ELSEVIER

Primal sketch: Integrating structure and texture ☆

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

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

Abstract

This article proposes a generative image model, which is called "primal sketch," following Marr's insight and terminology. This model combines two prominent classes of generative models, namely, sparse coding model and Markov random field model, for representing geometric structures and stochastic textures, respectively. Specifically, the image lattice is divided into structure domain and texture domain. The sparse coding model is used to represent image intensities on the structure domain, where edge and ridge segments are modeled by image coding functions with explicit geometric and photometric parameters. The edge and ridge segments form a sketch graph whose nodes are corners and junctions. The sketch graph is governed by a simple spatial prior model. The Markov random field model is used to summarize image intensities on the texture domain, where the texture patterns are characterized by feature statistics in the form of marginal histograms of responses from a set of linear filters. The Markov random fields in-paint the texture domain while interpolating the structure domain seamlessly. A sketch pursuit algorithm is proposed for model fitting. A number of experiments on real images are shown to demonstrate the model and the algorithm.
© 2006 Elsevier Inc. All rights reserved.

Keywords: Sparse coding; Markov random fields; Image primitives; Sketch graphs; Lossy image coding

1. Introduction

Geometric structures and stochastic textures are two ubiquitous classes of visual phenomena in natural scenes. Geometric structures appear simple, and can be represented by edges, ridges, and their compositions such as corners and junctions. Stochastic textures appear complex, and are often characterized by feature statistics. Despite their apparent distinctions, texture impressions are often caused by large number of object structures that are either too small or too distant relative to camera resolution. Moreover, with the change of viewing distance or camera resolution, the same group of objects may appear either as structures or textures. It is therefore desirable to integrate structures and textures in a common representational and computational framework.

In this article, a generative model call "primal sketch" is proposed, following the insight and terminology of Marr [15]. The model combines two prominent classes of generative models. One is sparse coding model, for representing geometric structures. The other is Markov random field model, for characterizing stochastic textures.

Specifically, the image lattice is divided into structure domain and texture domain. The sparse coding model is used to represent image intensities on the structure domain, where the most common structures are boundaries of objects that are above a certain scale. Following Elder and Zucker [7], we model the image intensities of object boundaries by a small number of edge and ridge coding

functions with explicit geometric and photometric parameters. For instance, an edge segment is modeled by an elongate step function convolved with a Gaussian kernel. A ridge segment is a composition of two parallel edge segments. These edge and ridge segments form a sketch graph, whose nodes are corners and junctions. The sketch graph is regulated by a simple spatial prior model. The form of our sparse coding model is similar to vector quantization [9], where the coding functions serve as coding vectors.

The Markov random field model is used to summarize image intensities on the texture domain, where the texture patterns are characterized by feature statistics in the form of marginal histograms of responses from a set of linear filters. The Markov random fields in-paint the texture domain while interpolating the structure domain seamlessly.

Fig. 1 shows an example. (a) is the observed image. (b) is the sketch graph, where each line segment represents an edge or ridge coding function. (c) is the reconstructed structure domain of the image using these edge and ridge coding functions. (d) is a segmentation of the remaining texture domain into a number of homogeneous texture regions, by clustering the local marginal histograms of filter responses. Here different regions are represented by different shades. (e) displays the synthesized textures in the segmented regions. (f) is the final synthesized image by putting (c) and (e) together. Because the textures are synthesized with the structure domain as boundary conditions, the textures interpolate the structure domain seamlessly.

In the primal sketch representation, the sparse coding model and the Markov random field model are intrinsically connected. The elongate and oriented linear filters, such as Gabor filters [6] or Difference of Gaussian filters [25], are used to detect edges or ridges at different frequencies. On the structure domain where edge and ridge segments are present, the filters have large responses along the edge or ridge directions, and the optimally tuned filters are highly connected and aligned across space and frequency. Such regularities and redundancies in filter responses can be accounted for by the image coding functions for edge and ridge segments. On the remaining texture domain where the filters fail to detect edges or ridges, the filter responses are weak and they are not well aligned over space or frequency. So they can be pooled into the marginal histograms of filter responses to form statistical summaries of the image intensities. In that sense, texture statistics arise from recycling the responses of filters that fail to detect edges or ridges. The sparse coding model and random field model represent two different schemes for grouping or combining filter responses.

The rest of the paper is organized as follows. Section 2 reviews sparse coding model and random field model to set the background, and then motivates the integrated modeling scheme. Section 3 presents the primal sketch representation. Section 4 describes the computational algorithm. Section 5 shows a set of experiments on real images. Section 6 concludes with discussion.

