Video Primal Sketch: A Generic Middle-Level Representation of Video

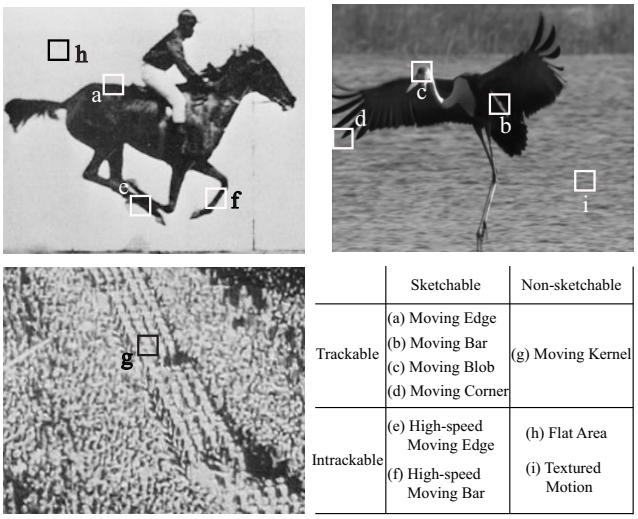
Zhi Han*† Zongben Xu* Song-Chun Zhu†
*Institute for Information and System Sciences
Xi'an Jiaotong University, China
†Department of Statistics
University of California, LA, USA
han.zhi@stu.xjtu.edu.cn zbxu@mail.xjtu.edu.cn sczhu@stat.ucla.edu

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 with spatio-temporal filters to implicitly represent textured motion, such as water and fire, by matching feature statistics, i.e. histograms. This paper makes three contributions: i) learning a dictionary of video primitives as parametric generative model; ii) studying the Spatio-Temporal FRAME (ST-FRAME) model for modeling and synthesizing textured motion; and iii) developing a parsimonious hybrid model for generic video representation. VPS selects the proper representation automatically and is compatible with high-level action representations. In the experiments, we synthesize a series of dynamic textures, reconstruct real videos and show varying VPS over the change of densities causing by the scale transition in videos.

1. Introduction

Videos of natural scenes contain vast varieties of motion patterns. Fig.1 shows some examples of different components in video. The simplest are sketchable and trackable motions, such as trackable corners, lines, and feature points, whose positions and shapes can be tracked during the movement. The most complex are textured motions, such as water, fire or grass. Essentially, these motion patterns can be classified based on their complexities measured by two criteria: i) sketchability[9], i.e. the possibility for representing a local patch by an explicit image primitive, and ii) trackability[8], i.e. the uncertainty of tracking an image patch using the entropy of posterior probability over velocities. Sketchable or trackable parts, defined as explicit region, are explicitly represented by primitives from a dictionary while non-sketchable and intrackable parts, defined as implicit region, are implicitly represented by pooling statistics.



	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

Figure 1. The four types of local video phenomenon characterized by two criteria, sketchability and trackability .

In the literature, there are two families of models for motion representations including trackable and intrackable motions respectively.

For trackable motions, there are mainly three types of representations, i.e. contours tracking[12], kernels tracking[5] and feature points (sparse[17] or dense[2]). Different primitives may be learned by generative models such as sparse coding[15, 11].

For textured motions or dynamic textures, numerous models have been studied, such as spatio-temporal auto-regressive (STAR) model[18], auto-regression moving-average (ARMA) model[7] or linear dynamical system (LDS) model[4][16] and mixed-state auto-models[3]. Although these models are successful in video synthesis, they have to record large amount of information. For example, the dynamic texture model[7] has to store a number of PCA components for synthesis and each of them is as large as an image frame. For a higher compression ratio, a parsimonious model has yet to be found to model textured motions.

