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 (ST-FRAME) and Motion-Appearance FRAME (MA-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.
Video Primal Sketch
A Unified Middle-Level Representation for Video

Zhi Han\textsuperscript{1,2} · Zongben Xu\textsuperscript{1} · Song-Chun Zhu\textsuperscript{2}

Received: date / Accepted: date

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 (ST-FRAME) and Motion-Appearance FRAME (MA-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

1 Introduction

1.1 Motivation

Videos of natural scenes contain vast varieties of motion patterns. We divide these motion patterns in a $2 \times 2$ table based on their complexities measured by two criteria: i) sketchability (Guo et al (2007)), i.e. whether a local patch can be represented explicitly by an image primitive from a sparse coding dictionary, and ii) intrackability (or trackability) (Gong and Zhu (2012)), which measures the uncertainty of tracking an image patch using the entropy of posterior probability on velocities. Fig.1 shows some examples of the different video patches in the four categories. Category A consists of the simplest vision phenomena, i.e. sketch-
able 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).

In the vision literature, as it was pointed out by (Shi and Zhu (2007)), there are two families of representations, which code images or videos by explicit and implicit functions respectively.

1. Explicit representations with generative models. (Olshausen (2003); Kim et al (2010)) learned an overcomplete set of coding elements from natural video sequences using the sparse coding model (Olshausen and Field (1996)). (Elder and Zucker (1998)) and (Guo et al (2007)) represented the image/video patches by fitting functions with explicit geometric and photometric parameters. (Wang and Zhu (2004)) synthesized complex motion, such as birds, snowflakes, and waves with a large mount of particles and wave components. (Black and Fleet (2000)) 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 contours tracking (Maccormick and Blake (2000)), kernels tracking (Comaniciu et al (2003)), and sparse or dense feature points tracking (Serby et al (2004), Black and Fleet (2000)).

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 (Szummer and Picard (1996); Campbell et al (2002)) were modelled 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. (Bothemy et al (2006)) proposed a mixed-state auto-models for motion textures by generalizing the auto-models in (Bosag (1974)). (Doretto et al (2003)) derived an auto-regression moving-average model for dynamic texture. (Chan and Vasconcelos (2008)) and (Ravichandran et al (2009)) 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) (Dalal and Triggs (2005)) for appearance and Histogram of Oriented Optical-Flow (HOOF) (N. Dalal and Schmid (2006); Chaudhry et al (2009)) 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 (Gong and Zhu (2012)) 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 (Marr (1982)), and is aimed at extending the primal sketch model proposed by (Guo et al (2007)) from images to videos. Our goal is not only to provide a parsimonious model for video compression and coding, but more importantly, to support and be compatible with high-level tasks such as motion tracking and action recognition.

Fig. 2. (a) 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 (Zhu et al (1998)) 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
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

| Parameter               | Value                                      |
|-------------------------|--------------------------------------------|
| 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     | 15×(11+12+5)=420                           |

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.

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 (Zhu et al (1998)) 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 (N. Dalal and Schmid (2006); Chaudhry et al (2009)).

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 (Comaniciu et al (2003)).

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 (Gong and Zhu (2012)), 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 (2012)) did not give a unified model for video representation and synthesis which is the focus on the current paper.

The remainder of this paper is organized as follows. In Section 2, we present the framework of video primal sketch. In Section 3, we explain the algorithms for explicit representation, textured motion synthesis and video synthesis, and show a series of experiments. The paper is concluded with a discussion in Section 4.
2 Video primal sketch model

In his monumental book (Marr (1982)), 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 (Guo et al (2007)), 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 \( I[1, m] = \{ I(t) \}_{t=1}^m \) be a video defined on a 3D lattice \( A \subset \mathbb{Z}^3 \). \( A \) is divided disjointly into explicit and implicit regions,

\[
A = A_{ex} \bigcup A_{im}, \quad A_{ex} \bigcap A_{im} = \emptyset. \tag{1}
\]

Then the video \( I \) is decomposed as two components

\[
I = (I_{A_{ex}}, I_{A_{im}}). \tag{2}
\]

In the following, we first present the two families of models for \( I_{A_{ex}} \) and \( I_{A_{im}} \) respectively, and then integrate them in the VPS model.

2.1 Explicit representation by sparse coding

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

\[
A_{ex} = \bigcup_{i=1}^{n_{ex}} A_{ex,i}. \tag{3}
\]

Here \( A_{ex,i} \subset A \) defines the domain of a “brick”. A brick, denoted by \( I_{A_{ex,i}} \), 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 section 1.