2. Background and motivation

Image modeling has been based on two alternative theories. The first theory originated from computational harmonic analysis. It represents an image deterministically by
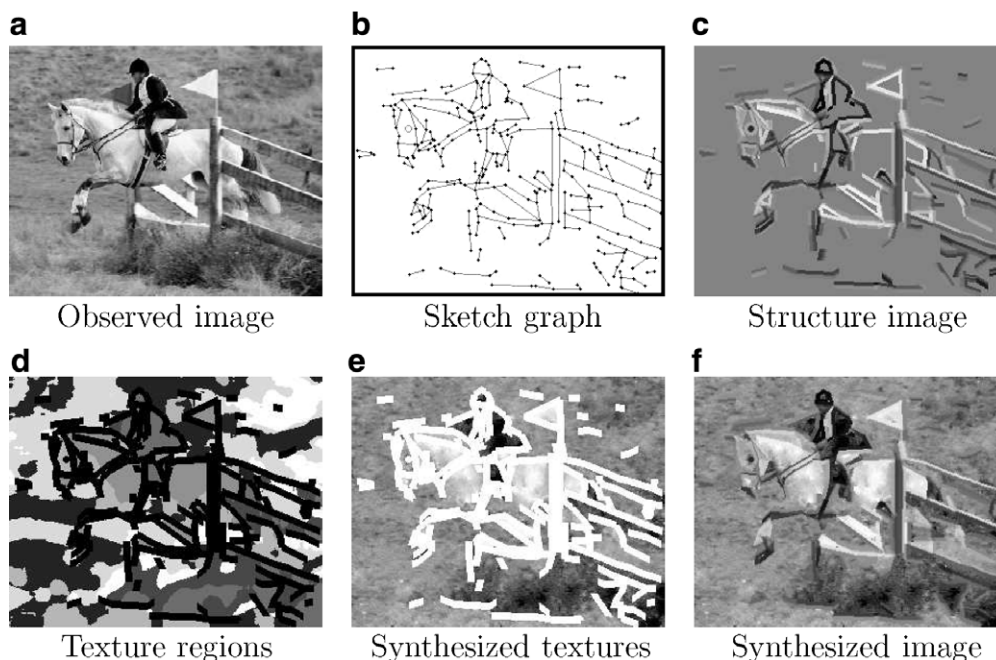

Fig. 1. An example of the primal sketch model. (a) An observed image. (b) The sketch graph computed from the image. (c) The structure domain of the image. (d) The remaining texture domain is segmented into a number of homogeneous texture regions. (e) Synthesized textures on these regions. (f) The final synthesized image that integrates seamlessly the structure and texture parts.

a linear superposition of wavelet functions. The second theory originated from statistical physics. It characterizes an image by a Markov random field, which describes the image in terms of a set of feature statistics.

Both theories involve a set of localized, oriented, elongated filters (as well as some isotropic filters) at different scales or frequencies. Two popular forms of the linear filters are Gabor functions [6] and Difference of Gaussian functions [25]. Such functions are localized in both spatial and frequency domains. Fig. 2 displays a set of Difference of Gaussian functions. Gabor filters are similar in shape. Both forms of functions are biologically motivated by observations on the simple cells in primary visual cortex. While the Difference of Gaussian filters emphasize the spatial domain concept of gradient and Hessian, the Gabor filters emphasize the frequency domain concept of local Fourier contents.

2.1. Sparse coding model

Let $\mathbf{I}(x,y),(x,y) \in \Lambda$ be an image defined on an image lattice Λ. Let $\{B_i(x,y), i = 1,\ldots,N\}$ be a set of coding elements such as those depicted in Fig. 2. The image $\mathbf{I}$ can be represented by

$$\mathbf{I}(x,y) = \sum_{i=1}^{N} c_i B_i(x,y) + \epsilon(x,y), \tag{1}$$

where c_i are the coefficients and ϵ is the residual image. This linear representation has been intensively studied in image coding and harmonic analysis. The interested reader is referred to the book of Mallat [13] and the references therein.

