Primal Sketch: Integrating Texture and Structure

Cheng-en Guo, Song-Chun Zhu, and Ying Nian Wu

Departments of Statistics and Computer Science
University of California, Los Angeles
Los Angeles, CA 90095

{*cguo*, *sczhu*, *ywu@stat.ucla.edu*}

Abstract

Following Marr's insight, we propose a generative image representation called *primal sketch*, which integrates two modeling components. The first component explains the structural part of an image, such as object boundaries, by a hidden layer of image primitives. The second component models the remaining textural part without distinguishable elements by Markov random fields that interpolate the structural part of the image. We adopt an artist's notion by calling the two components "sketchable" and "non-sketchable" parts respectively. A dictionary of image primitives are used for modeling structures in natural images, and each primitive is specified by variables for its photometric, geometric, and topological attributes. The primitives in the image representation are not independent but organized in an *sketch graph*. This sketch graph is modeled by a spatial Markov model that enforces Gestalt organizations. The inference of the sketch graph consists of two phases. Phase I sequentially adds the most prominent image primitives in a procedure similar to matching pursuit. Phase II edits the sketch graph by a number of graph operators to achieve good Gestalt organizations. Experiments show that the primal sketch model produces satisfactory results for a large number of generic images. The primal sketch model is not only a parsimonious image representation for lossy image coding, but also provides a meaningful mid-level generic representation for other vision tasks.

Preprint no. 416, Department of Statistics, UCLA, 2005. Some materials in this paper have been published in the Proceedings of the Int'l Conf. on Computer Vision, 2003, and the Workshop on Generative Model Based Vision, 2004.

1 Introduction

1.1 Motivation: image representation in early vision

In the early stage of visual perception, an image may be divided into two components – the structural part with noticeable elements called "textons" by Julesz [21] or "tokens" by Marr [25], and the textural part without distinguishable elements in preattentive vision. See Figure (1) for an illustration. The structural part are objects at near distance, such as tree trunks and branches, whose positions and shapes can be clearly perceived. In contrast, the textural part are objects at far distance whose structures become indistinguishable and thus yield various texture impressions. Obviously the notion of being far or near is relative to the object sizes. As Figure (1) illustrates, in natural scenes the two parts are not only seamlessly interweaved, but our perception can also switch between structure and texture depending on the viewing distance or even the change of our focus point (eye fixation) due to the non-uniform resolution of our retina.

Figure 1: Natural image with interweaving textures and structures.

The modeling of texture and structure has been a long standing puzzle in the study of early vision. Julesz first proposed in the 1960s a texture theory [20] and conjectured that a texture is a set of images sharing some common statistics on some features related to human perception. Later he switched to a texton theory [21] and identified bars, edges, terminators as textons – the atomic elements in early vision. Marr summarized Julesz's theories and other experimental results and proposed a primal sketch model in his book [25] as a "symbolic" or "token" representation in terms of image primitives. Marr argued that this symbolic representation should be parsimonious and sufficient to reconstruct the original image without much

perceivable distortion.

Despite many inspiring observations, Marr's description provided neither an explicit mathematical formulation nor a rigorous definition of the primal sketch model.

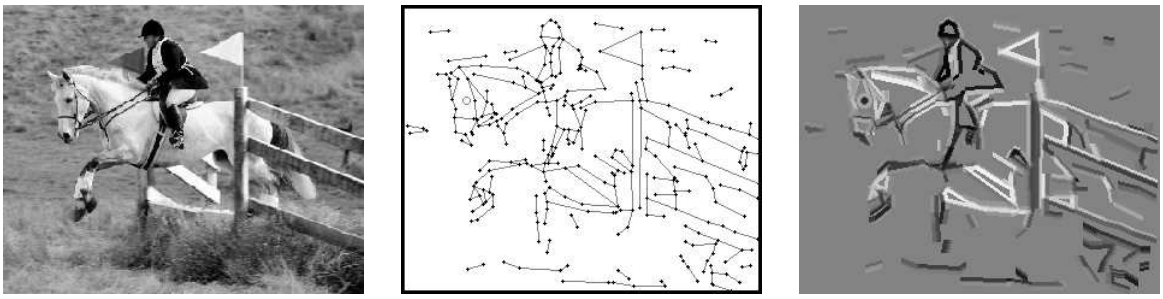
Since the 1980s, the studies of image modeling followed two distinct paths which represent two prevailing mathematical theories for generative image modeling respectively.

The first theory is a two-layer generative model originated from computational harmonic analysis which represents an image by a linear superposition of image bases selected from a dictionary – often over-complete like various wavelets [8], ridgelets [4], image pyramids [30], and sparse coding [27] [28]. Each image base is supposed to represent some image features with hidden variables describing their locations, orientations, scales, and intensity contrasts. The image is reconstructed with minimum error on the pixel intensities.

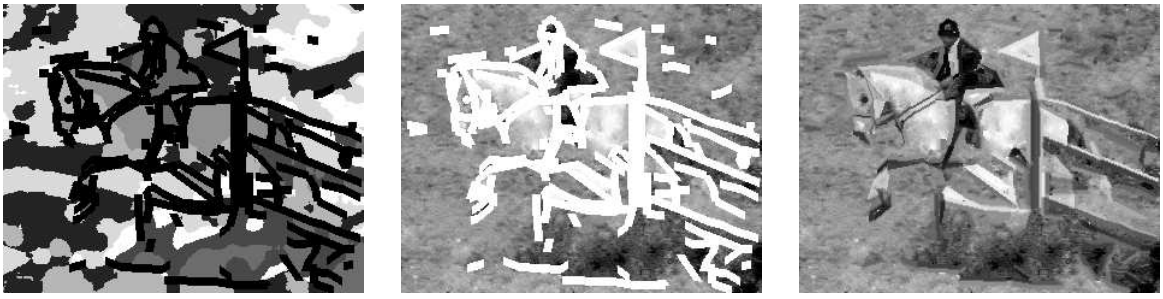
The second theory is the Markov random field (MRF) model originated from statistical mechanics. It represents a visual pattern by pooling the responses of a bank of filters over all locations into some statistical summary like the histograms that are supposed to represent our texture impressions. On large image lattices, a Julesz ensemble [32] is defined as a perceptual equivalence class where all images in the equivalence class share identical statistics. The statistics or texture impressions are the macroscopic properties and the differences between microscopic states (i.e. image instances in the Julesz ensemble) are ignored. In other words, all images in this equivalence class are perceptually the same, replacing one by the other does not cause perceivable distortion, although the two images may have large difference in pixel by pixel comparison. The image patches within local windows are shown to follow some MRF models [34].

1.2 Primal sketch model and an example

Following the insights of Marr and Julesz, we propose a *primal sketch model* that integrates both modeling schemes mentioned above as a mathematical formulation for Marr's primal sketch model. We adopt an artist's notion by calling the image portion with distinguishable structures as *sketchable*, and the remaining portion without distinguishable elements is said to be *non-sketchable*. An example is shown in Figure 2. Given an input image $\mathbf{I}$ in (a), which is defined on a lattice Λ, we compute a sketch graph S_{sk} in (b). Each vertex as well as line segment in this graph corresponds to an image primitive shown in Figure 3. These primitives are occluding patches with a number of landmarks (see the leftmost column of Figure 3) that are aligned to the graph. The variables describing the primitives become the attributes of the sketch graph. Thus



(a) input image $\mathbf{I}$ (b) sketch graph S_{sk} (c) sketchable image $\mathbf{I}_{\Lambda_{\text{sk}}}$



(d) texture regions $S_{\Lambda_{\text{nsk}}}$ (e) synthesized textures $\mathbf{I}_{\Lambda_{\text{nsk}}}$ (f) synthesized image $\mathbf{I}^{\text{syn}}$

Figure 2: An example of the primal sketch model. (a) An input image $\mathbf{I}$. (b) The sketch graph S_{sk} computed from the image $\mathbf{I}$. Each vertex in the graph correspond to an image primitive shown in Figure 3. These primitives are occluding patches rather than linear additive bases. (c) The sketchable part of the image by aligning the primitives to the graph. (d) The remaining non-sketchable portion is segmented into a small number of homogeneous texture regions. (e) Synthesized textures on these regions. (f) The final synthesized image integrating seamlessly the sketchable and non-sketchable parts.

we synthesize a partial image $\mathbf{I}_{\Lambda_{\text{sk}}}$ in (c) for the sketchable part of the image, where Λ_{sk} collects the sketchable pixels. Clearly this corresponds to the structural part of the image. The remaining textural part is said to be non-sketchable and is segmented into a small number of homogeneous texture regions. Each region is shown by a grey level in (c) and statistics (histograms of responses from 5-7 small filters) are extracted as the statistical summary. Then we synthesize textures on these regions by simulating the Markov random field (MRF) models which reproduce the statistical summaries in these regions. The MRF models interpolate the sketchable part of the image. The non-sketchable part of the image is denoted as $\mathbf{I}_{\Lambda_{\text{nsk}}}$, where Λ_{nsk} collects the non-sketchable pixels. The final synthesized image is shown in (f) which integrates seamlessly the sketchable and non-sketchable parts.

A set of image primitives are constructed for modeling the structures in natural images.

Some examples are shown in Figure 3. They are sorted in increasing order of connectivity degree. These primitives are described in a parametric form with topological (degree of connectivity), geometric (positions, orientations and scales) and photometric (contrast and blurring) attributes. The probability distributions of these attributes can be learned from a given image data set in supervised learning.

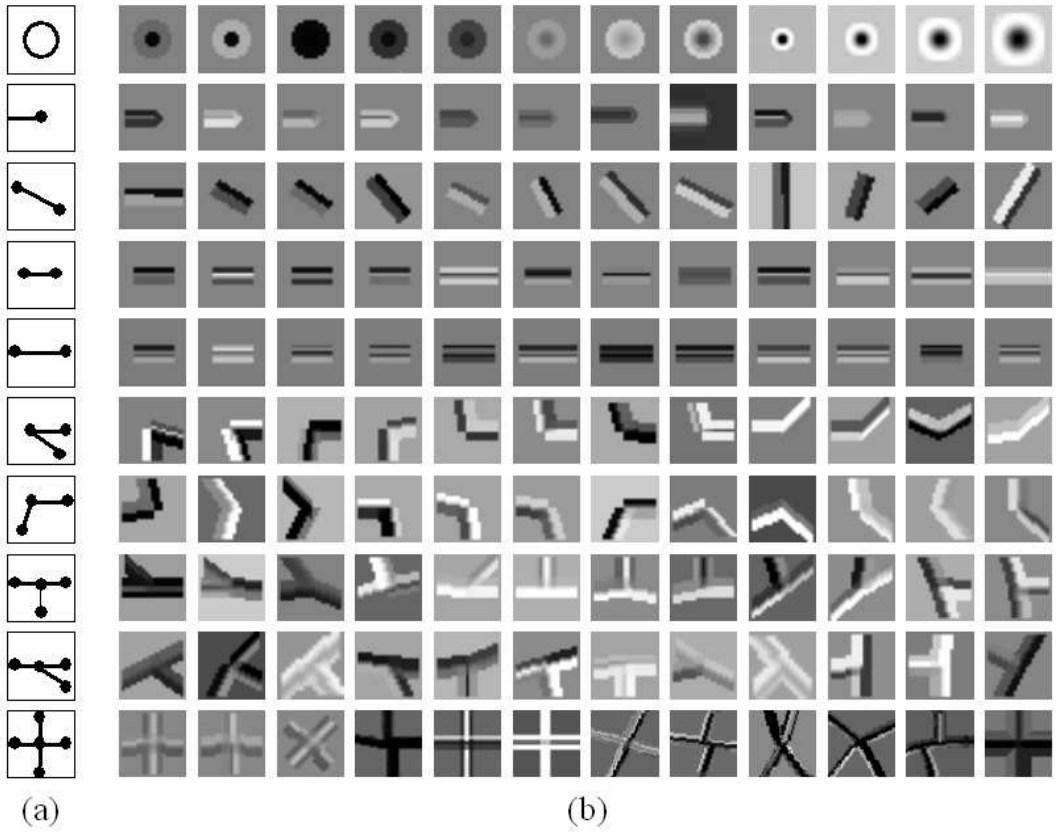


Figure 3: Samples from the visual primitive dictionary, consisting of eight types: blobs, end points, edges, ridges, multi-ridges, corners, junctions and crosses of different degrees. (a) The landmarks on the patches for topological and geometric attributes. (b) The photometric representation of the patches.