For example, \( A_{ex,i} \) includes \( 11 \times 11 \) pixels \( \times \) 3 frames for trackable bricks (sketchable or non-trackable), or \( 11 \times 11 \) pixels \( \times \) 1 frame for sketchable but intrackable bricks.

Fig.3 shows one example comparing the sketchable and trackable regions based on sketchability and trackability maps shown in Fig.2(b) and (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,

\[
I(x, y, t) = \alpha_i B_i(x, y, t) + \epsilon, \quad \forall (x, y, t) \in A_{ex,i}. \tag{4}
\]

\( B_i \) means the \( i \)th primitive from the primitive dictionary \( \Delta_B \), which fits the brick \( I_{A_{ex,i}} \) best. Here \( i \) indexes the parameters such as type, position, orientation and scale of \( B_i \). \( \alpha_i \) is the corresponding coefficient. \( \epsilon \) represents the residue, which is assumed to be i.i.d. Gaussian. For an 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 \( \Delta_B \) is composed of two categories:

- Common primitives \( \Delta_B^{common} \). These are primitives shared by most videos, such as blobs, edges, and ridges. They have explicit parameters for orientations and scales.
- Special primitives \( \Delta_B^{special} \). These bricks do not share common appearance and are limited to specific video frames. They are non-sketchable but trackable, and are recorded to code the specific video region.

Eqn (4) uses only one base function and thus is different from conventional linear additive model. Follows the Gaussian assumption for the residues, we have the
following probabilistic model for the explicit region $I_{A_{ex}}$

$$p(I_{A_{ex}}; B, \alpha) = \prod_{i=1}^{n_{ex}} \frac{1}{(2\pi)^{3} \sigma_{i}^{3}} \exp\{-E_{i}\}$$

$$E_{i} = \sum_{(x,y,t) \in A_{ex,i}} \frac{(I(x,y,t) - \alpha, A_{i}(x,y,t))^{2}}{2\sigma_{i}^{2}}.$$  \hspace{1cm} (5)

where $B = (B_{1}, ..., B_{n_{ex}})$ represents the selected primitive set and $n_{ex}$ is the number of selected primitives.

2.2 Implicit representations by FRAME models

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

$$A_{im} = \bigcup_{j=1}^{n_{im}} A_{im,j}.$$ \hspace{1cm} (6)

In texture modelling, researchers pool the histograms for a set of Gabor filters on the input image. 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. The FRAME model proposed in (Zhu et al. (1998)) reproduces the observed histograms as its marginal distributions 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 $I_{A_{im,j}}$ by an equivalence class of videos,

$$\Omega_{K}(h_{j}) = \{I_{A_{im,j}} : H_{k}(I_{A_{im,j}}) = h_{k,j}, k = 1, 2, ..., K\}.$$ \hspace{1cm} (7)

where $h_{j} = (h_{1,j}, ..., h_{K,j})$ 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 $h_{j}$ and synthesize the textured motion region $I_{A_{im,j}}$ by sampling from the set $\Omega_{K}(h_{j})$. As $I_{A_{im,j}}$ 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 section 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)(N. Dalal and Schmid (2006); Chaudhry et al (2009)) 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 section 2.4.

2.3 Implicit representation by spatio-temporal FRAME

ST-FRAME is an extension of the FRAME model (Zhu et al. (1998)) by adopting spatio-temporal filters.

A set of filters $F$ is selected from a filter bank $\Delta_{F}$. Fig. 5 illustrates the three types of filters in $\Delta_{F}$: 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 $F_{k} \in F$, the spatio-temporal filter response of $I$ at $(x, y, t) \in A_{im,j}$ is $F_{k} * I(x, y, t)$. The convolution is over spatial and temporal domain. By pooling the filter responses over all $(x, y, t) \in A_{im,j}$, we obtain a number of 1D histograms

$$H_{k}(I_{A_{im,j}}) = H_{k}(z; I_{A_{im,j}}) = \frac{1}{|A_{im,j}|} \sum_{(x,y,t) \in A_{im,j}} 1(z - F_{k} * I(x,y,t)), k = 1, ..., K.$$ \hspace{1cm} (8)

where $1()$ is an indicator function and $z$ is the index of the histogram bins in discrete form. Following the FRAME model, the statistical model of textured motion $I_{A_{im,j}}$ is written in the form of the following Gibbs

| $\Delta_{F}$ | Static Filter | Motion Filter | Flicker Filter |
|-------------|---------------|---------------|---------------|
| Log         | ![Log](image) | ![Motion](image) | ![Flicker](image) |
| Gabor       | ![Log](image) | ![Motion](image) | ![Flicker](image) |
| Intensity   | ![Log](image) | ![Motion](image) | ![Flicker](image) |
| Gradient    | ![Log](image) | ![Motion](image) | ![Flicker](image) |