The representations above are often manually selected

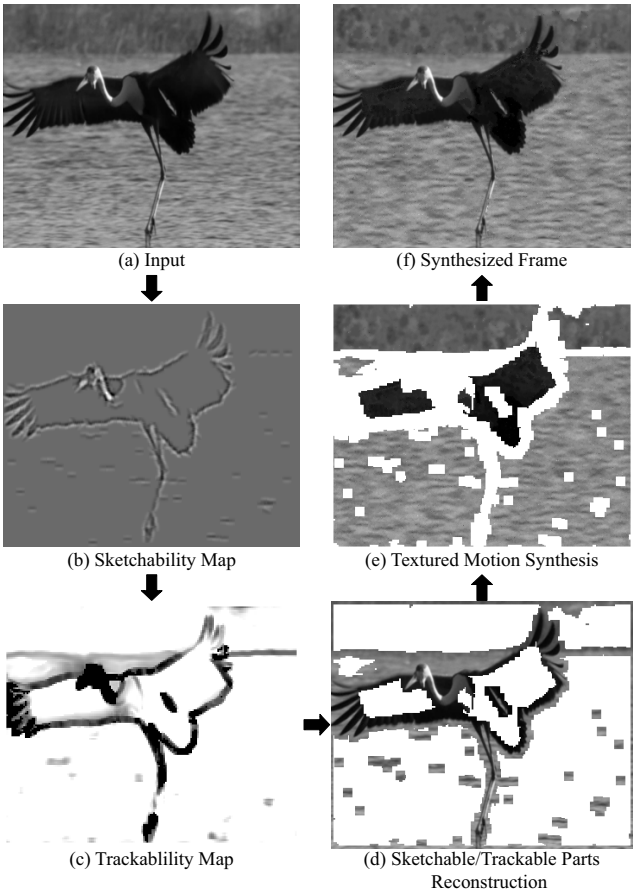


Figure 2. An example of Video Primal Sketch. (a) Input. (b) Sketchability map represented by filters. (c) Trackability map where heavier color means more trackable. (d) Reconstruction of explicit regions. (e) Synthesis for implicit regions (textured motions). (f) Synthesized frame by integrating explicit and implicit representations seamlessly.

for specific videos in different applications. There lacks a generic representation that can automatically select the proper models for different patterns of the video. Furthermore, both sketchability and trackability change over scales, densities, and dynamics, and thus a good video representation must change continuously in a long time-varying sequence.

In this work, we study a generic representation, called video primal sketch, by integrating two regimes of representations. Our goal is not only simply providing a parsimonious model for video compression and coding, but more importantly, it may support high level tasks such as motion tracking and action recognition. Fig.2 and Table 1 shows an example. An input frame from a video in (a) is separated into sketchable and non-sketchable areas by the sketchability map in (b), and trackable parts and intrackable regions by the trackability map in (c). Explicit regions including

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

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

sketchable or trackable parts are modeled by a sparse coding model and reconstructed with motion primitives in (d), and each implicit region of non-sketchable and intrackable parts has a textured motion which is synthesized by a generalized FRAME model in (e). The synthesis in (f) of this frame integrates the results from (d) and (e) seamlessly. The explicit representations are modeled with 3,600 parameters and the implicit representations are modeled with 420 parameters, which shows the parsimonious property of the model.

In this paper, we make the following contributions.

1. We define textured motions by spatio-temporal FRAME (ST-FRAME), which is a non-parametric MRF and generalizes the FRAME model[22] of texture with spatio-temporal filters to match the histograms of filter responses, which gives a much more parsimonious representation than the literature.

2. We learn a generative dictionary of motion primitives from videos, which is utilized for the reconstruction of explicit regions via a sparse coding model.

3. The two models are integrated in a hybrid representation, 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 and how it is compatible and consistent with high level action representation.

Our work is inspired by [8], which studies the statistical properties of videos over scale transition and defines intrackability as the entropy of local velocities, but does not give a unified model for video representation and synthesis. The latter is the focus of our work.

The remainder of this paper is organized as follows. In Section 2, we present the new framework of video primal sketch. Then we explain the algorithms for explicit representation, textured motion synthesis and video synthesis in Section 3, which followed by a series of experiments. The paper is concluded and discussed in Section 4.

2. Video primal sketch model

Marr conjectured a symbolic representation called primal sketch that should be parsimonious and sufficient to

reconstruct the original image without much perceivable distortions[14]. A mathematical model was later studied in [9], which successfully modeled hundreds of images by integrating sketchable structures and non-sketchable textures. In this section, we introduce 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 into explicit and implicit regions,

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

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

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

The explicit region Λ_{ex} and the implicit region Λ_{im} are modeled by sparse coding model and MRF model respectively in the following.

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 $n_{ex} = O(10^2)$),

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

Here $\Lambda_{ex,i} \subset \Lambda$ defines the domain of a "brick". A brick, denoted by $\mathbf{I}_{\Lambda_{ex,i}}$, is a spatio-temporal volume like a patch in images, eg. 11×11 pixels$\times$3 frames for trackable domains, or 11×11 pixels$\times$1 frame for intrackable domains, which can be represented by a motion primitive $B_i \in \Delta_B$,

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

B_i means the ith primitive from the primitive dictionary Δ_B, which fits the brick $\mathbf{I}_{\Lambda_{ex,i}}$ 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.