The dictionary of coding elements $\{B_i(x,y), i = 1,\ldots,N\}$ can be over-complete, in the sense that the number of coding functions N is greater than the number of pixels in image $\mathbf{I}$. The main principle in over-complete representation is sparsity. That is, the dictionary $\{B_i\}$ should be designed in such a way that for a typical natural image,
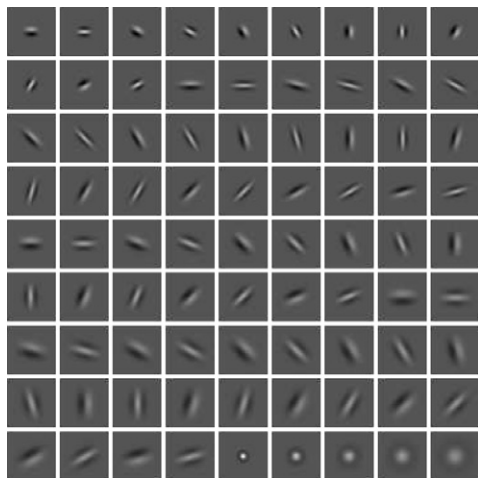

Fig. 2. A set of Difference of Gaussian filters, including elongate and oriented filters and isotropic filters.

only a small number of elements need to be chosen to approximate it within a small error, i.e., only a small number of coefficients in $\{c_i\}$ of Eq. (1) need to be significantly different from 0. Using this principle, Olshausen and Field [18] learned an over-complete set of coding elements that resemble the Gabor functions. The notion of sparsity can also be expressed statistically by assuming that $\{c_i\}$ follow independent long-tail distributions. See the paper by Pece [19] for a thorough review of this topic.

With a given over-complete dictionary of coding elements, the matching pursuit algorithm of Mallat and Zhang [14] can be employed to find a sparse representation of an image. The basic idea of this algorithm is to add one element at a time, and each time, the element that results in the maximum reduction in the L_2-norm of the reconstruction error is added to the representation.

We did some experiments using linear sparse coding with a set of Gabor functions as well as isotropic Laplacian of Gaussian functions. Since these functions can be centered at every pixel, the dictionary of all the elements is highly over-complete. Fig. 3 shows an example of sparse coding: (a) is an observed image of 128×128 pixels; (b) is the image reconstructed by 300 elements selected by the matching pursuit algorithm. Fig. 4 shows a second example with a symbolic representation, where each selected elongate and oriented Gabor function is represented by a bar at the same location, with the same elongation and orientation. The isotropic elements are represented by circles.

The linear sparse coding model can capture the image structures such as object boundaries with a small number of elements. However, a small number of elements cannot represent textures very well, since textures are often very random and are of high complexity. If the linear additive model is to be used to represent textures, the representation will not be sparse.

Moreover, the linear additive model is still not adequate for representing structures. From the reconstructed image of Fig. 4, one can see that a small number of elements cannot capture the boundaries very well, and they do not line up into lines and curves. There are no concepts of corners and junctions either.

2.2. Markov random field model

The set of Gabor or Difference of Gaussian filters are denoted by $\{F_k, k = 1,\ldots,K\}$, where k indexes the shape of the filter. The response of filter k at pixel (x,y) is denoted $[F_k * \mathbf{I}](x,y)$ or simply $F_k * \mathbf{I}(x,y)$. These operators are generalized versions of the ubiquitous gradient operator $\nabla \mathbf{I}$ that is used in Canny edge detection [3] and variational/PDE approaches to image processing (see the recent book of Aubert and Kornprobst [1], and the references therein).

Zhu et al. [26] proposed a Markov random field model [2] for textures that can be written in the form of the following Gibbs distribution:
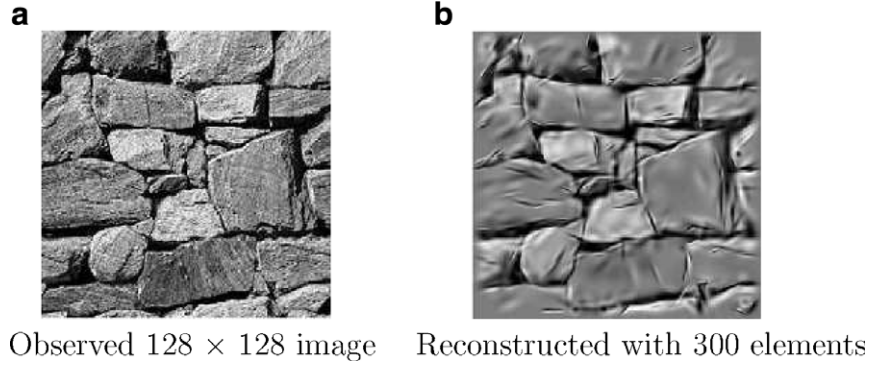
Fig. 3. A sparse coding example: the representation is computed by matching pursuit.
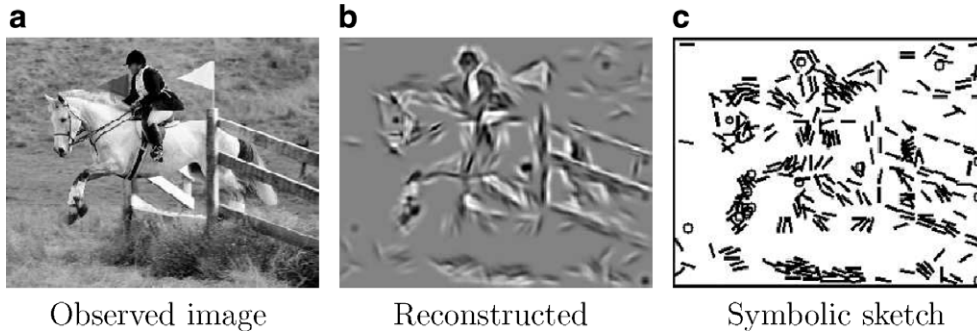

