distribution, the expression of the normalizing factor is $Z = \int_\Omega e^{-Q(\mathbf{X})} d\mathbf{X}$, where $\Omega$ is the sample space. Let $\Delta Q(\mathbf{X}_A)$ be a variation on the energy function, not necessarily small, due to an arbitrary variation over the values of the subfield $A \subset S$ and being a function only of $\mathbf{X}_A$. Then we can write the resulting partition function, $Z'$, as a function of the former one, $Z$,

$$\begin{aligned} Z' &= \int_\Omega e^{-Q'(\mathbf{X})} d\mathbf{X} = \int_\Omega e^{-Q(\mathbf{X}) - \Delta Q(\mathbf{X}_A)} d\mathbf{X} \\ &= \int_\Omega Z p(\mathbf{X}) e^{-\Delta Q(\mathbf{X}_A)} d\mathbf{X} = Z E_{\mathbf{X}_A}\left[e^{-\Delta Q(\mathbf{X}_A)}\right], \end{aligned} \tag{9.5}$$

where the expectation operator is applied w.r.t. the marginal probability distribution of $\mathbf{X}_A$. A similar calculation was previously proposed in [73] for estimating the unknown partition function from reference values using Monte Carlo integration.

Here, we use this result for the problem of segmentation. Available optimization methods for labeling problems are mostly based on iteratively computing energy changes as a result of adding or taking out points from a class. Defining $\Delta Q$ appropriately, we then have an expression for the change on the normalizing factor. Removing a point $x_i$ from a class is equivalent to discarding the cliques corresponding to that point. This change in $Z$ can be expressed by setting

$$\Delta Q(\mathbf{X}_A) = \Delta Q(x_i, \mathbf{X}_{\mathcal{N}_i}) = -\left(V_i(x_i) + \sum_{j \in \mathcal{N}_i} V_{i,j}(x_i, x_j)\right) - \log \mathbf{1}_0(x_i), \tag{9.6}$$

where $\log \mathbf{1}_0(x_i)$ is defined for convenience from $e^{\log \mathbf{1}_0(x_i)} = \mathbf{1}_0(x_i)$. This term allows integrating w.r.t. $x_i$ without changing the value of the integral. For the case of the Gaussian MS-MRF, $V_{i,j}(0, x_j) = 0$ and $\mathbf{S}(0) = 0$, which implies that $e^{-\Delta Q(\mathbf{X}_A)} = \mathbf{1}_0(x_i)$. We thus write

$$Z' = Z E_{\mathbf{X}_A}[\mathbf{1}_0(x_i)] = Z P(x_i = 0) \;\Rightarrow\; \log \frac{Z'}{Z} = \log P(x_i = 0). \tag{9.7}$$

Equivalently, we can calculate the change on the value of the partition function due to an extraction of an arbitrary subset $T$ of points from the field, redefining (9.6) adequately. Following the same reasoning, we arrive at $\log \frac{Z'}{Z} = \log P(\mathbf{X}_T = \mathbf{0})$. It is thus much easier to compute a relative change of the partition w.r.t. a current configuration than the very intricate (and still open) issue of evaluating the complete value of $Z$. This is exploited in what follows to define a simple energy minimization strategy.

**9.3. Energy minimization method.** Different methods for minimizing the energy (9.3) are available. Simulated annealing [34] is suitable for virtually any shape of energy function, but it is slow and not that efficient. For energy functions related to conditional models, i.e., those arising from MRFs, iterative conditional modes (ICM) [4] is based on maximizing the conditional local density for a point w.r.t. its label. While faster than simulated annealing, it relies on the strong assumption that, given the label field, the observations are conditionally independent. This is not the case of the segmentation method proposed here (see (9.1)), where we exploit the whole motion texture model for each class.

In recent years, a new family of energy optimization methods for some classes of energy functions [43] based on *graph-cuts* has been proposed and developed with a growing impact in many applications [6, 43, 68]. In this framework, one seeks the labeling $\mathbf{l}$ that minimizes energy functions of the type

$$E(\mathbf{l}) = \sum_{i \in S} D_i(l_i) + \sum_{<i,j>} E^{ij}(l_i, l_j), \tag{9.8}$$

where $D_i(l_i)$ is the data term and $E^{ij}(l_i, l_j)$ is usually called the smoothness or interaction term. The method relies on constructing a directed graph where the vertices are the image points plus two additional vertices corresponding to each binary label values. It is shown that there is a one-to-one correspondence between a partition of this graph and a complete binary labeling of the image. Then, assigning appropriate edge weights, obtaining a minimum cut is equivalent to optimizing $E(\mathbf{l})$. Thus, the formulation reduces to computing a min-cut/max-flow problem.

Among the methods that exploit this equivalence, a very efficient method is proposed in [6, 43], providing, at the same time, a general graph construction scheme. The method, called the *expansion move* algorithm, is valid for multilabel situations and is based on iteratively computing an expansion of the region occupied by a label class $l \in \{0, \ldots, c-1\}$ in the current labeling in such a way that this change generates a decrease in the global cost function. The expansion algorithm cycles through all the labels and finds the optimal expansion by application of the graph-cut method.

We propose using this technique for efficiently addressing the problem of motion texture segmentation. In our case we have to rewrite (9.3) in the form of (9.8). Let us consider the case of two classes ($c = 2$). Then we have

$$E(\mathbf{l}) = Q_1(\mathbf{x}_1) + Q_2(\mathbf{x}_2) + \log\left(Z_1(\mathbf{l})\right) + \log\left(Z_2(\mathbf{l})\right) - Q_S(\mathbf{l}), \tag{9.9}$$

where for each class $k \in \{0, 1\}$, $V_i^{(k)}$ and $V_{ij}^{(k)}$ are the corresponding potentials for each motion texture model. For the partition function, observe (9.7). Note that the probability $P(x_i = 0)$ is computed w.r.t. the marginal distribution for the site before taking out the point. At the same time, we can assume that this marginal density remains approximately constant as we successively take points from a class, as long as the parameters that describe the remaining field do not vary too much. Of course, the extreme case of leaving only one site in the class will violate this assumption. Finally, note that this approximation allows us to write $P(\mathbf{X}_T = \mathbf{0}) \approx [P(x_i = 0)]^{N_T}$, where $N_T$ is the number of extracted points. As $P(x_i = 0)$ is

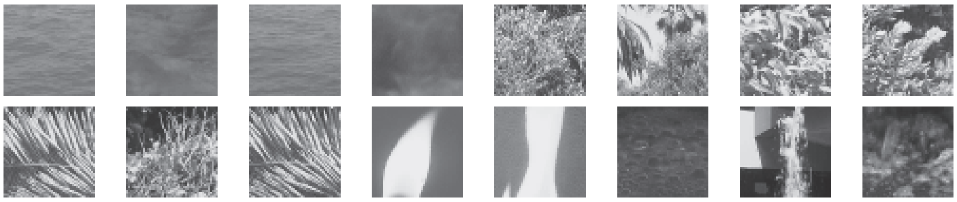


**Figure 3.** *Samples of video frames from the UCLA-*50 *dynamic texture database* [60].

considered constant, we can assume that each point contributes equally to the energy change in each iterative step of the expansion move algorithm, and we then define

$$D_i(l_i) = V_i^{(l_i)}(x_i) - \log P(x_i = 0 \mid l_i), \tag{9.10}$$

$$E^{ij}(l_i, l_j) = \psi(l_i, l_j)\left[V_{ij}^{(l_i)}(x_i, x_j) - \gamma\right]. \tag{9.11}$$

Note that the smoothness term may violate the regularity of the energy function in order to be *graph-representable* [43]. However, we have experimentally observed that, in our case, this occurs in less than 5% of the points, and the problem can be overcome with no impact on the final result, by simple truncation of these terms. More recent algorithms for minimizing nonsubmodular energy functions [42, 45] can be applied as well, but the benefit is marginal for our case.

## 10. Experimental results.

### 10.1. A brief review of dynamic texture datasets.

As it occurs in general with video content analysis applications, constructing a dynamic texture database is not an easy task. The semantics of a scene can hardly be completely reduced to a mathematical model, although one usually resorts to the latter in order to obtain a compact and treatable representation. With the first introduction of the concept of a dynamic texture in [28], this dynamic video content was formally defined as a sequence of color/intensity images that respond to an LDS. The dataset UCLA-50 presented in [60] contains spatially small and temporally long image volumes of size 48 × 48 × 75 carefully cropped from the original sequences to ensure key statistical and dynamical features (examples in Figure 3).

UCLA-50 is formed by 50 dynamic texture classes with four sequences each, giving a total of 200 original sequences. Many of these classes are clearly semantically equivalent and thus are derived in multiple subversions of this mother database into smaller and more meaningful datasets. Grouping them into nine classes, one obtains UCLA-9 [41, 56]. In [55] one of the classes is dropped, alleging that it has many samples w.r.t. the other classes. This leads to UCLA-8.

One could argue that these dynamic texture datasets well fit the family of LDS-based methods but could be less appropriate to assess the performance of an arbitrary recognition method. Indeed, attention must be paid to the type of data samples, the organization of the database, and the ground truth information. Specifically, in the context of our spatial two-frame motion texture framework, 48 × 48 sized frames are indeed small.

