J Math Imaging Vis
DOI 10.1007/s10851-015-0563-2

Video Primal Sketch: A Unified Middle-Level Representation for Video

Zhi Han · Zongben Xu · Song-Chun Zhu

Received: 20 December 2013 / Accepted: 20 January 2015

Abstract This paper presents a middle-level video representation named video primal sketch (VPS), which integrates two regimes of models: (i) sparse coding model using static or moving primitives to explicitly represent moving corners, lines, feature points, etc., (ii) FRAME /MRF model reproducing feature statistics extracted from input video to implicitly represent textured motion, such as water and fire. The feature statistics include histograms of spatio-temporal filters and velocity distributions. This paper makes three contributions to the literature: (i) Learning a dictionary of video primitives using parametric generative models; (ii) Proposing the spatio-temporal FRAME and motion-appearance FRAME models for modeling and synthesizing textured motion; and (iii) Developing a parsimonious hybrid model for generic video representation. Given an input video, VPS selects the proper models automatically for different motion patterns and is compatible with high-level action representations. In the experiments, we synthesize a number of textured motion; reconstruct real videos using the VPS; report a series of human perception experiments to verify the quality of reconstructed videos; demonstrate how the VPS changes over the scale transition in videos; and present the close connection between VPS and high-level action models.

Keywords Middle-level vision · Video representation · Textured motion · Dynamic texture synthesis · Primal sketch

Z. Han (✉) · Z. Xu
Institute for Information and System Sciences, Xi'an Jiaotong University, Xi'an, China
e-mail: han.zhi.hunt@gmail.com

Z. Xu
e-mail: zbxu@mail.xjtu.edu.cn

Z. Han · S.-C. Zhu
Department of Stat and CS, University of California, Los Angeles, USA
e-mail: sczhu@stat.ucla.edu

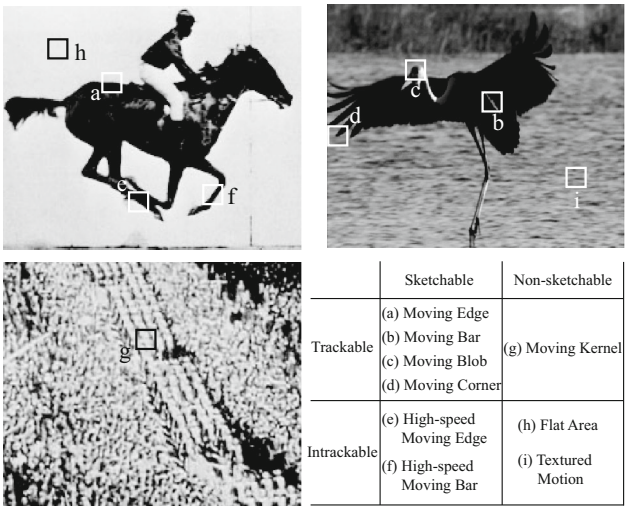
Z. Han
State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China

1 Introduction

1.1 Motivation

Videos of natural scenes contain vast varieties of motion patterns. We divide these motion patterns in a 2×2 table based on their complexities measured by two criteria: (i) sketchability [18], i.e., whether a local patch can be represented explicitly by an image primitive from a sparse coding dictionary, and (ii) intrackability (or trackability) [17], which measures the uncertainty of tracking an image patch using the entropy of posterior probability on velocities. Figure 1 shows some examples of the different video patches in the four categories. Category A consists of the simplest vision phenomena, i.e., sketchable and trackable motions, such as trackable corners, lines, and feature points, whose positions and shapes can be tracked between frames. For example, patches (a), (b), (c), and (d) belong to category A. Category D is the most complex and is called textured motions or dynamic texture in the literature, such as water, fire, or grass, in which the images have no distinct primitives or trackable motion, such as patches (h) and (i). The other categories are in between. Category B refers to sketchable but intrackable patches, which can be described by distinct image primitives but hardly be tracked between frames due to fast motion, for example, the patches (e) and (f) at the legs of the galloping horse. Finally, category C includes the trackable but non-sketchable patches, which are cluttered features or moving kernels, e.g., patch (g).

Published online: 04 February 2015



	Sketchable	Non-sketchable
Trackable	(a) Moving Edge (b) Moving Bar (c) Moving Blob (d) Moving Corner	(g) Moving Kernel
Intrackable	(e) High-speed Moving Edge (f) High-speed Moving Bar	(h) Flat Area (i) Textured Motion

Fig. 1 The four types of local video patches characterized by two criteria—sketchability and trackability

In the vision literature, as it was pointed out by [33], there are two families of representations, which code images or videos by explicit and implicit functions, respectively.

1. Explicit representations with generative models. Olshausen [27], Kim et al. [22] learned an over-complete set of coding elements from natural video sequences using the sparse coding model [28]. Elder and Zucker [15] and Guo et al. [18] represented the image/video patches by fitting functions with explicit geometric and photometric parameters. Wang and Zhu [36] synthesized complex motion, such as birds, snowflakes, and waves with a large mount of particles and wave components. Black and Fleet [4] represented two types of motion primitives, namely smooth motion and motion boundaries for motion segmentation. In higher level object motion tracking, people represented different tracking units depending on the underlying objects and scales, such as sparse or dense feature points tracking [4,32], kernels tracking [10,16], contours tracking [24], and middle-level pairwise-components generation [41].

2. Implicit representations with descriptive models. For textured motions or dynamic textures, people used numerous Markov models which are constrained to reproduce some statistics extracted from the input video. For example, dynamic textures [6,35] were modeled by a spatio-temporal auto-regressive (STAR) model, in which the intensity of each pixel was represented by a linear summation of intensities of its spatial and temporal neighbors. Bouthemy et al. [5] proposed mixed-state auto-models for motion textures by generalizing the auto-models in [3]. Doretto et al. [14] derived an auto-regression moving-average model for dynamic texture. Chan and Vasconcelos [7] and Ravichandran et al. [30] extended it to a stable linear dynamical system (LDS) model.

Recently, to represent complex motion, such as human activities, researchers have used Histogram of Oriented Gradients (HOG) [11] for appearance and Histogram of Oriented Optical Flow (HOOF) [8,12] for motion. The HOG and HOOF record the rough geometric information through the grids and pool the statistics (histograms) within the local cells. Such features are used for recognition in discriminative tasks, such as action classification, and are not suitable for video coding and reconstruction.

In the literature, these video representations are often manually selected for specific videos in different tasks. There lacks a generic representation and criterion that can automatically select the proper models for different patterns of the video. Furthermore, as it was demonstrated in [17] that both sketchability and trackability change over *scales*, *densities*, and *stochasticity of the dynamics*, a good video representation must adapt itself continuously in a long video sequence.

1.2 Overview and Contributions

Motivated by the above observations, we study a unified middle-level representation, called *video primal sketch* (VPS), by integrating the two families of representations. Our work is inspired by Marr's conjecture for a generic "token" representation called primal sketch as the output of early vision [26], and is aimed at extending the primal sketch model proposed by [18] from images to videos. Our goal is not only to provide a parsimonious model for video compression and coding, but also more importantly, to support and be compatible with high-level tasks such as motion tracking and action recognition.

Figure 2 overviews an example of the video primal sketch. Figure 2a is an input video frame which is separated into sketchable and non-sketchable regions by the sketchability map in (b), and trackable primitives and intrackable regions by the trackability map in (c). The sketchable or trackable regions are explicitly represented by a sparse coding model and reconstructed in (d) with motion primitives, and each non-sketchable and intrackable region has a textured motion which is synthesized in (e) by a generalized FRAME [42] model (implicit and descriptive). The synthesis of this frame is shown in (f) which integrates the results from (d) and (e) seamlessly.

As Table 1 shows, the explicit representations include 3,600 parameters for the positions, types, motion velocities, *etc.,* of the video primitives and the implicit representations have 420 parameters for the histograms of a set of filter responses on dynamic textures. This table shows the efficiency of the VPS model.

This paper makes the following contributions to the literature.

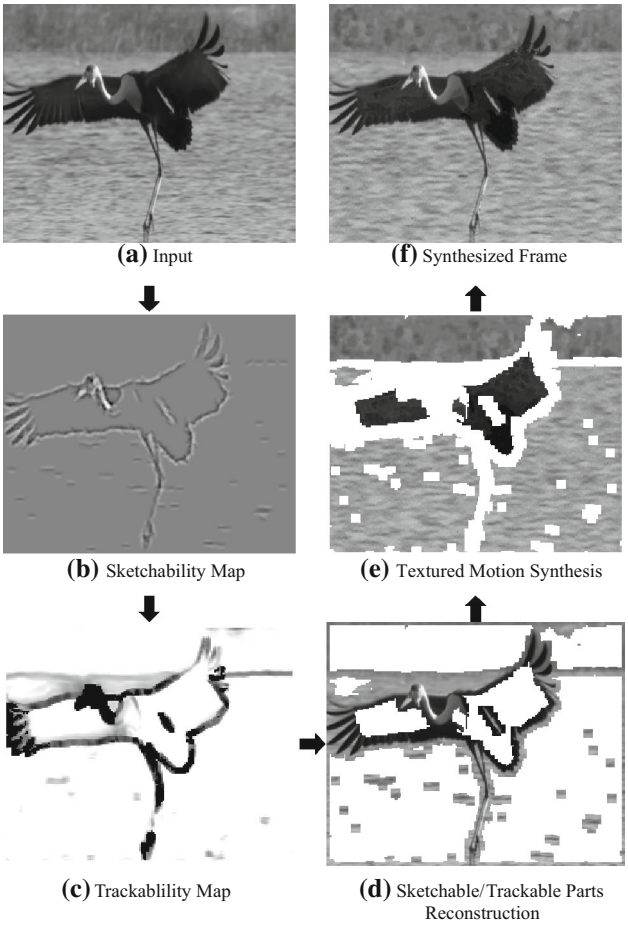


Fig. 2 An example of video primal sketch. **a** An input frame. **b** Sketchability map where dark means sketchable. **c** Trackability map where darker means higher trackability. **d** Reconstruction of explicit regions using primitives. **e** Synthesis for implicit regions (textured motions) by sampling the generalize FRAME model through Markov chain Monte Carlo using the explicit regions as boundary condition. **f** Synthesized frame by combining the explicit and implicit representations

Table 1 The parameters in video primal sketch model for the water bird video in Fig. 2

Video resolution	288 × 352 pixels
Explicit region	31,644 pixels ≈ 30 %
Primitive number	300
Primitive width	11 pixels
Explicit parameters	3,600 ≈ 3.6 %
Implicit parameters	420

1. We present and compare two different but related models to define textured motions. The first one is a spatio-temporal FRAME (ST-FRAME) model, which is a non-parametric Markov random field and generalizes the FRAME model [42] of texture with spatio-temporal filters. The ST-FRAME model is learned so that it has marginal probabilities that match the histograms of the responses from the spatio-temporal filters on the input video. The second one is a motion-appearance FRAME model (MA-FRAME), which not only matches the histograms of some spatio-temporal filter responses, but also matches the histograms of velocities pooled over a local region. The MA-FRAME model achieves better results in video synthesis than the ST-FRAME model, and it is, to some extent, similar to the HOOF features used in action classification [8,12].
2. We learn a dictionary of motion primitives from input videos using a generative sparse coding model. These primitives are used to reconstruct the explicit regions and include two types: (i) generic primitives for the sketchable patches, such as corners, bars etc; and (ii) specific primitives for the non-sketchable but trackable patches which are usually texture patches similar to those used in kernel tracking [10].
3. The models for implicit and explicit regions are integrated in a hybrid representation—the video primal sketch (VPS), as a generic middle-level representation of video. We will also show how VPS changes over information scales affected by distance, density, and dynamics.
4. We show the connections between this middle-level VPS representation and features for high-level vision tasks such as action recognition.

Our work is inspired by Gong's empirical study in [17], which revealed the statistical properties of videos over scale transitions and defined intrackability as the entropy of local velocities. When the entropy is high, the patch cannot be tracked locally and thus its motion is represented by a velocity histogram. Gong and Zhu [17] did not give a unified model for video representation and synthesis which is the focus on the current paper.

This paper extends a previous conference paper [19] in the following aspects:

1. We propose a new dynamic texture model, MA-FRAME, for better representing velocity information. Benefited from the new temporal feature, the VPS model can be applied to high-level action representation tasks more directly.
2. We compare spatial and temporal features with HOG [11] and HOOF [12] and discuss the connections between them.
3. We do a series of perceptual experiments to verify the high quality of video synthesis from the aspect of human perception.

The remainder of this paper is organized as follows. In Sect. 2, we present the framework of video primal sketch. In Sect. 3, we explain the algorithms for explicit represen-

tation, textured motion synthesis and video synthesis, and show a series of experiments. The paper is concluded with a discussion in Sect. 4.

2 Video Primal Sketch Model

In his monumental book [26], Marr conjectured a primal sketch as the output of early vision that transfers the continuous "analogy" signals in pixels to a discrete "token" representation. The latter should be parsimonious and sufficient to reconstruct the observed image without much perceivable distortions. A mathematical model was later studied by Guo et al. [18], which successfully modeled hundreds of images by integrating sketchable structures and non-sketchable textures. In this section, we extend it to video primal sketch as a hybrid generic video representation.

Let $\mathbf{I}[1, m] = \{\mathbf{I}_{(t)}\}_{t=1}^{m}$ be a video defined on a 3D lattice $\Lambda \subset Z^3$. Λ is divided disjointly into explicit and implicit regions,

$$\Lambda = \Lambda_{\text{ex}} \bigcup \Lambda_{\text{im}}, \ \ \Lambda_{\text{ex}} \bigcap \Lambda_{\text{im}} = \emptyset. \tag{1}$$

Then the video $\mathbf{I}$ is decomposed as two components

$$\mathbf{I}_{\Lambda} = (\mathbf{I}_{\Lambda_{\text{ex}}}, \mathbf{I}_{\Lambda_{\text{im}}}). \tag{2}$$

$\mathbf{I}_{\Lambda_{\text{ex}}}$ are defined by explicit functions $I = g(w)$, in which, each instance is corresponded to a different function form of $g()$ and indexed by a particular value of parameter w. And $\mathbf{I}_{\Lambda_{\text{im}}}$ are defined by implicit functions $H(I) = h$, in which, $H()$ extracts the statistics of filter responses from image I and h is a specific value of histograms.