Fig. 4. A sparse coding example computed by matching pursuit, (a) is the observed image, (b) is reconstructed with 300 elements, (c) is a symbolic representation where each selected elongate and oriented element is represented by a bar at the same location, with the same elongation and orientation. The isotropic elements are represented by circles.

$$p(\mathbf{I}) = \frac{1}{Z} \exp \left\{ -\sum_{x,y} \sum_k \phi_k(F_k * \mathbf{I}(x,y)) \right\}, \qquad (2)$$

where $\phi_k()$ are one-dimensional functions, and Z is the normalizing constant depending on $\{\phi_k()\}$. $\phi_k()$ can be further parameterized as piecewise constant functions. Model (2) is an embellished version of the early ϕ-model of Geman and Graffigne [8], which can be used as a prior distribution for Bayesian image processing. The reader is refer to the book of Winkler [23] for a review of Markov random fields and their applications in image processing. The energy function in model (2) can also be viewed as a generalized form of the regularization terms commonly used in variational/PDE methods for controlling image smoothness [1].

The above model has an interesting connection to feature statistics. For each operator F_k, the responses over the whole image lattice Λ, $\{F_k*\mathbf{I}(x,y), \ \forall(x,y) \in \Lambda\}$, can be summarized into a histogram $h_k(\mathbf{I})$, where for each bin of this histogram, the proportion of the responses falling into this bin is calculated, regardless of the positions of these responses. Specifically, $h_k = \{h_{k,z}, \ \forall z\}$, and

$$h_{k,z} = \frac{1}{|\Lambda|} \sum_{(x,y) \in \Lambda} \delta(z; F_k * \mathbf{I}(x,y)), \qquad (3)$$

where z indexes the histogram bins, and $\delta(z;x) = 1$ if x belongs to bin z, and $\delta(z;x) = 0$ otherwise.

These histograms are called the marginal histograms [10]. Let $h(\mathbf{I}) = (h_k(\mathbf{I}), k = 1, \ldots, K)$. Then let us consider the following image ensemble

$$\Omega(h) = \{\mathbf{I} : h(\mathbf{I}) = h\} = \{\mathbf{I} : h_k(\mathbf{I}) = h_k, k = 1, \ldots, K\}, \qquad (4)$$

that is, the set of images that produce the same histograms $h = (h_1, \ldots, h_K)$. This ensemble is termed micro-canonical ensemble in statistical physics.

According to the fundamental result of "equivalence of ensembles" in statistical physics [5], the Markov random field model or the Gibbs distribution (2) converges to the uniform distribution over the micro-canonical ensemble $\Omega(h)$ for some h, as the image lattice Λ goes to Z^2, i.e., the infinite lattice. Inversely, under the uniform distribution over the micro-canonical ensemble $\Omega(h)$, if h is fixed and $\Lambda \to Z^2$, then for any fixed local patch $\Lambda_0 \subset \Lambda$, the distribution of the image patch on Λ_0, i.e., $\mathbf{I}_{\Lambda_0}$, follows the Markov random field model (2) for some $\{\phi_k()\}$.