Table 1 counts the number of parameters for describing the primal sketch model presented in Figure 2. The input image is of 300×240 pixels, of which $18,185$ pixels (around 25%) are considered by our model as sketchable. The sketch graph has 152 vertices and 275 edges/ridges or strokes (primitives with degree 2) and the attributes are coded by $1,421$ bytes. Then the non-sketchable pixels are represented by 455 parameters or less. The parameters are 5 filters for 7 texture regions and each pools a 1D histogram of filter responses into 13 bins. Together with the codes for the region boundaries, total coding length for the textures is $1,628$ bytes.

The total coding length for the synthesized image in Figure 2.(f) is $3,049$ bytes or 0.04 byte per pixel. This is about $1:25$ compression ratio to JPG2000. It should be noted that the coding length is roughly computed here by treating the primitives as being independent. If one accounts for the dependence in the graph and applies some arithmetic compression schemes, a higher compression rate can be achieved. Similar results have been obtained for a large set of generic images.

	coding description	coding length (bits)
Sketch graph		
	vertices 152	152*2*9=2,736
	strokes (275)	275*2*4.7=2,585
Sketch image	profiles (275)	275*(2.4*8+1.4*2)=6,050
Total for sketchable	18,185 pixels	11,371 bits=1,421 bytes
Region boundaries	3659 pixels	3659*3 = 10,977
MRF parameters	texture regions (7)	7*5*13*4.5 = 2,048
Total for non-sketchable	41,815 pixels	13,025 bits = 1,628 bytes
Total for whole image	72,000 pixels	3,049 bytes, 0.04 byte/pixel

Table 1: The approximate coding length by the primal sketch model for the example in Figure 2.

We propose a *sketch pursuit* algorithm to compute the hidden variables – the attributed sketch graph. The algorithm consists of two phases. Phase I sequentially adds the most prominent strokes (edges/ridges) using a procedure similar to matching pursuit based on local fitness. Phase II edits the sketch graph by a number of graph operators to achieve good Gestalt organizations and resolve local ambiguities. Table 2 shows 10 pairs of reversible graph operators used in the graph editing process. It is observed in our experiments that the long edges in the graph can often be computed reliably. However the images around the vertices are often blurred and confusing, and thus the editing process uses global graph properties to fix ambiguities at vertices. Currently we adopt a greedy method, which accepts the graph operation if the coding length is decreased for that operator or a series (3 to 5) of operators.

We show a number of results for image sketching and reconstruction in Figures 15, 16 and 17.

1.3 Relations with previous work and contributions

The primal sketch model presented in this paper is related to many other generative models in the literature. In the following, we summarize the main contributions of our work in relation to previous work.

First, many image elements have been pursued in the generative models. These generative elements are mostly linear additive models, such as wavelets [8], image pyramid [30], sparse coding [27][28], various type of component analysis (PCA/ICA/TCA) [1][14], and textons [33]. In contrast, the image primitives in our primal sketch model are occluding patches with landmarks for topological and geometric alignment, and are far sparser than the linear additive models. In order words, the primal sketch model has a much more over-complete dictionary of primitives, and needs smaller (sparser) number of primitives to reconstruct an image. These primitives have landmarks that resemble the AAM (apparent active model) for face modeling using landmarks [9]. In comparison, the primitives are more generic and sufficient to construct any graphs and images for early vision tasks.

Second, the primal sketch model integrates two levels of Markov random fields within a generative model. This is a direct extension to the authors' previous work [18] on modeling texture patterns by integrating descriptive and generative models. The primal sketch model has a seamless integration while our previous work may have artifacts along the two textons/texture layers. The current model is generally applicable to all natural images and thus is broader in scope than previous work on textures.

Third, Gestalt field model on the sketch graph is what Mumford and Fridman called the mixed random field [15]. It is different from conventional MRF because of its two properties. (1) The vertices in the graph are inhomogeneous primitives. Each may have different degrees of connectivity, such as isolated point, terminators, bars, three-way junctions etc. (2) The neighborhood of each vertex has to be inferred from images as hidden variables in contrast to the fixed 4-nearest-neighbor system on lattices.

Fourth, the primal sketch model is not only a parsimonious image representation, but also provides a meaningful and generic representation for middle level vision– similar to the token representation conjectured by Marr [25]. Such representation can be used effectively for later vision tasks. We can study shape-from-X, such as stereo, shading and motion, based on the two level representation instead of working on the image lattices. Also by grouping subgraphs in the

sketch, we can represent a large range of objects for recognition.

The computed sketches could be used in a number of applications, such as low bit lossy image coding, automatic cartoon generation, non-photorealistic image rendering.

1.4 Organization of the rest of the paper

The rest of the paper is organized as follows. We first introduce the two generative modeling schemes for the structural and textural parts in Section 2. Section 3 presents the primal sketch model. Section 4 discusses the sketch pursuit algorithm. Section 5 shows the results of the experiments. Then we conclude the paper in Section 6.

2 Background: two generative modeling schemes

In this section, we review two popular schemes for generative modeling– image coding theory and Markov random field as the background for studying the primal sketch model.

2.1 Image coding theories: harmonic analysis, wavelets, and sparse coding

The modern image coding theories can be traced back to Fourier and harmonic analysis. Let $\mathbf{I}$ be an image defined on a lattice Λ, the image coding theory assumes that $\mathbf{I}$ is the weighted sum of a number of *image bases* B_k indexed by k for its position, scale, orientation etc. Thus one obtains a "generative model",

$$\mathbf{I} = \sum_{k=1}^{K} c_k B_k + \epsilon, \quad B_k \in \Delta_B,\ C = \{c_k\} \sim p(C), \tag{1}$$

where B_k are selected from a dictionary Δ_B, c_k are the coefficients, and ϵ is the residual error modeled by Gaussian white noise. When Δ_B consists of the orthonormal sine waves, the model reduces to Fourier analysis. In the literature, people often adopt over-complete basis (i.e. $|\Delta| \geq |\Lambda|$) with elements localized in space and frequency, so that $\mathbf{I}$ can be approximated by a small number of elements (i.e. $K \ll |\Lambda|$). In other words, only a fraction of elements in Δ_B is active in representing $\mathbf{I}$ – sparsity. Many dictionaries have been proposed in the literature and they can be used in combination, including various wavelets and wavelet packets [10] [8] [11]. For over-complete Δ_B, the coefficients $C = (c_1, ..., c_K)$ are no longer the inner products. They are computed by algorithms such as matching pursuit [24] and basis pursuit [7].

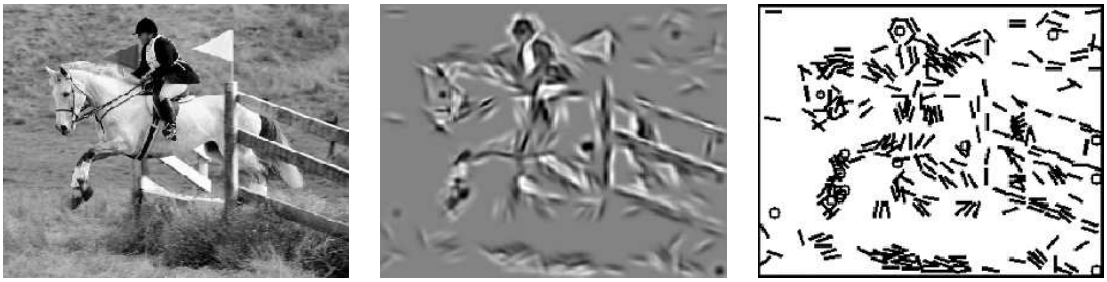
One influential piece of work [27] in sparse coding is to learn the dictionary of over-complete bases Δ_B from natural images. A crucial factor in obtaining a good dictionary is the prior model on the coefficients c_k, which has to follow a "super-Gaussian" distribution (long tails with peak at zero) [28], for instance,

$$p(C) = \prod_k p(c_k), \quad p(c_k) \propto e^{-\alpha|c_k|}. \tag{2}$$

(a) dictionary Δ_B (b) input image (c) reconstructed $K = 300$

Figure 4: A sparse coding example.

Figure 4 shows an example of sparse coding. Figure 4.(a) is the dictionary Δ_B with Gabor and Laplacian of Gaussian (LoG) bases, (b) is an observed image of 128×128 pixels, and (c) the image reconstructed by $K = 300$ bases selected from Δ_B. Figure 5 shows a second example with a symbolic representation in which a bar and a circle is used for a Gabor and LoG base respectively.



(a) input image (b) reconstructed $K = 300$ (c) symbolic sketch

Figure 5: A sparse coding example computed by matching pursuit. (c) is a symbolic representation where each base B_k is represented by a bar at the same location, with the same elongation and orientation. The isotropic LOG bases are represented by a circle.

The two examples show that the localized bases generally capture the image structures where

the intensity contrasts are high. However there are three obvious shortcomings.

Problem 1. The object boundaries are blurred. This indicates that we need to search for better dictionaries that is hyper-sparse. Currently each pixel in the image is explained by a small number of bases, but in a hyper-sparse representation, a pixel with high contrast should be represented by a single primitive.

Problem 2. The textures are not well represented. One may continue to add more bases in the matching pursuit process to code texture, but then the representation will not be very sparse (K is very large).

Problem 3. The bases (sketches) in Figure 5.(c) do not line up very well. In the wavelet literature, there are Markov tree models [29] for characterizing the relations between wavelet coefficients. But we need stronger model for regulating the spatial organization. In comparison, the symbolic sketch graph in Figure 2.(b) is much more meaningful than the sketches in Figure 5.(c).

The proposed primal sketch model shall resolve all of the above three problems.

2.2 Texture theories: MRF, FRAME, and statistical physics

In this paper, the term "texture" refers to image areas without distinguishable elements, i.e. what people called stochastic textures. Modern texture theories have become powerful enough to synthesize textures with structures for graphics purpose [12]. But the objective of the primal sketch is to represent the original image without noticeable perceptual distortion. Therefore, our texture corresponds to image areas where the wavelet coefficients are smaller than certain threshold, and their locations, orientations, and sizes are not perceivable. For example, there are very few image bases for the texture areas in Figure 5.(c). In such areas, human perception is said to capture some important statistics as a summary. It is long observed in psychophysics that preattentive vision cannot distinguish two texture regions if they share certain statistics measured by the neurons in early processing stage [3].