In the following, we first present the two families of models for $\mathbf{I}_{\Lambda_{\text{ex}}}$ and $\mathbf{I}_{\Lambda_{\text{im}}}$, respectively, and then integrate them in the VPS model.

2.1 Explicit Representation by Sparse coding

The explicit region Λ_{ex} of a video $\mathbf{I}$ is decomposed into n_{ex} disjoint domains (usually n_{ex} is in the order of 10^2),

$$\Lambda_{\text{ex}} = \bigcup_{i=1}^{n_{\text{ex}}} \Lambda_{\text{ex},i}. \tag{3}$$

Here $\Lambda_{\text{ex},i} \subset \Lambda$ defines the domain of a "brick". A brick, denoted by $\mathbf{I}_{\Lambda_{\text{ex},i}}$, is a spatio-temporal volume like a patch in images. These bricks are divided into the three categories A, B, and C as we mentioned in Sect. 1.

The size of $\Lambda_{\text{ex},i}$ influences the results of tracking and synthesis to some degree. The spatial size should depend on the scale of structures or the granularity of textures, and the temporal size should depend on the motion amplitude and frequency in time dimension, which are hard to estimate in real applications. However, a general size works well for most of cases, say 11 × 11 pixels × 3 frames for trackable bricks (sketchable or non-sketchable), or 11 × 11 pixels ×1 frame for sketchable but intrackable bricks. Therefore, in all the experiments of this paper, the size of $\Lambda_{\text{ex},i}$ is chosen as such.

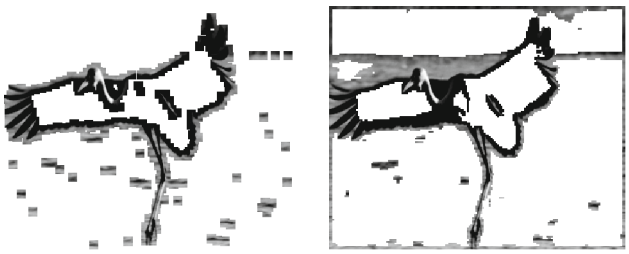


Fig. 3 Comparison between sketchable and trackable regions

Figure 3 shows one example comparing the sketchable and trackable regions based on sketchability and trackability maps shown in Fig. 2b, c, respectively. It is worth noting that the two regions overlap with only a small percentage of the regions is either sketchable or trackable.

Each brick can be represented by a primitive $B_i \in \Delta_B$ through an explicit function,

$$\mathbf{I}(x, y, t) = \alpha_i B_i(x, y, t) + \epsilon, \quad \forall (x, y, t) \in \Lambda_{\text{ex},i}. \tag{4}$$

B_i means the ith primitive from the primitive dictionary Δ_B, which fits the brick $\mathbf{I}_{\Lambda_{\text{ex},i}}$ best. Here i indexes the parameters such as type, position, orientation, and scale of B_i. α_i is the corresponding coefficient. ϵ represents the residue, which is assumed to be i.i.d. Gaussian. For a trackable primitive, $B_i(x, y, t)$ includes 3 frames and thus encodes the velocity (u, v) in the 3 frames. For sketchable but intrackable primitive, $B_i(x, y, t)$ has only 1 frame.

As Fig. 4 illustrates, the dictionary Δ_B is composed of two categories:

- Common primitives Δ_B^{common}. These are primitives shared by most videos, such as blobs, edges, and ridges. They have explicit parameters for orientations and scales. They are mostly belong to sketchable region as shown in Fig. 3.
- Special primitives $\Delta_B^{\text{special}}$. These bricks do not have common appearance and are limited to specific video frames. They are non-sketchable but trackable, and are recorded to code the specific video region. They mostly belong to trackable region but not included in sketchable region as shown in Fig. 3.

To be noted, the primitives and categories shown in Fig. 4 are some selected examples, but not the whole dictionary. The details for learning these primitives are introduced in Sect. 3.2.

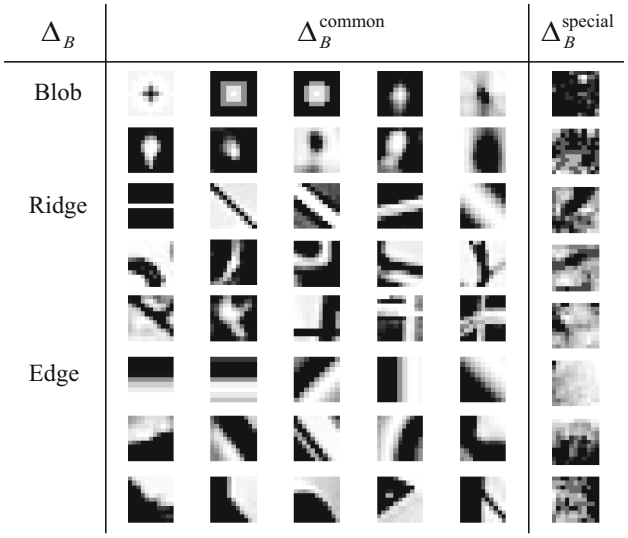


Fig. 4 Some selected examples of primitives. Δ_B is a dictionary of primitives with velocities (u,v) ((u,v) is not shown), such as blobs, ridges, edges, and special primitives

Equation (4) uses only one base function and thus is different from conventional linear additive model. Following the Gaussian assumption for the residues, we have the following probabilistic model for the explicit region $\mathbf{I}_{\Lambda_{\text{ex}}}$

$$p(\mathbf{I}_{\Lambda_{\text{ex}}}; \mathbf{B}, \alpha) = \prod_{i=1}^{n_{\text{ex}}} \frac{1}{(2\pi)^{\frac{n}{2}}\sigma_i^n} \exp\{-E_i\}$$
$$E_i = \sum_{(x,y,t)\in\Lambda_{\text{ex},i}} \frac{(\mathbf{I}(x,y,t) - \alpha_i B_i(x,y,t))^2}{2\sigma_i^2}. \tag{5}$$

where $\mathbf{B} = (B_1, \ldots, B_{n_{\text{ex}}})$ represents the selected primitive set, n is the size of each primitive, n_{ex} is the number of selected primitives, and σ_i is estimated standard deviation of representing natural videos by B_i.

2.2 Implicit Representations by FRAME Models

The implicit region Λ_{im} of video $\mathbf{I}$ is segmented into n_{im} (usually n_{im} is no more than 10) disjoint homogeneous textured motion regions,

$$\Lambda_{\text{im}} = \bigcup_{j=1}^{n_{\text{im}}} \Lambda_{\text{im},j}. \tag{6}$$

One effective approach for texture modeling is to pool the histograms for a set of filters (Gabor, DoG and DooG) on the input image [2,9,21,29,42]. Since Gabor filters model the response functions of the neurons in the primary visual cortex, two texture images with the same histograms of filter responses generate the same texture impression, and thus are considered perceptually equivalent [34]. The FRAME model proposed in [42] generates the expected marginal statistics to match the observed histograms through the maximum entropy principle. As a result, any images drawn from this model will have the same filtered histograms and thus can be used for synthesis or reconstruction.

We extend this concept to video by adding temporal constraints and define each homogeneous textured motion region $\mathbf{I}_{\Lambda_{\text{im},j}}$ by an equivalence class of videos,

$$\Omega_K(\mathbf{h}_j) = \{\mathbf{I}_{\Lambda_{\text{im},j}} : H_k(\mathbf{I}_{\Lambda_{\text{im},j}}) = h_{k,j}, k = 1, 2, \ldots, K\}. \tag{7}$$

where $\mathbf{h}_j = (h_{1,j}, \ldots, h_{K,j})$ is a series of 1D histograms of filtered responses that characterize the macroscopic properties of the textured motion pattern. Thus we only need to code the histograms $\mathbf{h}_j$ and synthesize the textured motion region $\mathbf{I}_{\Lambda_{\text{im},j}}$ by sampling from the set $\Omega_K(\mathbf{h}_j)$. As $\mathbf{I}_{\Lambda_{\text{im},j}}$ is defined by the implicit functions, we call it an implicit representation. These regions are coded up to an equivalence class in contrast to reconstructing the pixel intensities in the explicit representation.

To capture temporal constraints, one straightforward method is to choose a set of spatio-temporal filters and calculate the histograms of the filter responses. This leads to the spatio-temporal FRAME (ST-FRAME) model which will be introduced in Sect. 2.3. Another method is to compute the statistics of velocity. Since the motion in these regions is intrackable, at each point of the image, its velocity is ambiguous (large entropy). We pool the histograms of velocities locally in a way similar to the HOOF (Histogram of Oriented Optical Flow) [8,12] features in action classification. This leads to the motion-appearance FRAME (MA-FRAME) model which uses histograms of both appearance (static filters) and velocities. We will elaborate on this model in Sect. 2.4.

2.3 Implicit Representation by Spatio-Temporal FRAME

ST-FRAME is an extension of the FRAME model [42] by adopting spatio-temporal filters.

A set of filters $\mathbf{F}$ is selected from a filter bank Δ_F. Figure 5 illustrates the three types of filters in Δ_F: (i) the static filters for texture appearance in a single image; (ii) the motion filter with certain velocity; and (iii) the flicker filter that have zero velocity but opposite signs between adjacent frames. For each filter $F_k \in \mathbf{F}$, the spatio-temporal filter response of $\mathbf{I}$ at $(x, y, t) \in \Lambda_{\text{im},j}$ is $F_k * \mathbf{I}(x, y, t)$. The convolution is over spatial and temporal domain. By pooling the filter responses over all $(x, y, t) \in \Lambda_{\text{im},j}$, we obtain a number of 1D histograms

$$H_k(\mathbf{I}_{\Lambda_{\text{im},j}}) = H_k(z; \mathbf{I}_{\Lambda_{\text{im},j}})$$
$$= \frac{1}{|\Lambda_{\text{im},j}|} \sum_{(x,y,t)\in\Lambda_{\text{im},j}} \delta(z; F_k * \mathbf{I}(x,y,t)),$$
$$k = 1, \ldots, K. \tag{8}$$

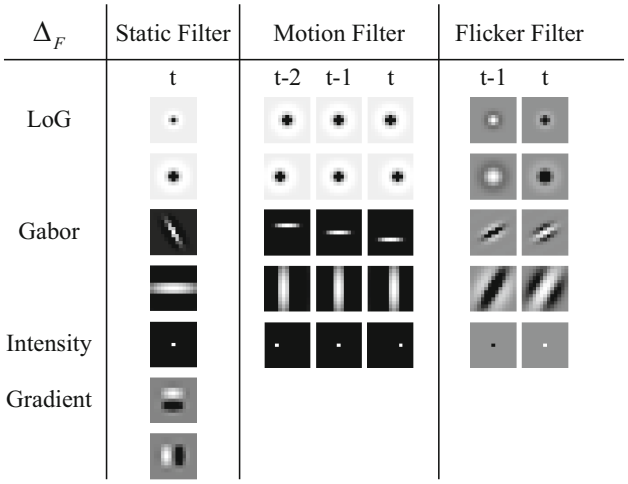


Fig. 5 Δ_F is a dictionary of spatio-temporal filters including static, motion, and flicker filters

where z indexes the histogram bins, and $\delta(z; x) = 1$ if x belongs to bin z, and $\delta(z; x) = 0$ otherwise. Following the FRAME model, the statistical model of textured motion $\mathbf{I}_{\Lambda_{\text{im},j}}$ is written in the form of the following Gibbs distribution,

$$p(\mathbf{I}_{\Lambda_{\text{im},j}}; \mathbf{F}, \beta) \propto \exp\left\{-\sum_k \langle \beta_{k,j}, H_k(\mathbf{I}_{\Lambda_{\text{im},j}})\rangle\right\}. \tag{9}$$

where $\beta_k = (\beta_{k,1}, \beta_{k,2}, \ldots, \beta_{k,3})$ are potential functions.

According to the theorem of ensemble equivalence [39], the Gibbs distribution converges to the uniform distribution over the set $\Omega_K(\mathbf{h}_j)$ in (7), when $\Lambda_{\text{im},j}$ is large enough. For any fixed local brick $\Lambda_0 \subset \Lambda_{\text{im},j}$, the distribution of I_{Λ_0} follows the Markov random field model (9). The model can describe textured motion located in an irregular shape region $\Lambda_{\text{im},j}$.

The filters in $\mathbf{F}$ are pursued one by one from the filter bank Δ_F so that the information gain is maximized at each step.

$$F_k^* = \arg\max_{F_k \in \Delta_F} \left\| H_k^{\text{syn}} - H_k^0 \right\|. \tag{10}$$

H_k^0 and H_k^{syn} are the response histograms of F_k before and after synthesizing $\mathbf{I}_{\Lambda_{\text{im},j}}$ by adding F_k, respectively. The larger the difference, the more important is the filter.

Following the distribution form of (9), the probabilistic model of implicit parts of $\mathbf{I}$ is defined as

$$p(\mathbf{I}_{\Lambda_{\text{im}}}; \mathbf{F}, \beta) \propto \prod_{j=1}^{n_{\text{im}}} p(\mathbf{I}_{\Lambda_{\text{im},j}}; \mathbf{F}, \beta). \tag{11}$$

where $\mathbf{F} = (F_1, \ldots, F_K)$ represents the selected spatio-temporal filter set.

In the experiments described later, we demonstrate that this model can synthesize a range of dynamic textures by matching the histograms of filter responses. The synthesis is done through sampling the probability by Markov chain Monte Carlo.

2.4 Implicit Representation by Motion-Appearance FRAME

Different from ST-FRAME, in which, temporal constraints are based on spatio-temporal filters, the MA-FRAME model uses the statistics of velocities, in addition to the statistics of filter responses for appearance.