It is worth noting that a minority of noisy bricks are trackable over time but not sketchable; thus we cannot find specific shared primitives to represent them. Then $\mathbf{I}_{\Lambda_{ex,i}}$ is represented by recording the region in the video as B_i for it. Therefore, Δ_B is composed of two categories, common primitives Δ_B^{common} for sketchable bricks and special primitives $\Delta_B^{special}$ for non-sketchable ones. Fig.3(a) shows some examples from Δ_B.

Based on the representation in eqn(4), the probabilistic model of trackable parts in $\mathbf{I}_{\Lambda_{ex}}$ is defined as

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

where $\mathbf{B} = (B_1, ..., B_{n_{ex}})$ represents the selected primitive set and $n = |\Lambda_{ex,i}|$.

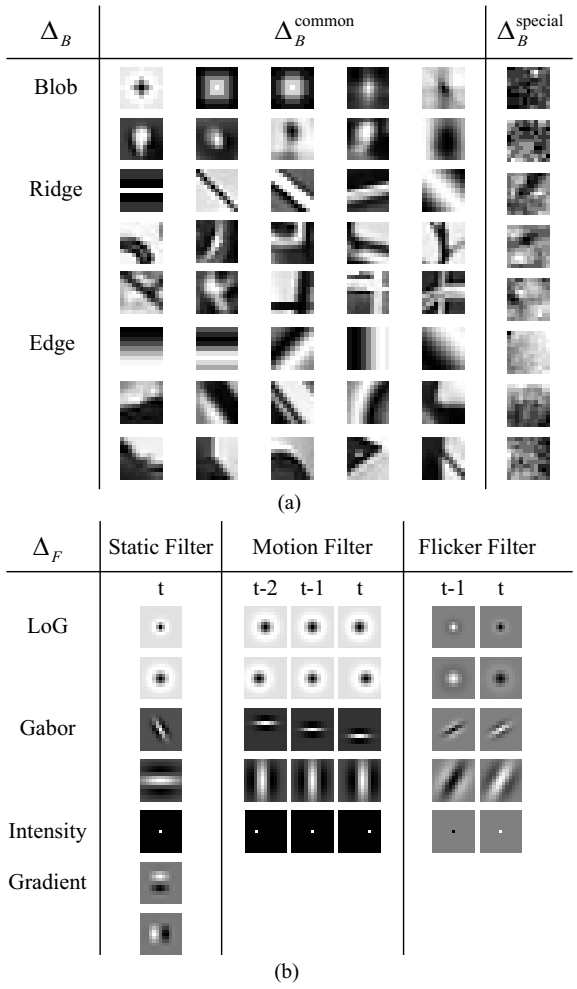


Figure 3. (a) Δ_B is a dictionary of primitives with velocities (u,v) ((u,v) is not shown), such as blobs, ridges, edges and special primitives; (b) Δ_F is a dictionary of spatio-temporal filters including static, motion and flicker filters.

2.2. Implicit representation by spatio-temporal FRAME

The implicit region Λ_{im} of video $\mathbf{I}$ can be segmented into n_{im} (usually $n_{im} = O(1)$) disjoint homogeneous textured motion regions,

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

According to the texture modeling literature[22], two textures are perceptually equivalent if they share the same histograms of a set of filters. Despite its success in modeling textures as a technically sound framework, it has not been applied to videos. We extend this concept to video by designing a dictionary of spatio-temporal filters Δ_F including three types shown as examples in Fig.3(b) and mentioned

in section 3.1. Each homogeneous textured motion region $\mathbf{I}_{\Lambda_{im,j}}$ is defined by a Julezs ensemble which is an equivalence class of videos,

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

where $\mathbf{h}_j = (h_{1,j}, ..., h_{K,j})$ is a series of 1D statistics characterizing the macroscopic properties of the textured motion pattern. $h_{k,j}$ is a histogram of filtered responses.