Therefore, for a relatively large image lattice, if the Markov random field model (2) is to be fitted to an observed image, and synthesize images are to be sampled from the fitted model, the marginal histograms of filter responses h can be estimated directly from the observed image, and synthesized images can be simulated by randomly sampling from the image ensemble $\Omega(h)$. This can be accomplished
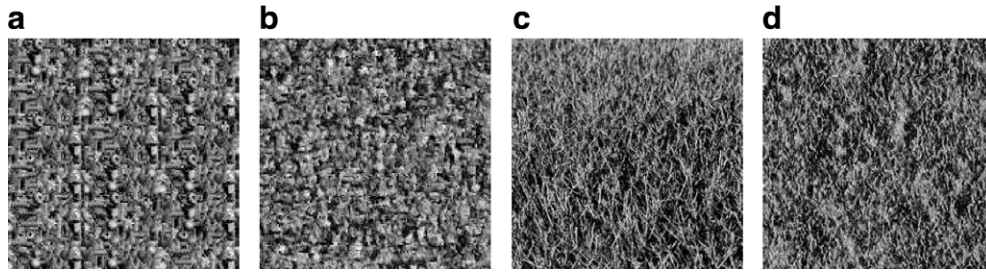
Fig. 5. Random field and marginal histograms: (a) and (c) are observed images. (b) and (d) are "reconstructed" by matching marginal histograms of filter responses.

using simulated annealing algorithm. See Wu et al. [24] for more details.

Experiments show that this model is quite effective in representing stochastic textures. See Fig. 5 for two examples. However, as is evident in Fig. 6, the random field models are ineffective in representing large image structures such as long edges.

2.3. Edge and ridge model

To understand the connection between the sparse coding model and the Markov random field model in the primal sketch representation, one may consider the most common image structures: edges and ridges. Elder and Zucker [7] represent an edge segment as an elongate step function convolved with a Gaussian kernel. A ridge is a composition of two parallel edges. See Fig. 7 for an illustration.

Consider, for example, a long step edge between two flat regions. Such an image structure evokes strong responses from Gabor filters (or Difference of Gaussian filters) across space and frequency (or scale). At each frequency or scale, the locations and orientations of the edge points can be detected by finding the optimally tuned elements in the same manner as Canny edge detection [3]. The orientations of these optimally tuned elements are highly aligned across space and frequency. That is, the linear filter responses from such an edge structure form a very regular pattern.

In the primal sketch representation scheme, after detecting the locations and orientations of the edge points using linear filters, the intensities of the edge segment are modeled explicitly by fitting the model of Elder and Zucker [7]. This is the domain where sparse coding model applies.

The optimally tuned Gabor elements can also be used to code the edge structure, but quite a number of Gabor
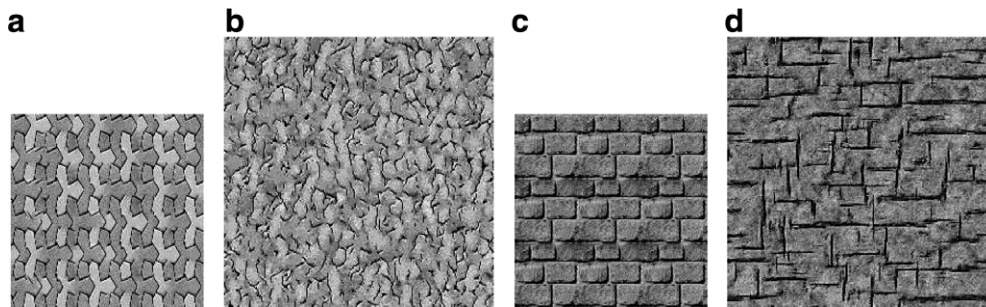

Fig. 6. Random field and marginal histograms: (a) and (c) are observed images, (b) and (d) are "reconstructed" by matching marginal histograms of filter responses.
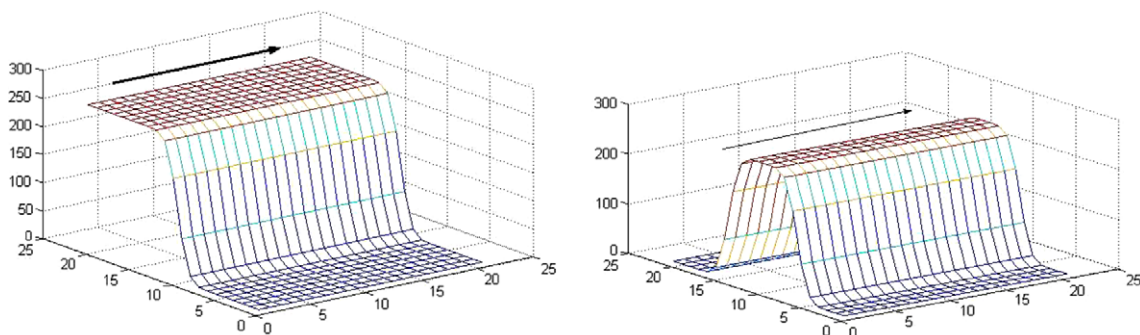