Let Δ_F be a set of filters, which can be Gabor or LoG (Laplacian of Gaussian) functions as in image coding (see Figure 4.(a)). For $F_i \in \Delta_F$, let $F_i * \mathbf{I}(x, y)$ be a filter response at $(x, y) \in \Lambda$. By pooling the filter responses over the lattice, one obtains a number of 1D histograms

$$h_i(\mathbf{I}) = h_i(z; \mathbf{I}) = \frac{1}{|\Lambda|} \sum_{(x,y)\in\Lambda} \delta(z - F_i * \mathbf{I}(x, y)), \quad i = 1, 2, ..., n, \tag{3}$$

where $\delta()$ is a Dirac delta function and z is the index of the histogram bins in discrete form. The

histograms may vary for small image areas even for the same texture. But in the limit when the image domain $\Lambda \to Z^2$, the statistical fluctuations diminish, and one arrives at the so-called Julesz ensemble studied by [32], which defines a texture pattern as an equivalence class,

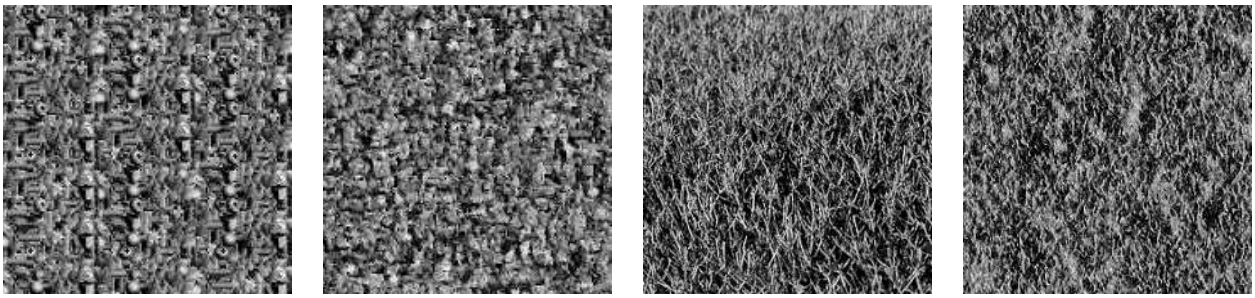
$$\text{A texture} = \Omega(h) = \{\mathbf{I} : h_i(\mathbf{I}) = h_i, i = 1, 2, ..., n, \Lambda \to Z^2\}, \tag{4}$$

where $h = (h_1, ..., h_n)$ are the 1D densities (statistics) characterizing the macroscopic properties of a texture pattern. As a principle used by Gibbs in 1902, all microscopic states $\mathbf{I} \in \Omega(h)$ are equally likely. This assumes a uniform density $q(\mathbf{I}; h)$ residing on the ensemble $\Omega(h)$ with $q(\mathbf{I}; h) = 0$ for $\mathbf{I} \notin \Omega(h)$.

If a large image $\mathbf{I}$ follows a Julesz ensemble $\Omega(h)$, then the image on any local patch $\Lambda_0 \subset \Lambda$ follows a Markov random field model conditioned on its local neighborhood $\partial\Lambda_0$,

$$\mathbf{I}_{\Lambda_0} \sim p(\mathbf{I}_{\Lambda_0}|\mathbf{I}_{\partial\Lambda_0}; \beta) = \frac{1}{Z} \exp\{-\sum_{i=1}^{n} \langle \beta_i, h_i(\mathbf{I}_{\Lambda_0}) \rangle\}, \tag{5}$$

where $\beta = (\beta_1, ..., \beta_n)$ are the potential functions, and Z is the normalizing constant. The Gibbs model in Equation (5) is called the FRAME model [34]. β are the Lagrange parameters learned from input images by maximum likelihood.



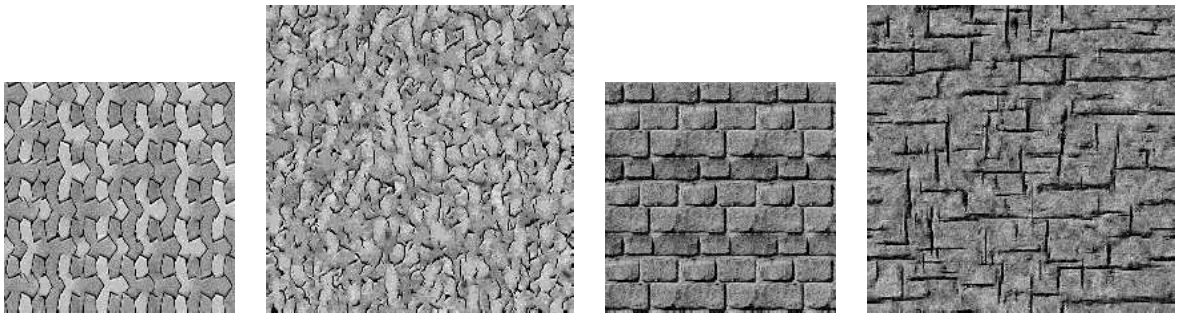
(a) (b) (c) (d)

Figure 6: Filter histograms: (a) and (c) are observed images. (b) and (d) are "reconstructed" by matching filter histograms.

Experiments showed that this model is quite effective in representing stochastic textures. See Figure (6). But it has two problems complementary to the sparse coding models, as indicated by Figure (7).

Problem 1. The FRAME model is ineffective in representing large image structures.

Problem 2. The model is built on pixels without introducing hidden (latent) variables (such as the bases and coefficients) which usually can drastically reduce the dimension of the image.



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

Figure 7: Filter histograms: (a) and (c) are observed images. (b) and (d) are "reconstructed" by matching filter histograms.

In this paper, we limit the filters to small window sizes which are sufficient for stochastic texture. Also these filters have small responses and thus it is computationally less expensive to simulate the textures.

3 The primal sketch representation

In this section, we introduce the primal sketch representation which integrates the two generative modeling schemes reviewed in the previous section. In the following, we present the primal sketch model and discuss the image primitives. Then in the next section, we shall present the sketch pursuit algorithm for computing the representation.

3.1 The primal sketch model

The image lattice Λ is divided into the sketchable and non-sketchable parts for the structural and textural parts respectively.

$$\Lambda = \Lambda_{\rm sk} \cup \Lambda_{\rm nsk}, \;\; \Lambda_{\rm sk} \cap \Lambda_{\rm nsk} = \emptyset. \tag{6}$$

The sketchable part is further divided into a number of disjoint patches with each patch being fitted by an image primitive.

$$\Lambda_{\rm sk} = \cup_{k=1}^{K} \Lambda_{{\rm sk},k}, \quad \Lambda_{{\rm sk},k_1} \cap \Lambda_{{\rm sk},k_2} = \emptyset, k_1 \neq k_2. \tag{7}$$

Some examples of the image primitives are shown in Figure 3. These primitives are aligned through their landmarks to form a sketch graph $S_{\rm sk}$. We index the selected image primitives by

$k = 1, ..., K$,

$$k = (\theta_{\text{topological}}, \theta_{\text{geometric}}, \theta_{\text{photometric}}), \tag{8}$$

where $\theta_{\text{topological}}$ is the type (degree of arms) of the primitive (blob, terminator, corner, junctions etc), $\theta_{\text{geometric}}$ collects the locations of the landmarks of the primitive, and $\theta_{\text{photometric}}$ collects the intensity profiles of the arms of the primitive. The sketch graph is a layer of hidden representation which has to be inferred from the image,

$$S_{\text{sk}} = (K, (\Lambda_{\text{sk},k}, B_k, a_k), k = 1, 2, ..., K),$$

where S_{sk} decides the sketchable part of the image, B_k is the image patch for primitive k, and a_k is the address variable pointing to the neighbors of the vertex $S_{\text{sk},k} = (\Lambda_{\text{sk},k}, B_k)$. We adopt the following generative image model on Λ_{sk}

$$\mathbf{I}_{\Lambda_{\text{sk},k}} = B_k + \mathbf{n}, \quad k = 1, 2, ..., K. \tag{9}$$

We shall discuss the representation of the primitives $B_k, k = 1, 2, ..., K$ in next subsection.

The non-sketchable part is divided into a small number $M = 3 \sim 7$ disjoint homogeneous texture regions by clustering the filter responses,

$$\Lambda_{\text{nsk}} = \cup_{m=1}^{M} \Lambda_{\text{nsk},m}, \quad \Lambda_{\text{nsk},m_1} \cap \Lambda_{\text{nsk},m_2} = \emptyset, m_1 \neq m_2. \tag{10}$$

$\mathbf{h}_{mi}, m = 1, 2, ..., M, i = 1, 2, ..., n$ are the texture statistics in each texture region,

$$h_i(\mathbf{I}_{\Lambda_{\text{nsk},m}}) = \mathbf{h}_{mi}, \quad m = 1, 2, ..., M. \tag{11}$$

This yields the FRAME model in equation (5) for small window size. The model learns the Lagrange parameters β_{mi} for the statistics h_{mi}. We denote the texture region labeling by

$$S_{\text{nsk}} = (M, (\Lambda_{\text{nsk},m}, h_{mi} \leftrightarrow \beta_{mi}), m = 1, 2, ...M, i = 1, 2, ..., n). \tag{12}$$

In summary, we have the following probability model for the primal sketch representation,

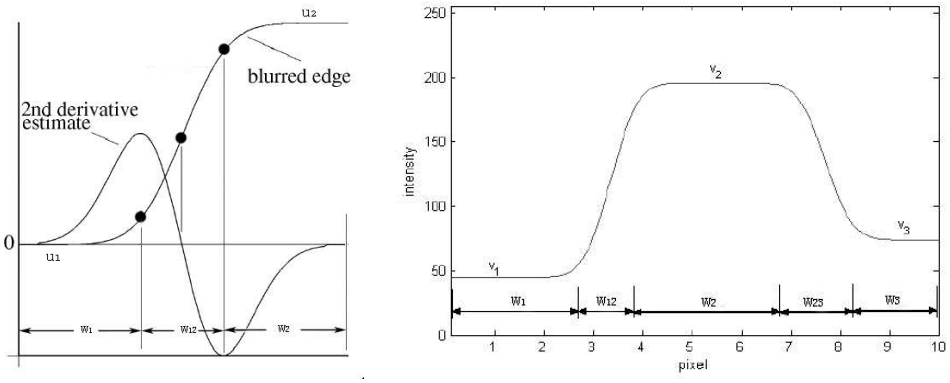
$$\begin{aligned} & p(\mathbf{I}_\Lambda, S_{\text{sk}}, S_{\text{nsk}}) \\ = \; & \frac{1}{Z} \exp\{-\frac{1}{2\sigma_o^2} \sum_{k=1}^{K} \sum_{(u,v) \in \Lambda_{\text{sk},k}} (\mathbf{I}(u,v) - B_k(u,v))^2 - \sum_{m=1}^{M} \sum_{i=1}^{n} \langle \beta_{mi}, h_i(\mathbf{I}_{\Lambda_{\text{nsk},m}}) \rangle \\ & - E(S_{\text{sk}}) - E(S_{\text{nsk}})\}, \end{aligned} \tag{13}$$

where $E(S_{\text{sk}})$ and $E(S_{\text{nsk}})$ are prior energy functions defined on the sketch graph S_{sk} and texture regions S_{nsk}.

In the following two sections, we shall introduce the primitive dictionary Δ_{sk} and the prior energy for the sketches S_{sk} – the mixed random field for the Gestalt organization of the sketch graph and a simple energy on S_{nsk} for texture clustering.