Fig. 5 $\Delta_{F}$ is a dictionary of spatio-temporal filters including static, motion and flicker filters.
distribution,
\[ p(\Lambda_{im,j}; F, \beta) \propto \exp\left\{-\sum_k \langle \beta_{k,j}, H_k(\Lambda_{im,j}) \rangle \right\}. \]  
(9)

According to the theorem of ensemble equivalence (Wu et al. (2000)), the Gibbs distribution converges to the uniform distribution over the set \( \Omega_K(\mathbf{h}_1) \) in eqn(7), when \( \Lambda_{im,j} \) is large enough. For any fixed local brick \( \Lambda_0 \subset \Lambda_{im,j} \), the distribution of \( I_{\Lambda_0} \) follows the Markov random field model (9). The model can describe textured motion located in an irregular shape region \( \Lambda_{im,j} \).

The filters in \( F \) are pursued one by one from the filter bank \( \Delta_F \) so that the information gain is maximized at each step.

\[ F_k^* = \arg\max_{F_k \in \Delta_F} \| H_k^{syn} - H_k^0 \|. \]  
(10)

\( H_k^0 \) and \( H_k^{syn} \) are the response histograms of \( F_k \) before and after synthesizing \( I_{\Lambda_{im}} \) by adding \( F_k \) respectively. The larger the difference, the more important is the filter.

Following the distribution form of eqn(9), the probabilistic model of implicit parts of \( I \) is defined as

\[ p(\Lambda_{im}; F, \beta) \propto \prod_{j=1}^{\text{num}} p(\Lambda_{im,j}; F, \beta). \]  
(11)

where \( F = (F_1, ..., F_K) \) 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 eqn(10)

\[ H_k^{(s)}(I_{\Lambda_{im,j}}) = H_k^{(s)}(z; I_{\Lambda_{im,j}}) \]  
(12)

\[ = \frac{1}{|\Lambda_{im,j}|} \sum_{(x,y,t) \in \Lambda_{im,j}} 1(z - F_k \ast I(x,y,t)), k = 1, ..., K. \]

where the filter set \( F \) includes static and flicker filters in \( \Delta_F \).

For the motion constraints, the velocity distribution of each local patch is estimated via the calculation of trackability (Gong and Zhu (2012)). We compute the probability of the local velocity \( v \),

\[ p(v|I(x, y, t - 1), I(x, y, t)) \]

\[ \propto \exp\left\{-\frac{\| I_{\partial(I(x-y-vx,y-vy,t) - I_{\partial(I(x,y,t-1)}}^2}{2\sigma_v^2} \right\}. \]  
(13)

The statistical information of velocities for a certain area of texture is approximated by averaging the velocity distribution over region \( \Lambda_{im,j} \)

\[ H^{(v)}(v_{\Lambda_{im,j}}) \]  
(14)

\[ = \sum_{(x,y,t) \in \Lambda_{im,j}} p(v|I(x, y, t - 1), I(x, y, t)). \]

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

\[ p(I_{\Lambda_{im,j}}; F, \beta) \propto \exp\left\{-\langle \beta, H(I_{\Lambda_{im,j}}, v_{\Lambda_{im,j}}) \rangle \right\}. \]  
(15)

Here, \( \beta \) is the parameter of the model.

According to the theorem of ensemble equivalence (Wu et al. (2000)), the Gibbs distribution converges to the uniform distribution over the set \( \Omega_K(\mathbf{h}_1) \) in eqn(7), when \( \Lambda_{im,j} \) is large enough. For any fixed local brick \( \Lambda_0 \subset \Lambda_{im,j} \), the distribution of \( I_{\Lambda_0} \) follows the Markov random field model (9). The model can describe textured motion located in an irregular shape region \( \Lambda_{im,j} \).

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.

2.5 Hybrid model for video representation

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

\[ I_{\Lambda_{im}} \sim p(I_{\Lambda_{im}}; F, \beta), I_{\Lambda_{im}} \sim p(I_{\Lambda_{im}}|I_{\partial(A_{im}; F, \beta)}). \]  
(17)

Here, \( I_{\Lambda_{im}} \) represents the boundary condition of \( I_{\Lambda_{im}} \), which belong to the reconstruction of \( I_{\Lambda_{im}} \). It leads to seamless boundaries in the synthesis.
By integrating the explicit and implicit representation, the video primal sketch has the following probability model,

\[ p(\mathbf{I}, \mathbf{F}, \alpha, \beta) = \]

\[ \frac{1}{Z} \exp\left\{ - \sum_{i=1}^{n_{ex}} \left( \mathbf{I}(x,y,t) - \alpha \mathbf{B}(x,y,t) \right)^2 - \sum_{j=1}^{n_{im}} \sum_{k=1}^{K} \langle \beta_{k,j}, H_k(\mathbf{I}_{A_{im,j}})|\mathbf{I}_{B_{A_{im,j}}}) \rangle \right\} \]

where \( Z \) is the normalizing constant.