For the appearance constraints, the filter response histograms $H^{(s)}$ are obtained similarly as ST-FRAME in (8)

$$\begin{aligned} H_k^{(s)}(\mathbf{I}_{\Lambda_{\text{im},j}}) &= H_k^{(s)}(z; \mathbf{I}_{\Lambda_{\text{im},j}}) \\ &= \frac{1}{|\Lambda_{\text{im},j}|} \sum_{(x,y,t)\in\Lambda_{\text{im},j}} \delta(z; F_k * \mathbf{I}(x, y, t)), \\ & k = 1, \ldots, K. \end{aligned} \tag{12}$$

where the filter set $\mathbf{F}$ includes static and flicker filters in Δ_F.

For the motion constraints, the velocity distribution of each local patch is estimated via the calculation of trackability [17], in which, each patch is compared with its spatial neighborhood in adjacent frame and the probability of the local velocity v is computed as

$$\begin{aligned} &p(v|I(x, y, t-1), I(x, y, t)) \\ &\propto \exp\left\{-\frac{\|I_{\partial(x-v_x, y-v_y, t)} - I_{\partial(x,y,t-1)}\|^2}{2\sigma^2}\right\}. \end{aligned} \tag{13}$$

Here, σ is the standard deviation of the differences between local patches from adjacent frames based on various velocities. The statistical information of velocities for a certain area of texture is approximated by averaging the velocity distribution over region $\Lambda_{\text{im},j}$

$$H^{(t)}(\mathbf{v}_{\Lambda_{\text{im},j}}) = \sum_{(x,y,t)\in\Lambda_{\text{im},j}} p(v|I(x, y, t-1), I(x, y, t)). \tag{14}$$

Let $\mathbf{H} = (\mathbf{H}^{(s)}(\mathbf{I}_{\Lambda_{\text{im},j}}), \mathbf{H}^{(t)}(\mathbf{v}_{\Lambda_{\text{im},j}}))$ collect the filter responses and velocities histograms of the video. The statistical model of textured motion $\mathbf{I}_{\Lambda_{\text{im},j}}$ can be written in the form of the following joint Gibbs distribution,

$$p(\mathbf{I}_{\Lambda_{\text{im},j}}; \mathbf{F}, \beta) \propto \exp\left\{-\langle \boldsymbol{\beta}, \mathbf{H}(\mathbf{I}_{\Lambda_{\text{im},j}}, \mathbf{v}_{\Lambda_{\text{im},j}})\rangle\right\}. \tag{15}$$

Here, β is the parameter of the model.

In summary, the probabilistic model for the implicit regions of $\mathbf{I}$ is defined as

$$p(\mathbf{I}_{\Lambda_{\text{im}}}; \mathbf{F}, \beta) \propto \prod_{j=1}^{n_{\text{im}}} p(\mathbf{I}_{\Lambda_{\text{im},j}}; \mathbf{F}, \beta). \tag{16}$$

where $\mathbf{F} = (F_1, \ldots, F_K)$ represents the selected filter set.

In the experiment section, we show the effectiveness of the MA-FRAME model and its advantages over the ST-FRAME model.

2.5 Hybrid Model for Video Representation

The ST or MA-FRAME models for the implicit regions $\mathbf{I}_{\Lambda_{\text{im}}}$ use the explicit regions $\mathbf{I}_{\Lambda_{\text{ex}}}$ as boundary conditions, and the probabilistic models for $\mathbf{I}_{\Lambda_{\text{ex}}}$ and $\mathbf{I}_{\Lambda_{\text{im}}}$ are given by (5) and (11), respectively

$$\mathbf{I}_{\Lambda_{\text{ex}}} \sim p(\mathbf{I}_{\Lambda_{\text{ex}}}; \mathbf{B}, \alpha), \mathbf{I}_{\Lambda_{\text{im}}} \sim p(\mathbf{I}_{\Lambda_{\text{im}}} | \mathbf{I}_{\partial\Lambda_{\text{im}}}; \mathbf{F}, \beta). \tag{17}$$

Here, $\mathbf{I}_{\partial\Lambda_{\text{im}}}$ represents the boundary condition of $\mathbf{I}_{\Lambda_{\text{im}}}$, which belongs to the reconstruction of $\mathbf{I}_{\Lambda_{\text{ex}}}$. It leads to seamless boundaries in the synthesis.

By integrating the explicit and implicit representation, the video primal sketch has the following probability model,

$$\begin{aligned} &p(\mathbf{I}|\mathbf{B}, \mathbf{F}, \alpha, \beta) \\ &= \frac{1}{Z} \exp\left\{ -\sum_{i=1}^{n_{\text{ex}}} \sum_{(x,y,t)\in\Lambda_{\text{ex},i}} \frac{(\mathbf{I}(x,y,t) - \alpha_i B_i(x,y,t))^2}{2\sigma_i^2} \right. \\ &\left. - \sum_{j=1}^{n_{\text{im}}} \sum_{k=1}^{K} \langle \beta_{k,j}, H_k(\mathbf{I}_{\Lambda_{\text{im},j}} | \mathbf{I}_{\partial\Lambda_{\text{im},j}}) \rangle \right\}, \end{aligned} \tag{18}$$

where Z is the normalizing constant.

We denote by $\text{VPS} = (\mathbf{B}, \mathbf{H})$ the representation for the video $\mathbf{I}_\Lambda$, where $\mathbf{H} = (\{h_{k,1}\}_{k=1}^{K_1}, \ldots, \{h_{k,n_{\text{im}}}\}_{k=1}^{K_{n_{\text{im}}}})$ includes the histograms described by $\mathbf{F}$ and $\mathbf{V}$; and $\mathbf{B}$ includes all the primitives with parameters for their indexes, position, orientation, and scales. $p(\text{VPS}) = p(\mathbf{B}, \mathbf{H}) = p(\mathbf{B})p(\mathbf{H})$ gives the prior probability of video representation by VPS. $p(\mathbf{B}) \propto \exp\{-|\mathbf{B}|\}$, in which, $|\mathbf{B}|$ is the number of primitives. $p(\mathbf{H}) \propto \exp\{-\gamma_{\text{tex}}(\mathbf{H})\}$, in which, $\gamma_{\text{tex}}(\mathbf{H})$ is the energy term and for instance, $\gamma_{\text{tex}}(\mathbf{H}) = \rho n_{\text{im}}$ to penalize the number of implicit regions. Thus, the best video representation VPS^* is obtained by maximizing the posterior probability,

$$\begin{aligned} \text{VPS}^* &= \arg\max_{\text{VPS}} p(\text{VPS}|\mathbf{I}_\Lambda) = \arg\max_{\text{VPS}} p(\mathbf{I}_\Lambda|\text{VPS}) p(\text{VPS}) \\ &= \arg\max_{\mathbf{B},\mathbf{H}} \frac{1}{Z} \exp\left\{ -\sum_{i=1}^{n_{\text{ex}}} \sum_{(x,y,t)\in\Lambda_{\text{ex},i}} \frac{(\mathbf{I}(x,y,t) - \alpha_i B_i(x,y,t))^2}{2\sigma_i^2} \right. \\ &\left. - |\mathbf{B}| - \sum_{j=1}^{n_{\text{im}}} \sum_{k=1}^{K} \langle \beta_{k,j}, H_k \rangle - \gamma_{\text{tex}}(\mathbf{H}) \right\}. \end{aligned} \tag{19}$$

following the video primal sketch model in (18).

Table 1 shows an example of VPS. For a video of the size of 288×352 pixels, about 30 % of the pixels are represented explicitly by $n_{\text{ex}} = 300$ motion primitives. As each primitive needs 11 parameters (the side length of the patch according to the primitive learning process in Sect. 3.2) to record the profile and 1 more to record the type, the number of total parameters for the explicit representation is 3,600. $n_{\text{im}} = 3$ textured motion regions are represented implicitly by the histograms, which are described by $K_1 = 11$, $K_2 = 12$, and $K_3 = 5$ filters, respectively. As each histogram has 15 bins, the number of the parameters for the implicit representation is 420.

2.6 Sketchability and Trackability for Model Selection

The computation of the VPS involves the partition of the domain Λ into the explicit regions Λ_{ex} and implicit regions Λ_{im}. This is done through the sketchability and trackability maps. In this subsection, we overview the general ideas and refer to previous work on sketchability [18] and trackability [17] for details.

Let us consider one local volume $\Lambda_0 \subset \Lambda$ of the video $\mathbf{I}$. In the video primal sketch model, $\mathbf{I}_{\Lambda_0}$ may be modeled either by the sparse coding model in (5) or by the FRAME model in (11). The choice is determined via the competition between the two models, i.e., comparing which model gives shorter coding length [33] for representation.

If $\mathbf{I}_{\Lambda_0}$ is represented by the sparse coding model, the posterior probability is calculated by

$$p(\mathbf{B}|\mathbf{I}_{\Lambda_0}) = \frac{1}{(2\pi)^{n/2}\sigma^n} \exp\left\{ -\sum_i \frac{\|\mathbf{I}_{\Lambda_0} - \alpha_i B_i\|^2}{2\sigma^2} \right\}. \tag{20}$$

where $n = |\Lambda_0|$. The coding length is

$$\begin{aligned} L_{\text{ex}}(\mathbf{I}_{\Lambda_0}) &= \log \frac{1}{p(\mathbf{B}|\mathbf{I}_{\Lambda_0})} \\ &= \frac{n}{2} \log 2\pi\sigma^2 + \sum_i \frac{\|\mathbf{I}_{\Lambda_0} - \alpha_i B_i\|^2}{2\sigma^2}. \end{aligned}$$

Since σ^2 is estimated via the given data temporarily in real application, $\frac{1}{n}\sum_i \|\mathbf{I}_{\Lambda_0} - \alpha_i B_i\|^2 = \sigma^2$ holds by definition. As a result, the coding length is derived as,

$$L_{\text{ex}}(\mathbf{I}_{\Lambda_0}) = \frac{n}{2}\left(\log 2\pi\sigma^2 + 1\right). \tag{21}$$

If $\mathbf{I}_{\Lambda_0}$ is described by the FRAME model, the posterior probability is calculated by

$$p(\mathbf{F}|\mathbf{I}_{\Lambda_0}) \propto \exp\left\{ -\sum_{k=1}^{K} \langle \beta_k, H_k(\mathbf{I}_{\Lambda_0}) \rangle \right\}. \tag{22}$$

The coding length is estimated through a sequential reduction process. When $K = 0$, with no constraints, the FRAME model is a uniform distribution, and thus the coding length is $\log|\Omega_0|$ where $|\Omega_0|$ is the cardinality of the space of all videos in Λ. Suppose the intensities of the video range from 0 to 255, then $\log|\Omega_0| = 8 \times |\Lambda_0|$. By adding each constraint, the equivalence $\Omega(K)$ will shrink in size, and the ratio of the

compression $\log \frac{|\Omega_{K-1}|}{|\Omega_K|}$ is approximately equal to the information gain in (10). Therefore, we can calculate the coding length by

$$L_{\text{im}}(\mathbf{I}_{\Lambda_0}) = \log|\Omega_0| - \log\frac{|\Omega_0|}{|\Omega_1|} - \cdots - \log\frac{|\Omega_{K-1}|}{|\Omega_K|}. \quad (23)$$

By comparing $L_{\text{im}}(\mathbf{I}_{\Lambda_0})$ and $L_{\text{ex}}(\mathbf{I}_{\Lambda_0})$, whoever has the shorter coding length will win the competition and be chosen for $\mathbf{I}_{\Lambda_0}$.

In practice, we use a faster estimation which utilizes the relationship between the coding length and the entropy of the local posterior probabilities.

Consider the entropy of $p(B|\mathbf{I}_{\Lambda_0})$,

$$\mathcal{H}(B|\mathbf{I}_{\Lambda_0}) = -E_{p(B,\mathbf{I}_{\Lambda_0})}[\log p(B|\mathbf{I}_{\Lambda_0})]. \quad (24)$$

It measures the uncertainty of selecting a primitive in Δ_B for representation. The sharper the distribution $p(B|\mathbf{I}_{\Lambda_0})$ is, the lower the entropy $\mathcal{H}(B_k|\mathbf{I}_{\Lambda_0})$ will be, which gives smaller $L_{\text{ex}}(\mathbf{I}_{\Lambda_0})$ according to (21). Hence, $\mathcal{H}(B_k|\mathbf{I}_{\Lambda_0})$ reflects the magnitude of $L_{\text{diff}}(\mathbf{I}_{\Lambda_0}) = L_{\text{ex}}(\mathbf{I}_{\Lambda_0}) - L_{\text{im}}(\mathbf{I}_{\Lambda_0})$. Set an entropy threshold $\mathcal{H}_0$ on $\mathcal{H}(B_k|\mathbf{I}_{\Lambda_0})$, ideally, $\mathcal{H}(B_k|\mathbf{I}_{\Lambda_0}) = \mathcal{H}_0$ if and only if $L_{\text{diff}}(\mathbf{I}_{\Lambda_0}) = 0$. Therefore, when $\mathcal{H}(B_k|\mathbf{I}_{\Lambda_0}) < H_0$, we consider $L_{\text{ex}}(\mathbf{I}_{\Lambda_0})$ is lower and $\mathbf{I}_{\Lambda_0}$ is modeled by the sparse coding model, else it is modeled by the FRAME model.

It is clear that $\mathcal{H}(B_k|\mathbf{I}_{\Lambda_0})$ has the same form and meaning with sketchability [18] in appearance representation and trackability [17] in motion representation. Therefore, sketchability and trackability can be used for model selection for each local volume. Figure 2b, c shows the sketchability and trackability maps calculated by the local entropy of posteriors. The two maps decide the partition of the video into the explicit implicit regions. Within the explicitly regions, they also decide whether a patch is trackable (using primitives with size of 11×11 pixels $\times 3$ frames) or intrackable (using primitives with 11×11 pixels $\times 1$ frame).