The filter set $\mathbf{F}$ is selected from Δ_F and for each of the filters $F_k \in \mathbf{F}$, the spatio-temporal filter response of $\mathbf{I}$ at $(x, y, t) \in \Lambda_{im,j}$ is $F_k * \mathbf{I}(x, y, t)$. The convolution is over spatial and temporal domain. By pooling the filter responses over all $(x, y, t) \in \Lambda_{im,j}$, we can obtain a number of 1D histograms

$$H_k(\mathbf{I}_{\Lambda_{im,j}}) = H_k(z; \mathbf{I}_{\Lambda_{im,j}}) \quad (7)$$
$$= \frac{1}{|\Lambda_{im,j}|} \sum_{(x,y,t)\in\Lambda_{im,j}} \mathbf{1}(z - F_k * \mathbf{I}(x, y, t)), k = 1, ..., K.$$

where $\mathbf{1}()$ is an indicator function and z is the index of the histogram bins in discrete form. Following the FRAME model[22], the statistical model of textured motion $\mathbf{I}_{\Lambda_{im,j}}$ can be written in the form of the following Gibbs distribution,

$$p(\mathbf{I}_{\Lambda_{im,j}}; \mathbf{F}, \beta) \propto \exp\{-\sum_k \langle \beta_{k,j}, H_k(\mathbf{I}_{\Lambda_{im,j}})\rangle\}. \quad (8)$$

The filters in $\mathbf{F}$ are chosen one by one from the filter bank Δ_F by maximizing the information gain provided by the filters

$$F_k^* = \arg\max_{F_k\in\Delta_F} \|H_k^{syn} - H_k^0\|. \quad (9)$$

H_k^0 and H_k^{syn} are the response histograms of F_k before and after synthesizing $\mathbf{I}_{\Lambda_{im,j}}$ by adding F_k respectively. This shows the process of reproducing the observed histogram of the filter responses which describe the texture best.

Following the distribution form of eqn(8), the probabilistic model of implicit parts of $\mathbf{I}$ is defined as

$$p(\mathbf{I}_{\Lambda_{im}}; \mathbf{F}, \beta) \propto \prod_{j=1}^{n_{im}} p(\mathbf{I}_{\Lambda_{im,j}}; \mathbf{F}, \beta). \quad (10)$$

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

In the experiments described later, we demonstrate this model can synthesize a range of dynamic textures by matching the histograms of filter responses.

2.3. Hybrid model for video representation

In summary, by taking the explicit parts as boundary conditions for the implicit regions, the probabilistic models for $\mathbf{I}_{\Lambda_{ex}}$ and $\mathbf{I}_{\Lambda_{im}}$ are given by eqn(5) and eqn(10) respectively

$$\mathbf{I}_{\Lambda_{ex}} \sim p(\mathbf{I}_{\Lambda_{ex}}; \mathbf{B}, \alpha), \mathbf{I}_{\Lambda_{im}} \sim p(\mathbf{I}_{\Lambda_{im}} | \mathbf{I}_{\partial\Lambda_{im}}; \mathbf{F}, \beta). \quad (11)$$

As an extension of the image primitive sketch model[9], we have the following probability model for the video primal sketch representation,

$$p(\mathbf{I}|\mathbf{B}, \mathbf{F}, \alpha, \beta) = \quad (12)$$
$$\frac{1}{Z} \exp\{-\sum_{i=1}^{n_{ex}} \sum_{(x,y,t)\in\Lambda_{ex,i}} \frac{(\mathbf{I}(x, y, t) - \alpha_i B_i(x, y, t))^2}{2\sigma_i^2}$$
$$- \sum_{j=1}^{n_{im}} \sum_{k=1}^{K} \langle \beta_{k,j}, H_k(\mathbf{I}_{\Lambda_{im,j}} | \mathbf{I}_{\partial\Lambda_{im,j}})\rangle\}.$$

where Z is the normalizing constant.

We denote by $VPS = (\mathbf{B}, \mathbf{H})$ as the representation for the video $\mathbf{I}_\Lambda$, where $\mathbf{H} = (\{h_{k,1}\}_{k=1}^{K_1}, ..., \{h_{k,n_{im}}\}_{k=1}^{K_{n_{im}}})$ are the histograms described by $\mathbf{F}$. The solution of VPS is obtained by maximizing the posterior probability

$$VPS^* = \arg\max_{VPS} p(VPS|\mathbf{I}_\Lambda). \quad (13)$$

following the video primal sketch model in eqn(12).

Table 1 gives an example of VPS composition. For a given frame of the size 288×352, 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 $K_1 = 11$, $K_2 = 12$ and $K_3 = 5$ filters respectively. As each histogram has 15 bins, the number of the parameters for the implicit representation is 420.

3. Algorithms and experiments

3.1. Spatio-temporal filters

In the literature, spatio-temporal filters were used for motion information extraction[1] and optical flow estimation[10], pattern categorization[20], dynamic texture recognition[6]. In the experiments, we choose spatio-temporal filters Δ_F as shown in Fig.3(b). It includes three types:

Static filters. Laplacian of Gaussian (LoG), Gabor, gradient, or intensity filter on a single frame. They capture statistics of spatial features.