We denote by \( VPS = (\mathbf{B}, \mathbf{H}) \) the representation for the video \( \mathbf{I}_A \), where \( \mathbf{H} = \{h_{k,1}, h_{k,2}, \ldots, h_{k,n_{im}} \} \) 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 etc. The \( VPS \) is obtained by maximizing the posterior probability,

\[ VPS^* = \text{arg max}_{VPS} p(VPS|\mathbf{I}_A). \]  

following the video primal sketch model in eqn(18).

Table 1 shows an example of \( VPS \). For a video of the size of 288 \times 352 pixels, about 30% of the pixels are represented explicitly by \( n_{ex} = 300 \) motion primitives. As each primitive needs \( 11 \) parameters to record the profile and \( 1 \) more to record the type, the number of total parameters for the explicit representation is \( 3,600. \), \( n_{im} = 3 \) textured motion regions are represented implicitly by the histograms, which are described by \( K_1 = 11, K_2 = 12, K_3 = 5 \) 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 \( A \) into the explicit regions \( A_{ex} \) and implicit regions \( A_{im} \). This is done through the sketchability and trackability maps. In this subsection, we overview the general ideas and refer to previous work on sketchability (Guo et al (2007)) and trackability (Gong and Zhu (2012)) for details.

Let’s consider one local volume \( A_0 \subset A \) of the video \( \mathbf{I} \). In the video primal sketch model, \( \mathbf{I}_{A_0} \) may be modeled either by the sparse coding model in eqn(5) or by the FRAME model in eqn(11). The choice is determined via the competition between the two models, i.e. comparing which model gives shorter coding length (Shi and Zhu (2007)) for representation.

If \( \mathbf{I}_{A_0} \) is represented by the sparse coding model, the posterior probability is calculated by

\[ p(\mathbf{B}|\mathbf{I}_{A_0}) = \]

\[ \frac{1}{(2\pi)^{n/2}\sigma^n} \exp\left\{ - \sum_{(x,y,t)\in A_0} \left( \mathbf{I}(x,y,t) - \alpha \mathbf{B}(x,y,t) \right)^2 \right\}. \]

where \( n = |A_0| \). The coding length is

\[ L_{ex}(\mathbf{I}_{A_0}) = \log \frac{1}{p(\mathbf{B}|\mathbf{I}_{A_0})} \]

\[ = \frac{n}{2} \log 2\pi\sigma^2 + \sum_{(x,y,t)\in A_0} \left( \mathbf{I}(x,y,t) - \alpha \mathbf{B}(x,y,t) \right)^2 \]

\[ = \frac{n}{2} (\log 2\pi\sigma^2 + 1). \]

If \( \mathbf{I}_{A_0} \) is described by the FRAME model, the posterior probability is calculated by

\[ p(\mathbf{F}|\mathbf{I}_{A_0}) \propto \exp\left\{ - \sum_{k=1}^{K} \langle \beta_k, H_k(\mathbf{I}_{A_0}) \rangle \right\}. \]

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 \( \Lambda \). Suppose the intensities of the video range from 0 to 255, then \( \log |\Omega_0| = 8 \times |A_0| \). 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 eqn.(10). Therefore we can calculate the coding length by

\[ L_{im}(\mathbf{I}_{A_0}) = \log |\Omega_0| - \log \frac{|\Omega_0|}{|\Omega_1|} - \ldots - \log \frac{|\Omega_{K-1}|}{|\Omega_K|}. \]  

By comparing \( L_{im} \mathbf{(I}_{A_0} \) and \( L_{ex}(\mathbf{I}_{A_0} \), whoever has the shorter coding length will win the competition and be chosen for \( \mathbf{I}_{A_0} \).