3 Algorithms and Experiments

3.1 Spatio-Temporal Filters

In the vision literature, spatio-temporal filters have been widely used for motion information extraction [1], optical flow estimation [20], multi-scale representation of temporal data [23], pattern categorization [38], and dynamic texture recognition [13]. In the experiments, we choose spatio-temporal filters Δ_F as shown in Fig. 5. It includes three types:

1 **Static filters.** Laplacian of Gaussian (LoG), Gabor, gradient, or intensity filter on a single frame. They capture statistics of spatial features.
2 **Motion filters.** Moving LoG, Gabor or intensity filters in different speeds and directions over three frames. Gabor motion filters move perpendicularly to their orientations.
3 **Flicker filters.** One static filter with opposite signs at two frames. They contrast the static filter responses between two consequent frames and detect the change of dynamics.

For implicit representation, the filters are 7×7 pixels in size and have 6 scales, 12 directions, and 3 speeds. Each type of filter has a special effect in textured motion synthesis, which will be discussed in Sect. 3.3 and shown in Fig. 8.

3.2 Learning Motion Primitives and Reconstructing Explicit Regions

After computing the sketchability and trackability maps of one frame, we extract explicit regions in the video. By calculating all the coefficients of each part with motion primitives from the primitive bank, $\alpha_{i,j} = \langle \mathbf{I}_{\Lambda_{\text{tr},i}}, B_j \rangle$, all the $\alpha_{i,j}$ are ranked from high to low. Each time, we select the primitive with the highest coefficient to represent the corresponding domain and then do local suppression to its neighborhood to avoid excessive overlapping of extracted domains. The algorithm is similar to matching pursuit [25] and the primitives are chosen one by one.

In our work, in order to alleviate computational complexity, $\alpha_{i,j}$ are calculated by filter responses. The filters used here are 11×11 pixels and have 18 orientations and 8 scales. The fitted filter F_j gives a raw sketch of the trackable patch and extracts property information, such as type and orientation, for generating the primitive. If the fitted filter is a Gabor-like filter, the primitive B_j is calculated by averaging the intensities of the patch along the orientation of F_j, while if the fitted filter is a LoG-like filter, B_j is calculated by averaging the intensities circularly around its center. Then B_j is added to the primitive set $\mathbf{B}$ with its motion velocities calculated from the trackability map. It is also added into Δ_B for the dictionary buildup. The size of each primitive is 11×11, the same as the size of the fitted filter. And the velocity (u, v) are two parameters for recording motion information. In Fig. 4, we show some examples of different types of primitives, such as blob, ridge, and edge. Figure 6 shows some examples of reconstruction by motion primitives. In each group, the original local image, the fitted filter, the generated primitive, and the motion velocity are given. In the frame, each patch is marked by a square with a short line for representing its motion information.

Through the matching pursuit process, the sketchable regions are reconstructed by a set of common primitives. Figure 7 shows an example of the sketchable region reconstruction by using a series of common primitives. By comparing the observed frame (a) and reconstructed frame (b), (c) shows the error of reconstruction. The more detailed quan-

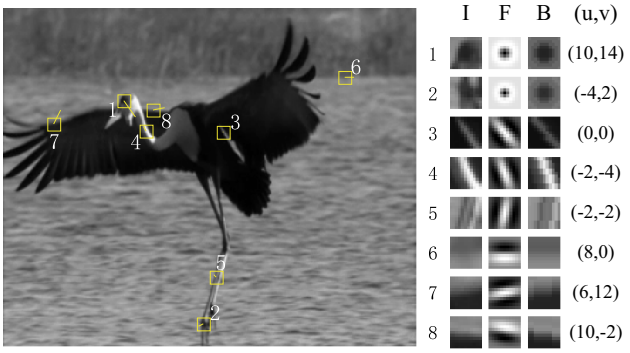


Fig. 6 Some examples of primitives in a frame of video. Each group shows the original local image **I**, the best fitted filter **F**, the fitted primitive $\mathbf{B} \in \Delta_B$ and the velocity (u, v), which represents the motion of **B**

titative assessment is given in Sect. 3.7. It is evident that a rich dictionary of video primitives can lead to a satisfactory reconstruction of explicit regions of videos.

For non-sketchable but trackable regions, based on the trackability map, we get the motion trace of each local trackable patch. Because each patch cannot be represented by a shared primitive, we record the whole patch and motion information as a special primitive for video reconstruction. It is obvious that special primitives increase model complexity compared with common primitives. However, as stated in Sect. 2.1, the percentage of special primitives for the explicit region reconstruction of one video is very small (around 2–3 %), hence it will not affect the final storage space significantly.

3.3 Synthesizing Textured Motions by ST-FRAME

Each local volume $\mathbf{I}_{\Lambda_0}$ of textured motion located at Λ_0 follows a Markov random field model conditioned on its local neighborhood $\mathbf{I}_{\partial\Lambda_0}$ following (9),

$$p(\mathbf{I}_{\Lambda_0}|\mathbf{I}_{\partial\Lambda_0}; \mathbf{F}, \beta) \propto \exp\left\{-\sum_k \langle \beta_k, H_k(\mathbf{I}_{\Lambda_0})\rangle\right\}, \tag{25}$$

where Lagrange parameters $\beta_k = \{\beta_k^{(i)}\}_{i=1}^L \in \beta$ are the discrete form of potential function $\beta_k()$ learned from input videos by maximum likelihood,

$$\begin{aligned}\hat{\beta} &= \arg\min_{\beta} \log p(\mathbf{I}_{\Lambda_0}|\mathbf{I}_{\partial\Lambda_0}; \beta, \mathbf{F}) \\ &= \arg\max_{\beta}\left\{-\log Z(\beta) - \sum_k < \beta_k, H_k(\mathbf{I}_{\Lambda_0}) >\right\}\end{aligned} \tag{26}$$

But the closed form of β is not available in general. So it can be solved iteratively by

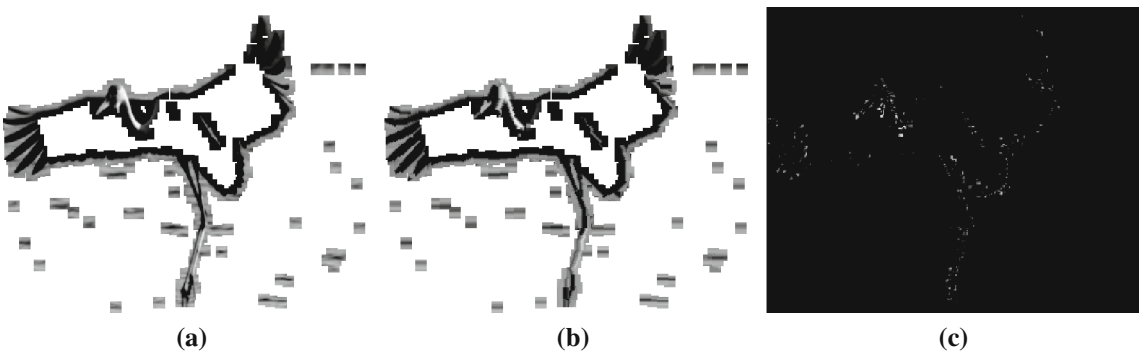
$$\frac{\mathrm{d}\beta^{(i)}}{\mathrm{d}t} = E_{p(\mathbf{I};\beta,\mathbf{F})}[H^{(i)}] - H^{\mathrm{obs}(i)} \tag{27}$$

In order to draw a typical sample frame from $p(\mathbf{I}; \mathbf{F}, \beta)$, we use the Gibbs sampler which simulates a Markov chain. Starting from any random image, e.g., a white noise, it converges to a stationary process with distribution $p(\mathbf{I}; \mathbf{F}, \beta)$. Therefore, we get the final converged results dominated by $p(\mathbf{I}; \beta, \mathbf{F})$, which characterizes the observed dynamic texture.

In summary, the process of textured motion synthesis is given by the following algorithm.

Algorithm 1. Synthesis for Textured Motion by ST-FRAME

Input video $\mathbf{I}^{\mathrm{obs}} = \{\mathbf{I}_{(1)}, \ldots, \mathbf{I}_{(m)}\}$.
Suppose we have $\mathbf{I}^{\mathrm{syn}} = \{\mathbf{I}^{\mathrm{syn}}_{(1)}, \ldots, \mathbf{I}^{\mathrm{syn}}_{(m-1)}\}$, our goal is to synthesize the next frame $\mathbf{I}^{\mathrm{syn}}_{(m)}$.
Select a group of spatio-temporal filters from a filter bank $\mathbf{F} = \{F_k\}_{k=1}^K \in \Delta_F$.
Compute $h_k, k = 1, \ldots, K$ of $\mathbf{I}^{\mathrm{obs}}$.
Initialize $\beta_k^{(i)} \leftarrow 0, k = 1, \ldots, K, i = 1, \ldots, L$.
Initialize $\mathbf{I}^{\mathrm{syn}}_{(m)}$ as a uniform white noise image.
Repeat
  Calculate $h_k^{\mathrm{syn}}, k = 1, 2, \ldots, K$ from $\mathbf{I}^{\mathrm{syn}}$.
  Update $\beta_k, k = 1, \ldots, K$ and $p(\mathbf{I}; \mathbf{F}, \beta)$.
  Sample $\mathbf{I}^{\mathrm{syn}}_{(m)} \sim p(\mathbf{I}; \mathbf{F}, \beta)$ by Gibbs sampler.
Until $\frac{1}{2}\sum_{i=1}^L |h_k^{(i)} - h_k^{\mathrm{syn}(i)}| \le \epsilon$ for $k = 1, 2, \ldots, K$.



(a) **(b)** **(c)**

Fig. 7 The reconstruction effect of sketchable regions by common primitives. **a** The observed frame. **b** The reconstructed frame. **c** The errors of reconstruction

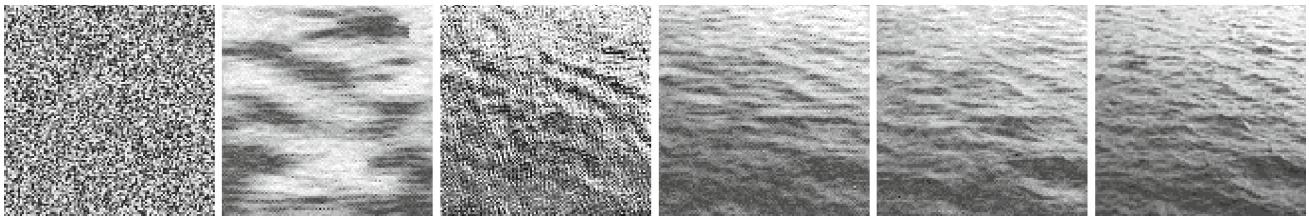


Fig. 8 Synthesis for one frame of the ocean textured motion. **a** Initial uniform white noise image. **b** Synthesized frame with only static filters. **c** Synthesized frame with only motion filters. **d** Synthesized frame with both of static and motion filters. **e** Synthesized frame with all of the 3 types of filters. **f** The original observed frame

Figure 8 shows an example of the synthesis process. (f) is one frame from textured motion of ocean. Starting from a white noise frame in (a), (b) is synthesized with only 7 static filters. It shows high smoothness in spatial domain, but lacks temporal continuity with previous frames. However, in (c) the synthesis with only 9 motion filters has similar macroscopic distribution to the observed frame, but appears quite grainy over local spatial relationship. By using both static and motion filters, the synthesis in (d) performs well on both spatial and temporal relationships. Compared with (d), the synthesis by 2 extra flicker filters in (e) shows more smoothness and more similar to the observed frame.

In Fig. 9, we show four groups of textured motion (4 bits) synthesis by Algorithm 1: ocean (a), water wave (b), fire (c), and forest (d). In each group, as time passes, the synthesized frames are getting more and more different from the observed one. It is caused by the stochasticity of textured motions. Although the synthesized and observed videos are quite different on pixel level, the two sequences are perceived extremely identical by human after matching the histograms of a small set of filter responses. This conclusion can be further supported by perceptual studies in Sect. 3.9. Figure 10 shows that as $\mathbf{I}^{\text{syn}}_{(m)}$ changes from white noise (Fig. 8a) to the final synthesized result (Fig. 8e), the histograms of filter responses become matched with the observed ones.

Table 2 shows the comparison of compression ratios between ST-FRAME and the dynamic texture model [14]. It has a significantly better compression ratio than the dynamic texture model, because the dynamic texture model has to record PCA components as large as the image size.

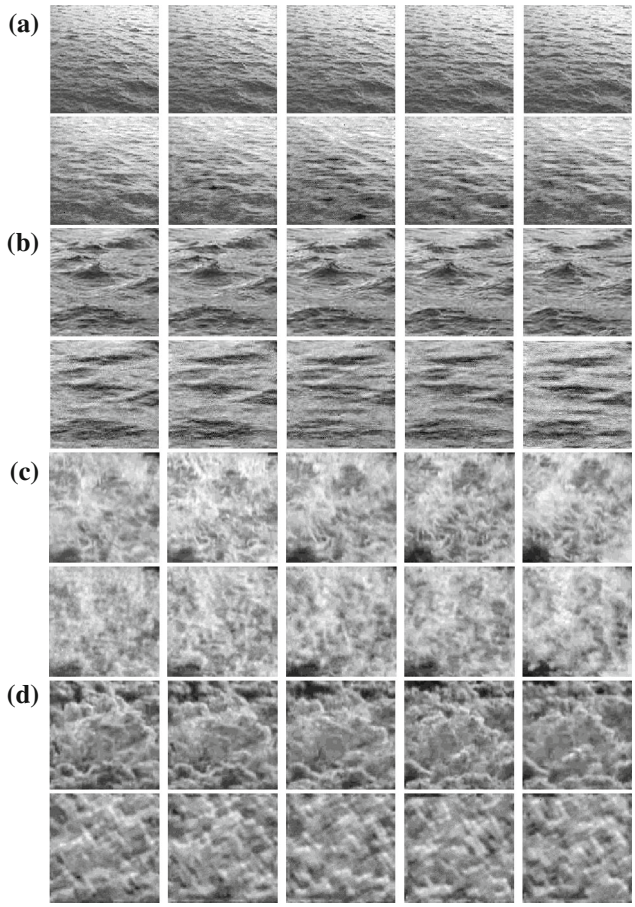


Fig. 9 Textured motion synthesis examples. For each group, the *top row* is the original videos and the *bottom row* shows the synthesized ones. **a** Ocean. **b** Water wave. **c** Fire. **d** Forest

3.4 Computing Velocity Statistics

One popular method for velocity estimation is optical flow. Based on the optical flow, HOOF features extract the motion statistics by calculating the distribution of velocities in each region. Optical flow is an effective method for estimating the motions at trackable areas, but does not work for the intrackable dynamic texture areas. The three basic assumptions for optical flow equations, i.e., brightness constancy between matched pixels in consecutive frames, smoothness among adjacent pixels and slow motion, are violated in these areas due to the stochastic nature of dynamic textures. Therefore, we go for a different velocity estimation method.