3.2 The dictionary of image primitives

As shown in Figure 3, the dictionary of image primitives designed for the sketch graph S_{sk} consists of eight types of primitives in increasing degree of connection – blob, terminators, edge/ridge/multi-ridge, corner, junction and cross. These primitives have a center landmark and $l = 0 \sim 4$ axes (arms) for connecting with other primitives. These primitives are represented by geometric parameters $\theta_{\text{geometric}}$ and photometric parameters $\theta_{\text{photometric}}$. There is no arm for blobs. Terminators have only one arm, while edge/ridge/multi-ridge and corners have two. Junctions and crosses have three and four arms respectively. For arms, the photometric property is represented by the intensity profiles.



(a) Edge profile (b) Ridge profile

Figure 8: (a) The edge profile is represented by 5 parameters. The illustration of the computing of the scale for a blurred edge. The blurring scale is measured by the distance between the extremes of the second derivative. (b) The representation of a ridge profile with 8 parameters.

We model the intensity of the arm profile in a parametric way, following [13]. For the edge profiles, as shown in Figure 8.(a), we use five parameters (u_1, u_2, w_1, w_{12}, w_2), which denotes the left intensity, the right intensity, the width of the left intensity (from the leftmost to the

left extreme of the second derivative), the blurring scale, and the width of the right intensity, respectively. The total width of the edge is $W = w_1 + w_{12} + w_2$. The intensity in between is modeled by the Gaussian cumulative density function model, and a look-up table is built for computational efficiency.

For the ridge (bar) profile, we use eight parameters: (u_1, u_2, u_3, w_1, w_{12}, w_2, w_{23}, w_3) as shown in Figure 8.(b). A segment with constant intensity value (e.g. u_1, u_2, u_3) is called one flat segment. Such photometric representation can be extended to multi-ridges which have more than three flat segments.

Around the center of a primitive, the arms may overlap with each other. The intensities on the overlapping area interpolate the intensities of the arms. Specifically, for a pixel p, suppose it is covered by L arms, and the intensity functions for the arms are A_1, A_2, ..., A_L. If we denote the intensities of the profiles at this pixel p as A_1^p, A_2^p, ..., A_L^p, and the distances from the point p to the center lines of these arms as d_1, d_2, ..., d_L, the interpolated intensity is a weighted sum:

$$B_k(p) = \frac{1}{D} \sum_{l=1}^{L} \frac{A_l^p}{d_l + 1}, \tag{14}$$

where $D = \sum_{l=1}^{L} \frac{1}{d_l+1}$.

Figure 9 shows an example of a T-junction where three strokes meet. For a pixel p, each stroke contributes the intensity of A_1^p, A_2^p and A_3^p respectively. Then the interpolated intensity for pixel p will be the weighted sum of the three as in Equation (14) (in this example $d_1 = 0$).

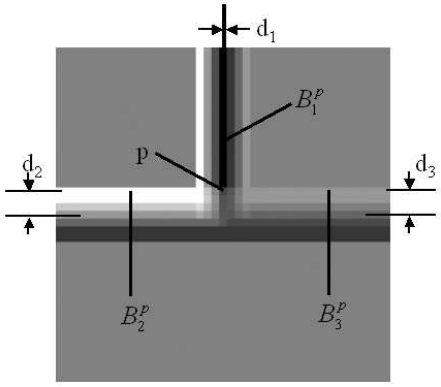


Figure 9: The illustration of the mixed intensity model at vertices.

3.3 The mixed random field for the sketch graph

Now we specify the Gestalt field for the sketch graph $S_{\text{sk}} = (K, (\Lambda_{\text{sk},k}, B_k, a_k), k = 1, ..., K)$, which follows a so-called mixed random field [15]. The address variables a_k's represent dynamic linking of the primitives $V = (S_{\text{sk},k}, k = 1, ..., K)$. It differs from conventional Markov random field in two aspects.

1. The vertices are inhomogeneous with different degrees of connections and are inferred from the images.

2. The neighborhood of each vertex is no longer fixed but inferred as address variables, which yields the graph structure.

The sketch graph follows some Gestalt organizations which are enforced by our explicit description of vertex types. We use a simple energy function which penalizes different vertices with different weights.

We divide the set of vertices V into 5 subsets according to their degrees of connection,

$$V = V_0 \cup V_1 \cup V_2 \cup V_3 \cup V_4, \tag{15}$$

where V_i is the set of vertices with degree i. Then we have

$$E(S_{\text{sk}}) = \sum_{d=0}^{4} \lambda_d |V_d|, \tag{16}$$

where $|V_d|$ is the cardinality of the set V_d, and λ_d can be interpreted as the coding length associated with each types of vertices. In our experiments we choose $\lambda_0 = 1.0$, $\lambda_1 = 5.0$, $\lambda_2 = 2.0$, $\lambda_3 = 3.0$,$\lambda_4 = 4.0$. The reason that we choose $\lambda_1 = 5.0$ for terminators is that from Gestalt laws, the closure and continuity are preferred in the perceptual organization.

The energy for the texture regions are also very simple. We choose the Potts model for $E(S_{\text{nsk}})$ in equation (13),

$$E(S_{\text{nsk}}) = - \sum_{p_1 \sim p_2, p_1 \in \Lambda_{\text{nsk},i}, p_2 \in \Lambda_{\text{nsk},j}} \lambda_{\text{nsk}} \delta(i, j), \tag{17}$$

where $p_1 \sim p_2$ means that p_1 and p_2 are two pixels that are neighbors of each other. $\lambda_{\text{nsk}} (\geq 0)$ is the parameter for the Potts model which favorites identical labelling for neighboring pixels. $\delta(i, j) = 0$, if $i = j$. Otherwise $\delta(i, j) = 1$.

4 The Sketch Pursuit Algorithm

The primal sketch algorithm includes three parts. (i) Deterministic pursuit of the sketch graph $S_{\rm sk}$ in a procedure similar to matching pursuit. It sequentially add new strokes (primitives of edges/ridges) that are most prominent. (ii) Refine the sketch graph $S_{\rm sk}$ to achieve better Gestalt organization by reversible graph operators, in a process of maximizing a posterior probability (MAP). (iii) A simple texture clustering algorithm computing the texture regions $S_{\rm nsk}$.

4.1 The sketch pursuit phase I

To speed up the computation, in our sketch pursuit process, we first adopt a procedure similar to matching pursuit [24], which adds new strokes sequentially from a proposal map. In this phase, we use a simplified primal sketch model,

$$p_I(\mathbf{I}_\Lambda, S_{\rm sk}, S_{\rm nsk}; \Delta_{\rm sk}) \approx \frac{1}{Z}\exp\{-\frac{1}{2\sigma_o^2} \quad (\sum_{k=1}^{K}\sum_{(u,v)\in\Lambda_{{\rm sk},k}} (\mathbf{I}(u,v) - B_k(u,v))^2 + \sum_{(u,v)\in\Lambda_{\rm nsk}} (\mathbf{I}(u,v) - \mu(u,v))^2)\}, \tag{18}$$

in which only image coding for the sketchable part $\Lambda_{\rm sk}$ is kept, the prior models on $S_{\rm sk}$ and $S_{\rm nsk}$ are ignored, and a simple Gaussian model is applied to the pixels in the texture regions $\Lambda_{\rm nsk}$. $\mu(u,v)$ is the local intensity mean around pixel (u,v). From the simplified model (18), when we add a stroke $S_{{\rm sk},K+1}$ into $S_{\rm sk}$, then $S'_{\rm sk} = S_{\rm sk} \cup S_{{\rm sk},K+1}$, $\Lambda'_{\rm nsk} = \Lambda_{\rm nsk} - \Lambda_{{\rm sk},K+1}$, and the probability changes to

$$p_I(\mathbf{I}_\Lambda, S'_{\rm sk}, S'_{\rm nsk}; \Delta_{\rm sk}) \approx \frac{1}{Z}\exp\{-\frac{1}{2\sigma_o^2} \quad (\sum_{k=1}^{K+1}\sum_{(u,v)\in\Lambda_{{\rm sk},k}} (\mathbf{I}(u,v) - B_k(u,v))^2 + \sum_{(u,v)\in\Lambda'_{\rm nsk}} (\mathbf{I}(u,v) - \mu(u,v))^2)\}, \tag{19}$$

Comparing (19) and (18), define

$$\begin{aligned} \Delta\mathcal{L} &= \log\frac{p(\mathbf{I}_\Lambda, S'_{\rm sk}, S'_{\rm nsk}; \Delta_{\rm sk})}{p(\mathbf{I}_\Lambda, S_{\rm sk}, S_{\rm nsk}; \Delta_{\rm sk})} \\ &= \frac{1}{2\sigma_o^2}\{\sum_{(u,v)\in\Lambda_{{\rm sk},K+1}} (\mathbf{I}(u,v) - \mu(u,v))^2 - (\mathbf{I}(u,v) - B_{(K+1)}(u,v))^2\}, \end{aligned} \tag{20}$$

which is called *image coding length gain* by adding a new stroke.

At the beginning of the computation, we only have the input image $\mathbf{I}^{\rm obs}$, and the sketch graph is empty $S_{\rm sk} = \emptyset$. We propose a method called *blob-edge-ridge detection* which provides a

proposal map S_{sk}^{∞} for the strokes. From the proposal map, strokes are pursued sequentially by maximizing the probability (18), or equivalently obtaining highest image coding length gain of (20) by adding strokes one by one.

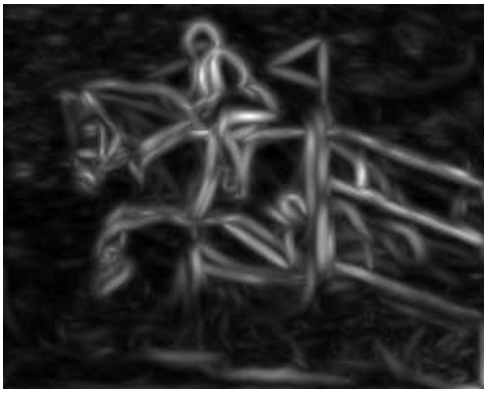
Blob-Edge-Ridge (BER) Detector Since the desired primal sketches are mostly blobs, edges, and ridges, we adopt a step called blob-edge-ridge (BER) detection to provide the proposal (denoted by S_{sk}^{∞}) for the sketch pursuit process. For an input image $\mathbf{I}^{\text{obs}}$, the blob-edge-ridge (BER) detection algorithm works as follows.

First the input image is convolved with a set of filters which include Laplacian of Gaussian (LoG), elongated filters of first derivative of Gaussian (DG) and second derivative of Gaussian (D2G). The filters are similar to the set of bases for the sparse coding as shown in Figure 4.(a). Filters of DG and D2G are chosen at several scales (e.g. three to five) and a number of orientations (e.g. 18) to detect edges and ridges. Several scales of LoG are used to detect blobs. For each pixel, we compute the combined response which is the sum of the squares of DG and D2G responses. The maximum of the combined responses is considered as the edge/ridge strength at that pixel (see Figure 10.(b)). The local orientation (Figure 10.(c)) is decided by the orientation of the maximum response filter. Second, the non-maxima suppression method in Canny edge detector [5] is used to compute the maxima as the proposed sketch S_{sk}^{∞} (see Figure 10.(d)). The blob strength is measured by the maximum responses of the LoG filters and is shown in Figure 10.(e). We apply a local maxima searching to get the proposed blobs shown in Figure 10.(f).