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 \( p(\mathbf{B}|\mathbf{I}_{A_0}) \).

\[ H(\mathbf{B}|\mathbf{I}_{A_0}) = -E_p(\mathbf{B}|\mathbf{I}_{A_0}) \log p(\mathbf{B}|\mathbf{I}_{A_0})]. \]

It measures the uncertainty of selecting a primitive in \( \Delta \) for representation. The sharper the distribution \( p(\mathbf{B}|\mathbf{I}_{A_0}) \) is, the lower the entropy \( H(\mathbf{B}|\mathbf{I}_{A_0}) \) will be, which gives smaller \( L_{ex}(\mathbf{I}_{A_0}) \) according to eqn(21). Hence, \( H(\mathbf{B}_k|\mathbf{I}_{A_0}) \) reflects the magnitude of \( L_{dif}(\mathbf{I}_{A_0}) = L_{ex}(\mathbf{I}_{A_0}) - L_{im}(\mathbf{I}_{A_0}) \). Set an entropy threshold \( H_0 \) on \( H(\mathbf{B}_k|\mathbf{I}_{A_0}) \), ideally, \( H(\mathbf{B}_k|\mathbf{I}_{A_0}) = H_0 \) if and only if \( L_{dif}(\mathbf{I}_{A_0}) = 0 \). Therefore, when \( H(\mathbf{B}_k|\mathbf{I}_{A_0}) < H_0 \), we consider \( L_{ex}(\mathbf{I}_{A_0}) \)
is lower and \( I_{4b} \) is modeled by the sparse coding model, else it is modeled by the FRAME model.

It is clear that \( \mathcal{H}(B_j|I_{4b}) \) has the same form and meaning with sketchability (Guo et al. (2007)) in appearance representation and trackability (Gong and Zhu (2012)) in motion representation. Therefore, sketchability and trackability can be used for model selection for each local volume. Fig.2 (b) and (c) show 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 \( 11 \times 11 \) pixels \( \times 3 \) frames) or intrackable (using primitives with \( 11 \times 11 \) 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 (Adelson and Bergen (1985)), optical flow estimation (Heeger (1987)), pattern categorization (Wildes and Bergen (2000)), and dynamic texture recognition (Derpanis and Wildes (2010)). In the experiments, we choose spatio-temporal filters \( \Delta F \) 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 velocities and orientations 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 \( 7 \times 7 \) pixels in size and have 6 scales, 12 moving orientations and 3 velocities. Each type of filter has a special effect in textured motion synthesis, which will be discussed in section 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, \( \alpha_{i,j} = \langle I_{A_{\alpha},1}, B_j \rangle \), 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 (Mallat and Zhang (1993)) 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 \( 11 \times 11 \) 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 \( B \) with its motion orientation and velocity calculated from the trackability map. It is also added into \( \Delta B \) for the dictionary buildup. The size of each primitive is \( 11 \times 11 \), the same as the size of the fitted filter. In Fig.4, we show some examples of different types of primitives, such as blob, ridge and edge. Fig.6 shows some examples of reconstruction by motion primitives. In each group, the original local image, the best fitted filter and the motion velocity \((u,v)\), which represents the motion of \( B \).

![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 B ∈ ΔB and the velocity (u, v), which represents the motion of B.](image-url)
The reconstruction effect of sketchable regions by common primitives. (a) The observed frame. (b) The reconstructed frame.

By comparing the observed frame (a) and reconstructed frame (b), it is noted that there are only very tiny differences in very detailed parts of the video frame. The more detailed quantitative assessment is given in section 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 section 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 $I_{A_0}$ of textured motion located at $A_0$ follows a Markov random field model conditioned on its local neighborhood $I_{V_{A_0}}$ following eqn(9),

$$p(I_{A_0}|I_{V_{A_0}}; F, \beta) \propto \exp\{- \sum_k \langle \beta_k, H_k(I_{A_0}) \rangle\},$$  \hspace{1cm} (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,

$$\hat{\beta} = \arg \min_{\beta} \log p(I_{A_0}|I_{V_{A_0}}; \beta, F)$$

$$= \arg \max_{\beta} \{- \log Z(\beta) - \sum_k \langle \beta_k, H_k(I_{A_0}) \rangle\}$$ \hspace{1cm} (26)

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

$$\frac{d\beta(i)}{dt} = E_p(I, \beta, F)[H(i)] - H_{obs}(i)$$ \hspace{1cm} (27)

In order to draw a typical sample frame from $p(I; F, \beta)$, 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 $p(I; F, \beta)$. Therefore, we get the final converged results dominated by $p(I; \beta, F)$, 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 $I_{obs} = \{I(1), ..., I(m)\}$.

Suppose we have $I_{syn} = \{I_{syn}^{(1)}, ..., I_{syn}^{(m)}\}$, our goal is to synthesize the next frame $I_{syn}^{(m+1)}$.

Select a group of spatio-temporal filters from a filter bank $F = \{F_k\}_{k=1}^K \in \Delta F$.

Compute $h_k$, $k = 1, ..., K$ of $I_{obs}$.

Initialize $\beta_k(i) \leftarrow 0$, $k = 1, ..., K$, $i = 1, ..., L$.