Considering one pixel $I(x, y, t)$ at (x, y) in frame t, we denote its neighborhood as $I_{\partial \Lambda_{x,y,t}}$. Comparing patch $I_{\partial \Lambda_{x,y,t}}$ with all the patches in the previous frame within a searching radius, each patch corresponding to one velocity $v = (v_x, v_y)$, we obtain a distribution

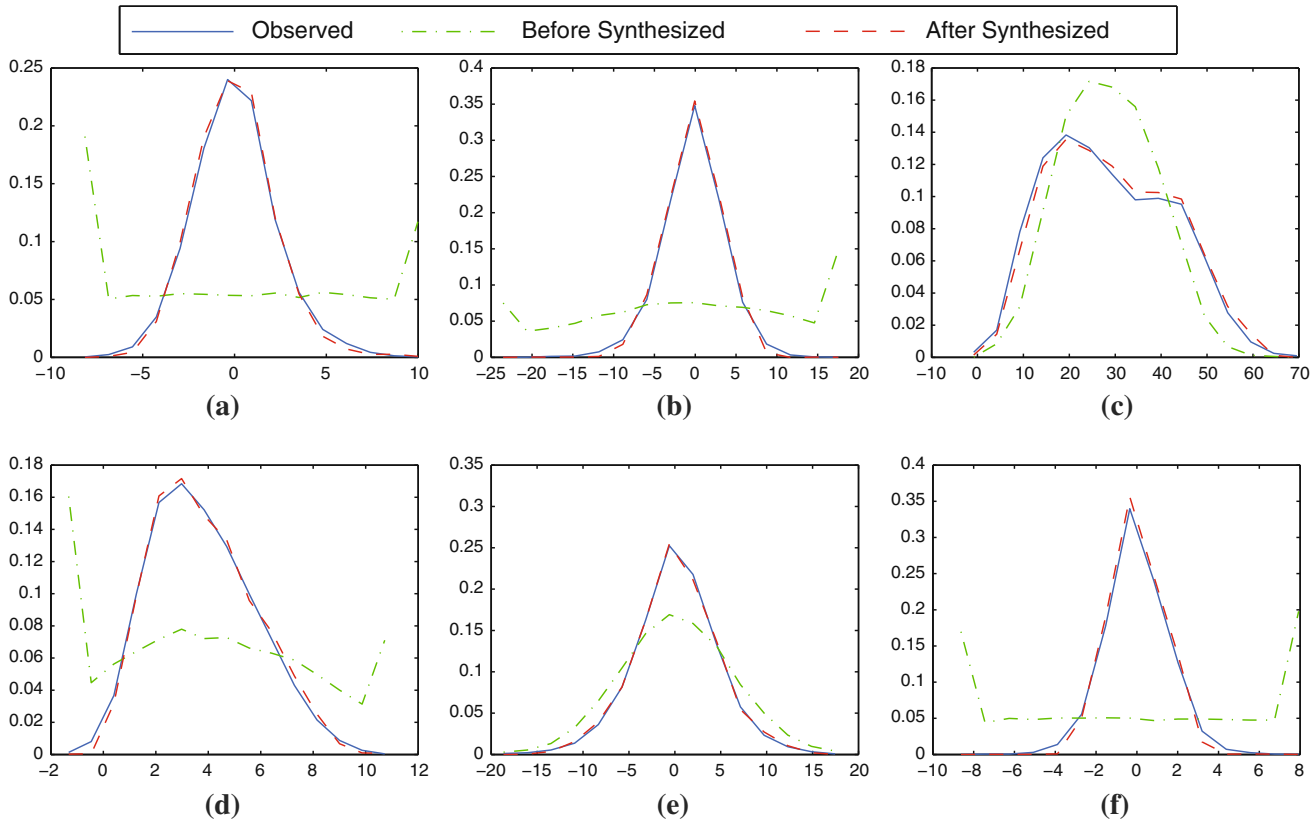


Fig. 10 Matching of histograms of spatio-temporal filter responses for Ocean. The filters are **a** Static LoG (5 × 5). **b** Static gradient (vertical). **c** Motion gabor (6,150°). **d** Motion gabor (2,60°). **e** Motion gabor (2,0°). **f** Flicker LoG (5 × 5)

Table 2 The number of parameters recorded and the compression ratios for synthesis of 5-frame textured motion videos by ST-FRAME and the dynamic texture model ([14])

Example	Size	ST-FRAME	Dynamic texture
Ocean	112 × 112 × 5	558 (0.89 %)	25,096 (40.01 %)
Water wave	105 × 105 × 5	465 (0.84 %)	22,058 (40.01 %)
Fire	110 × 110 × 5	527 (0.87 %)	24,210 (40.02 %)
Forest	110 × 110 × 5	465 (0.77 %)	24,210 (40.02 %)

$$p(v) \propto \exp\left\{-\|I_{\partial\Lambda_{x,y,t}} - I_{\partial\Lambda_{x-v_x,y-v_y,t-1}}\|^2\right\} \tag{28}$$

This distribution describes the probability of the origin of the patch, i.e., the location where the patch $I_{\partial\Lambda_{x,y,t}}$ moves from. Equivalently, it reflects the average probability of the motions of the pixels in the patch. Therefore, by clustering all the pixels according to their velocity distribution, the cluster center of each cluster gives the velocity statistics of all the pixels in this cluster approximately, which reflects the motion pattern of these clustered pixels. Figures 14 and 15 show some examples of velocity statistics, in which the brighter, the higher probability, while the darker, the lower probability. The meanings of these two figures are explained later.

Compared to HOOF, the estimated velocity distribution is more suitable for modeling textured motion. Firstly, the velocity distribution is estimated pixel wisely. Hence it can depict more non-smooth motions. Secondly, although it seeks to compare the intensity pattern around a point to nearby regions at a subsequent temporal instance, which seems to also take brightness constancy assumption into account, the difference here is that it calculates the probability of motions rather than the single pixel correspondence. As a result, the constraints by the assumption are weakened, and it has the ability to represent stochastic dynamics.

3.5 Synthesizing Textured Motions by MA-FRAME

In MA-FRAME model, similar to ST-FRAME, each local volume I_{Λ_0} of textured motion follows a Markov random field model. However, the difference is that MA-FRAME extracts motion information via the distribution of velocities v.

In the sampling process, $I_{\Lambda_{\text{im},j}}$ and $v_{\Lambda_{\text{im},j}}$ are sampled simultaneously following the joint distribution in (15),

$$p(\mathbf{I}_{\Lambda_{\text{im},j}}; \mathbf{F}, \beta) \propto \exp\left\{-\langle \boldsymbol{\beta}, \mathbf{H}(\mathbf{I}_{\Lambda_{\text{im},j}}, \mathbf{v}_{\Lambda_{\text{im},j}})\rangle\right\}.$$

In experiments, we design an effective way for sampling from the above model. For each pixel, we build a 2D-distribution matrix, whose two dimensions are velocities and intensities, respectively, to guide the sampling process.

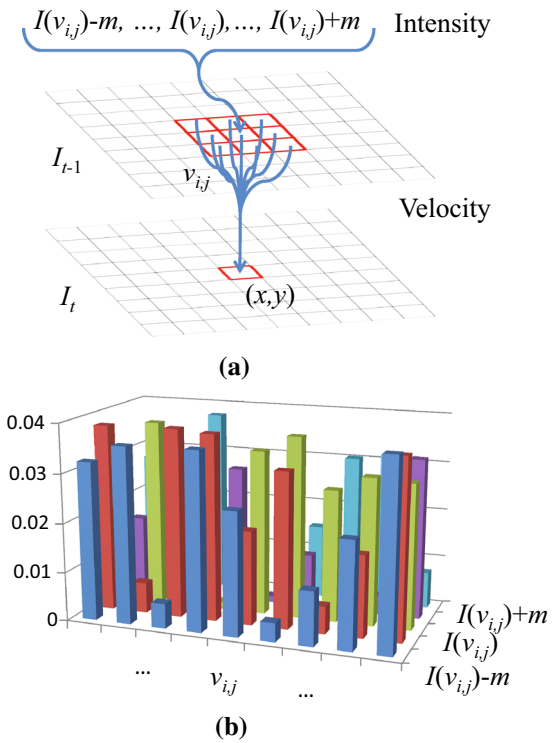


Fig. 11 Sampling process of MA-FRAME. **a** For each pixel of current frame $I_t(x, y)$, the sample candidates are perturbation intensities of its neighborhood pixels in previous frame I_{t-1} dominated by different velocities. **b** The velocity list and intensity perturbations construct two dimensions of the 2D distribution matrix, which is used for sampling $I_t(x, y)$. Here, $I(v_{i,j})$ is short for $I_{t-1}(x - v_{x,i}, y - v_{y,j})$ and i, j are indexes for different velocities in the neighborhood area

The sampling probability for every candidate (labeled by one velocity and one intensity) is obtained by integrating motion score, appearance score, and multiplying smoothness weight,

$$\text{SCORE}(v, I) \propto \exp\Big\{-\omega(v)(\langle\beta^{(t)}, \|H^{(t)} - H^{(t)\text{obs}}\|^2\rangle + \langle\beta^{(s)}, \|H^{(s)} - H^{(s)\text{obs}}\|^2\rangle)\Big\}.$$

The details are explained with the illustration by Fig. 11 for the sampling method at one pixel. For each pixel (x, y) of the current frame I_t, we consider its every possible velocity $v_{i,j} = (v_{x,i}, v_{y,j})$ within the range $-v_{max} \le v_{x,i}, v_{y,j} \le v_{\max}$. Each velocity corresponds to a position $(x - v_{x,i}, y - v_{y,j})$ in the previous frame I_{t-1}. Under velocity $v_{i,j}$, the perturbation range of $I_{t-1}(x - v_{x,i}, y - v_{y,j})$ yields the intensity candidates for $I_t(x, y)$ which is a smaller interval than [0, 255] and thus saves computational complexity. In the shown example (Fig. 11a), $v_{max} = 1$ and the perturbed intensity range is $[I_{t-1}(x - v_{x,i}, y - v_{y,j}) - m, I_{t-1}(x - v_{x,i}, y - v_{y,j}) + m]$. Therefore, we have 9 velocity candidates and $2m+1$ intensity candidates for each velocity, hence the size of the sampling matrix is $9 \times (2m + 1)$ (Fig. 11b). With the motion constraints given by matching the velocity statistics, the velocity candidates have their motion scores. With the appearance constraints given by matching the filter response histograms, intensity candidates have their appearance scores. By integrating the two sets of scores, we obtain a preliminary sampling matrix shown in Fig. 11b.

In order to guarantee the motion of each pixel is as consistent as possible with its neighborhoods to make the macroscopic motion smooth enough, we add a set of weights on the distribution matrix, in which each multiplier for candidates of one velocity is calculated by

$$\omega(v_{x,i}, v_{y,j}) = \sum_{(x_k, y_k) \in \partial\Lambda_{(x,y)}} \|v_{x,i} - v_{x_k}\|^2 + \|v_{y,j} - v_{y_k}\|^2.$$

The weights encourage the velocity candidate which is closer to the velocities of its neighbors. With the weights, the sampled velocities are prone to be regarded as "blurred" optical flow. The main difference is that it preserves the uncertainty of dynamics in a texture motion, but not definite velocities of every local pixel.

After multiplying the weights to the preliminary matrix, we get the final sampling matrix. Although the main purpose of MA-FRAME is sampling intensities of each pixel from a textured motion, the sampling for intensities is highly related to velocities, and the sampling process is actually based on the joint distribution of velocity and intensity.

In summary, textured motion synthesis by MA-FRAME is given as follows

Algorithm 2. Synthesis for Textured Motion by MA-FRAME

Input video $\mathbf{I}^{\text{obs}} = \{\mathbf{I}_{(1)}, \ldots, \mathbf{I}_{(n)}\}$.
Suppose we have $\mathbf{I}^{\text{syn}} = \{\mathbf{I}^{\text{syn}}_{(1)}, \ldots, \mathbf{I}^{\text{syn}}_{(m-1)}\}$, our goal is to synthesize the next frame $\mathbf{I}^{\text{syn}}_{(m)}$.
Select a group of static and flicker filters from a filter bank $\mathbf{F} = \{F_k\}_{k=1}^{K_s} \in \Delta_F$, where K_s is the number of selected filters.
Compute $\{h_k^{(s)}, k = 1, \ldots, K_s\}$, $\{h_k^{(v)}, k = 1, \ldots, K_v\}$ of $\mathbf{I}^{\text{obs}}$, where K_v is the number of velocity clusters.
Initialize $\beta_k^{(i)} \leftarrow 0, k = 1, \ldots, K, i = 1, \ldots, L$.
Initialize velocity vector $V = (v(x, y))_{(x,y)\in\Lambda}$ uniformly, and initialize $\mathbf{I}^{\text{syn}}_{(m)}$ by choosing intensities based on V.
Repeat
  Calculate $\{h_k^{\text{syn}(s)}, k = 1, 2, \ldots, K_s\}$ and $\{h_k^{\text{syn}(v)}, k = 1, 2, \ldots, K_v\}$ from $\mathbf{I}^{\text{syn}}$.
  Update $\beta_k, k = 1, \ldots, K$ and $p(\mathbf{I}; \mathbf{F}, \beta)$.
  Sample $(\mathbf{I}^{\text{syn}}_{(m)}, \mathbf{V}^{\text{syn}}_{(m)}) \sim p(\mathbf{I}, \mathbf{V}; \mathbf{F}, \beta)$ by Gibbs sampler.
Until $\frac{1}{2}\sum_{i=1}^{L} |h_k^{(i)} - h_k^{\text{syn}(i)}| \le \epsilon$ for $k = 1, 2, \ldots, K_s + K_v$.