Fig. 7. An edge is modeled by a step function convolved with a Gaussian kernel. A ridge is modeled by double edges.

elements are needed. This is because a Gabor element is localized in both spatial and frequency domains, whereas a long step edge between two flat regions spans large ranges in both spatial and frequency domains. In order to capture the sparsity of the edge structure, the highly regular pattern of these Gabor elements and their coefficients must be modeled. In this article, we choose not to do that, but to fit the parametric functions of edges and ridges directly.

On the part of the image where no edges or ridges are present, the filter responses are usually weak and are not well aligned in either spatial or frequency domain. In that case, the marginal histograms of filter responses can be pooled to summarize the texture patterns. The marginal histograms are used because the filter responses are not well aligned into joint patterns. This is the domain where the Markov random field model applies.

3. Primal sketch model

According to the model, the image is generated as a mosaic as follows: the image lattice Λ is divided into a structure domain Λ_{str} and a textured domain Λ_{tex}. The image intensities on the structure domain are represented by a set of coding functions for edges and ridges. The image intensities on the texture domain are characterized by Markov random fields that interpolate the structure domain of the image.

3.1. Structure domain

The model for the structure domain of the image is

$$\mathbf{I}(x,y) = \sum_{i=1}^{n} B(x,y|\theta_i) + \epsilon(x,y), \quad (x,y) \in \Lambda_{str}. \quad (5)$$

The coding functions $B(x,y|\theta_i)$ are used to represent edge and ridge segments (as well as blobs) in the image, where θ_i are geometric and photometric parameters of these coding functions. Let $\Lambda_{str,i}$ be the set of pixels coded by $B(x,y|\theta_i)$. They do not overlap each other except over the small number of pixels where they join each other to form

corners and junctions. Therefore, $B(x,y|\theta_i)$ is similar to coding vectors in vector quantization.

An edge segment is modeled by a 2D function that is constant along the edge, and has a profile across the edge. Specifically,

$$B(x,y \mid \theta) = f((x-u)\sin\alpha - (y-v)\cos\alpha), \quad (6)$$

where

$$-l \leqslant (x-u)\cos\alpha + (y-v)\sin\alpha \leqslant l,$$
$$-w \leqslant (x-u)\sin\alpha - (y-v)\cos\alpha \leqslant w.$$

That is, the function $B(x,y|\theta)$ is supported on a rectangle centered at (u,v), with length $2l+1$, width $2w+1$, and orientation α.

For the profile function $f()$, let $f_0(x) = -1/2$ for $x < 0$ and $f_0(x) = 1/2$ for $x \geqslant 0$, and let $g_s()$ be a Gaussian function of standard deviation s. Then $f() = a + b f_0()*g_s()$. This is the model proposed by Elder and Zucker [7]. The convolution with Gaussian kernel is used to model the blurred transition of intensity values across the edge, caused by the three dimensional shape of the underlying physical structure, as well as the resolution and focus of the camera. As proposed by Elder and Zucker [7], the parameter s can be determined by the distance between the two extrema of the second derivative $f''()$. See Fig. 8a for an illustration.

Thus in the coding function $B(x,y|\theta)$ for an edge segment, $\theta = (t,u,v,\alpha,l,w,s,a,b)$, namely, type (which is edge in this case), center, orientation, length, width, sharpness, average intensity, intensity jump. θ captures geometric and photometric aspects of an edge explicitly, and the coding function is non-linear in θ.