Motion filters. Moving LoG, Gabor or intensity filters in different velocities and orientations over three frames. Specifically, Gabor motion filters move perpendicularly to their orientations.

Flicker filters. One static filter with opposite signs at two frames. It contrasts the static filter responses between two consequent frames and detect the change of dynamics.

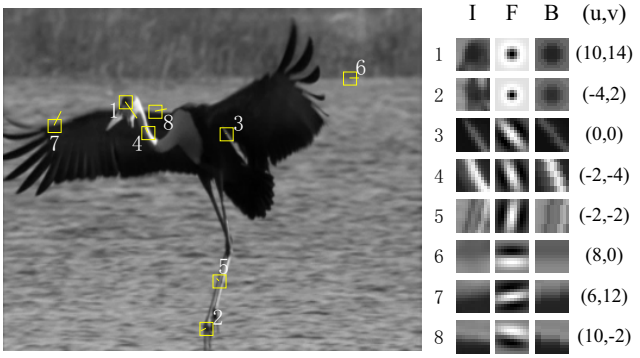


Figure 4. Some examples of primitives in a frame of video. Each group shows the original local image $\mathbf{I}$, the best fitted filter $\mathbf{F}$, the fitted primitive $\mathbf{B} \in \Delta_B$ and the velocity (u, v), which represents the motion of $\mathbf{B}$.

For implicit representation, the filters are 7×7 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.5.

3.2. Learning motion primitives

After computing the sketchability[9] and trackability[8] of one frame, we can extract explicit regions in the video. By calculating all the coefficients of each part with motion primitives from the primitive bank, $\alpha_{i,j} = \langle \mathbf{I}_{\Lambda_{tr,i}}, B_j \rangle$, all the $\alpha_{i,j}$ are ranked from high to low. We select the primitive with the highest coefficient each time to represent the corresponding domain. The algorithm is similar to matching pursuit[13].

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 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 orientation and velocity 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. In Fig.3, we show some examples of different types of primitives, such as blob, ridge and edge. Fig.4 shows some examples of reconstruction by motion primitives. In each group, the original local image, the filter that is supposed to fit, 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.

3.3. Synthesizing textured motions

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 eqn(8), $p(\mathbf{I}_{\Lambda_0}|\mathbf{I}_{\partial\Lambda_0}; \mathbf{F}, \beta) \propto \exp\{-\sum_k \langle \beta_k, H_k(\mathbf{I}_{\Lambda_0})\rangle\}$, where parameters $\beta_k = \{\beta_k^{(i)}\}_{i=1}^L \in \beta$ are the discrete forms of the potential function $\beta_k()$ learned from input videos[22].

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 can converge to a stationary process with distribution $p(\mathbf{I}; \mathbf{F}, \beta)$.

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

Algorithm 1. Synthesis for Textured Motion

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

Fig.5 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.6, we show three groups of textured motion (4 bits) synthesis by algorithm 1: ocean (a), water wave (b) and fire (c). In each group, as time passes, the synthesized frames are getting different more and more from the observed one. It is caused by the stochasticity of textured motions. However, the synthesized and observed sequences are quite similar to human perception after matching the histograms of a small set of filter responses. Fig.7 shows that as $\mathbf{I}^{syn}_{(m)}$ changes from white noise (Fig.5(a)) to the final synthesized result (Fig.5(e)), the histograms of filter responses become matched with the observed ones.

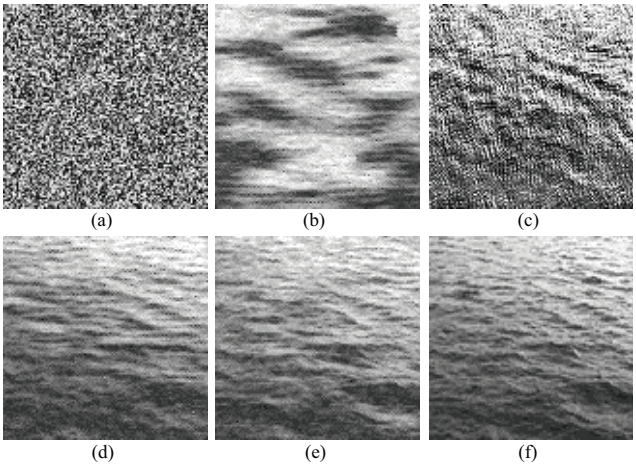


Figure 5. Synthesis for one frame of the ocean textured motion. (a) Initial uniform white noise image. (b) Synthesized frame with only static filters. (c) Synthesized frame with only motion filters. (d) Synthesized frame with both of static and motion filters. (e) Synthesized frame with all of the 3 types of filters. (f) The original observed frame.