Sequential Sketch Pursuit From the proposal map S_{sk}^{∞}, we can find the highest strength position (x_0, y_0). From (x_0, y_0) the connected points in S_{sk}^{∞} are collected in two directions until a straight line segment (linelet) is formed within a pre-specified average fitting error per pixel (e.g. 1.0 /pixel). The profile model A_0 is computed from the averaged profile perpendicular to the linelet. Then an edge/ridge primitive is generated as $S_{\text{sk},0}$. The image coding length gain $\Delta\mathcal{L}$ (equation (20)) by introducing a new stroke $S_{\text{sk},0}$ can be evaluated. If $\Delta\mathcal{L} < \epsilon$, where ϵ is a pre-specified threshold, then the proposed $S_{\text{sk},0}$ is not accepted. Otherwise we accept the proposed $S_{\text{sk},0}$ and set $S_{\text{sk}} \leftarrow S_{\text{sk}} \cup S_{\text{sk},0}$. Also $S_{\text{sk},0}$ is removed from the proposal S_{sk}^{∞} by setting $S_{\text{sk}}^{\infty} \leftarrow S_{\text{sk}}^{\infty} - S_{\text{sk},0}$, which means $S_{\text{sk},0}$ is not in the proposal set anymore.

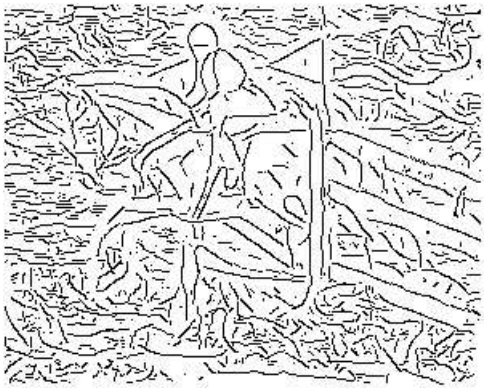
The above operation is called *creation* and defined as graph operator O_1 as shown in Table 2. The reverse operation O_1' proposes to remove one stroke. More graph operators will be introduced for the sketch pursuit phase II.

(a) original image

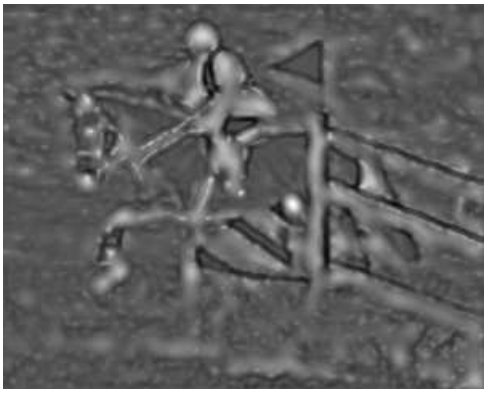


(b) edge/ridge strength

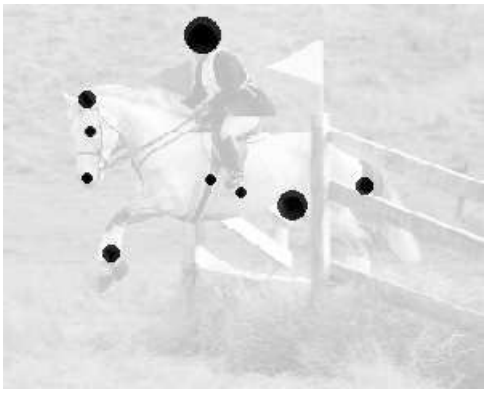
(c) local orientation



(d) proposed sketches



(e) blob strength

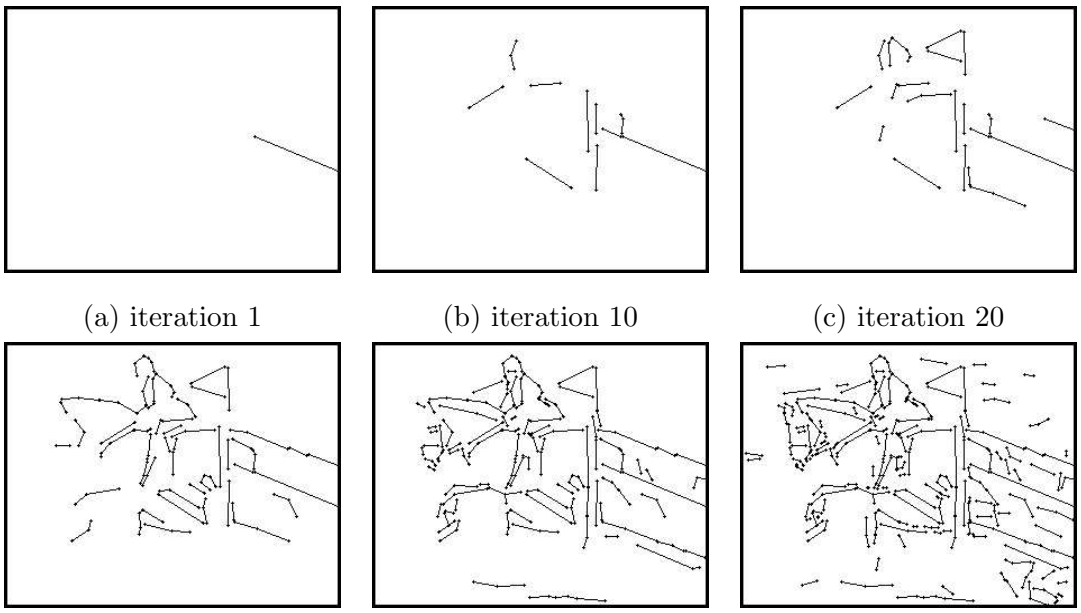


(f) proposed blobs

Figure 10: The blob-edge-ridge detector. For the original image (a) a set of filters are applied to get the edge/ridge strength (b) and the local orientation (c). The proposal sketch (d) is computed by non-maxima suppression. (e) shows the blob strength and (f) shows the proposed blobs after non-maxima suppression and thresholding.

From each end of the accepted stroke $S_{\text{sk},K}$, we search the connected points in S_{sk}^{∞} until a linelet is formed within a pre-specified average fitting error per pixel. We denote it as a new proposed stroke $S_{\text{sk},K+1}$. Similar to the proposal of $S_{\text{sk},0}$, the image coding length gain $\Delta\mathcal{L}$ can be computed. We decide whether to accept the proposed $S_{\text{sk},K+1}$ or not by testing whether $\Delta\mathcal{L} > \epsilon$. This operation is called *growing* and defined as graph operator O_2. This operator can be applied iteratively until no proposal is accepted. Then a curve (with one or several strokes, two terminators and possible corners) is obtained. The reverse operation which removes a stroke from the end of a curve is called *shrinking* and will be denoted as O_2'.

The sketch pursuit phase I applies operators O_1 and O_2 iteratively until no more strokes are accepted. Figure 11 shows the results of the sketch pursuit phase I on the horse-riding image. The sketch graphs are obtained after 1, 10, 20, 50, 100, and 180 iterations. The image coding length gain $\Delta\mathcal{L}$ for each iteration is plotted in Figure 12. Phase I provides an initialization state for sketch pursuit phase II.



(a) iteration 1 (b) iteration 10 (c) iteration 20

(d) iteration 50 (e) iteration 100 (f) iteration 180

Figure 11: The process of applying operators O_1 and O_2 iteratively for the horse-riding image at iterations of 1, 10, 20, 50, 100, and 180.

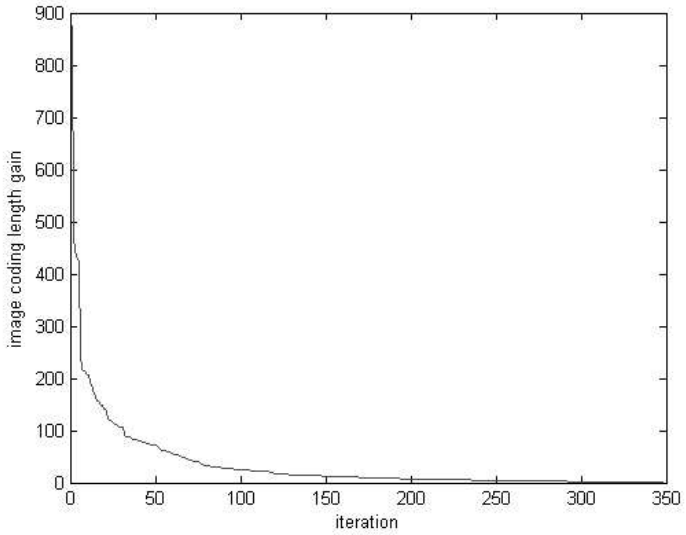


Figure 12: The image coding length gain $\Delta\mathcal{L}$ by applying operators O_1 and O_2 verses the iteration.

4.2 The sketch pursuit phase II

From the initialization result of the sketch pursuit phase I, by applying a set of graph operators, the sketch pursuit phase II maximizes a simplified version of the joint probability (13).

$$p_{II}(\mathbf{I}_\Lambda, S_{\rm sk}, S_{\rm nsk}; \Delta_{\rm sk}) \approx \frac{1}{Z}\exp\{-\frac{1}{2\sigma_o^2} \quad (\sum_{k=1}^{K}\sum_{(u,v)\in\Lambda_{{\rm sk},k}} (\mathbf{I}(u,v) - B_k(u,v))^2 + \sum_{(u,v)\in\Lambda_{\rm nsk}} (\mathbf{I}(u,v) - \mu(u,v))^2) - E(S_{\rm sk})\}, \quad (21)$$

in which the image coding for the sketchable part $\Lambda_{\rm sk}$ and the prior model on $S_{\rm sk}$ are kept. The prior model on $S_{\rm nsk}$ is ignored, and a simple Gaussian model is applied to the pixels in the texture regions $\Lambda_{\rm nsk}$ as in phase I.

Reversible Graph Operators Two pairs of reversible graph operators O_1 & O_1' and O_2 & O_2' have been introduced in sketch pursuit phase I. An additional eight pairs of graph operators are proposed and utilized in the sketch pursuit phase II. These operators are summarized in Table 2 and explained below. These graph operators facilitate the sketch pursuit process to transverse the sketch graph space.

The third graph operator O_3 is called *connection*, which proposes to connect two vertices by a new stroke. A set of nearest neighbors for each vertex $S_{{\rm sk},k}$ are found. The operator proposes to connect each of them to $S_{{\rm sk},k}$ by introducing a new stroke. The operator O_3 are repeatedly

checked for all possible connections. The reverse operation O_3' is to disconnect two vertices by removing the connecting stroke.

The fourth and fifth graph operators O_4 and O_5 are called *extension and cross*, which propose to extend one (O_4) or two (O_5) existing strokes and check if they are crossed with other strokes and form junctions (O_4) or corners (O_5). In this situation, one new vertex and one or two more new strokes may be proposed. The reverse operators O_4' and O_5' are to disconnect vertices by removing one or two strokes.

Graph operators O_6 and O_7 are called *combination*, which combines two connected strokes (O_6) or parallel strokes (O_7) into one. The reverse operators O_6' and O_7' are to break or split one stroke into two.

Graph operator O_8 works on graph vertices and called *merging*, which merges two nearby vertices into one. This proposal will remove one stroke and one vertex. The neighborhood of related strokes will also be changed as shown in Table 2. The reverse operator O_8' is to split one vertex into two.

The last two graph operators O_9 and O_{10} are for blobs, which propose to create a blob or change one stroke or several strokes into one blob. The reverse operations O_9' and O_{10}' are to remove a blob and change a blob into a stroke.

An example of applying graph operators is show in Figure 14. The ten pairs of reversible graph operators change the topological property of the sketch graph.