A more realistic dataset is the DynTex dynamic texture database [52]. It is larger (650 original sequences), the images are of higher quality and colored, and fundamentally, the image

**Table 1**

*Dynamic texture datasets used in the state-of-the-art methods for recognition and classification. UCLA-N: Original UCLA-*50 [60] *reorganized to N classes. UCLA-pan: UCLA-*50 *with artificially introduced camera panning. UCLA-SIR: Shift-invariant recognition* [27]*. DynTex-N: N class DynTex. DynTex++: Annotated* 36 *classes. MIT-*10*: Temporal texture database with* 10 *classes. YUVL: York University Vision Lab spacetime texture database* [27]*. Traffic:* 3 *class traffic sequences (see* [10]*).*

| Dataset<br>Method | UCLA-50 | UCLA-9 | UCLA-8 | UCLA-7 | UCLA-4 | UCLA-pan | UCLA-SIR | DynTex-26 | DynTex-3 | DynTex++ | MIT-10 | YUVL | Traffic |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [9] |  |  | √ |  | √ |  |  |  |  |  |  |  |  |
| [11] | √ |  |  |  |  | √ |  |  |  |  |  |  |  |
| [27] | √ |  |  | √ |  |  | √ |  |  |  |  | √ |  |
| [63] |  |  |  |  |  |  |  |  |  |  | √ |  |  |
| [33] |  |  |  |  |  |  |  | √ |  |  | √ |  |  |
| [41] | √ | √ | √ | √ |  |  | √ |  |  |  |  |  |  |
| [57] |  |  | √ |  |  |  |  |  | √ |  |  |  | √ |
| [10] | √ |  |  |  |  |  |  |  |  |  |  |  | √ |
| [56] |  | √ | √ |  |  |  |  |  |  |  |  |  |  |
| [55] | √ | √ | √ |  | √ |  |  |  |  |  |  |  |  |
| [60] | √ |  |  |  |  |  |  |  |  |  |  |  |  |
| [35] |  |  |  |  |  |  |  |  |  | √ |  |  |  |

frame covers the whole spatiotemporal region of interest (recall that UCLA-50 is restricted to $48 \times 48 \times 75$ patches). The spread of DynTex still remains limited today; nonetheless, in our opinion, it appears to be a richer compendium of dynamic texture sequences.

Having this said, it is interesting to take a precise look at the different choices that the state-of-the-art methods have taken while choosing an appropriate dataset for benchmarking their dynamic texture recognition (DTR) performance. In Table 1 we display a certainly incomplete but meaningful list of state-of-the-art references on DTR and the databases they have considered. Clearly, none of these dynamic texture datasets can be considered as a single well-established reference for testing methods.

There are surely two main dynamic texture databases that are considered in some form by almost every recognition method: UCLA-$N$ ($N$ stands for the number of classes) and DynTex. Only one method, that of [57], among those presented in Table 1, shows results on both but reorganizing DynTex to only three classes (DynTex-3). We propose testing our motion texture classification method on a 10-class DynTex subset and on several subsets of UCLA-50. This is explained in detail in what follows.

### 10.2. Motion texture classification.

*DynTex*-10. We first took motion textures extracted from the DynTex dynamic texture database [52] where the homogeneity assumption was mostly valid. Although a large part of the images is occupied by the dynamic texture of interest, some of the samples display regions where the homogeneity is broken (Figure 4). We divided them into 10 different classes: *Steam, Straw, Traffic, Water, Candles, Shower, Flags, Water-Rocks, Waves, and Fountain.* A total of 30 different sequences were considered, and for each one, five pairs of consecutive images

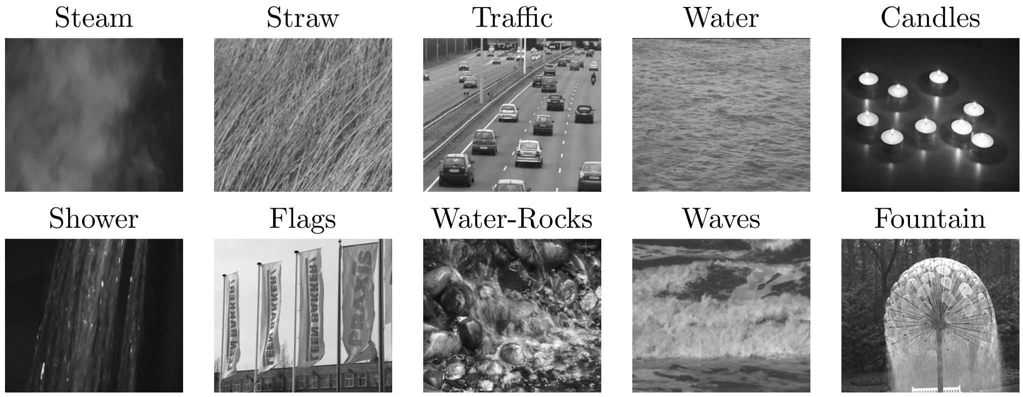


**Figure 4.** *Sample images from the DynText-10 motion texture classes used for the recognition experiments. For some of the samples the dynamic texture of interest does not occupy the whole image, violating the assumption of spatial homogeneity. However, our model is sufficiently robust to deliver a correct classification.*

were selected at frames 1, 20, 40, 60, 80 for a total of 150 samples. Each motion texture class parameter set was learned from a single pair of images picked from only one of the sequences belonging to each type of motion textures. All sequences were composed of gray scale images with a resolution of 720 × 576 pixels, given at a rate of 25 frames per second. The original images were filtered and subsampled to a resolution of 180 × 144 pixels. On one side, this reduces the processing time. But it also makes the motion measurements more reliable, as the normal flow computation is valid under the hypothesis of small displacements.

We estimate the reference model parameters for each class with only one training sample. We then estimate $\boldsymbol{\phi}$ for each test sample and compute the KL distance (8.3) with each learned class parameter vector.

Table 2 contains the confusion matrix for the 10 motion texture classes. A correct recognition is considered when both the test sample and the closest reference parameter vector belong to the same class. Of course, training samples are not considered for testing. An overall classification rate of 90.7% was achieved. As for the confusion matrix, let us note that it is likely that waves are classified as Water or Water-Rocks as they correspond to similar dynamic contents. Straw may be confused with Shower (they have similar vertical orientation), and Candles can be classified as Traffic, as both classes show a motion pattern consisting of isolated blobs. The nonsymmetry of the confusion matrix is due to the nature of the tested data set, where for some classes the tested sequences have a closer resemblance to the training sample, while for others there are notorious intraclass variations that may lead to a misclassification.

Reported experiments for dynamic texture recognition on DynTex data are found in [57] for a three-class joint segmentation/classification task (Waves, Flags, Fountain) for which they achieve 72.5% of correctness. The result of [35] on the 36-class DynTex++ version is 63.7%. This shows how challenging this database is and that our method provides top-performing results, though a direct quantitative comparison is not currently possible.

*UCLA database.* Different reorganizations of the original UCLA-50 dynamic texture dataset permit us to further study the recognition performance of our approach. It should be

**Table 2**

*Motion texture class confusion matrix obtained by our mixed-state motion texture model on the DynTex-*10 *dataset. Each row indicates how the samples for a class were classified. Overall rate:* 90.7%*.*

| | Steam | Straw | Traffic | Water | Candles | Shower | Flags | Water-rocks | Waves | Fountain |
|---|---|---|---|---|---|---|---|---|---|---|
| Steam | 1.0 | - | - | - | - | - | - | - | - | - |
| Straw | - | .93 | - | - | - | .07 | - | - | - | - |
| Traffic | - | - | .87 | - | - | - | - | .13 | - | - |
| Water | - | - | - | 1.0 | - | - | - | - | - | - |
| Candles | - | - | .13 | - | .74 | - | - | .13 | - | - |
| Shower | .20 | - | - | - | - | .80 | - | - | - | - |
| Flags | - | - | - | - | - | - | 1.0 | - | - | - |
| Water-rocks | - | - | - | - | - | - | - | .93 | - | .07 |
| Waves | - | - | - | .07 | - | - | - | .13 | .80 | - |
| Fountain | - | - | - | - | - | - | - | - | - | 1.0 |

noted that a unique manual classification of the sequences into distinctive classes is somehow subjective and might even be senseless under the hypothesis of a given method. Table 1 is a clear illustration of the difficulty of establishing an objective and unique benchmark.

For the set of experiments that follow we have generated several subsets and recategorizations of UCLA-50, with variable numbers of classes and different organization criteria. On the other hand, the original 48 × 48 sized sequences would not be sufficiently large for our spatial model. To cope with this situation, we first generate new 96 × 96 pixels sequences by spatially concatenating four original shorter 48 × 48 subsequences of the same video class. This has the goal of augmenting the number of points while estimating the parameters, assuming the effect of borders is negligible. Temporal length is not an issue as we need only two frames to learn and recognize. Moreover, several UCLA-50 sequences are shot with three zooming levels (far, mid, near). Handling large motion scale invariability is outside the scope of this paper, and thus, while reorganizing the database into classes, we did not include the three versions but discarded the least numerous between far and near.

We start with a two-class classification problem between classes Fountain (12 samples) and Water-Falls (16 samples) as proposed in [55], which by the way are close dynamic textures with similar appearance and dynamics. For testing, we took three pairs of frames at instants 10, 30, and 60, while learning was done at instant 0. Our result is 96.4% using only a pair of frames for learning from only one of the sequences of each class (chosen randomly), while [55] reported 98% on average. Next, we learn several models for each class while still estimating each model from one image pair of the training sequence. We take 50% of the sequences of each class as training sequences. We test the rest, and we achieve 97.6% (only one misclassified sample). Although the effectiveness of the best LDS approaches is similar to ours, the method proposed here has a big advantage, which is that we need only two consecutive frames to estimate and recognize the mixed-state models.

Next, we took the organization of UCLA-4 into Fountain (12), Sea (12), Water (12), and Water-Falls (16). We obtain 87.2% on this set learning on 50% of the sequences (Table 3a).