A ridge segment has the same functional form as (6), where the profile $f()$ is a composition of two edge profiles. See Fig. 8b for an illustration, where there are three flat regions. The profile of a multi-ridge is a composition of two or more ridge profiles. A blob function is modeled by rotating an edge profile, more specifically, $B(x, y \mid \theta) = f(\sqrt{(x-u)^2 + (y-v)^2} - r)$, where $(x-u)^2 + (y-v)^2 \leqslant R^2$, and again $f() = a + b f_0()*g_s()$ being a step edge
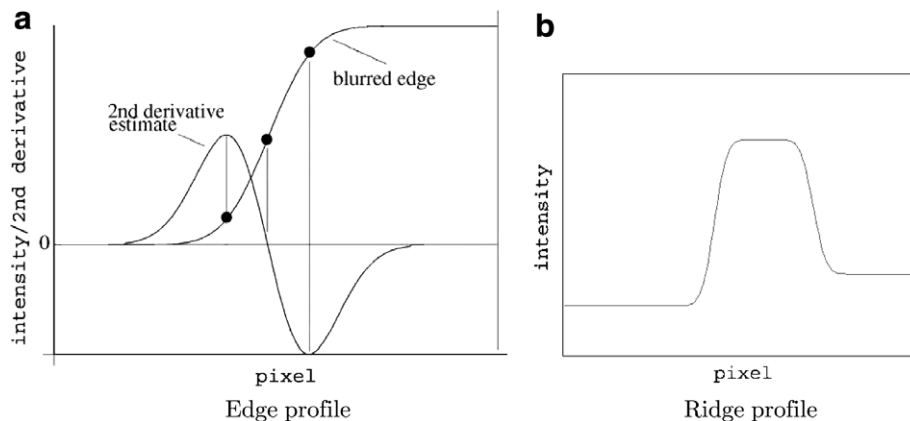

Fig. 8. (a) An edge profile and its second derivative. The blurring scale is determined by the distance between the extrema of the second derivative. (b) A ridge profile is a composition of two edge profiles.

convolved with a Gaussian kernel. This function is supported on a disk area centered at (u,v) with radius R. The transition of intensity occur at the circle of radius $r < R$.

The corners and junctions are important structures in images. They are modeled as compositions of edge or ridge functions. When a number of such coding functions join to form a corner or a junction, the image intensities of the small number of overlapping pixels are modeled as averages of these coding functions. The end point of a ridge is modeled by a half blob.

See Fig. 9 for a sample of local structure elements, which are the coding functions and their combinations. There are eight types of elements: blobs, end points, edges, ridges, multi-ridges, corners, junctions and crosses. Fig. 9a shows the symbolic representations of these elements. Fig. 9b displays the image patches of these elements.

Let $S_{str} = (\theta_i, i = 1, \ldots, n)$ be the sketch graph formed by these coding functions. The graph has a set of nodes or vertices $V = \cup_{d=0}^{4} V_d$, where V_d is the set of nodes with degree d, i.e., the nodes with d arms. For instance, a blob node has degree 0, an end point has degree 1, a corner has degree 2, a T-junction has degree 3, and a cross has degree 4. Nodes with more than 4 arms are not allowed. S_{str} is regularized by a simple spatial prior model:

$$p(S_{str}) \propto \exp\left\{-\sum_{d=0}^{4} \lambda_d |V_d|\right\}, \quad (7)$$

where $|V_d|$ is the number of nodes with d arms. The prior probability or the energy term $\gamma_{str}(S_{str}) = \sum_{d=0}^{4} \lambda_d |V_d|$ penalizes free end points by setting λ_1 at a large value. The concrete parameter values will be given in Section 4.

3.2. Texture domain

The texture domain Λ_{tex} is segmented into m regions of homogenous texture patterns, $\Lambda_{tex} = \cup_{j=1}^{m} \Lambda_{tex,j}$. Within each region j, the marginal histograms of the responses from the K filters, $h_j = (h_{j,k}, k = 1, \ldots, K)$, are pooled,

$$h_{j,k,z} = \frac{1}{|\Lambda_{tex,j}|} \sum_{(x,y) \in \Lambda_{tex,j}} \delta(z; F_k * \mathbf{I}(x,y)), \quad (8)$$

where z indexes the histogram bins, and $\delta(z;x) = 1$ if x belongs to bin z, and $\delta(z;x) = 0$ otherwise.

According to the previous section, this is equivalent to a Markov random field model for each texture region:

$$p(\mathbf{I}_{\Lambda_{tex,j}}) \propto \exp\left\{-\sum_{(x,y) \in \Lambda_{tex,j}} \sum_{k=1}^{K} \phi_{j,k}(F_k * \mathbf{I}(x,y))\right\}. \quad (9)$$

These Markov random fields have the structure domain as boundary conditions, because when the filters F_k are applied on the pixels in Λ_{tex}, these filters may also cover some pixels in Λ_{str}. These Markov random fields in-paint the texture domain Λ_{tex} while interpolating the structure domain Λ_{str}, and the in-painting is guided by the marginal histograms of linear filters within each region. This point of view is closely related to the in-painting work of Chan and Shen [4].