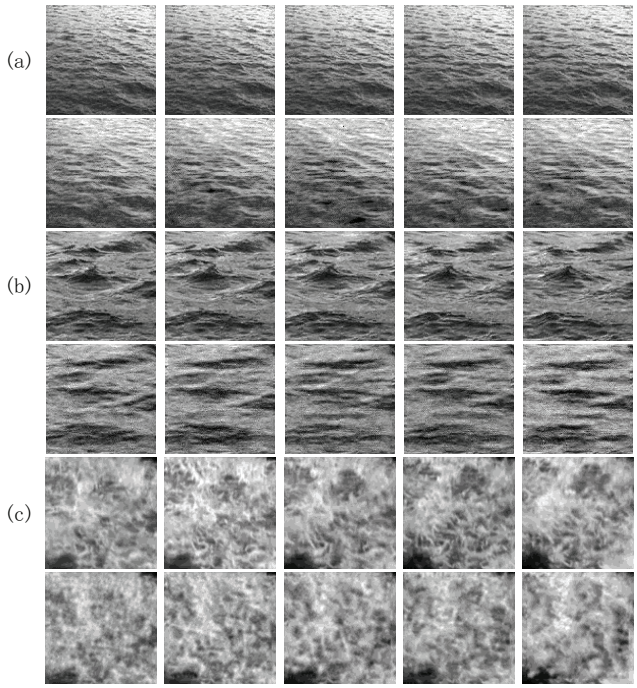


Figure 6. Textured motion synthesis examples. For each group, the top row are the original videos and the bottom row shows the synthesized ones. (a) Ocean. (b) Water wave. (c) Fire.

Table 2 shows the comparison of compression ratios between ST-FRAME and the dynamic texture model[7]. 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.

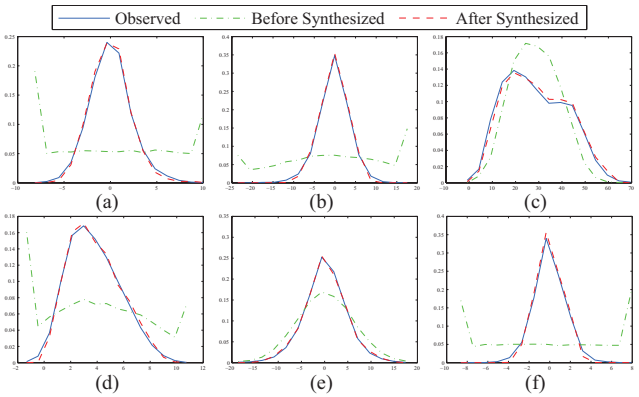


Figure 7. Matching of histograms of spatio-temporal filter responses for Ocean. The filters are (a) Static LoG(5×5). (b) Static Gradient(vertical). (c) Motion Gabor(6,150°). (d) Motion Gabor(2,60°). (e) Motion Gabor(2,0°). (f) Flicker LoG(5×5).

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%)

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[7].

3.4. Synthesizing videos with VPS

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

Algorithm 2. Video Primal Sketch

Input a video $\mathbf{I}^{obs}$.

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

Reconstruct $\mathbf{I}_{\Lambda_{ex}}$ by the sparse coding model with the selected primitives $\mathbf{B}$ chosen from the dictionary Δ_B.

For each region of homogeneous textured motion, using $\mathbf{I}_{\Lambda_{ex}}$ as boundary condition, synthesize $\mathbf{I}_{\Lambda_{im}}$ by ST-FRAME model with the selected spatio-temporal filters $\mathbf{F}$ chosen from the filter bank Δ_F.

Integrate the explicit representations and implicit representations to get the synthesized video $\mathbf{I}^{syn}$.

Fig.2 shows this process as we introduced in section 1. Fig.8 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 see 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[19]. The error for explicit representations is measured by the difference of pixel intensities $err^{ex} = \frac{1}{|\Lambda_{ex}|}\sum_{\Lambda_{ex}} \|I^{syn} - I^{obs}\|$, while for implicit representations, the error is given by the difference of filter response histograms $err^{im} = \frac{1}{n_{im}\times K}\sum_{n_{im},K} \|H_k(I^{syn}_{\Lambda_{im,j}}) - H_k(I^{obs}_{\Lambda_{im,j}})\|$. Table 4 shows the quality assessments of the synthesized videos.