**Table 3**

*Motion texture class confusion matrices obtained by our mixed-state motion texture model on different reorganizations of UCLA-*50.

| | Fountain | Sea | Water | Water falls |
|---|---|---|---|---|
| Fountain | .89 | - | .03 | .08 |
| Sea | - | .92 | .08 | - |
| Water | - | - | .86 | .14 |
| Water-Fall | - | - | .15 | .85 |

(a) Confusion matrix for our method on the UCLA-4 dataset used in [55]. Overall rate: 87.2%.

| | Turb. water | Fire flames | Swaying veg. | Smoke | Calm water |
|---|---|---|---|---|---|
| Turb. water | .93 | - | .05 | .02 | - |
| Fire flames | .37 | .50 | .12 | - | - |
| Swaying veg. | .06 | - | .94 | - | - |
| Smoke | .75 | - | - | .25 | - |
| Calm water | - | - | - | .05 | .95 |

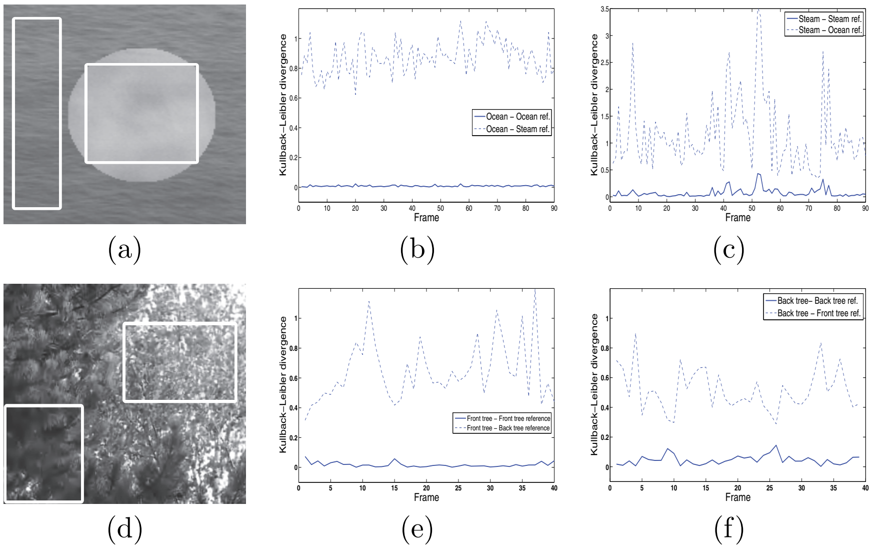
(b) Confusion matrix for our method on the UCLA-Motion dataset. Overall rate: 89%.

| | Flames | Fountain | Smoke | Turb. Water | Water waves | Water falls | Swaying veg. |
|---|---|---|---|---|---|---|---|
| Flames | .70 | - | .06 | .06 | - | .06 | .12 |
| Fountain | - | .88 | - | .12 | - | - | - |
| Smoke | | .12 | .38 | .12 | | .38 | |
| Turb. Water | - | - | - | .65 | - | .35 | - |
| Water waves | - | - | - | .04 | .96 | - | - |
| Water falls | - | - | - | - | - | .89 | .11 |
| Swaying veg. | - | .005 | .005 | - | .005 | .07 | .91 |

(c) Confusion matrix for our method on the UCLA-7 dataset introduced in [27]. Overall rate: 84%.

In [55] the best classification result is 89%, while they report baseline results with the model used in [60] of 52.2%. The best published result on this set, to the best of our knowledge, is obtained by [9] with an overall 95%.

A different categorization of dynamic textures can be formulated in terms of motion properties, and indeed this is particularly interesting for our motion texture model. We have extracted five classes from UCLA-50 in terms of its dynamics which we call UCLA-Motion: *Turbulent water* (44)*, Fire flames* (8)*, Swaying vegetation* (120)*, Smoke* (4)*, Calm water* (24)*.* Our method achieves 89% of classification rate (Table 3b). In [27] a similar dataset is used but with 7 classes (*Flames, Fountain, Smoke, Water turbulence, Water waves, Waterfall, Windblown vegetation*). They achieve 92.3% while for the two-frame MS-MRF motion texture model we have 84% (Table 3c). On the other side, their model requires processing a large spatiotemporal volume from which features are extracted. Note that the results for the Smoke class are not satisfactory. The same issue was previously reported in [55], putting in evidence the complexity of this particular class.



(a) (b) (c)

(d) (e) (f)

**Figure 5.** *Temporal stability of the estimated motion texture model.* (a) *Ocean-Steam sequence. The white rectangles indicate manually delineated regions in which the parameters for each class were estimated.* (b) *KL distance between the parameters for the Ocean class at each time instant and the reference model obtained from the first pair of frames for Ocean (solid line) and Steam (dashed line).* (c) *KL distance between Steam and the reference model for Steam (solid line) and Ocean (dashed line).* (d) *Trees sequence, where we have a Front tree and a Back tree.* (e) *KL distance Front tree-Front tree reference (solid line) and Front tree-Back tree reference (dashed line).* (f) *KL distance Back tree-Back tree reference (solid line) and Back tree-Front tree reference (dashed line).*

**10.3. Two-frame model estimation and temporal stability.** Our method is based on the modeling of the spatial distribution of mixed-state motion values (normal flows) for a given motion texture, and only two frames are required to estimate the model parameters. The previous experiments on motion texture classification have shown that training a motion texture class from a single pair of images is sufficient to achieve a high recognition rate. On the other side, for the segmentation method one could use the estimated parameters for each region at a given instant of time to also recognize the intervening motion texture classes. Yet, a question arises about the stability of the estimated model parameters over time. In other words, would the model estimated from a given pair of frames out of the video sequence containing the dynamic texture (which can be of a much longer temporal extent) be different if it were estimated from another image pair in the sequence? Also, in what follows, we show that for temporally stationary motion textures, the model estimated from only two consecutive frames is well representative of the rest of the sequence; that is, the mixed-state model parameters are stable (consistent) over time.

We take two sequences also used later for the segmentation experiments, each one composed of two different motion textures (Figure 5): Trees (two different kinds of trees moved by the wind, and at different depth with respect to the camera) and Ocean-Steam[1] (a circular region comprising steam superimposed to a sequence of ocean waves). Then, we estimate the

[1] Copyright 2003, UCLA Vision Lab. Thanks to Daniel Cremers and Stefano Soatto for providing this sequence.

set of parameters over manually delineated regions (Figures 5(a) and 5(d)) corresponding to each of the motion texture classes. From the first pair of frames we obtain a set of so-called reference parameters for each class. Finally, we compute the KL divergence between the reference models obtained at $t = 1$ and the models corresponding to the parameters estimated within each region along the rest of the sequence (for each $t > 1$).

In Figure 5(b) we plot the KL distance between the parameters estimated for the Ocean motion texture at each instant, and the reference models for Ocean (solid line) and Steam (dashed line). We see that the parameters successively estimated over the sequence for Ocean remain very close to the reference Ocean model extracted from the first pair of frames. At the same time, the distance between Ocean and Steam is far higher along the whole sequence. Figure 5c displays the reciprocal experiment. The parameters for Steam were estimated for each instant, and we plot the KL distance with respect to the reference models. Again, the parameters for the Steam class at each time instant $t > 1$ keep close to the reference Steam model at $t = 1$ (solid line) while being quite different from the Ocean reference model (dashed line).

Finally, the same behavior is confirmed for the Trees sequence, where we have two motion textures: a Front tree and a Back tree. In Figure 5(e) we display the KL distance between the Front tree model computed at each $t > 1$ within the manually delineated region and the two respective reference models, and in Figure 5(f) we display the distance between the successively estimated Back tree models and the two respective reference models.

These experiments demonstrate that the motion texture model parameters estimated at each time instant are stable for a temporally stationary motion texture. Thus, we can state that the model parameters estimated, for instance, from the first image pair of the dynamic texture clip are well representative of the whole clip. Conversely, our method can easily handle nonstationary temporal textures (e.g., varying temporal textures). The estimated model parameters will evolve accordingly. In the latter case, several sets of parameters would be necessary to correctly represent the dynamic texture. To summarize, we need only two frames to estimate the MS-MRF model of a temporally stationary dynamic texture, whatever its length is; we can easily deal with nonstationary temporal textures since we only need two frames to estimate the motion texture model.

**10.4. Dynamic texture change detection.** A fundamental consequence arises from our two-frame modeling framework, that is, from its ability to characterize a motion texture for a precise time instant based on an image pair, and its property of temporal stability of the estimated parameters, as discussed before. Such a model together with our similarity measure (i.e., the KL distance) can be readily applied for motion texture change detection. The following experiments are thus performed and depicted in Figure 6. Each pair of motion texture classes for which a misclassification was found in Table 2 was used to construct a new 40-frame sequence by taking a 20-frame segment of each of the two classes and concatenating them. Thus, we have built a set of seven challenging test videos. A motion texture model is estimated from the first pair of images, and the KL distance is computed w.r.t. the motion texture model estimated at every subsequent time instant. For all but one (Shower-Steam), the significant increase in the KL distance at the instant of change of motion texture (frame 20) is indeed a key feature for this detection task.