Initialize $I_{syn}^{(m+1)}$ as a uniform white noise image.

Repeat

Calculate $h_k^{syn}$, $k = 1, 2, ..., K$ from $I_{syn}$.

Update $\beta_k$, $k = 1, ..., K$ and $p(I; F, \beta)$.

Sample $I_{syn}^{(m+1)} \sim p(I; F, \beta)$ by Gibbs sampler.

Until $\frac{1}{L} \sum_{i=1}^L |h_k(i) - h_k^{syn}(i)| \leq \epsilon$ for $k = 1, 2, ..., K$.

Fig.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 section 3.8. Fig.10 shows that as $I_{syn}^{(m)}$ changes from white noise (Fig.8(a)) to the final synthesized result (Fig.8(e)), the histograms...
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.

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

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 (Doretto et al (2003)).

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

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. However, optical flow is an approximate estimation of global motion trend, espe-
cially efficient for trackable area in videos, but unsuit- able for textured motion. For example, it can hardly detect high-frequency variation of moving direction and speed in dynamic textures. It becomes ineffective when the velocity is non-smooth. 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 \(I_{0\Lambda_{x,y},t}\). Comparing patch \(I_{0\Lambda_{x,y},t}\) 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

\[\displaystyle p(v) \propto \exp\{-\|I_{0\Lambda_{x,y},t} - I_{0\Lambda_{x-v_x,y-v_y},t-1}\|^2\}\]  

(28)

This distribution describes the probability of the origin of the patch, i.e. the location where the patch \(I_{0\Lambda_{x,y},t}\) 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 approximately, which reflects the motion pattern of these clustered pixels. Fig.14 and Fig.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.

3.5 Synthesizing textured motions by MA-FRAME

In MA-FRAME model, similar to ST-FRAME, each local volume \(I_{t_0}\) 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, \(I_{\Lambda_{m_{ij}}}\) and \(v_{\Lambda_{m_{ij}}}\) are sampled simultaneously following the joint distribution in eqn(15).

\[\displaystyle p(I_{\Lambda_{m_{ij}}}, F, \beta) \propto \exp\{-\beta \, H(I_{\Lambda_{m_{ij}}}, v_{\Lambda_{m_{ij}}})\}\} \]

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. 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.

Fig.11 illustrates the sampling method at a 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 \(-v_{max} \leq v_{x,i}, v_{y,j} \leq v_{max}\). 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.11(a)), \(v_{max} = 1\) 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.11(b)). 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.11(b).

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.
Fig. 12 Texture synthesis for 24 frames of ocean video by MA-FRAME.

by
\[
\omega(x, y) = \exp\left\{- \sum_{(x, y) \in \partial \Lambda(x, y)} \|v_x - v_x^k\|^2 + \|v_y - v_y^k\|^2\right\}
\]
The weights encourage the velocity candidate which is closer to the velocities of its neighbours. 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 \(\mathbf{I}_{\text{obs}} = \{I_1, ..., I_n\}\).
Suppose we have \(\mathbf{I}_{\text{syn}} = \{I_{\text{syn}}^{(1)}, ..., I_{\text{syn}}^{(m-1)}\}\), our goal is to synthesize the next frame \(I_{\text{syn}}^{(m)}\).
Select a group of static and flicker filters from a filter bank \(\mathbf{F} = \{F_k\}_{k=1}^{K_{v}} \in \Delta_{F}\), where \(K_{v}\) is the number of selected filters.
Compute \(\{h_{k}^{(s)}, k = 1, ..., K_{s}\}, \{h_{k}^{(v)}, k = 1, ..., K_{v}\}\) of \(\mathbf{I}_{\text{obs}}\), where \(K_{s}\) is the number of velocity clusters.
Initialize \(\beta_{k}^{(i)} \leftarrow 0, k = 1, ..., K; i = 1, ..., L\).
Initialize velocity vector \(V = \{v(x, y)\}_{(x, y) \in A}\) uniformly, and initialize \(I_{\text{syn}}^{(m)}\) by choosing intensities based on \(V\).
Repeat
Calculate \(\{h_{k}^{\text{syn}(s)}, k = 1, 2, ..., K_{s}\}\) and \(\{h_{k}^{\text{syn}(v)}, k = 1, 2, ..., K_{v}\}\) from \(I_{\text{syn}}\).
Update \(\beta_{k}, k = 1, ..., K\) and \(p(I; \mathbf{F}, \beta)\).
Sample \((I_{(m)}^{\text{syn}}, V_{(m)}^{\text{syn}}) \sim p(I; V; \mathbf{F}, \beta)\) by Gibbs sampler.
Until \(\frac{1}{2} \sum_{i=1}^{L} |h_{k}^{(i)} - h_{k}^{\text{syn}(i)}| \leq \epsilon\) for \(k = 1, 2, ..., K_{s} + K_{v}\).