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

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

Table 4. Error assessment of synthesized videos.

3.5. VPS over scales, densities, dynamics

As it is observed in [8], the optimal visual representation at a region is affected by distance, density and dynamics. In Fig.9, 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.

3.6. VPS supports action representation

VPS is also compatible with high-level action representation. By grouping meaningful explicit parts in a principled way, it will represent an action template. In Fig.10, (b) is the action template given by the deformable action template model[21] from the video shown in (a). (c) shows a rough 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. This demonstrates that VPS has a shared representation of videos for high-level tasks.

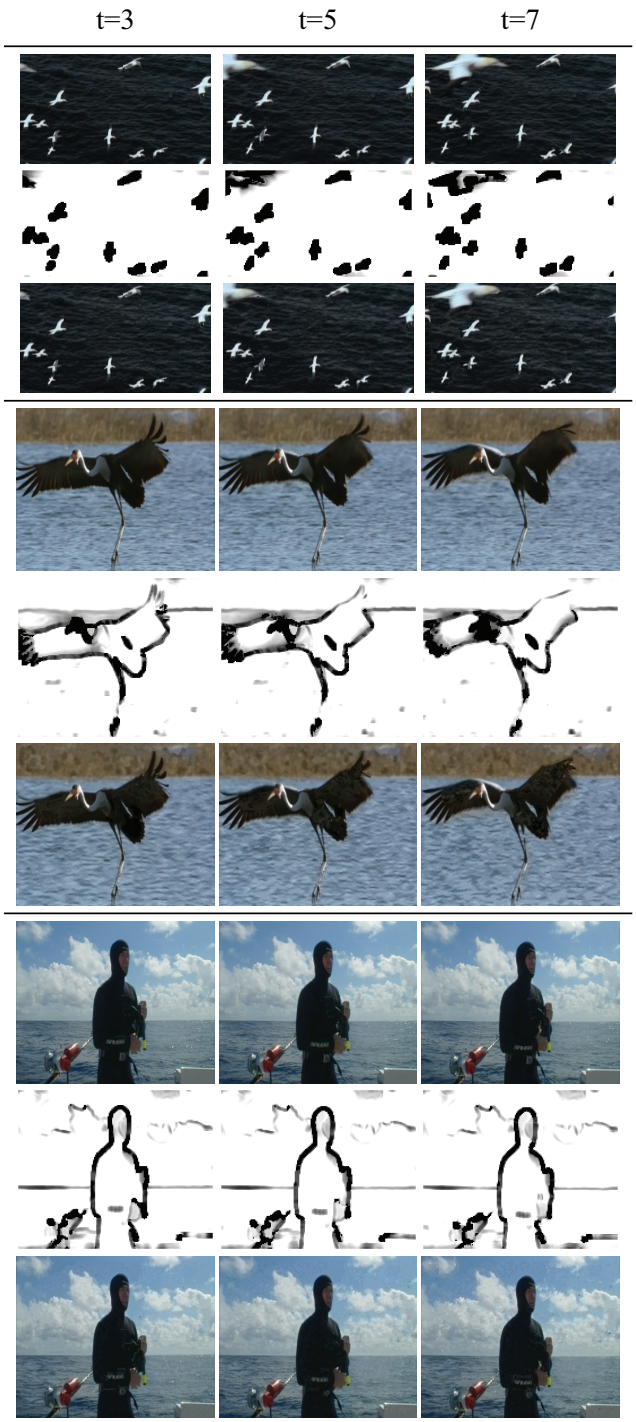


Figure 8. Video synthesis. For each experiment, Row 1: original frames; Row 2: trackability maps; Row 3: synthesized frames.

4. Conclusion and discussion

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

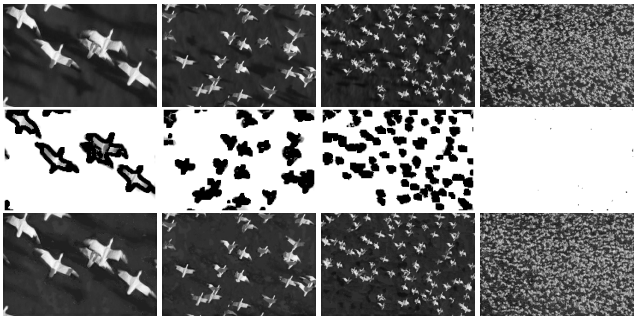


Figure 9. Representation switches triggered by scale. Row 1: observed frames; Row 2: trackability maps; Row 3: synthesized frames.