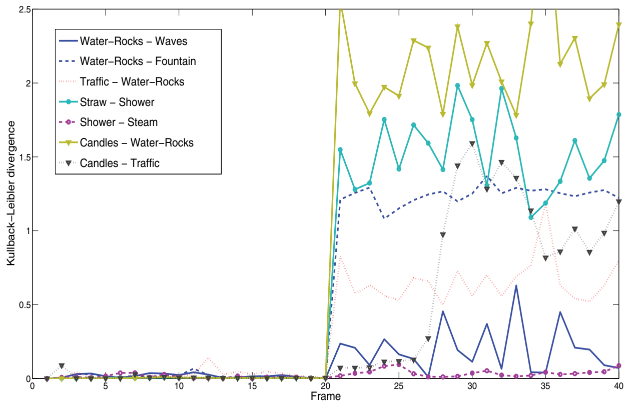
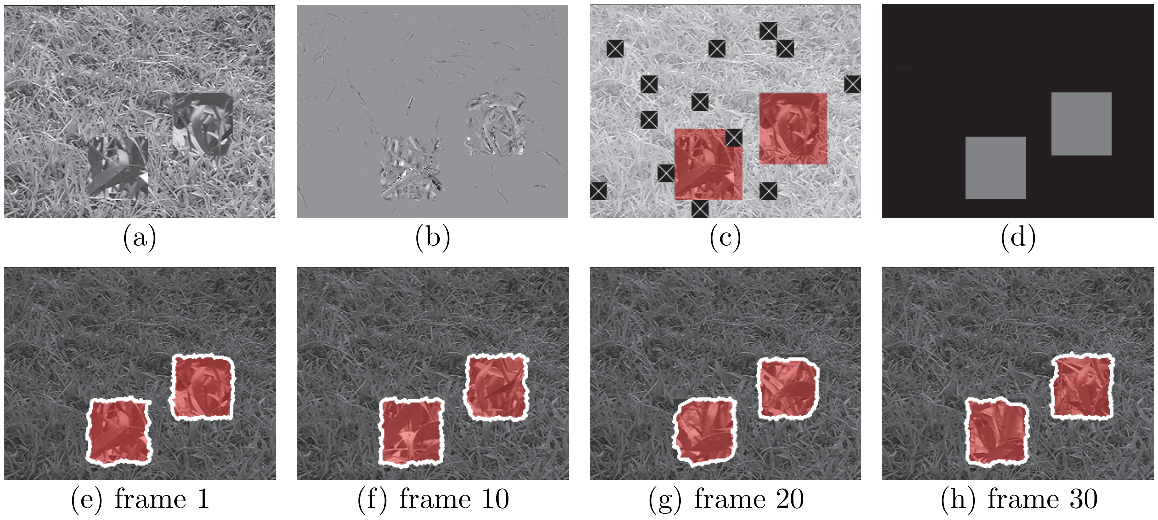


**Figure 6.** *Temporal motion texture change detection. For each pair of motion texture classes we construct a new* 40*-frame sequence by concatenation of two* 20*-frame segments of said classes, for a total of seven test videos. Curves show the evolution of the KL distance along the new sequence w.r.t. a mixed-state motion texture model learned from the first pair of frames. The sudden increase in the KL distance exactly at the instant of change of content shows the ability of the model to detect the change of motion texture, without the need of recognizing the intervening classes.*

**10.5. Segmentation of dynamic textures.** In this section we present results on segmentation of motion texture videos. Figure 7 displays an artificially composed sequence consisting of a motion texture of grass and two small regions of moving leaves (Grass-Leaves sequence). The location and the shape of these two regions are fixed over time. Each image is of size $320 \times 240$ pixels. The motion texture is shown in Figure 7(b). In Figure 7(c) we give the result of the initialization stage of the algorithm where blocks of size $20 \times 20$ were used. From this first segmentation, we estimated the motion texture class parameters. The corresponding values are given in Figure 7(i). Some of the blocks were discarded during the clustering as explained in subsection 9.1, and their points were assigned a random initial label. Then, a fine segmentation is obtained by minimizing the MAP energy (9.9). For this and the following segmentation experiments, a value of $\gamma \in [0.4, 0.8]$ was used for the smoothness term and set manually. For processing subsequent frames, the previous segmentation is used as an initialization as well as for re-estimating the parameters. Thus, the block-based initialization is applied once at the very beginning of the sequence. The segmentation result is consistent along the whole sequence and is close to the ground truth.

The second experiment (Figure 8) corresponds to the Ocean-Steam sequence. The images are of size $150 \times 150$ pixels. For the initialization of the algorithm, blocks of $30 \times 30$ pixels were used. The algorithm is well initialized by the block clustering strategy, as depicted in Figure 8(c), where a block located at the border between both motion textures was discarded. As for the estimated parameter values, note the value of $h_H$ for the Ocean texture class. Its high value indicates the horizontal orientation of the motion map, which is much stronger than for the Steam class. Also, Ocean is a more uniform motion texture, which is reflected in the lower variance $\sigma^2 = 1/2b$. These kinds of properties are well captured by the model and permit to differentiate the motion texture classes.



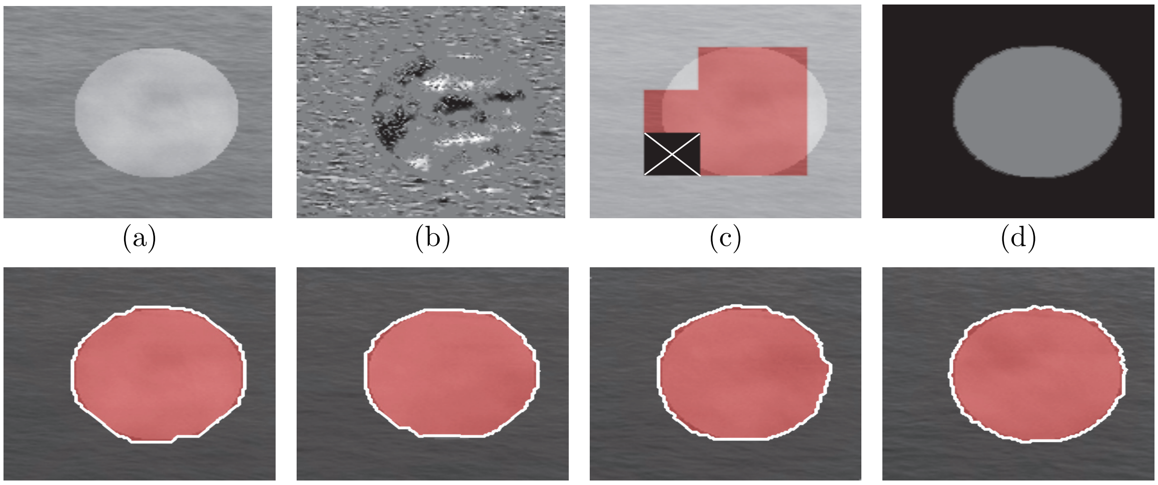
(i)

| *Class* | $a$ | $b$ | $c$ | $d_H$ | $h_H$ | $d_V$ | $h_V$ | $d_D$ | $h_D$ | $d_{AD}$ | $h_{AD}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Leaves | -5.06 | 1.98 | -0.004 | 0.27 | 0.79 | 2.20 | 0.59 | 1.03 | 0.16 | 1.12 | 0.41 |
| Grass | 1.34 | 60.15 | -0.03 | 1.92 | 32.6 | 1.39 | 11.2 | 0.17 | 0.00 | -0.04 | 1.12 |

**Figure 7.** *Grass-Leaves sequence* ($320 \times 240$ *pixels*). (a) *First frame from the original sequence.* (b) *Motion texture obtained by computing the proposed scalar normal flow between frames* 1 *and* 2 *(we mapped the motion measurements to the range of gray* $[0, 255]$ *where* 128 *corresponds to null motion).* (c) *Initial segmentation by clustering blocks of size* $20 \times 20$ *using a simplified KL distance. Some blocks were automatically discarded (white cross) from the initial parameter estimation.* (d) *Ground-truth.* (e)–(h) *Result of the proposed segmentation method for different time instants. The delineated squares correspond to the Leaves class.* (i) *Motion-texture model parameters estimated for the first pair of frames.*

We have compared our segmentation result with two other methods: the method by Doretto et al. [29] based on linear dynamic models of intensity and a level-set approach for estimating the boundaries (Figure 9(a)) and the method by Chan and Vasconcelos [12] exploiting the mixture-of-dynamic-textures model (Figure 9(b)). Our result is more accurate in locating the Steam boundary than [29] and performs similarly to [12]. Moreover, for these two methods the final segmentation is achieved by processing about 100 frames of the sequence, while in our case we need only two frames to get the segmentation at each time instant.

In Figure 10 we show the results for a real sequence of two motion textures—the Trees sequence ($320 \times 240$ pixels). During the initialization stage, blocks of size $40 \times 40$ were used. Recall that in this sequence we have two trees moved by the wind, but of different kinds and at different depths with respect to the camera. This is a complex scene since the trees have not only similar intensity textures but close motion textures. Both motion textures exhibit no predominant orientation, which is confirmed by observing that the horizontal and vertical model coefficients have similar values. One of the main differences between the front and the back tree is that the front tree involves large compact regions of null motion mixed with areas of motion. Our explicit modeling of discrete and continuous values is then important to distinguish the two dynamic textures. Note that the bottom-right area of the image (marked with a white square in Figure 10(e)) appears to be included in the wrong texture class. This



(a) (b) (c) (d)

(e) frame 1 (f) frame 10 (g) frame 20 (h) frame 30

(i)

| *Class* | $a$ | $b$ | $c$ | $d_H$ | $h_H$ | $d_V$ | $h_V$ | $d_D$ | $h_D$ | $d_{AD}$ | $h_{AD}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Ocean | -0.10 | 10.53 | 0.02 | 1.03 | 10.43 | 0.43 | 0.00 | 0.11 | 0.00 | 0.06 | 0.00 |
| Steam | -0.15 | 2.38 | 0.03 | 0.21 | 1.07 | 0.13 | 0.46 | 0.23 | 0.35 | 0.11 | 0.47 |

**Figure 8.** *Ocean-steam sequence (*$150 \times 150$ *pixels).* (a) *First frame from the original sequence.* (b) *Motion texture obtained by computing the proposed scalar normal flow between frames* 1 *and* 2. (c) *Initial segmentation by clustering blocks of size* $30 \times 30$ *using a simplified KL distance. One block was automatically discarded (white cross) from the initial parameter estimation.* (d) *Ground-truth.* (e)–(h) *Result of the proposed segmentation method for different time instants. The red circular region corresponds to the Steam class.* (i) *Motion-texture model parameters estimated for the first pair of frames.*

is due to the fact that this area has a different motion pattern than the rest of the back tree, with a lower density of moving points. Then, it is confused with the front tree. Overall, the segmentation is well obtained and the border between both classes is correct, although even for the human eye it is difficult to separate both regions. Also it proves robustness to a wrong classification of initial blocks (Figure 10(c)).