We modify the geometric and photometric properties of the sketch graph by a graph operation called *diffusion*, which proposes to change the geometric and photometric properties such as the vertex position, and the stroke profiles. For example, to defuse the positions of the vertices, we perturb the position (x, y) in an area of $[x \pm dx, y \pm dy]$ and select the position with the highest coding length gain.

Figure 13 shows an example of the sketch pursuit process. Figure 13.(a) is an input image. Figure 13.(b) and (c) is the sketch graph after the sketch pursuit phase I and phase II respectively. Figure 13.(d) is the reconstructed image from the primal sketch model. From the sketch graphs in Figure 13.(b) and (c), it can be seen that some of the corners/junctions are missed in phase I due to the weak edge/ridge strength, while they are recovered in phase II.

Sketch Pursuit by Reversible Graph Operators In the sketch pursuit phase II, the sketch graph S_{sk} is refined to achieve better Gestalt organization by the ten pairs of the reversible graph operators discussed above, in a process of maximizing a posterior (MAP).

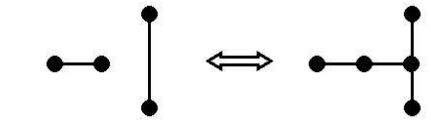
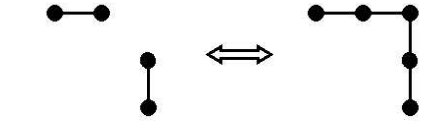
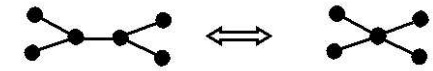
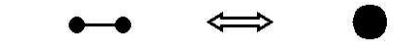
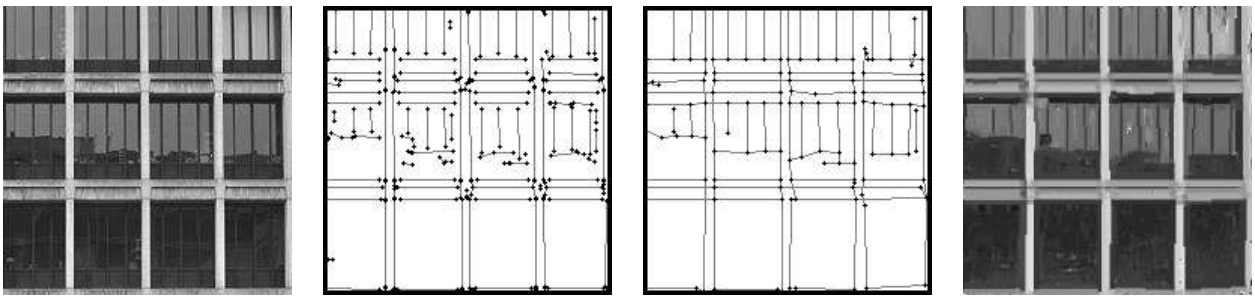
operators	graph change	illustration
O_1, O_1'	create / remove a stroke	
O_2, O_2'	grow / shrink a stroke	
O_3, O_3'	connect / disconnect vertices	
O_4, O_4'	extend one stroke and cross / disconnect and combine	
O_5, O_5'	extend two strokes and cross / disconnect and combine	
O_6, O_6'	combine two connected strokes / break a stroke	
O_7, O_7'	combine two parallel strokes / split one into two parallel	
O_8, O_8'	merge two vertices / split a vertex	
O_9, O_9'	create / remove a blob	
O_{10}, O_{10}'	switch between a stroke(s) and a blob	

Table 2: The 10 pairs of reversible graph operators used in the sketch pursuit process.



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

Figure 13: One result of the primal sketch model. (a) input image; (b) raw sketch graph after sketch pursuit phase I; (c) final sketch graph after sketch pursuit phase II; (d) reconstructed image from our primal sketch model.

$$
\begin{aligned}
(S_{\rm sk}, S_{\rm nsk})^* &= \arg\max p_{II}(S_{\rm sk}, S_{\rm nsk}|\mathbf{I}_\Lambda; \Delta_{\rm sk}) && (22)\\
&= \arg\max p_{II}(\mathbf{I}_\Lambda, S_{\rm sk}, S_{\rm nsk}; \Delta_{\rm sk}) && (23)\\
&= \arg\min \frac{1}{2\sigma_o^2}(\sum_{k=1}^{K}\sum_{(u,v)\in\Lambda_{{\rm sk},k}}(\mathbf{I}(u,v)-B_k(u,v))^2\\
&+ \sum_{(u,v)\in\Lambda_{\rm nsk}}(\mathbf{I}(u,v)-\mu(u,v))^2) + E(S_{\rm sk})\\
&= \arg\min \mathcal{L}_A + \mathcal{L}_S\\
&= \arg\min \mathcal{L}(S_{\rm sk}, S_{\rm nsk}) && (24)
\end{aligned}
$$

where $\mathcal{L}_A = \frac{1}{2\sigma_o^2}(\sum_{k=1}^{K}\sum_{(u,v)\in\Lambda_{{\rm sk},k}}(\mathbf{I}(u,v)-B_k(u,v))^2 + \sum_{(u,v)\in\Lambda_{\rm nsk}}(\mathbf{I}(u,v)-\mu(u,v))^2)$ is called image coding length, $\mathcal{L}_S = E(S_{\rm sk})$ is called sketch coding length, $\mathcal{L}(S_{\rm sk}, S_{\rm nsk}) = \mathcal{L}_A + \mathcal{L}_S$ is called total coding length.

As illustrated in Figure 14, from a current sketch graph $S_{\rm sk}$, a local subgraph is examined. All of the ten pairs of graph operators are checked for that local graph. We exam all the new subgraphs after three to five operations. All possible graph change candidates $(S'_{\rm sk}, S'_{\rm nsk})$ in these three to five operations (usually around 5 to 20) are evaluated in terms of total coding length gain $\Delta\mathcal{L} = \mathcal{L}(S_{\rm sk}, S_{\rm nsk}) - \mathcal{L}(S'_{\rm sk}, S'_{\rm nsk})$. Currently we adopt a greedy method, by which if the best $\Delta\mathcal{L} > \delta$, where δ is pre-specified threshold, the local graph will be modified. This process is repeated until no modification can be accepted.

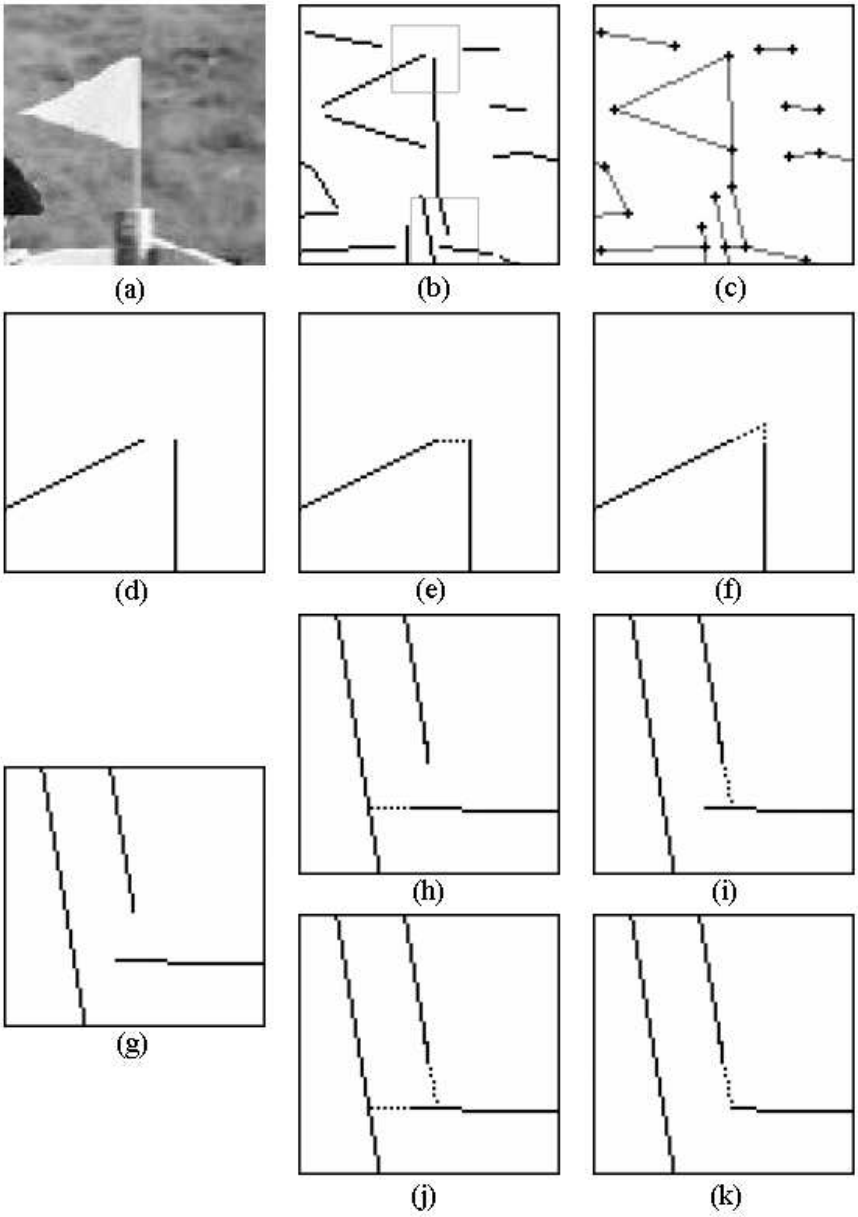


Figure 14: An example of applying graph operators. (a) a local image patch from the horse-riding image; (b) the sketch graph after sketch pursuit phase I; (c) the sketch graph after sketch pursuit phase II; (d) the zoom-in view of the upper rectangle in (b); (e) applying graph operator O_3 – connecting two vertices to (d); (f) applying graph operator O_5 – extending two strokes and cross; (g) the zoom-in view of the lower rectangle in (b); (h) applying graph operator O_4 – extending one stroke and cross; (i) applying graph operator O_4 to a second stroke; (j) combining (h) and (i); (k) applying graph operator O_4 – extending one stroke and O_1' – removing one stroke.

4.3 The texture clustering process

In the sketch pursuit process, we use a simplified model for the texture regions, which is a Gaussian model with a local mean of $\mu(u,v)$ and variance σ_0^2. After the sketch pursuit process is finished, we have the lattice Λ_{nsk} for the textures. We use a texture clustering method to divide them into regions. The feature $h(u,v)$ which is chosen for clustering is the histogram of a set of pre-selected filters within a local window (e.g. 7x7). For example, if we use seven filters and if 7 bins are used for each of the filter response histogram, then totally we have a 49-Dimensional feature $h(u,v)$ for each pixel (u,v) in Λ_{nsk}. The clustering process is also maximizing a posterior,

$$\begin{aligned} S^*_{\text{nsk}} &= \arg\max p(S_{\text{nsk}}|I_{\Lambda_{\text{nsk}}}) \\ &= \arg\max p(I_{\Lambda_{\text{nsk}}}|S_{\text{nsk}})p(S_{\text{nsk}}) \\ &= \arg\min \sum_{m=1}^{M} E(I_{\Lambda_{\text{nsk},m}}|S_{\text{nsk},m}) + E(S_{\text{nsk}}), \qquad (25) \\ E(I_{\Lambda_{\text{nsk},m}}|S_{\text{nsk},m}) &= -\frac{1}{2}\log|\Sigma_m| - \frac{1}{2}\sum_{(u,v)\in\Lambda_{\text{nsk},m}} (h(u,v)-h_m)^T\Sigma_m^{-1}(h(u,v)-h_m), \qquad (26) \end{aligned}$$

where $E(S_{\text{nsk}})$ follows the Potts model (17). The texture regions are modeled by multivariate Gaussian with the mean h_m and the covariance Σ_m for region $\Lambda_{\text{nsk},m}$. After the texture regions are obtained, we have a FRAME model for each region and draw samples from the FRAME models to synthesize textures for the final sketch pursuit results.