Figures 12 and 13 show two examples of textured motion synthesis by MA-FRAME. Different from the synthesis results by ST-FRAME, it can deal with videos of larger size,

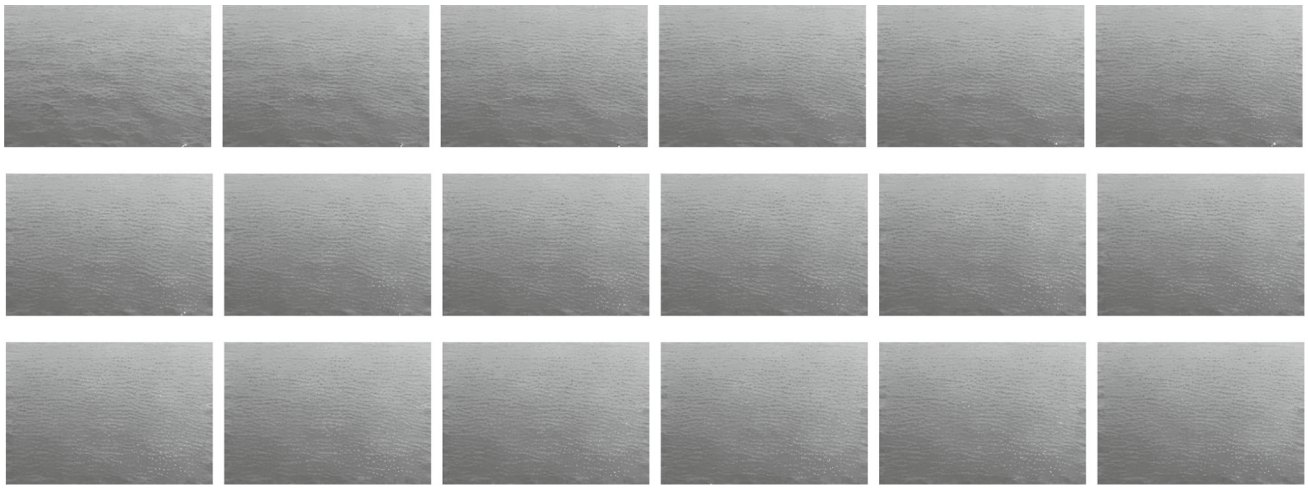


Fig. 12 Texture synthesis for 18 frames of ocean video (from *top-left* to *bottom-right*) by MA-FRAME

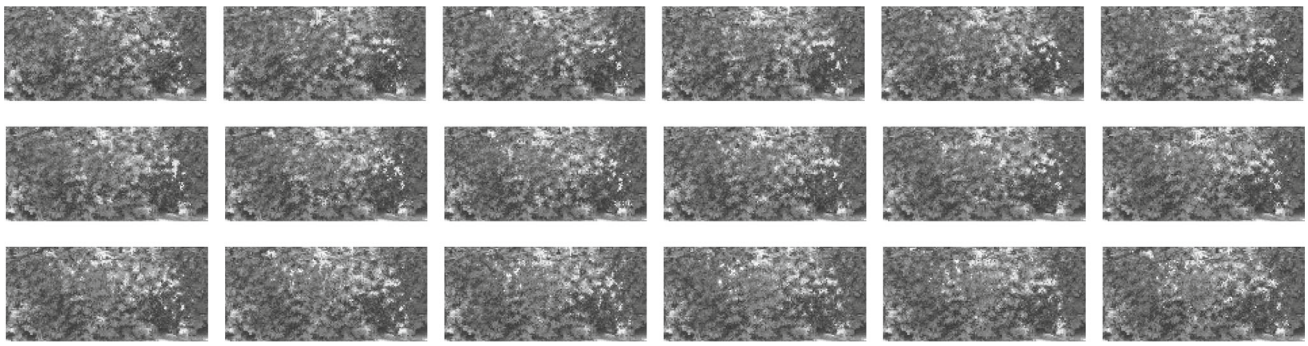


Fig. 13 Texture synthesis for 18 frames of bushes video (from *top-left* to *bottom-right*) by MA-FRAME

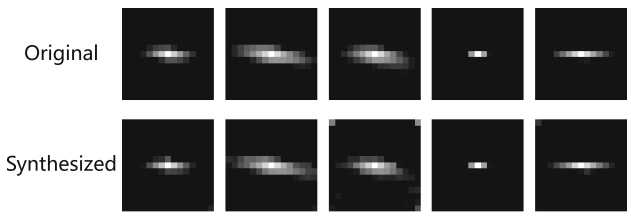


Fig. 14 Five pairs of global statistics of velocities for comparison. Each patch corresponds to the neighborhood lattices as shown in Fig. 11a. The brighter means higher motion probability, while the darker means lower probability

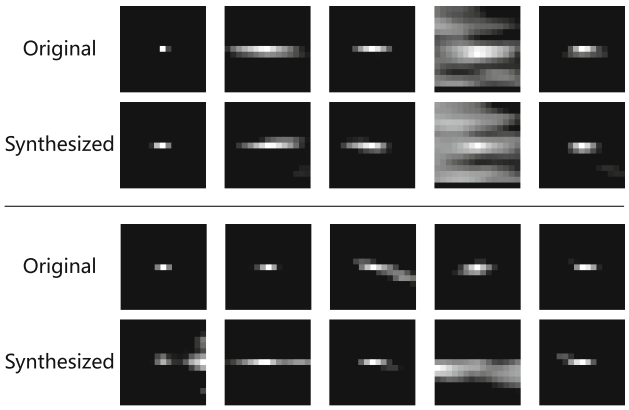


Fig. 15 Ten pairs of local statistics of velocities for comparison. Upper row: original; lower row: synthesis. Each patch corresponds to the neighborhood lattices as shown in Fig. 11a. The brighter means higher motion probability, while the darker means lower probability

higher intensity level (8 bits here compared to 4 bits in ST-FRAME experiments) and more frames because of its smaller sample space and higher temporal continuity. Furthermore, it generates better motion pattern representations.

Figure 14 shows the comparison of velocities statistics between the original video and the synthesized video of different textured motion clusters, the brighter, the higher motion probability, while the darker, the lower probability. It is easy to tell that they are quite consistent, which means the original and synthesized videos have similar macroscopical motion properties.

We also test local motion consistency between observed and synthesized videos by comparing velocity distributions

of every pair of corresponding pixels. Figure 15 shows the comparisons of ten pairs of randomly chosen pixels. Most of them match well. It demonstrated that the motion distributions of most of local patches also preserve well during the synthesis procedure.

3.6 Dealing with Occlusion Parts in Texture Synthesis

Before providing the full version of computational algorithm for VPS, we first introduce how to deal with occluded areas.

In video, dynamic background textures are often occluded by the movement of foreground objects. Synthesizing background texture by ST-FRAME uses histograms of spatio-temporal filter responses. When a textured region becomes occluded, the pattern no longer belongs to the same equivalence class. In this event, the spatio-temporal responses are not precise enough for matching the given histograms, and may cause a deviation in the synthesis results. These errors may accumulate over frames and the synthesis will ultimately degenerate completely. Synthesis by MA-FRAME has a greater problem because the intensities in the current frame are selected from small perturbations in intensities from the previous frame. If a pixel cannot be found from the neighborhood in the previous frame that belongs to the same texture class, the intensity it adopts may be incompatible with other pixels around it.

In order to solve this problem, occluded pixels are sampled separately by the original (spatial) FRAME model, which means, we have two classes of filter response histograms

1. Static filter response histograms H^S. Histograms are calculated by summarizing static filter responses of all the textural pixels;
2. Spatio-temporal filter response histograms H^{ST}. Histograms are calculated by summarizing spatio-temporal filter responses of all the non-occluded textured pixels.

Therefore, in the sampling process, the occluded pixels and non-occluded pixels are treated differently. First, their statistics are constrained by different sets of filters; second, in MA-FRAME, the intensities of non-occlude pixels are sampled from the intensity perturbation of their neighborhood locations in previous frame, while the intensities of occluded pixels are sampled from the whole intensity space, say 0–255 for 8 bits gray levels.

3.7 Synthesizing Videos with VPS

In summary, the full version of the computational algorithm for video synthesis of VPS is presented as follows.

Algorithm 3. Video Synthesis via Video Primal Sketch

Input a video $\mathbf{I}^{\text{obs}} = \{\mathbf{I}^{\text{obs}}_{(1)}, \ldots, \mathbf{I}^{\text{obs}}_{(m)}\}$.

Compute sketchability and trackability for separating $\mathbf{I}^{\text{obs}}$ into explicit region $\mathbf{I}_{\Lambda_{\text{ex}}}$ and implicit region $\mathbf{I}_{\Lambda_{\text{im}}}$.

for t=1:m

Reconstruct $\mathbf{I}^{\text{obs}}_{(t)\Lambda_{\text{ex}}}$ by the sparse coding model with the selected primitives $\mathbf{B}$ chosen from the dictionary Δ_B to get $\mathbf{I}^{\text{syn}}_{(t)\Lambda_{\text{ex}}}$.

For each region of homogeneous textured motion $\Lambda_{\text{im},j}$, using $\mathbf{I}^{\text{syn}}_{(t)\Lambda_{\text{ex}}}$ as boundary condition, synthesize $\mathbf{I}^{\text{obs}}_{(t)\Lambda_{\text{im},j}}$ by ST-FRAME model or MA-FRAME with the selected filter set $\mathbf{F}$ chosen from the filter bank Δ_F to get $\mathbf{I}^{\text{syn}}_{(t)\Lambda_{\text{im}}}$.

The synthesis of the ith frame of the video $\mathbf{I}^{\text{syn}}_{(t)}$ is given by aligning $\mathbf{I}^{\text{syn}}_{(t)\Lambda_{\text{ex}}}$ and $\mathbf{I}^{\text{syn}}_{(t)\Lambda_{\text{im}}}$ together seamlessly.

end for

Output the synthesized video $\mathbf{I}^{\text{syn}}$.

Figure 2 shows this process as we introduced in Sect. 1. Figure 16 shows three examples of video synthesis (YCbCr color space, 8 bits for gray level) by VPS frame by frame. In every experiment, observed frames, trackability maps, and final synthesized frames are shown. In Table 3, H.264 is selected as the reference of compression ratio compared with VPS, from which we can tell VPS is competitive with state-of-art video encoder on video compression.

For assessing the quality of the synthesized results quantitatively, we adopt two criteria for different representations, rather than the traditional approach based on error-sensitivity as it has a number of limitations [37]. The error for explicit representations is measured by the difference of pixel intensities

$$\text{err}^{\text{ex}} = \frac{1}{|\Lambda_{\text{ex}}|} \sum_{\Lambda_{\text{ex}}} \left\| I^{\text{syn}} - I^{\text{obs}} \right\|, \tag{29}$$

while for implicit representations, the error is given by the difference of filter response histograms,

$$\text{err}^{\text{im}} = \frac{1}{n_{\text{im}} \times K} \sum_{n_{\text{im}}, K} \left\| H_k\left(I^{\text{syn}}_{\Lambda_{\text{im},j}}\right) - H_k\left(I^{\text{obs}}_{\Lambda_{\text{im},j}}\right) \right\|. \tag{30}$$

Table 4 shows the quality assessments of the synthesis, which demonstrates good performance of VPS on synthesizing videos.

3.8 Computational Complexity Analysis

In this subsection, we analyze the computational complexity of the algorithms studies in this paper. We discuss the complexity for four algorithms in the following. The implementation environment is the desktop computer with Intel Core i7 2.9 GHz CPU, 16GB memory and Windows 7 operating system.