Fig. 12 and Fig. 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, 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.

Fig. 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. Fig. 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 grey 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 $I^{obs} = \{I^{obs}_{1}, \ldots, I^{obs}_{m}\}$.

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

for $t=1:m$  
Reconstruct $I^{obs}_{(t)\Lambda_{ex}}$ by the sparse coding model with the selected primitives $B$ chosen from the dictionary $\Delta_B$ to get $I^{syn}_{(t)\Lambda_{ex}}$.

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

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

end for

Output the synthesized video $I^{syn}$.

Fig.2 shows this process as we introduced in section 1. Fig.16 shows three examples of video synthesis (YCbCr color space, 8 bits for grey 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 (Wang et al (2004)). The error for explicit representations is measured by the difference of pixel intensities

$$err^{ex} = \frac{1}{|\Lambda_{ex}|} \sum_{\Lambda_{ex}} \|I^{syn}_{\Lambda_{ex}} - I^{obs}_{\Lambda_{ex}}\|. \quad (29)$$

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

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

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

3.8 Perceptual Study

The error assessment of VPS is consistent with human perception. To support this claim, in this subsection, we
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%) | 20.82 (2.22%) |
| 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($I_{A_{in}}$) | Error($I_{A_{out}}$) |
|---------|--------------|---------------------|---------------------|
| 1       | 190x330      | 5.37%               | 0.59%               |
| 2       | 288x352      | 3.07%               | 0.16%               |
| 3       | 288x352      | 2.8%                | 0.17%               |

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

| Original | Synthesis |
|----------|-----------|
| (a) 44 | 37 | 6.7 | 12.3 | (a) 42 | 36 | 7 | 15 |
| (b) 32 | 45.3 | 8 | 14.7 | (b) 32.7 | 43.3 | 8.7 | 15.3 |
| (c) 9.7 | 7 | 4.4 | 39.3 | (c) 10 | 7.7 | 44.3 | 38 |
| (d) 10.3 | 12.3 | 35 | 42.3 | (d) 10.7 | 13.7 | 35.3 | 40.3 |

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

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

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

Table 5 The ANOVA results of analyzing recognition accuracies of original and synthesized textures. For each texture in every group, the corresponding F and p values are shown respectively.

| F/p | Group 1 | Group 2 | Group 3 |
|-----|---------|---------|---------|
| (a) 1.34/0.2520 | 0.06/0.3222 | 0.02/0.8813 |
| (b) 0.96/0.3305 | 0.70/0.4065 | 0.20/0.6583 |
| (c) 0.06/0.5100 | 0.15/0.6993 | 0.03/0.8563 |
| (d) 1.45/0.2366 | 1.08/0.3088 | 0.26/0.6151 |

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 Fig. 17 to 19, as the size of cropped videos gets larger, the confusion rate becomes lower, and actually when the size goes larger than $25 \times 25$ 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.

Table 6 The accuracy of differentiating the original video from the synthesized one in different scales. As the percentage is getting closer to 50%, it means it is harder to discriminate the original and synthesized videos for observers.

| 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      |

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 are 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 approach 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.

3.9 VPS adapting over scales, densities and dynamics

As it was observed in (Gong and Zhu (2012)) 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. Fig. 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 blob-like structures are much more prominent in small scale. So from this experiment, it is evident that VPS can choose proper representations automatically and furthermore, the representation patterns may reflect the scale of the videos.

3.10 VPS supporting action representation

VPS is also compatible with high-level action representation. By grouping meaningful explicit parts in a prin-
Table 7 The comparisons between the number of blob-like and edge-like primitives in 3 scales. For each scale, the numbers are compared in first 50, 100, 150 and 200 primitives respectively.

| 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    |

In order to show the relationship between VPS representation and effective high-level features, we take an KTH video (Schuldt et al (2004)) as an example. Fig.23 and Fig.24 show the spatial and temporal features of explicit regions respectively. In Fig.23, we compare VPS spatial descriptor with well-known HOG feature (Dalal and Triggs (2005)), 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 correspond-
ing 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 re-
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 re-
sults in 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 represen-
tations. Especially, it is very close to HOG and HOOF
descriptors, which are proven effective spatial and tem-
poral features respectively. The main difference is VPS
makes a local decision to give a more compact expres-
sion 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 fea-
tures 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 (Guo et al (2007)).
It can choose appropriate models automatically for video
representation.