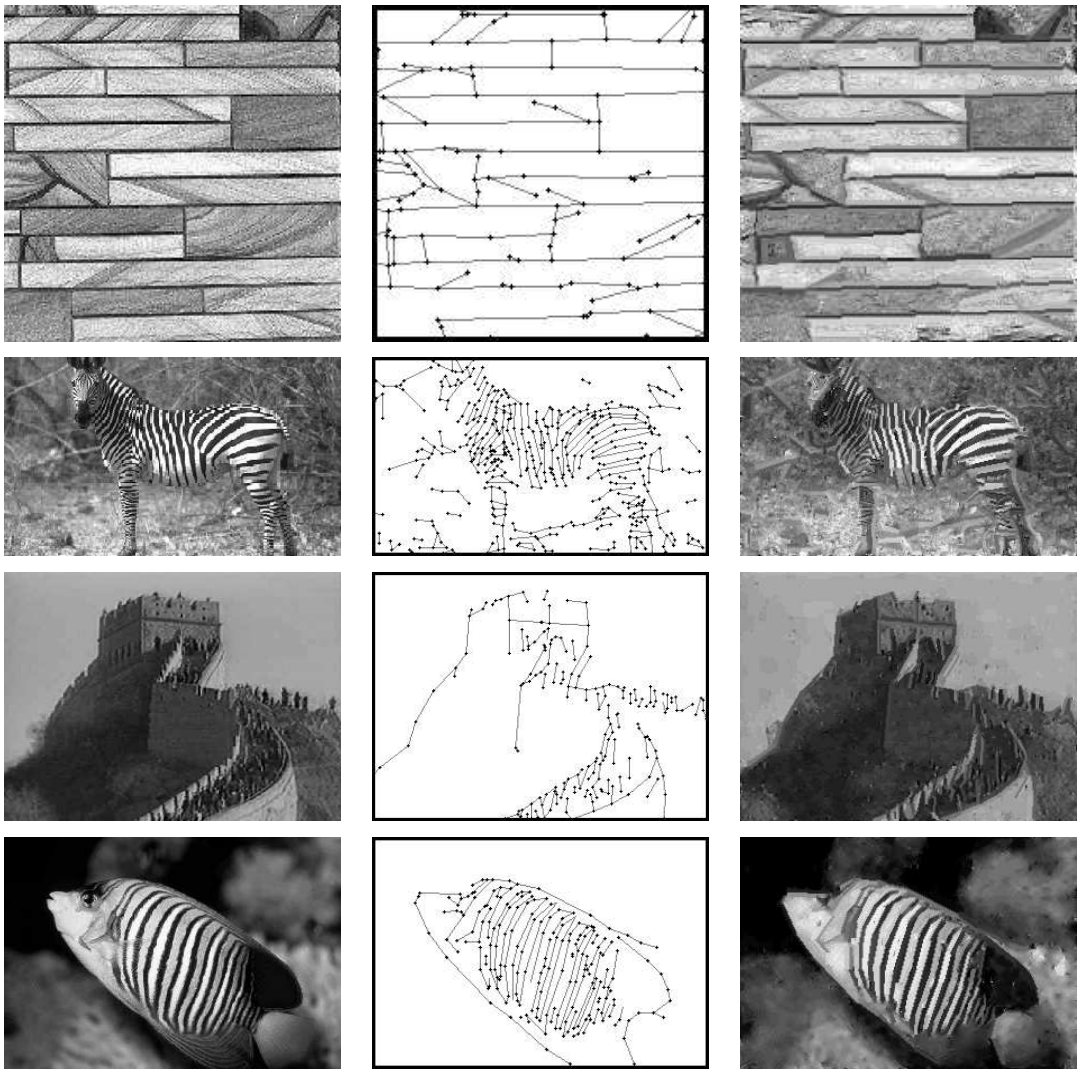
5 Experiments

5.1 Sketch pursuit results

We run our algorithm on hundreds of images. Figures 15, 16, and 17 show some results of the sketch pursuit process. From these results, it can be seen that the sketch graphs capture the main structures in the images while the leftover textures are modeled perceptually well by MRF models.

5.2 Comparison with edge detection

Canny edge detector can be considered as a special case of the proposal for the primal sketch pursuit, where the filters used in the canny edge detector can be incorporated as a subset of the filters used in the proposal for the primal sketch pursuit. From the proposal, the sketch pursuit

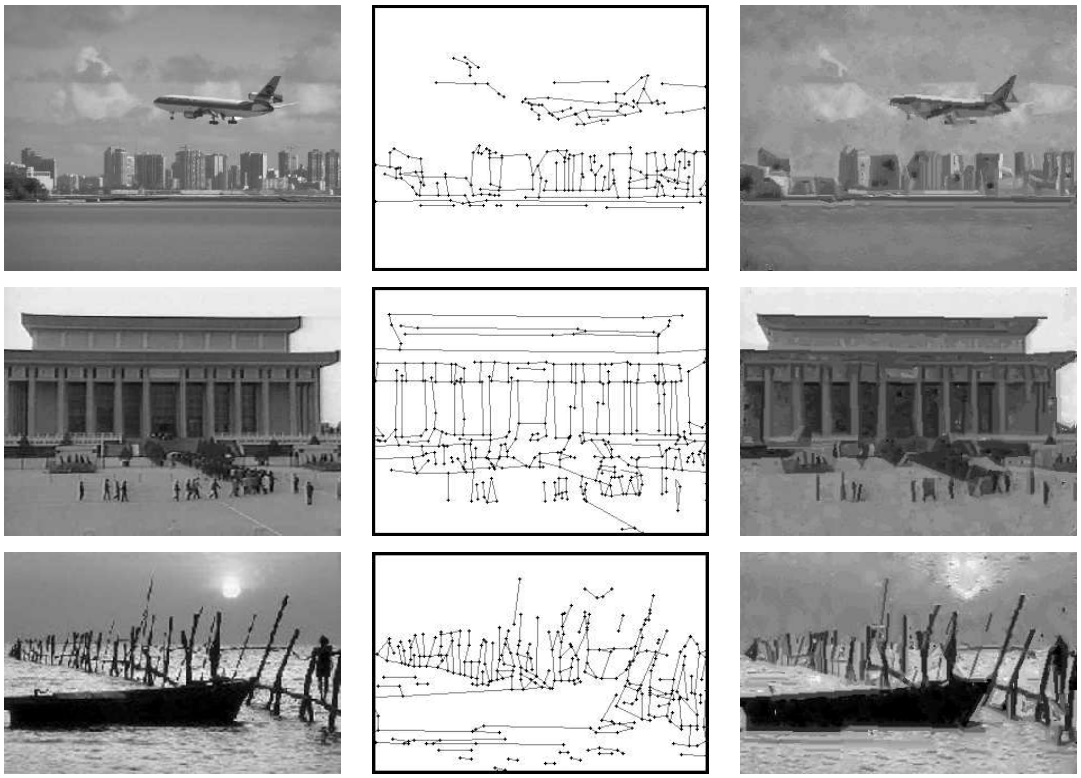


(a) (b) (c)

Figure 15: More results of the primal sketch model. (a)) input image; (b) sketch graph; (c) reconstructed image from our primal sketch model.

process uses a generative model and prior model to construct the sketch graph, which consists of visual primitives such as corners and junctions. The sketch graph representation is not pixel level representation any more. It is far sparser than the edge map, which has no concepts such as corners, junctions, and line segments.

There are three parameters for the Canny edge detection [5]: the Gaussian blur scale σ, the low threshold θ_l and high threshold θ_h. Some researchers set $\theta_l = 0.4 * \theta_h$ and reduce them to two parameters. Figure 18 shows the Canny edge maps for the horse riding image with three

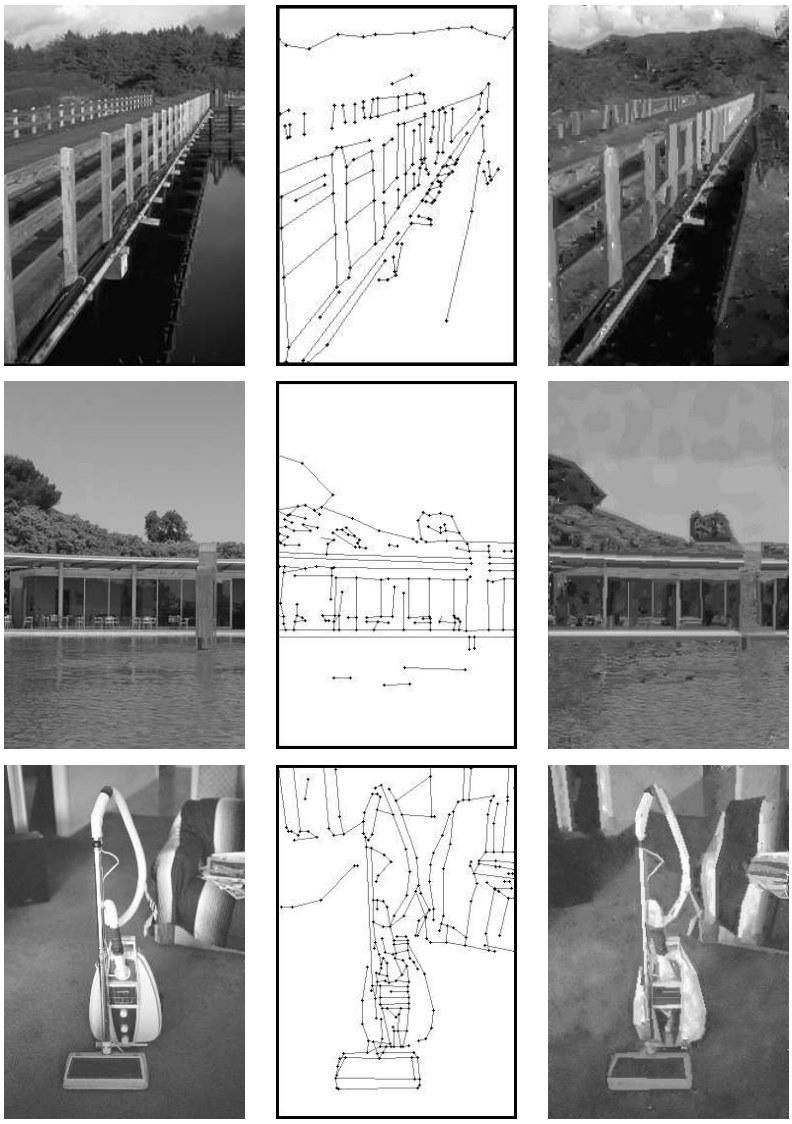


(a) (b) (c)

Figure 16: More results of the primal sketch model. (a)) input image; (b) sketch graph; (c) reconstructed image from our primal sketch model.

Gaussian blur scales $\sigma = 0.6, 1.2, 1.8$. For each scale, we choose the best thresholds such that the edge map looks the best (not too many noisy edges on textures, but the main structures are kept.) We also set $\theta_l = 0.4 * \theta_h$.