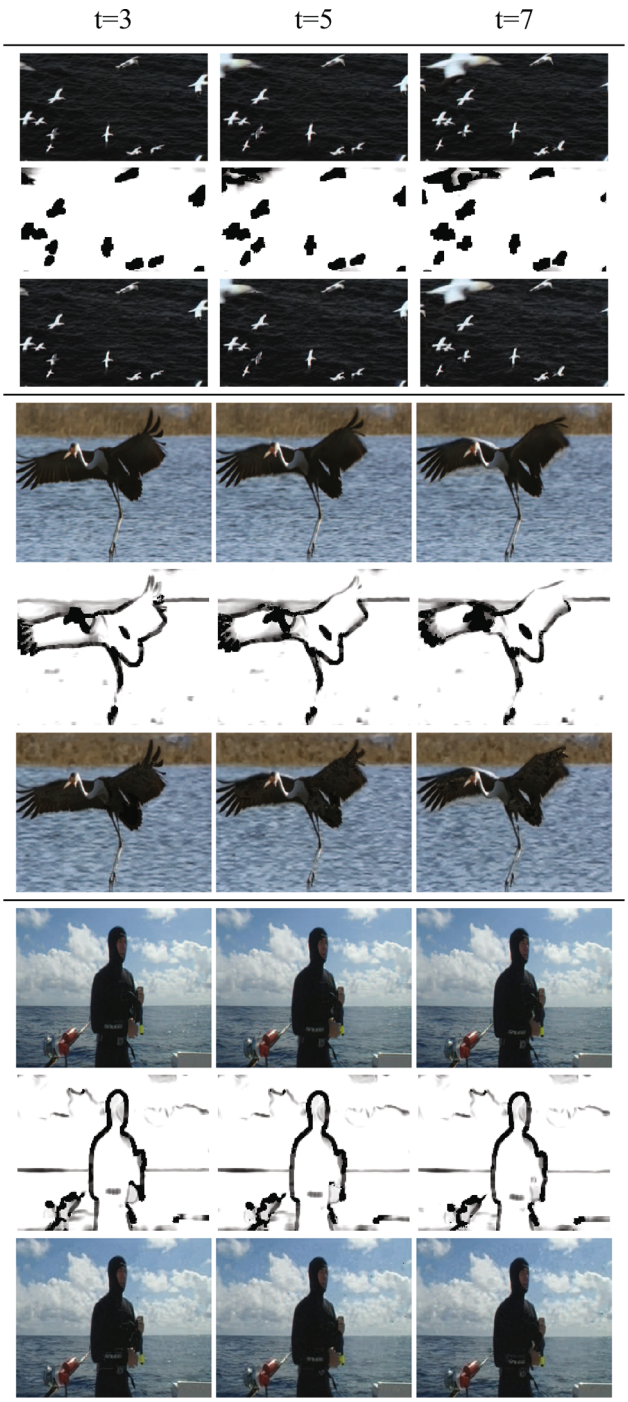


Fig. 16 Video synthesis. For each experiment, *Row 1* original frames; *Row 2* trackability maps; *Row 3* synthesized frames

Table 3 Compression ratio of video synthesis by VPS and H.264 to raw image sequence

Example	Raw (Kb)	VPS (Kb)	H.264 (Kb)
1	924	16.02 (1.73 %)	20.8 (2.2 %)
2	1,485	26.4 (1.78 %)	24 (1.62 %)
3	1,485	28.49 (1.92 %)	18 (1.21 %)

Table 4 Error assessment of synthesized videos

Example	Size (pixels)	Error ($I_{\Lambda_{\text{ex}}}$)	Error ($I_{\Lambda_{\text{im}}}$)
1	$190 \times 330 \times 7$	5.37 %	0.59 %
2	$288 \times 352 \times 7$	3.07 %	0.16 %
3	$288 \times 352 \times 7$	2.8 %	0.17 %

1) Video modeling by VPS. Suppose one frame of a video contains N pixels, of which, N_{ex} pixels belong to explicit regions and N_{im} in implicit regions. Let the size of the filter dictionary be N_F and the filter size be S_F, the computational complexity for calculating filter responses is $O(NN_FS_F)$. For extracting and learning explicit bricks, the complexity is no more than $O(N_{\text{ex}}\sqrt{S_F})$. For calculating the response histograms of K chosen filters within the implicit regions, the complexity is no more than $O(N_{\text{im}}Kk)$ if there are k homogeneous textural area in the regions. To sum up, the total computation complexity for video coding is no more than $O(NN_FS_F + N_{\text{ex}}\sqrt{S_F} + N_{\text{im}}Kk)$. In our experiments, for coding one frame of the video with the size of 288×352, the time consumption is less than 0.5 seconds.

2) Reconstruction of explicit regions. Because the information of all the basis for explicit regions is recorded and there needs no additional computations for reconstructing, the computational complexity can be regarded as $O(1)$ and the reconstruction costs no time in comparison to other components.

3) Synthesis of implicit regions by Gibbs sampling by ST-FRAME. For one round sampling, each of the N_{im} pixels will be sampled in the range of the overall intensity levels, say L. For every sampling candidate, i.e., one intensity, the score is calculated via the change of synthesized filter response histograms. To reduce the computation burden, we can simply update the change of filter responses caused by the change of the intensity on the current pixel. This operation requires KS_F times of multiplications. As a result, the computational complexity for one round sampling of one frame is $O(N_{\text{im}}LKS_F)$. In the experiments of this paper, one frame will be sampled for about 20 rounds. Then the running time is about 2 min if the image is 4 bits and the size of implicit region is 10,000 pixels.

4) Synthesis of implicit regions by Gibbs sampling by MA-FRAME. The computational complexity of MA-FRAME is quite similar with ST-FRAME. The biggest difference is the number of sampling candidates. As the number of velocity candidates is N_v and the intensity perturbation range is $[-m, m]$, the computational complexity is $O(N_{\text{im}}N_vmKS_F)$, which is on the same level with ST-FRAME. However, in real application, because the intensities of the neighborhood of one pixel are not far away, the intensities of the candidates with different velocities are quite

redundant. As a result, MA-FRAME may save a lot of time compared with ST-FRAME, especially when the intensity level is high. For one frame with 8 bits and 60, 000 pixels, the running time is about 4 minutes within 20 rounds sampling.

In summary, the computational complexity of video modeling / coding by VPS is small, but that of video synthesis is quite large. It is because of texture synthesis procedure. In VPS, the textures are modeled by MRF and synthesized via a Gibbs sampling process, which is well known as a computational costing method. However, the video synthesis is only one of the applications of VPS and is used for verifying the correctness of the model. As a result, it is not the very important issue we care about here.

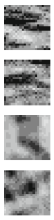
3.9 Perceptual Study

The error assessment of VPS is consistent with human perception. To support this claim, in this subsection, we present a series of human perception experiments and explore the relationship between perception accuracy. In the experiments below, the 30 participants include graduate students and researchers from mathematics, computer science, and medical science. The age range is from 22 to 39, and they all have normal or corrected-to-normal vision.

In the first experiment, we randomly crop several clips of videos with different sizes from the four synthesized textured motion examples and their corresponding original videos (as shown in the left side of Figs. 17, 18 and 19, each video is shown one frame as an example which is marked by (a), (b), (c), and (d), respectively, and they are in different sizes but shown in the same size after zooming for better shows). And then for original and synthesized examples, respectively, each participant is shown 40 clips one by one (10 clips from each texture) and is required to guess which texture they come from. We show 3 representative groups of results below for demonstration, in which the sizes of cropped examples are 5×5, 10×10, and 20×20, respectively. Both of the confusion rates (%) of original and synthesized examples are shown in the tables on the right side in Figs. 17, 18, and 19. Each row gives the average confusion rates, which the video clip labeled by the row title is judged coming from textures labeled by the column titles. In order to test if the syntheses are perceived the same with the original videos, we compare the original and synthesis confusion tables in each group. From the results, we can tell that the confusion tables are mostly consistent. For more precise quantitative estimation, we also analyze the recognition accuracies by ANOVA in Table 5, in which, each row shows the corresponding F and p values for each texture in all the three groups. The results show that the recognition accuracies on original and synthesized textures do not differ significantly.

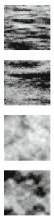
	Original					Synthesis				
		(a)	(b)	(c)	(d)		(a)	(b)	(c)	(d)
(a)	(a)	44	37	6.7	12.3	(a)	42	36	7	15
(b)	(b)	32	45.3	8	14.7	(b)	32.7	43.3	8.7	15.3
(c)	(c)	9.7	7	44	39.3	(c)	10	7.7	44.3	38
(d)	(d)	10.3	12.3	35	42.3	(d)	10.7	13.7	35.3	40.3

Fig. 17 Perceptual confusion test on original and synthesized textured motions, respectively. The size of cropped examples is 5×5



	Original					Synthesis				
		(a)	(b)	(c)	(d)		(a)	(b)	(c)	(d)
(a)	(a)	54.3	36.3	2	7.3	(a)	53	36	3	8
(b)	(b)	32	56	2.7	9.3	(b)	32	54.3	3.7	10
(c)	(c)	3.7	1.7	59	35.7	(c)	3.3	3	59.7	34
(d)	(d)	2.7	13.7	34	49.7	(d)	3	15.7	33.7	47.7

Fig. 18 Perceptual confusion test on original and synthesized textured motions, respectively. The size of cropped examples is 10×10



	Original					Synthesis				
		(a)	(b)	(c)	(d)		(a)	(b)	(c)	(d)
(a)	(a)	82	15	0.7	2.3	(a)	81.7	14	1	3.3
(b)	(b)	13.3	84.3	0.3	2	(b)	14	83.3	0.7	2
(c)	(c)	2.3	0.7	77.3	19.7	(c)	2	0.7	77.7	19.7
(d)	(d)	1.3	3.7	16.7	78.3	(d)	1.7	4.3	16.7	77.3

Fig. 19 Perceptual confusion test on original and synthesized textured motions, respectively. The size of cropped examples is 20×20

Table 5 The ANOVA results of analyzing recognition accuracies of original and synthesized textures

F/p	Group 1	Group 2	Group 3
(a)	1.34/0.2520	0.65/0.4222	0.02/0.8813
(b)	0.96/0.3305	0.70/0.4065	0.20/0.6583
(c)	0.06/0.8100	0.15/0.6993	0.03/0.8563
(d)	1.43/0.2366	1.08/0.3088	0.26/0.6151

For each texture in every group, the corresponding F and p values are shown, respectively

Also, it is noted that texture (a) and (b) appear similarly, while (c) and (d) tend to be confused with each other. Therefore, the confusion rates between (a) and (b), (c), and (d) are apparently larger. However, from Figs. 17 to 19, as the size

of cropped videos gets larger, the confusion rate becomes lower, and actually when the size goes larger than 25×25 in this experiment, the accuracies get very close to 100 %. This experiment demonstrates the fact that the dynamic textures synthesized by the statistics of dynamic filters can be well discriminated by human vision, although the synthesized one and the original one are totally different on pixel level. Therefore, it is evident that the approximation of filter response histograms reflects the quality of video synthesis. Furthermore, it is proved that larger area textures give much better perception effect because human can extract more macroscopic statistical information and motion-appearance characteristics, while small size local areas can only provide salient structural information which may be shared by a various of different videos.

In the second experiment, we test if the synthesized video by VPS gives similar vision impact compared with the original video. Each time we provide the original and the synthesized videos to one participant in the same scale. The videos are played synchronously and the participants are required to point out which is the original video in 5 seconds. Each pair of videos is tested in four scales, 100, 75, 50, and 25 %. The accuracy is shown in Table 6. From the result, when the videos are shown in larger scales, it is easier to discriminate the original and synthesized videos, because a lot of structural details can be noticed by the observers. But as the scale gets smaller, the macroscopic information gives the major impact to the vision system, therefore, the original and synthesized video are perceived almost the same, so that the accuracy gets lower and approaches to 50 %. From this experiment, it is evident that although VPS cannot give the complete reconstruction of a video on pixel level, especially for dynamic textures, but the synthesis gives human similar vision impact, which means most of the key information for perception are kept via VPS model.

3.10 VPS Adapting Over Scales, Densities and Dynamics

As it was observed in [17] that the optimal visual representation at a region is affected by distance, density, and dynamics. In Fig. 20, we show four video clips from a long video sequence. As the scale changes from high to low over time, the birds in the videos are perceived by lines of boundary, groups of kernels, dense points, and dynamic textures, respectively. We show the VPS of each clip and demonstrate that the proper representations are chosen by the model. Figure 21 shows the types of chosen primitives for explicit representations, in which circles represent blob-like type, while short lines represent edge-like type primitives. Table 7 gives corresponding comparisons between the number of blob-like and edge-like primitives in each scale. For each scale, the comparison is within first 50, 100, 150, and 200 chosen primitives, respectively. It is quite obvious that the percentage of chosen edge-like primitives in large scale frame is much higher than that in small scale. Meanwhile, in large scale frame, the blob-like primitives start to appear very late, which shows the fact that edge-like primitives are much more important in this scale for representing videos. But in small scale frame, the blob-like primitives possess a large percentage at the very beginning, and the number increase of edge-like primitives gets quicker and quicker while more and more primitives are chosen. This phenomenon demonstrates that blob-like structures are much more prominent in small scale. So from this experiment, it is evident that VPS

Table 6 The accuracy of differentiating the original video from the synthesized one in different scales

Video	Scale 100 %	Scale 75 %	Scale 50 %	Scale 25 %
1	66.7	56.7	46.7	50
2	100	90	73.3	63.3
3	73.3	63.3	50	53.3

As the percentage is getting closer to 50 %, it means it is harder to discriminate the original and synthesized videos for observers

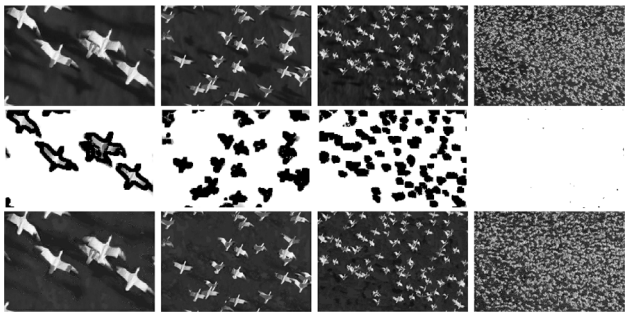


Fig. 20 Representation switches triggered by scale. *Row 1* observed frames; *Row 2* trackability maps; *Row 3* synthesized frames

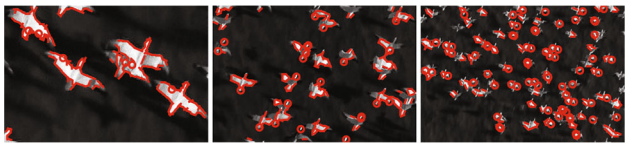


Fig. 21 Representation types in different scale video frames, where circles represent blob-like type and short lines represent edge-like type

Table 7 The comparisons between the number of blob-like and edge-like primitives in 3 scales

Scale	First 50	First 100	First 150	First 200
1	0/50	0/100	1/149	6/194
2	0/50	6/94	16/134	23/177
3	19/31	37/63	58/92	71/129

For each scale, the numbers are compared in first 50, 100, 150, and 200 primitives, respectively

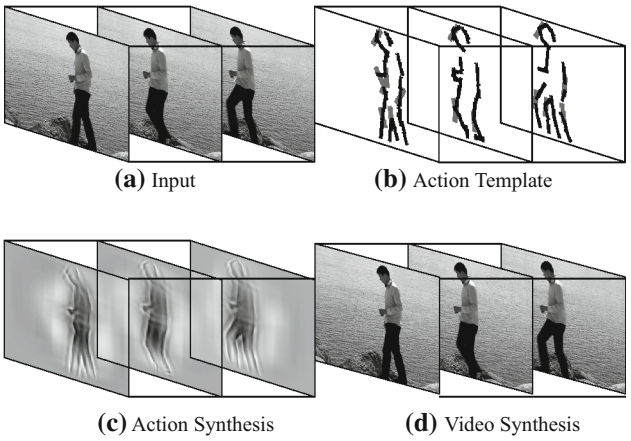


Fig. 22 Action representation by VPS. **a** The input video. **b** Action template obtained by the deformable action template [40]. **c** Action synthesis by filters. **d** Video synthesis by VPS