**10.6. MS-MRF versus classical MRF model.** One may ask whether the synthetic sequences (such as the Ocean-Steam sequence) are indeed so complex to justify the need of our motion modeling approach. To this end, we consider a standard Gaussian MRF motion model instead of the MS-MRF motion model by eliminating the discrete component at the null value. Then, we estimated the remaining model parameters. We apply the same segmentation strategy. The configuration of the experiment is that of Figure 8. The result is shown in Figure 11, where we notably observe a dramatic drop in the segmentation quality. The spatial structure of the motion texture (Figure 7(b)) is indeed fairly complex despite the illusory simplicity of the original image. Observe how the simple MRF model seems to delineate the small homogeneous motion blobs distinguishable in the motion texture sequence (Figure 8(b)), while it is largely perturbed by the ubiquitous null value present in the motion observations.

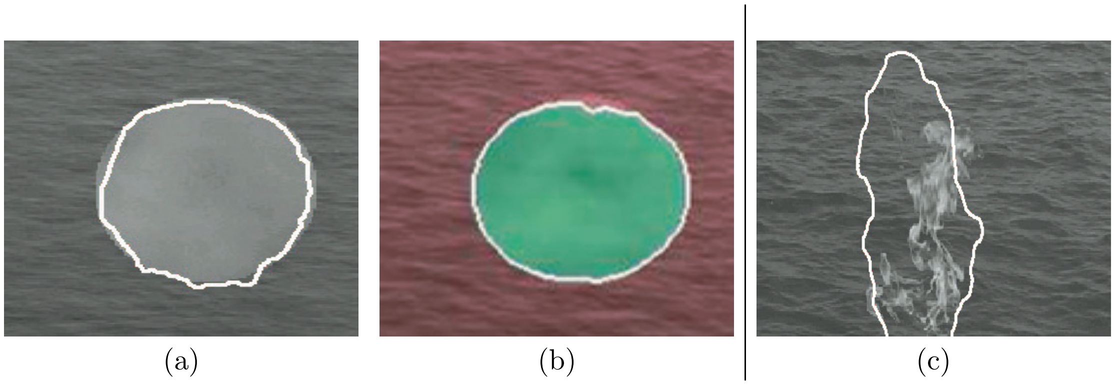


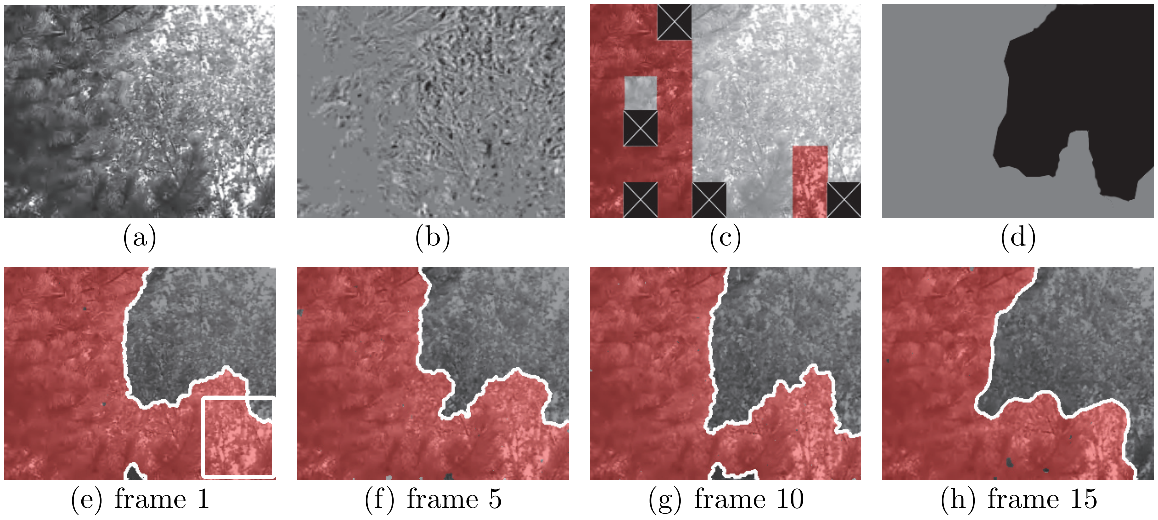
**Figure 9.** *Results of other dynamic texture segmentation methods.* (a) *Result for the Ocean-Steam sequence by Doretto et al. presented in* [29]*.* (b) *Result for the Ocean-Steam sequence by Chan and Vasconcelos presented in* [12]*. For both methods, the segmentation is obtained by processing the whole sequence of more than* 100 *frames, and the estimated boundary contour is the same for each frame.* (c) *Result for the Ocean-Fire sequence by Doretto et al. presented in* [29]*. The same remark applies here.*

**10.7. Segmentation of dynamic textures with changing boundaries.** Motion (or dynamic) textures are related to phenomena that have very different dynamics compared to rigid bodies. Thus, considering that the region occupied by a dynamic texture is rigid and constant over time may not be a valid hypothesis for segmenting real sequences. In our approach, this is not the case, as we segment instantaneous motion maps instead of a whole temporal clip. We then show how the proposed algorithm can be applied to the segmentation of motion textures with deformable spatial supports over time.

In Figure 12, we first present a sequence consisting of a fountain that suddenly appears, grows, and progressively diminishes over a static background. Note that, in our model, the static background can be considered as a motion texture class for which the probability of null motion values is very high. Thus, it can be segmented in the same manner as any motion texture. The images are of size 320 × 240 pixels, and 40 × 40 blocks were used during the initialization stage. For the first frame, the algorithm produces a few errors in the segmented regions of water. The method needs a sufficient spatial extent of the motion texture, which is not the case at the very beginning of the sequence, in order to identify its parameters and obtain a correct segmentation. As the fountain appears in the sequence, the segmentation result improves. The regions corresponding to the Fountain class may seem to be wrongly merged together in large blobs instead of detecting the vertical water jets. However, within the proposed model, the jets and the spaces (null motion regions) between them can be described by a single set of parameters. The motion texture of Fountain is consequently characterized by a pattern of vertical spikes and is correctly segmented by our method. We moreover capture the temporal variations of the boundaries.

Finally, we display the results for a very challenging sequence with deformable motion textures in Figure 13. In this case, a dynamic texture of a fire flame was combined with a dynamic texture of water to form the *Ocean-Fire* sequence[2]. The boundaries of the flame are

[2] Copyright 2003, UCLA Vision Lab.



| | *Class* | $a$ | $b$ | $c$ | $d_H$ | $h_H$ | $d_V$ | $h_V$ | $d_D$ | $h_D$ | $d_{AD}$ | $h_{AD}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (i) | Front Tree | -2.29 | 15.9 | -0.06 | 1.09 | 7.92 | 1.12 | 7.89 | 0.28 | 0.00 | 0.21 | 0.00 |
| | Back Tree | -1.23 | 5.24 | 0.01 | 0.64 | 2.42 | 0.69 | 2.76 | 0.22 | 0.00 | 0.18 | 0.00 |

**Figure 10.** *Trees sequence (*320 × 240 *pixels).* (a) *First frame from the original sequence.* (b) *Motion texture obtained by computation of the scalar normal flow between frames* 1 *and* 2. (c) *Initial segmentation by clustering blocks of size* $40 \times 40$ *using a simplified KL distance. Some blocks were automatically discarded (white cross) from the initial parameter estimation.* (d) *Ground-truth.* (e)–(h) *Result of the proposed segmentation method for different time instants. The biggest region corresponds to the Front tree. The white square in* (e) *indicates a region where the algorithm confuses the classes.* (i) *Motion-texture model parameters estimated for the first pair of frames.*

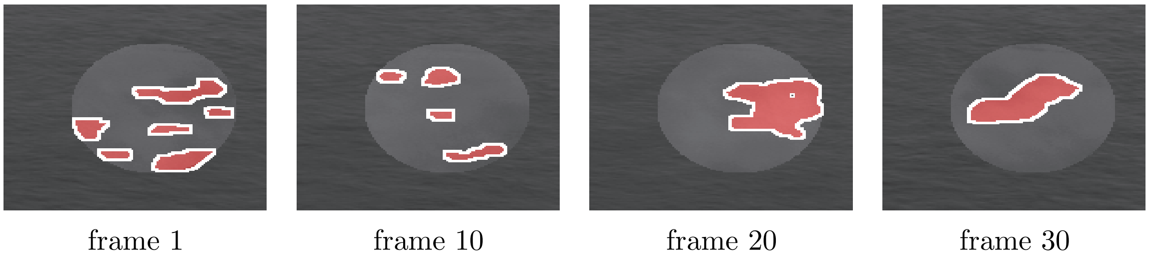


**Figure 11.** *Results of motion texture segmentation using a Gaussian MRF motion model (non–mixed-state) for the Ocean-Steam sequence. The mixed-state nature of the motion observations is not well handled in this case.*

continuously and rapidly changing in time. The images are of size 360 × 280, and blocks of $40 \times 40$ pixels were used. First, we compare our result with that presented for the same sequence in [29], as depicted in Figure 9(c). In [29], the linear dynamical models of intensity [28] assume that each dynamic texture is not varying its shape over time so that the model can be coherently estimated over a temporal window (120 frames in this case). The algorithm in [29] performs an average segmentation over the time window, and the boundary contour between flame and water is adjusted for the whole sequence, not capturing the boundary variations.