Some results are shown in Figure 18. We can see: (1) For the eye of the horse (marked 1), it is represented as a blob in the primal sketch, while a set of edges are detected by Canny edge detector; (2) For the ridge (marked 2), it is represented by one sketch with a ridge profile in the primal sketch, while double Canny edges are detected; (3) For the corners and junctions (marked 3, 4 and 5), the primal sketch explicitly captures them more accurately than the Canny edges since a generative model and a spatial MRF model are adopted.



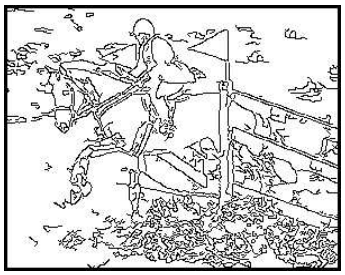
(a) (b) (c)

Figure 17: More results of the primal sketch model. (a)) input image; (b) sketch graph; (c) reconstructed image from our primal sketch model.

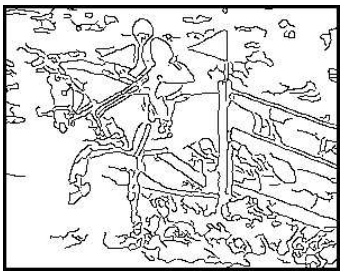
5.3 Applications

As a low-level image representation, the primal sketch model provides a common platform for image coding, processing, and high level tasks.

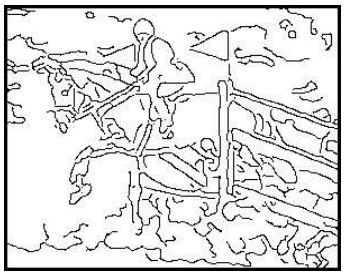
Our primal sketch model directly leads to a lossy image coding scheme. Since we use the FRAME model for the non-skechable texture regions, those regions are coded semantically (perceptually) by the histograms of the filter responses. The coding error per pixel might be



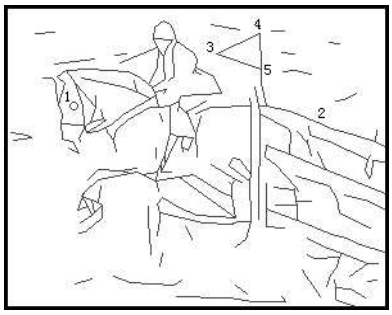
(a) $\sigma = 0.6, \theta_h = 0.80$



(b) $\sigma = 1.2, \theta_h = 0.75$



(c) $\sigma = 1.8, \theta_h = 0.70$



(d) primal sketch graph

Figure 18: Comparison between Canny edges and the primal sketch. Canny edges for the horse riding image with different parameters. For the eye of the horse (marked 1), it is represented as a blob in the primal sketch, while a set of edges from the Canny edge detection are produced. For the ridge (marked 2), it is represented by one sketch with a ridge profile in the primal sketch, while double Canny edges are detected. For the corners and junctions (marked 3, 4 and 5), the primal sketch explicitly captures them more accurately than the Canny edges.

(a) original image (b) sketch image

(c) interpolation (d) residue

(e) with 800 bases (f) residue

(g) with 1600 bases (h) residue

Figure 19: One example of the lossless image coding from the primal sketch model.

very high. However,the perception of textures is kept and very high compression ratio (more than 1 : 25 are achieved.) We explain this coding scheme by an example in Figure 2. Table 1 shows the approximate coding length for each step.

For the sketch graph, it is more efficient to code the junctions first, then code the corner vertices since the junctions are less randomly/independently distributed in the image, while the

corner vertices can be coded relatively to the junctions.

From the primal sketch model, we propose a lossless coding scheme as follows. For both the sketchable and non-sketchable parts, sparse coding are used. We use the constructed visual primitive dictionary for the sketchable portion and use Gabor wavelets for the non-sketchable portion. As shown in Figure 19, from the reconstructed sketchable part of the image, we run the linear interpolation to fill-in the non-sketchable portion with the sketchable part as boundary conditions. The interpolation result is shown in Figure 19.(c). We run the matching pursuit process on the residual image after interpolation (shown in Figure 19.(d)). As more and more bases are used, the residual image approaches Gasssian noise as shown in Figure 19.(e) to (j).

6 Discussion

In this paper, we present a primal sketch model that integrates three components: a texture model (Julesz ensemble), a generative model with image primitives (textons), and a mixed random field for the sketch graph.

Our model can be traced back to the line process or weak membrane models used in (Mumford and Shah, 89, Blake and Zisserman, 87). They employed a set of edges B that breaks the smoothness energy,

$$p(\mathbf{J}, B) \propto \exp\{-\int_{\Lambda/B} ||\nabla \mathbf{J}(x,y)||^2 dxdy - \lambda |B|\}.$$

The sketch graph S_{sk} can be viewed as an extension of the edges B, and the texture model is an extension of the smoothness term by both the filters and potentials. Along this vein, our work is interestingly related to the inpainting work (Chan and Shen, 01 and others), which adopts a PDE for filling in scratched pictures. The inpainting work is a variational method for minimizing the smoothness term. Our method is much more general in the potential formulation and simulates the texture by sampling, instead of maximization.

Our model also have some implications for the roles of neurons in V1. It is well known that V1 cells in primate visual cortex have Gabor like functions, but it is puzzling what roles the cells play in visual representation, because a Gabor function can be used as *filters* for pooling information to form the texture perception or be used as a linear *base* for representing the image primitive. We believe that the V1 cells can switch between the two roles when a critical sketchability condition occurs.

The primal sketch is a generic token representation of natural images. As Marr predicted, many middle level vision tasks can be built on this representation. Some most recent work in our lab have demonstrated that the primal sketch representation provides a representation for shape from shading, stereo, motion etc., which is more effective than the MRF on pixels, because of its power of knowledge representation and effective inference.

References

[1] A. J. Bell and T. J. Sejnowski. "An information maximization approach to blind separation and blind deconvolution." *Neural Computation*, **7**:1129–1159, 1995.

[2] A. Blake and A. Zisserman. *Visual Reconstruction*. Cambridge, MA. MIT press, 1987.

[3] J.R. Bergen and E. H. Adelson, "Theories of visual texture perception", *Spatial Vision*, D. Regan(eds.), CRC Press, 1991.

[4] E. J. Candes and D. L. Donoho. "Ridgelets: a key to higher-dimentional intermittency?" *Phil. Trans. R. Soc. Lond. A.*, **357**:2495–509, 1999.

[5] J. Canny. "A computational approach to edge detection." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **8**:679–698, 1986.

[6] T. Chan and J. Shen, "Local inpainting model and TV inpainting", *SIAM J. of Appl. Math*, 62:3, 1019-43, 2001.

[7] S. Chen, D. Donoho, and M. A. Saunders. "Atomic decomposition by basis pursuit." *SIAM Journal on Scientific Computing*, **20**:33–61, 1999.

[8] R. R. Coifman and M. V. Wickerhauser. "Entropy based algorithms for best basis selection." *IEEE Trans. on Information Theory*, **38**:713–18, 1992.

[9] T. F. Cootes, G. J. Edwards, and C. J. Taylor. "Active appearance models". *Proc. European Conf. on Computer Vision*, volume 2, pages 484-498. Springer, 1998

[10] I. Daubechies, "Orthogonal bases of compactly supported wavelets", *Comm. Pur. Appl. Math.*, volume 41, pages 909-996, 1998.

[11] R. A. Devore and B. Jawerth and B. J. Lucier, "Image composition through wavelet transform coding", *IEEE Trans. Information Theory*, volume 38(2), pages 719-746, 1992.

[12] A. A. Efros and W. T. Freeman, "Image Quilting for Texture Synthesis and Transfer", *SIGGRAPH*, 2001.

[13] J.H. Elder and S. W. Zucker, "Local scale control for edge detection and blur estimation." *IEEE Trans. PAMI*, vol. 20, no. 7, 699-716, 1998.

[14] B. Frey and N. Jojic. "Transformed component analysis: joint estimation of spatial transforms and image components." In *Proceedings of Int'l Conference on Computer Vision*, 1999.

[15] A. Fridman, "Mixed Markov Models", *Ph.D. Thesis, Department of Mathematics, Brown University*, 2000.

[16] S. Geman and D. Geman. "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **6**:721–741, 1984.

[17] C. E. Guo, S. C. Zhu, and Y. N. Wu, "Visual Learning by Integrating descriptive and Generative Methods" *Proc. 8th ICCV*, 2001.

[18] C.E. Guo, S. C. Zhu, and Y. N. Wu, "Modeling Visual Patterns by Integrating Descriptive and Generative Methods", *International Journal of Computer Vision*, Vol. 53, No. 1, pp.5-29, 2003.

[19] D. J. Heeger and J. R. Bergen, "Pyramid Based Texture Analysis/Synthesis", *Computer Graphics Proc.*, pp. 229-238, 1995.

[20] B. Julesz, "Visual pattern discrimination". *IRE Transactions onInformation Theory*, IT-8, pp. 84-92, 1962.

[21] B. Julesz, "Textons, the elements of texture perception and their interactions", *Nature*, Vol. 290, pp. 91-97, 1981.

[22] K. Koffka, *Principles of Gestalt Psychology*, 1935.

[23] J. Malik and R. Perona, "Preattentive Texture Discrimination with Early Vision Mechanisms", *J. Opt. Soc. AM* , Vol. 7, No. 5, pp. 923-932, 1990.

[24] S. Mallat and Z. Zhang, "Matching Pursuit in a Time-Frequency Dictionary", *IEEE Sig. Proc.*, 41, 3397-415, 1993.

[25] D. Marr, *Vision,* W. H. Freeman and Company, 1982.

[26] D. Mumford and J. Shah, "Optimal Approx. by Piecewise Smooth Functions and Assoc. Variational Prob.," *Comm. in Pure and Appl. Math*, 42(5), 577-685, 1989.

[27] B. A. Olshausen and D. J. Field, "Emergence of Simple-cell Receptive Field Properties by Learning a Sparse Code for Natural Images," *Nature*, Vol. 381, pp. 607-609, 1996.

[28] A. Pece, "The Problem of Sparse Image Coding," *Journal of Mathematical Imaging and Vision*, vol. 17(2), pp. 89-108, 2002.

[29] J. Romberg, H. Choi, R. Baraniuk, "Bayesian Tree-Structured Image Modeling using Wavelet Domain Hidden Markov Models", *IEEE Transactions on Image Processing*, Volume 10, No. 7, pp. 1056-1068, July 2001.

[30] E.P. Simoncelli, W.T. Freeman, E.H. Adelson, and D.J. Heeger, "Shiftable multiscale transforms", *IEEE Trans. on Info. Theory*, 38(2):587-607, 1992.

[31] Z. W. Tu and S. C. Zhu, "Image Segmentation by Data Driven Markov Chain Monte Carlo," *IEEE Tran. PAMI*, 24(5), 657-673, 2002.

[32] Y. N. Wu and S. C. Zhu "Equivalence of Julesz and Gibbs Ensembles", *Proc. of ICCV*, Corfu, Greece, 1999.

[33] S.C. Zhu, C.E. Guo, Y. Z Wang, and Z.J. Xu, "What are Textons?" *International Journal of Computer Vision,* vol.62(1-2), pp.121-143, April/May, 2005.

[34] S. C. Zhu, Y. N. Wu, and D. Mumford, "Minimax Entropy Principle and Its Applications in Texture Modeling", *Neural Computation*, 9(8), 1627-1660, 1997.