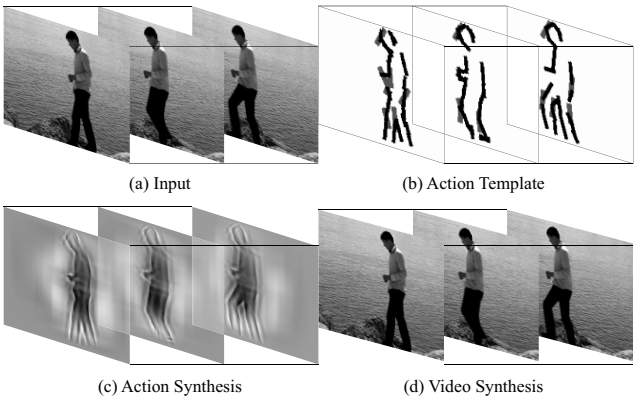


Figure 10. Action representation by VPS. (a) The input video. (b) Action template obtained by the deformable action template[21]. (c) Action synthesis by filters. (d) Video synthesis by VPS.

model for explicitly representing sketchable and trackable regions and a ST-FRAME model for implicitly representing textured motions.

This model is also compatible with high-level representations, e.g. action recognition where the popular features are HOG (Histogram of Oriented Gradients) for appearance and HoF (Histogram of Optical-flow) for motion. Specifically, sketchability and trackability in VPS provide spatial and temporal statistical information of the video respectively as the HOG and HoF features do. The difference is that VPS moves one step further by making local decisions to represent those regions, which have low entropy in their appearance or motion statistics, with explicit primitives. In future works, we will learn a richer dictionary of the video primitives and improve the model by adding histograms of velocities as a stronger temporal constraint, which is consistent with the concept of trackability. This will hopefully give better time-continuity over frames in video synthesis and be used for action recognition and representation in the high level.

Acknowledgements: 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 two grants in China: 2007CB31100 and NSFC 60832004.

References

[1] E. Adelson and J. Bergen. Spatiotemporal energy models for the perception of motion. *JOSA A*, 2(2), 1985.
[2] M. J. Black and D. J. Fleet. Probabilistic detection and tracking of motion boundaries. *IJCV*, 38(3), 2000.
[3] P. Bouthemy, C. Hardouin, G. Piriou, and J. Yao. Mixed-state auto-models and motion texture modeling. *Journal of Mathematical Imaging and Vision*, 25(3), 2006.
[4] A. B. Chan and N. Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *PAMI*, 30(5), 2008.
[5] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *PAMI*, 25(5), 2003.
[6] K. G. Derpanis and R. P. Wildes. Dynamic texture recognition based on distributions of spacetime oriented structure. *CVPR*, 2010.
[7] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *IJCV*, 51(2), 2003.
[8] H. F. Gong and S. C. Zhu. Intrackability : characterizing video statistics and pursuing video representations. Technical Report, UCLA.
[9] C. Guo, S. C. Zhu, and Y. N. Wu. Primal sketch: integrating texture and structure. *CVIU*, 106(1), 2007.
[10] D. Heeger. Model for the extraction of image flow. *JOSA A*, 4(8), 1987.
[11] T. Kim, G. Shakhnarovich, and R. Urtasun. Sparse coding for learning interpretable spatio-temporal primitives. *NIPS*, 2010.
[12] J. Maccormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. *IJCV*, 39(1), 2000.
[13] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE TSP*, 41(12), 1993.
[14] D. Marr. Vision. *W. H. Freeman and Company*, 1982.
[15] B. A. Olshausen. Learning sparse, overcomplete representations of time-varying natural images. *ICIP*, 2003.
[16] A. Ravichandran, R. Chaudhry, and R. Vidal. View-invariant dynamic texture recognition using a bag of dynamical systems. *CVPR*, 2009.
[17] D. Serby, S. Koller-Meier, and L. V. Gool. Probabilistic object tracking using multiple features. *ICPR*, 2004.
[18] M. Szummer and R. W. Picard. Temporal texture modeling. *ICIP*, 1996.
[19] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error measurement to structural similarity. *IEEE TIP*, 13(4), 2004.
[20] R. Wildes and J. Bergen. Qualitative spatiotemporal analysis using an oriented energy representation. *ECCV*, 2000.
[21] B. Yao and S. C. Zhu. Learning deformable action templates from cluttered videos. *ICCV*, 2009.
[22] S. C. Zhu, Y. N. Wu, and D. B. Mumford. Filters, random field and maximum entropy (FRAME): towards a unified theory for texture modeling. *IJCV*, 27(2), 1998.