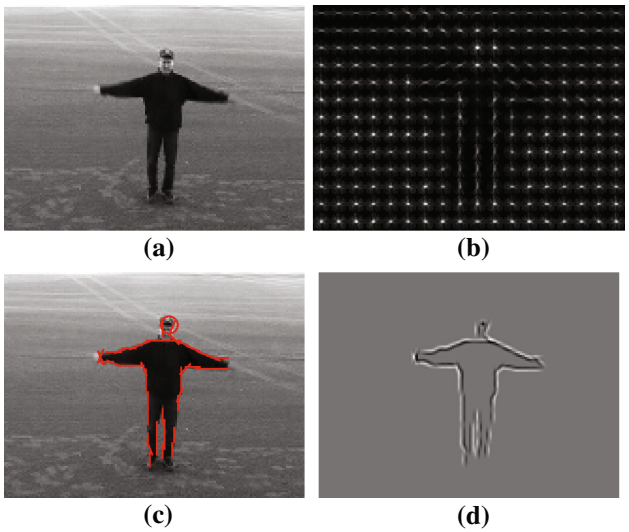


Fig. 23 Structural information extracted by HOG and VPS. **a** The input video frame. **b** HOG descriptor. **c** VPS feature. **d** Boundary synthesis by filters

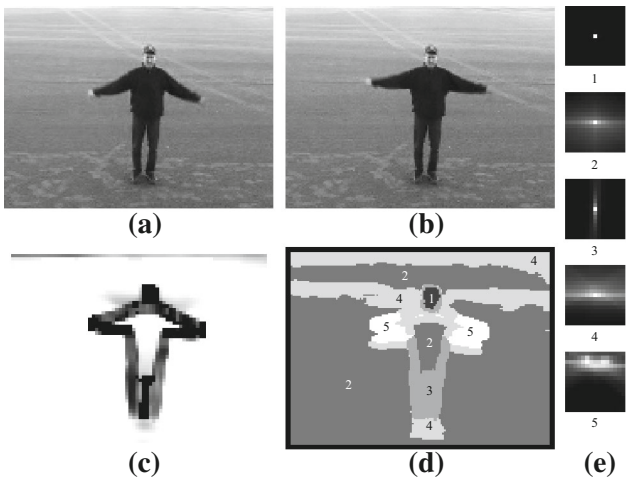


Fig. 24 Motion statistics by VPS. **a** and **b** two continuous video frames of waving hands. **c** Trackability map. **d** Clustered motion style regions. **e** Corresponding motion statistics of each region

can choose proper representations automatically and furthermore, the representation patterns may reflect the scale of the videos.

3.11 VPS Supporting Action Representation

VPS is also compatible with high-level action representation. By grouping meaningful explicit parts in a principled way, it represents an action template. In Fig. 22b is the action template given by the deformable action template model [40] from the video shown in (a). The action template is essentially the sketches from the explicit regions. (c) shows an action synthesis with only filters from a matching pursuit process. While in (d), following the VPS model, the action parts and a few sketchable background are reconstructed by the explicit representation, and the large region of water is synthesized by the implicit representation; thus we get the synthesis of the whole video. Here, the explicit regions correspond to meaningful "template" parts, while the implicit regions are auxiliary background parts.

In order to show the relationship between VPS representation and effective high-level features, we take an KTH video [31] as an example. Figures 23 and 24 show the spatial and temporal features of explicit regions, respectively. In Fig. 23, we compare VPS spatial descriptor with well-known HOG feature [11], which has been widely used for object representation recently. (b) is the HOG descriptor for the human in one video frame (a). (c) shows structural features extracted by VPS, where circles and short edges represent 53 local descriptors. Compared with HOG in (b), VPS makes a local decision on each area based on statistics of filter responses; therefore, it provides shorter coding length than HOG. Furthermore, it gives more precise description than HOG, e.g., the head part is represented by a circle descriptor, which contains more information than pure filter response histogram like HOG. And (d) gives a synthesis with corresponding filters, which shows the human boundary precisely.

In Fig. 24, we show the motion information between two continuous frames (a) and (b) extracted by MA-FRAME in VPS. (d) gives the clustered motion styles in the current video. The motion statistics of the five styles are shown in (e), respectively. It is obvious that region 1 represents the area of head, which is almost still in the waving motion, while region 5 is for two arms, which shows definite moving direction. Region 3 represents the legs, which is actually an oriented trackable area. Region 2 and 4 are relatively ambiguous in motion direction, which are basically background of textures in the video. After giving the trackability map shown in (c) based on these motion styles, the motion template pops up.

In summary, the information extracted by VPS is compatible with high-level object and motion representations. Especially, it is very close to HOG and HOOF descriptors, which are proven effective spatial and temporal features, respectively. The main difference is VPS makes a local decision to give a more compact expression and be better for visualization. Therefore, VPS does not only give a middle-level representation for video, but also has strong connection with low-level vision features and high-level vision templates.

4 Discussion and Conclusion

In this paper, we present a novel video primal sketch model as a middle-level generic representation of video. It is generative and parsimonious, integrating a sparse coding model for explicitly representing sketchable and trackable regions and extending the FRAME models for implicitly representing textured motions. It is a video extension of the primal sketch model [18]. It can choose appropriate models automatically for video representation.

Based on the model, we design an effective algorithms for video synthesis, in which, explicit regions are reconstructed by learned video primitives and implicit regions are synthesized through a Gibbs sampling procedure based on spatio-temporal statistics. Our experiments show that VPS is capable for video modeling and representation, which has high compression ratio and synthesis quality. Furthermore, it learns explicit and implicit expressions for meaningful low-level vision features and is compatible with high-level structural and motion representations, therefore, provides a unified video representation for all low, middle, and high-level vision tasks.

In ongoing work, we will strengthen our work from several aspects, especially enhance the connections with low-level and high-level vision tasks. For low-level study, we are learning a much richer dictionary of Δ_B for video primitives, which is more comprehensive. For high-level application, we are applying the VPS features to object and action representation and recognition.

Acknowledgments This work is done when Han is a visiting student at UCLA. We thank the support of an NSF grant DMS 1007889 and ONR MURI grant N00014-10-1-0933 at UCLA. The authors also thank the support by four grants in China: NSFC 61303168, 2007CB311002, NSFC 60832004, NSFC 61273020.

References

1. Adelson, E., Bergen, J.: Spatiotemporal energy models for the perception of motion. JOSA A **2**(2), 284–299 (1985)
2. Bergen, J.R., Adelson, E.H.: In: Regan, D. (ed.) Theories of Visual Texture Perception. Spatial Vision. CRC Press, Boca Raton, FL (1991)
3. Besag, J.: Spatial interactions and the statistical analysis of lattice systems. J. R. Stat. Soc. Ser. B **36**, 192–236 (1974)
4. Black, M.J., Fleet, D.J.: Probabilistic detection and tracking of motion boundaries. IJCV **38**(3), 231–245 (2000)
5. Bouthemy, P., Hardouin, C., Piriou, G., Yao, J.: Mixed-state auto-models and motion texture modeling. J. Math. Imaging Vis. **25**(3) (2006)
6. Campbell, N.W., Dalton, C., Gibson, D., Thomas, B. : Practical generation of video textures using the auto-regressive process. In: Proceedings of British Machine Vision Conference, pp 434–443 (2002)
7. Chan, A.B., Vasconcelos, N.: Modeling, clustering, and segmenting video with mixtures of dynamic textures. PAMI **30**(5), 909–926 (2008)
8. Chaudhry, R., Ravichandran, A., Hager, G., Vidal, R. : Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. CVPR (2009)
9. Chubb, C., Landy, M.S.: Orthogonal distribution analysis: a new approach to the study of texture perception. In: Landy, M.S., et al. (eds.) Proceedings of the Comp Models of Visual. MIT Press, Cambridge, MA (1991)
10. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. PAMI **25**(5), 564–577 (2003)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. CVPR (2005)
12. Dalal, N., Triggs, B., Schmid, C. : Human detection using oriented histograms of flow and appearance. ECCV (2006)
13. Derpanis, K.G., Wildes, R.P. : Dynamic texture recognition based on distributions of spacetime oriented structure. CVPR (2010)
14. Doretto, G., Chiuso, A., Wu, Y.N., Soatto, S.: Dynamic textures. IJCV **51**(2), 91–109 (2003)
15. Elder, J., Zucker, S.: Local scale control for edge detection and blur estimation. PAMI **20**(7), 699–716 (1998)
16. Fan, Z., Yang, M., Wu, Y., Hua, G., Yu, T. : Effient optimal kernel placement for reliable visual tracking. CVPR (2006)
17. Gong, H.F., Zhu, S.C.: Intrackability: characterizing video statistics and pursuing video representations. IJCV **97**(33), 255–275 (2012)
18. Guo, C., Zhu, S.C., Wu, Y.N.: Primal sketch: integrating texture and structure. CVIU **106**(1), 5–19 (2007)
19. Han, Z., Xu, Z., Zhu, S.C.: Video primal sketch: a generic middle-level representation of video. ICCV (2011)
20. Heeger, D.: Model for the extraction of image flow. JOSA A **4**(8), 1455–1471 (1987)
21. Heeger, D.J., Bergen, J.R.: Pyramid-based texture analysis/synthesis. SIGGRAPH (1995)
22. Kim, T., Shakhnarovich, G., Urtasun, R.: Sparse coding for learning interpretable spatio-temporal primitives. NIPS (2010)
23. Lindeberg, T., Fagerstrm, D.: Scale-space with casual time direction. ECCV (1996)
24. Maccormick, J., Blake, A.: A probabilistic exclusion principle for tracking multiple objects. IJCV **39**(1), 57–71 (2000)
25. Mallat, S., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE TSP **41**(12), 3397–3415 (1993)
26. Marr, D.: Vision. W H Freeman and Company, San Francisco, CA (1982)
27. Olshausen, B.A.: Learning sparse, overcomplete representations of time-varying natural images. ICIP (2003)
28. Olshausen, B.A., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381 (1996)
29. Portilla, J., Simoncelli, E.: A parametric texture model based on joint statistics of complex wavelet coefficients. IJCV **40**(1), 49–71 (2000)

30. Ravichandran, A., Chaudhry, R., Vidal, R.: View-invariant dynamic texture recognition using a bag of dynamical systems. CVPR (2009)
31. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local svm approach. ICPR (2004)
32. Serby, D., Koller-Meier, S., Gool, L.V.: Probabilistic object tracking using multiple features. ICPR (2004)
33. Shi, K., Zhu, S.C.: Mapping natural image patches by explicit and implicit manifolds. CVPR (2007)
34. Silverman, M.S., Grosof, D.H., Valois, R.L.D., Elfar, S.D.: Spatial-frequency organization in primate striate cortex. Proc. Natl. Acad. Sci. **86**, 711–715 (1989)
35. Szummer, M., Picard, R.W.: Temporal texture modeling. ICIP (1996)
36. Wang, Y.Z., Zhu, S.C.: Analysis and synthesis of textured motion: particles and waves. PAMI **26**(10), 1348–1363 (2004)
37. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error measurement to structural similarity. IEEE TIP **13**(4) (2004)
38. Wildes, R., Bergen, J.: Qualitative spatiotemporal analysis using an oriented energy representation. ECCV (2000)
39. Wu, Y.N., Zhu, S.C., Liu, X.W.: Equivalence of julesz ensemble and frame models. IJCV **38**(3), 247–265 (2000)
40. Yao, B., Zhu, S.C.: Learning deformable action templates from cluttered videos. ICCV (2009)
41. Yuan, F., Prinet, V., Yuan, J.: Middle-level representation for human activities recognition: the role of spatio-temporal relationships. ECCVW (2010)
42. Zhu, S.C., Wu, Y.N., Mumford, D.B.: Filters, random field and maximum entropy (FRAME): towards a unified theory for texture modeling. IJCV **27**(2), 107–126 (1998)

Zhi Han received the B.Sc., M.Sc., and Ph.D. degrees in applied mathematics from Xi'an Jiaotong University, China in 2005, 2007, and 2012, respectively. From 2009 to 2011, he was a joint Ph.D. candidate of statistics in University of California, Los Angeles, USA. He is currently an assistant professor of State Key Laboratory of Robotics in Shenyang Institute of Automation, Chinese Academy of Sciences. His research interests include image/video modeling, low-rank matrix recovery and illumination processing.

Zongben Xu received the Ph.D. degree in mathematics from Xian Jiaotong University, Xian, China, in 1987. He currently serves as a Vice President with Xian Jiaotong University, the Academician of the Chinese Academy of Sciences, the Chief Scientist of the National Basic Research Program of China (973 Project), and the Director of the Institute for Information and System Sciences of the University. His current research interests include nonlinear functional analysis and intelligent information processing. Dr. Xu was a recipient of the National N atural Science Award of China in 2007 and was a winner of the CSIAM Su B uchin Applied Mathematics Prize in 2008. He delivered a talk at the Inter national Congress of Mathematicians 2010.

Song-Chun Zhu received a Ph.D. degree from Harvard University in 1996. He is currently a professor of Statistics and Computer Science at UCLA, and the director of the Center for Vision, Cognition, Learning and Art. He has published over 160 papers in computer vision, statistical modeling and learning, cognition, and visual arts. He received a number of honors, including the J.K. Aggarwal prize from the Intl Association of Pattern Recognition in 2008, the David Marr Prize in 2003 with Z. Tu et al.,twice Marr Prize honorary nominations in 1999 and 2007, He received the Sloan Fellowship in 2001, a US NSF Career Award in 2001, and an US ONR Young Investigator Award in 2001. He received the Helmholtz Test-of-time award in ICCV 2013, and he is a Fellow of IEEE since 2011.