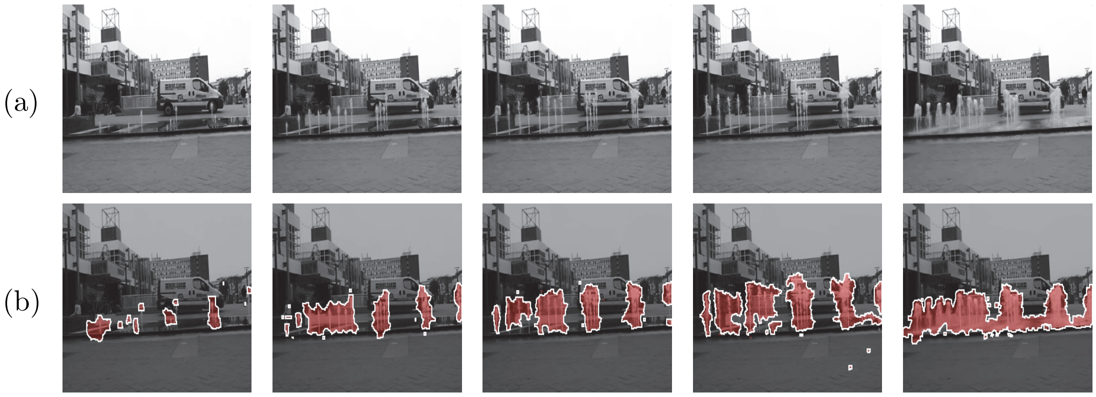


**Figure 12.** *Fountain sequence (*$320\times240$ *pixels).* (a) *Original frames from the video sequence corresponding to a deformable dynamic texture.* (b) *Result of the proposed segmentation method which is applied for each time instant between two consecutive frames.*

Next, in Figure 13(b) we display the result presented by Chan and Vasconcelos in [12] using mixtures of dynamic textures. They used spatiotemporal patches of five frames to track the dynamic texture deformation. However, oversmoothing occurs on the estimated boundary since it cannot be adjusted rapidly to changes that happen within the five-frame interval. In contrast, the segmentation strategy proposed here, and, in particular, our modeling approach, is able to perform a frame-by-frame segmentation with accurate results for each time instant. In the original images, the regions of Fire show considerable variations in terms of size and shape, and also they can occupy different nonconnected areas, as seen in Figure 13(e). Facing this highly dynamic behavior, the segmentation is accurately obtained by our method, which outperforms other state-of-the-art segmentation approaches.

**10.8. Spatiotemporal segmentation of motion textures.** We have shown that our mixed-state motion texture model can be efficiently exploited to perform two complex tasks—motion texture segmentation and recognition. The two-frame approach is especially attractive for dealing with instantaneous changes (moving boundaries and content changes). Our unique parametric representation can thus be exploited for a combined video sequence analysis task, where (1) motion texture regions are determined in a frame-by-frame basis by a spatial segmentation and (2) the motion texture model for each segment is tested at each time instant to determine if a change of content has occurred. To illustrate, we have concatenated the sequences Ocean-Steam and Ocean-Fire, one after the other, at frame 20. The segmentation algorithm is applied at each instant to obtain a sequence of motion texture segmentation maps. For the first segmentation map (at time 0) we learn the motion texture model parameters for each region and save them as reference models (regions A and B in Figure 14). Then, for each subsequent segmentation map we compute the KL distance between the motion texture model for each segment (regions 0 and 1) w.r.t. A and B. In this manner, at each time we obtain four values according to the four possible combinations of reference model and segment model (D0A, D0B, D1A, D1B). It is worthwhile first to analyze each curve:

- D0A: The initially learned model A is the background ocean motion texture of Ocean-Steam (region A). Until frame 20, regions A and 0 correspond to the same dynamic

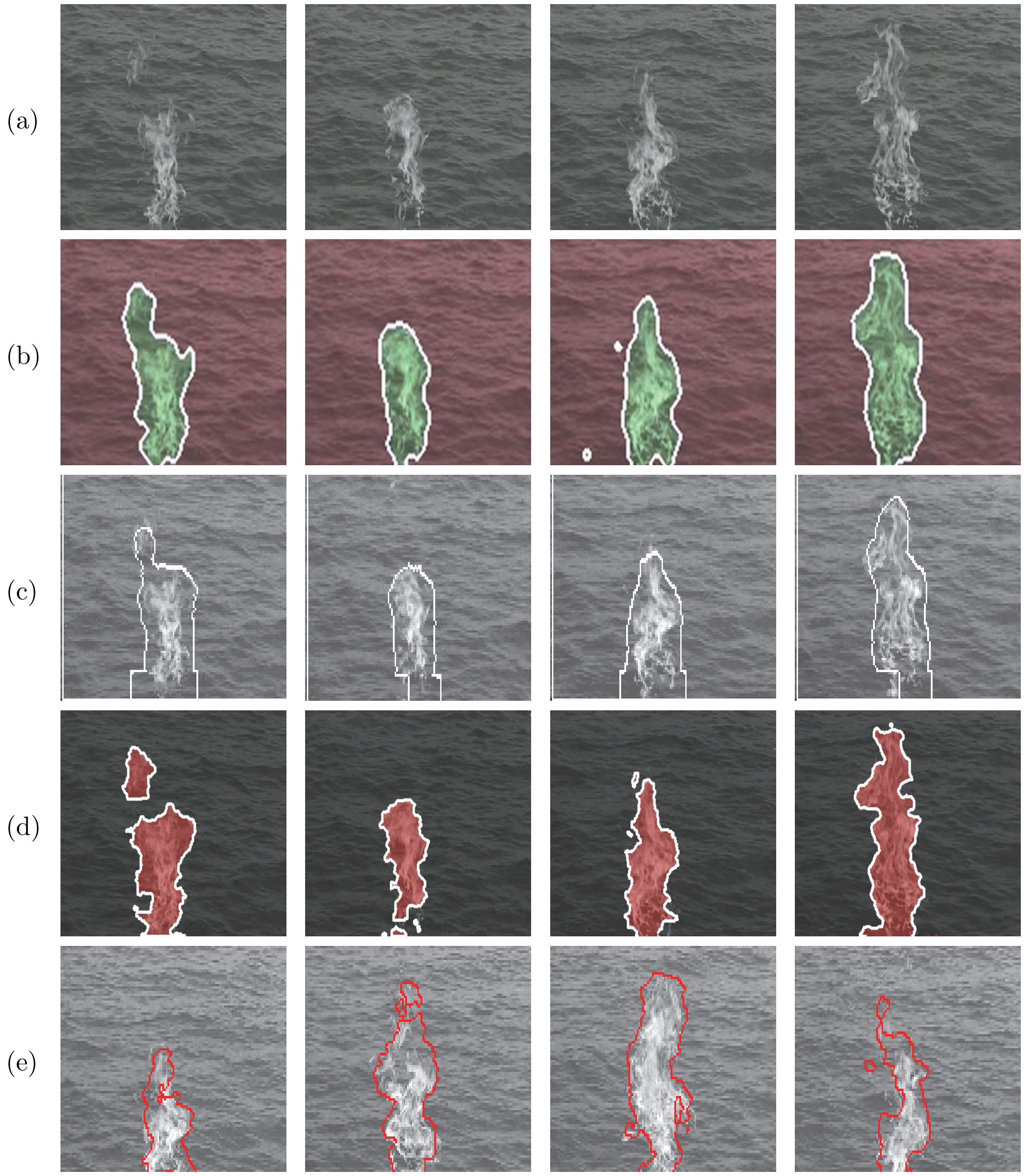


**Figure 13.** *Fire flame sequence (*360×280 *pixels).* (a) *Original frames from the video sequence corresponding to a deformable dynamic texture of fire over ocean waves.* (b) *Results for the method by Chan and Vasconcelos presented in* [12] *using spatiotemporal patches of five frames.* (c) *Results for the method by Chen et al.* [15] *using local binary patterns.* (d) *Results of our segmentation method, which is applied to each time instant between two consecutive frames.* (e) *Results by Ghoreyshi and Vidal* [36] *using Ising descriptors and ARX models (due to unavailability we present results for different instants).*

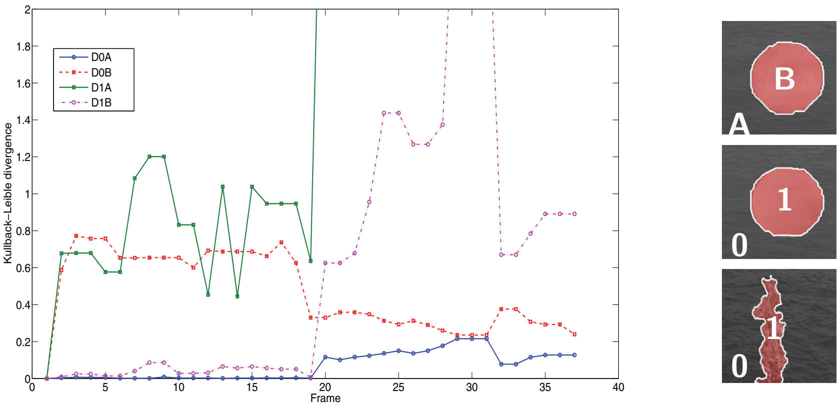


**Figure 14.** *Concatenation of the two-class sequences Ocean-Steam and Ocean-Fire. The curves show the KL distance between mixed-state motion texture models, where DXY stands for the KL distance between the model estimated in region X at the current frame and reference model learned from region Y at the initial instant (see text).*

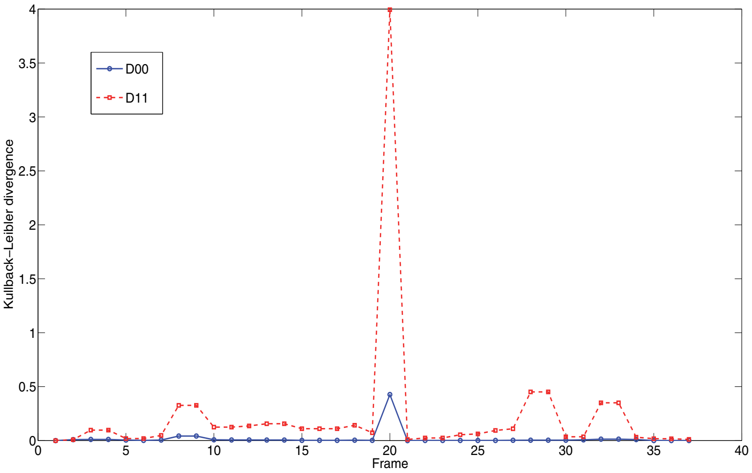


**Figure 15.** *Concatenation of the two-class sequences Ocean-Steam and Ocean-Fire. The curves show the instantaneous KL distance between the foreground regions segmented at two consecutive instants* (D11)*, and the background regions segmented at two consecutive instants* (D00)*. A peak appears when the class Steam changes to Fire in, say, the foreground (region* 1*), while the values remain low between the background regions with two different instances of Ocean.*

content. The distance values D0A are in this case very low. From frame 20, region 0 corresponds to the ocean motion texture of Ocean-Fire. Both ocean textures are similar but with some variations, which leads to slightly higher D0A values, but still indicates that both models are very close.