Based on the model, we design an effective algo-
rithms for video synthesis, in which, explicit regions
are reconstructed by learned video primitives and im-
PLICIT regions are synthesized through a Gibbs sampling
procedure based on spatio-temporal statistics. Our ex-
periments shows that VPS is capable for video mod-
ing and representation, which has high compression
ratio and synthesis quality. Furthermore, it learns ex-
plicit and implicit expressions for meaningful low-level
vision features and is compatible with high-level struc-
tural 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 $\Delta R$ for video
primitives, which is more comprehensive. For high-level
application, we are applying the VPS features to object
and action representation and recognition.

Acknowledgements This work is done when Han is a visit-
ing 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 three grants
in China: 2007CB311002, NSFC 60832004, NSFC 61273020.

References

Adelson E, Bergen J (1985) Spatiotemporal energy
models for the perception of motion. JOSA A 2(2)
Besag J (1974) Spatial interactions and the statistical
analysis of lattice systems. J Royal Statistics Soc,
Series B 36
Black MJ, Fleet DJ (2000) Probabilistic detection and
tracking of motion boundaries. IJCV 38(3)
Bouthemy P, Hardouin C, Piriou G, Yao J (2006)
Mixed-state auto-models and motion texture mod-
ing. Journal of Mathematical Imaging and Vision
25(3)
Campbell NW, Dalton C, Gibson D, Thomas B (2002)
Practical generation of video textures using the auto-
regressive process. Proceedings of British Machine
Vision Conference pp 434–443
Chan AB, Vasconcelos N (2008) Modeling, clustering,
and segmenting video with mixtures of dynamic tex-
tures. PAMI 30(5)
Chaudhry R, Ravichandran A, Hager G, Vidal R (2009)
Histograms of oriented optical flow and binet-cauchy
kernels on nonlinear dynamical systems for the recog-
nition of human actions. CVPR
Comaniciu D, Ramesh V, Meer P (2003) Kernel-based
object tracking. PAMI 25(5)
Dalal N, Triggs B (2005) Histograms of oriented gradi-
ents for human detection. CVPR
Derpanis KG, Wildes RP (2010) Dynamic texture
recognition based on distributions of spacetime ori-
ented structure. CVPR
Doretto G, Chiuso A, Wu YN, Soatto S (2003) Dynamic
textures. IJCV 51(2)
Elder J, Zucker S (1998) Local scale control for edge
detection and blur estimation. PAMI 20(7)
Gong HF, Zhu SC (2012) Intratrackability: Characterizing
video statistics and pursuing video representations.
IJCV 97(33)
Guo C, Zhu SC, Wu YN (2007) Primal sketch: integrating texture and structure. CVIU 106(1)
Heeger D (1987) Model for the extraction of image flow. JOSA A 4(8)
Kim T, Shakhnarovich G, Urtasun R (2010) Sparse coding for learning interpretable spatio-temporal primitives. NIPS
Maccormick J, Blake A (2000) A probabilistic exclusion principle for tracking multiple objects. IJCV 39(1)
Mallat S, Zhang Z (1993) Matching pursuits with time-frequency dictionaries. IEEE TSP 41(12)
Marr D (1982) Vision. W H Freeman and Company
M N Dalal BT, Schmid C (2006) Human detection using oriented histograms of flow and appearance. ECCV
Olshausen BA (2003) Learning sparse, overcomplete representations of time-varying natural images. ICIP
Olshausen BA, Field DJ (1996) Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381
Ravichandran A, Chaudhry R, Vidal R (2009) View-invariant dynamic texture recognition using a bag of dynamical systems. CVPR
Schuldt C, Laptev I, Caputo B (2004) Recognizing human actions: a local svm approach. ICPR
Serby D, Koller-Meier S, Gool LV (2004) Probabilistic object tracking using multiple features. ICPR
Shi K, Zhu SC (2007) Mapping natural image patches by explicit and implicit manifolds. CVPR
Szummer M, Picard RW (1996) Temporal texture modeling. ICIP
Wang YZ, Zhu SC (2004) Analysis and synthesis of textured motion: particles and waves. PAMI 26(10)
Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error measurement to structural similarity. IEEE TIP 13(4)
Wildes R, Bergen J (2000) Qualitative spatiotemporal analysis using an oriented energy representation. ECCV
Wu YN, Zhu SC, Liu XW (2000) Equivalence of julesz ensemble and frame models. IJCV 38(3)
Yao B, Zhu SC (2009) Learning deformable action templates from cluttered videos. ICCV
Zhu SC, Wu YN, Mumford DB (1998) Filters, random field and maximum entropy (FRAME): towards a unified theory for texture modeling. IJCV 27(2)