- D0B: This curve always corresponds to different classes. Region B is the steam motion texture, and region 0 corresponds to Ocean along the whole sequence. Large distance values observed before frame 20 evidence this situation, with lower but still significant values afterwards. Note that the D0B curve is always over D0A.

- D1A: This curve is the complementary of D0B where the initially learned model corresponds to Ocean and region 1 is the foreground motion texture (Steam or Fire). Clearly, the distance values are always large, as region A and region 1 always refer to different contents.
- D1B: This curve is probably the most interesting from the point of view of the application. It shows that the initially learned model in region B corresponds to the same content as region 1 up to frame 20 (Steam). After this time instant, region 1 changes from Steam to Fire, which is correctly reflected by a sudden increase in the D1B curve.

In practice, this process is done sequentially; that is, at the current frame we first segment the motion textures, and then the four KL distances are computed before passing to the next frame. Regarding the reference models, they can either be estimated at the initial frame or they can be learned off-line with explicit reference to a particular content class.

This experiment shows the discriminating power of the mixed-state motion texture model not only spatially (segmentation) but also temporally (change detection) and how both tasks can be combined into a single and unified motion texture analysis framework. Undiscussed aspects such as determining an optimal decision rule on the values of the KL distance depend on the concrete application and are to be studied in the future.

Finally, in a practical scenario where the detection is performed on-line and one has to account for successive changes of content, one would imagine the following setting. The KL distance is computed between the model parameters estimated from consecutive motion textures instead of considering a reference model at the beginning of the sequence. In this case, a content change detection is identified by observing a KL distance peak as shown in Figure 15 when the class Steam changes to Fire in, say, the foreground (region 0) and the values remain low for the background Ocean.

## 11. Conclusion.

We have proposed a new approach to dynamic texture modeling, based on a spatial statistical parametric model of the apparent motion extracted from video sequences. The mixed-state motion texture model has shown to be a compact and powerful representation for describing complex dynamic content with only a few parameters. One of the main advantages of our proposal is that we can model, learn, and identify the motion texture on a two-frame basis. As a consequence we can properly handle the segmentation of motion textures with moving or deformable spatial support over time. Moreover, the temporal consistency of estimated motion texture model parameters was demonstrated, along with an accurate classification rate, proving a high discrimination power from a limited training step. In contrast, several existing state-of-the-art methods need to take into account long sequences of images in order to estimate hundreds or thousands of parameters.

We have obtained state-of-the-art results for the recognition and classification issues; while our model is parsimonious, the learning stage is very light, and the recognition is on a two-frame basis. Furthermore, we provide a survey of the datasets used by the different works on dynamic texture recognition and classification. We have also introduced two new issues for the dynamic texture domain: dynamic texture change detection and space-time segmentation of temporal textures. We can straightforwardly and successfully handle them by using the KL distance between two MS-MRF dynamic texture models and by combining our spatial segmentation and classification methods.

We have shown that motion information is powerful in order to classify and segment dynamic textures and that an MS-MRF is an adequate model regarding the specific nature of motion observations and the local interactions involved. Due to the nature of the moving scenes encountered for dynamic textures (close to an "ergodic" property), a spatial modeling can be sufficient, and our MS-MRF model enables us to design an efficient, flexible, and accurate two-frame approach for dynamic texture content analysis. Extensions to other types of motion information such as optical flow could also be envisaged.

## REFERENCES

[1] S. Baker, D. Scharstein, J.P. Lewis, S. Roth, M.J. Black, and R. Szeliski, *A database and evaluation methodology for optical flow*, Int. J. Comp. Vision, 92 (2011), pp. 1–31.

[2] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, *Performance of optical flow techniques*, Int. J. Comp. Vision, 12 (1994), pp. 43–77.

[3] J. Besag, *Spatial interaction and the statistical analysis of lattice systems*, J. Roy. Statist. Soc. Ser. B, 36 (1974), pp. 192–236.

[4] J. Besag, *On the statistical analysis of dirty pictures*, J. Roy. Statist. Soc. Ser. B, 48 (1986), pp. 259–302.

[5] P Bouthemy, C Hardouin, G Piriou, and J.-F. Yao, *Mixed-state auto-models and motion texture modeling*, J. Math. Imaging Vision, 25 (2006), pp. 387–402.

[6] Y. Boykov, O. Veksler, and R. Zabih, *Efficient approximate energy minimization via graph cuts*, IEEE Trans. Pattern Anal. Machine Intell., 20 (2001), pp. 1222–1239.

[7] A. Bruhn, J. Weickert, , C. Feddern, T. Kohlberger, and C. Schnorr, *Variational optical flow computation in real time*, IEEE Trans. Image Process., 14 (2005), pp. 608–615.

[8] B. Cernuschi-Frías, *Mixed-states Markov random fields with symbolic labels and multidimensional real values*, Rapport de Recherche INRIA 6255, 2007. http://hal.inria.fr/docs/00/16/59/37/PDF/RR-6255.pdf.

[9] A.B. Chan, E. Coviello, and G.R. Lanckriet, *Clustering dynamic textures with the hierarchical EM algorithm*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10), IEEE, Washington, DC, 2010, pp. 2022–2029.

[10] A.B. Chan and N. Vasconcelos, *Mixtures of dynamic textures*, in Proceedings of the IEEE International Conference on Computer Vision (ICCV'05), IEEE, Washington, DC, 2005, pp. 641–647.

[11] A.B. Chan and N. Vasconcelos, *Classifying video with kernel dynamic textures*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07), IEEE, Washington, DC, 2007.

[12] A.B. Chan and N. Vasconcelos, *Modeling, clustering, and segmenting video with mixtures of dynamic textures*, IEEE Trans. Pattern Anal. Machine Intell., 30 (2008), pp. 909–926.

[13] A.B. Chan and N. Vasconcelos, *Layered dynamic textures*, IEEE Trans. Pattern Anal. Machine Intell., 31 (2009), pp. 1862–1879.

[14] A.B. Chan and N. Vasconcelos, *Variational layered dynamic textures*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09), IEEE, Washington, DC, 2009, pp. 1062–1069.

[15] J. Chen, G. Zhao, M. Salo, E. Rahtu, and M. Pietikainen, *Automatic dynamic texture segmentation using local descriptors and optical flow*, IEEE Trans. Image Process., 22 (2013), pp. 326–339.

[16] D. Chetverikov, S. Fazekas, and M. Haindl, *Dynamic texture as foreground and background*, Machine Vision Appl., 22 (2011), pp. 741–750.

[17] D. Chetverikov and R. Peteri, *A brief survey of dynamic texture description and recognition*, in Proceedings of the International Conference on Computer Recognition Systems (CORES'05), Springer Advances in Soft Computing, Springer, New York, 2005, pp. 17–26.

[18] L. Cooper, J. Liu, and K. Huang, *Spatial segmentation of temporal texture using mixture linear models*, in Dynamical Vision, WDV 2005/2006, Lecture Notes in Comput. Sci. 4358, Springer, New York, 2007, pp. 142–150.

[19] T. Corpetti, E. Memin, and P. Perez, *Dense estimation of fluid flows*, IEEE Trans. Pattern Anal. Machine Intell., 24 (2002), pp. 365–380.

[20] T. Cover and J. Thomas, *Elements of Information Theory*, John Wiley and Sons, New York, 1991.

[21] T. Crivelli, P. Bouthemy, B. Cernuschi-Frías, and J.-F. Yao, *Simultaneous motion detection and background reconstruction with a conditional mixed-state Markov random field*, Int. J. Comp. Vision, 94 (2011), pp. 295–316.

[22] T. Crivelli, B. Cernuschi-Frías, P. Bouthemy, and J.-F. Yao, *Mixed-state Markov random fields for motion texture modeling and segmentation*, in Proceedings of the IEEE International Conference on Image Processing (ICIP'06), IEEE, Washington, DC, 2006, pp. 1857–1860.

[23] T. Crivelli, B. Cernuschi-Frías, P. Bouthemy, and J.-F. Yao, *Segmentation of motion textures using mixed-state Markov random fields*, in Proceedings of SPIE, Vol. 6315, SPIE, Bellingham, WA, 2006, 63150J.

[24] T. Crivelli, B. Cernuschi-Frías, P. Bouthemy, and J.-F. Yao, *Mixed-state causal modeling for statistical kl-based motion texture tracking*, Pattern Recognition Lett., 31 (2010), pp. 2286–2294.

[25] T. Crivelli, G. Piriou, P. Bouthemy, B. Cernuschi-Frías, and J.-F. Yao, *Simultaneous motion detection and background reconstruction with a mixed-state conditional Markov random field*, in European Conference on Computer Vision (ECCV'08), Lecture Notes in Comput. Sci. 5302, Springer, New York, 2008, pp. 113–126.

[26] G.R Cross and A.K. Jain, *Markov random field texture models*, IEEE Trans. Pattern Anal. Machine Intell., 5 (1983), pp. 25–39.

[27] K.G. Derpanis and R.P. Wildes, *Spacetime texture representation and recognition based on a spatiotemporal orientation analysis*, IEEE Trans. Pattern Anal. Machine Intell., 34 (2012), pp. 1193–1205.

[28] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto, *Dynamic textures*, Int. J. Comp. Vision, 51 (2003.), pp. 91–109.

[29] G. Doretto, D. Cremers, P. Favano, and S. Soatto, *Dynamic texture segmentation*, in Proceedings of the International Conference on Computer Vision (ICCV'03), IEEE, Washington, DC, 2003, pp. 1236–1242.

[30] S. Dubois, R. Péteri, and M. Ménard, *Decomposition of dynamic textures using morphological component analysis*, IEEE Trans. Circuits Systems Video Tech., 22 (2012), pp. 188–201.

[31] I. Elfadel and R. Picard, *Gibbs random fields, cooccurrences, and texture modeling*, IEEE Trans. Pattern Anal. Machine Intell., 16 (1994), pp. 24–37.

[32] R. Fablet and P. Bouthemy, *Motion recognition using non-parametric image motion models estimated from temporal and multiscale co-ocurrence statistics*, IEEE Trans. Pattern Anal. Machine Intell., 25 (2003), pp. 1619–1624.

[33] S. Fazekas and D. Chetverikov, *Normal versus complete flow in dynamic texture recognition: A comparative study*, in Texture 2005: 4th International Workshop on Texture Analysis and Synthesis at ICCV'05, Beijing, China, 2005, pp. 37–42.

[34] S. Geman and D. Geman, *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*, IEEE Trans. Pattern Anal. Machine Intell., 6 (1984), pp. 721–741.

[35] B. Ghanem and N. Ahuja, *Maximum margin distance learning for dynamic texture recognition*, in Proceedings of the European Conference on Computer Vision (ECCV'10), Vol. 2, Crete, Greece, 2010, pp. 223–236.

[36] A. Ghoreyshi and R. Vidal, *Segmenting dynamic textures with Ising descriptors, ARX models and level sets*, in Dynamical Vision, WDV 2005/2006, Lecture Notes in Comput. Sci. 4358, Springer, New York, 2007, pp. 127–141.

[37] C. Hardouin and J.-F. Yao, *Spatial modelling for mixed-state observations*, Electron. J. Stat., 2 (2008), pp. 213–233.

[38] B. Horn and B. Schunck, *Determining optical flow*, Artificial Intelligence, 17 (1981), pp. 185–203.

[39] J. Hsieh, S. Yu, and Y. Chen, *Motion-based video retrieval by trajectory matching*, IEEE Trans. Circuits Systems Video Tech., 16 (2006), pp. 396–409.

[40] M. Jain, H. Jégou, and P. Bouthemy, *Better exploiting motion for better action recognition*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR'13), Portland, OR, 2013.

[41] H. Ji, X. Yang, H. Ling, and Y. Xu, *Wavelet domain multifractal analysis for static and dynamic texture classification*, IEEE Trans. Image Process., 22 (2013), pp. 286–299.

[42] V. Kolmogorov and C. Rother, *Minimizing nonsubmodular functions with graph cuts-a review*, IEEE Trans. Pattern Anal. Machine Intell., 29 (2007), pp. 1274–1279.

[43] V. Kolmogorov and R. Zabih, *What energy functions can be minimized via graph cuts?*, IEEE Trans. Pattern Anal. Machine Intell., 26 (2004), pp. 147–159.

[44] S. Krishnamachari and R. Chellappa, *Multiresolution Gauss-Markov random field models for texture segmentation*, IEEE Trans. Image Process., 6 (1997), pp. 251–267.

[45] V. S. Lempitsky, C. Rother, S. Roth, and A. Blake, *Fusion moves for Markov random field optimization*, IEEE Trans. Pattern Anal. Machine Intell., 32 (2010), pp. 1392–1405.

[46] R. Lizarraga-Morales, Y. Guo, G. Zhao, and M. Pietikinen, *Dynamic texture synthesis in space with a spatio-temporal descriptor*, in Asian Conference on Computer Vision 2012 Workshops (ACCVW'12), Vol. 7728, Daejeon, Korea, 2013, pp. 38–40.

[47] Z. Lu, W. Xie, J. Pei, and J. Huang, *Dynamic texture recognition by spatio-temporal multiresolution histograms*, in IEEE Workshop on Motion and Video Computing (WACV/MOTION), IEEE, Washington, DC, 2005, pp. 241–246.

[48] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, *Anomaly detection in crowded scenes*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10), IEEE, Washington, DC, 2010, pp. 1975–1981.

[49] A. Mitiche and P. Bouthemy, *Computation and analysis of image motion: A synopsis of current problems and methods*, Int. J. Comp. Vision, 19 (1996), pp. 29–55.

[50] R. Nelson and R. Polana, *Qualitative recognition of motion using temporal texture*, CVGIP Image Understanding, 56 (1992), pp. 78–89.

[51] R. Peteri and D. Chetverikov, *Dynamic texture recognition using normal flow and texture regularity*, in Proceedings of IbPRIA 2005, Estoril, Portugal, pp. 223–230.

[52] R. Peteri, M. Huiskes, and S. Fazekas, *Dyntex: A comprehensive database of dynamic textures (http://projects.cwi.nl/dyntex/index.html)*, Pattern Recognition Lett., 31 (2010), pp. 1627–1632.

[53] G. Piriou, P. Bouthemy, and J.-F. Yao, *Recognition of dynamic video contents with global probabilistic models of visual motion*, IEEE Trans. Image Process., 15 (2006), pp. 3417–3430.

[54] A. Rahman and M. Murshed, *Real-time temporal texture characterisation using block based motion co-occurrence statistics*, in Proceedings of the IEEE International Conference on Image Processing (ICIP'04), IEEE, Washington, DC, 2004, pp. 1593–1596.

[55] A. Ravichandran, R. Chaudry, and R. Vidal, *View-invariant dynamic texture recognition using a bag of dynamical systems*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09), IEEE, Washington, DC, 2009, pp. 1651–1657.

[56] A. Ravichandran, R. Chaudry, and R. Vidal, *Categorizing dynamic textures using a bag of dynamical systems*, IEEE Trans. Pattern Anal. Machine Intell., 35 (2013), pp. 342–353.

[57] A. Ravichandran and R. Vidal, *A unified approach to segmentation and categorization of dynamic textures*, in Asian Conference on Computer Vision (ACCV'10), Vol. 1, Queenstown, New Zealand, 2010, pp. 425–438.

[58] A. Ravichandran and R. Vidal, *Video registration using dynamic textures*, IEEE Trans. Pattern Anal. Machine Intell., 33 (2011), pp. 158–171.

[59] J. Ren, X. Jiang, and J. Yuan, *Dynamic texture recognition using enhanced lbp features*, in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'13), IEEE, Washington, DC, 2013.

[60] P. Saisan, G. Doretto, Y. Wu, and S. Soatto, *Dynamic texture recognition*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01), Hawaii, 2001, pp. 58–63.

[61] F. Salzenstein and C. Collet, *Fuzzy Markov random fields versus chains for multispectral image segmentation*, IEEE Trans. Pattern Anal. Machine Intell., 28 (2006), pp. 1753–1767.

[62] F. Salzenstein and W. Pieczynski, *Parameter estimation in hidden fuzzy Markov random fields and image segmentation*, Graphical Models Image Process., 59 (1997), pp. 205–220.

[63] M. Szummer and R. Picard, *Temporal texture modelling*, in Proceedings of the IEEE International Conference on Image Processing (ICIP'96), IEEE, Washington, DC, 1995, pp. 823–826.

[64] B.U. Toreyin, Y. Dedeoglu, and A.E. Cetin, *Flame detection in video using hidden Markov models*, in Proceedings of the IEEE International Conference on Image Processing (ICIP'05), IEEE, Washington, DC, 2005.

[65] R. Vidal and A. Ravichandran, *Optical flow estimation and segmentation of multiple moving dynamic textures*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05), Vol. 2, IEEE, Washington, DC, 2005, pp. 516–521.

[66] R. Vikas, C. Sanderson, and B. Lovell, *Improved anomaly detection in crowded scenes via cell-based analysis of foreground speed, size and texture*, in IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW'11), 2011, pp. 55–61.

[67] Y. Wang and S.-C. Zhu, *Analysis and synthesis of textured motion: Particles and waves*, IEEE Trans. Pattern Anal. Machine Intell., 26 (2004), pp. 1348–1363.

[68] J. Xiao and M. Shah, *Motion layer extraction in the presence of occlusion using graph cuts*, IEEE Trans. Pattern Anal. Machine Intell., 10 (2005), pp. 1644–1659.

[69] J. Xu, S. Denman, S. Sridharan, C. Fookes, and R. Rana, *Dynamic texture reconstruction from sparse codes for unusual event detection in crowded scenes*, in Joint ACM Workshop on Modeling and Representing Events, ACM, New York, 2011, pp. 25–30.

[70] R. Xu and D. Wunsch, *Survey of clustering algorithms*, IEEE Trans. Neural Networks, 16 (2005), pp. 645–678.

[71] L. Yuan, F. Wen, C. Liu, and H. Shum, *Synthesizing dynamic textures with closed-loop linear dynamic systems*, in European Conference on Computer Vision (ECCV'04), Lecture Notes in Comput. Sci. 3022, Springer, New York, 2004, pp. 603–616.

[72] G. Zhao and M. Pietikainen, *Dynamic texture recognition using local binary patterns with an application to facial expressions*, IEEE Trans. Pattern Anal. Machine Intell., 29 (2007), pp. 915–927.

[73] S. Zhu and X. Liu, *Learning in Gibbsian fields: How accurate and how fast can it be?*, IEEE Trans. Pattern Anal. Machine Intell., 24 (2002), pp. 1001–1006.