Let $S_{tex} = (\Lambda_{tex,j}, j = 1, \ldots, m)$ denotes the segmentation of the texture domain. S_{tex} follows a prior model $p(S_{tex}) \propto \exp\{-\gamma_{tex}(S_{tex})\}$, for instance, $\gamma_{tex}(S_{tex}) = \rho m$ to penalize the number of regions. More sophisticated models can be used for segmentation (see, e.g., Tu et al. [22]).
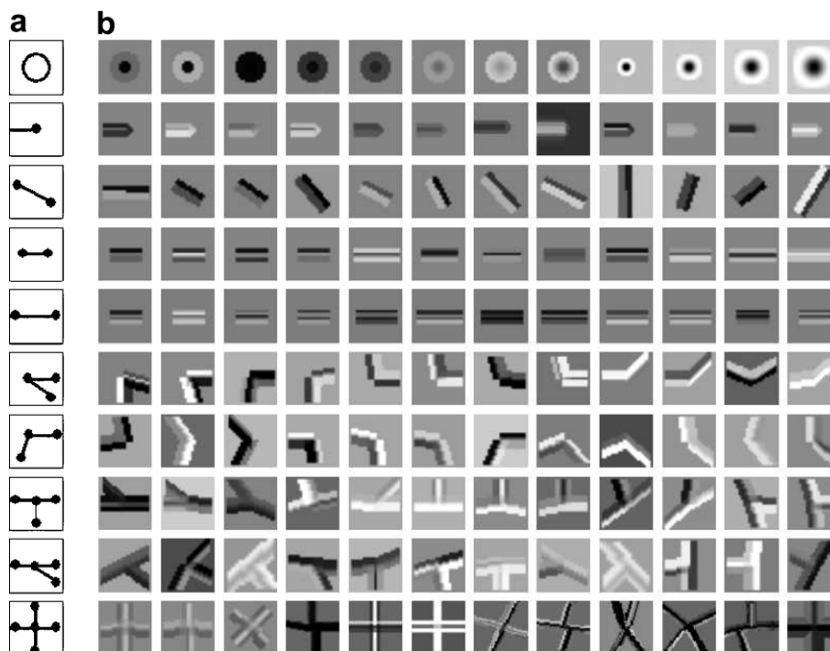

Fig. 9. A collection of local structure elements produced by our model. There are eight types of elements: blobs, end points, edges, ridges, multi-ridges, corners, junctions and crosses. (a) The symbolic representation. (b) The photometric representation.

3.3. Integrated model

Formally, the structure model (5) and the texture model (9) can be integrated into a probability distribution. Our inspiration for such an integration comes from the model of Mumford and Shah [17]. In their method, the prior model for the noiseless image can be written as

$$p(\mathbf{I}, S) = \frac{1}{Z} \exp\left\{ -\sum_{(x,y)\in A/S} \lambda |\nabla \mathbf{I}(x,y)|^2 - \gamma|S| \right\}, \qquad (10)$$

where S is a set of pixels of discontinuity that correspond to the boundaries of objects, and $|S|$ is the number of pixels in S. In model (10), S is the structure domain of the image, and the remaining part is the texture domain.

The primal sketch model can be viewed as an extension of the Mumford-Shah model. Let $S = (S_{\text{str}}, S_{\text{tex}})$,

$$
\begin{aligned}
p(\mathbf{I}, S) = \frac{1}{Z} \exp\Big\{ &-\sum_{i=1}^{n} \sum_{(x,y)\in A_{\text{str},i}} \frac{1}{2\sigma^2}(\mathbf{I}(x,y) \\
&- B_i(x, y \mid \theta_i))^2 - \gamma_{\text{str}}(S_{\text{str}}) \\
&- \sum_{j=1}^{m} \sum_{(x,y)\in A_{\text{tex},j}} \sum_{k=1}^{K} \phi_{j,k}(F_k * \mathbf{I}(x,y)) \\
&- \gamma_{\text{tex}}(S_{\text{tex}}) \Big\}.
\end{aligned}
\qquad (11)
$$

Compared to Mumford–Shah model, model (11) is more sophisticated in both structure part and texture part.

4. Computational algorithm

This section details the algorithm for computing $S = (S_{\text{str}}, S_{\text{tex}})$ from an observed image $\mathbf{I}$. Ideally, the algorithm should maximize the logarithm of $p(\mathbf{I}, S)$ defined in (11) with respect to S. Unfortunately the computation can be too expensive. So an approximate but efficient algorithm is adopted. The algorithm consists of two parts. The first part is called sketch pursuit, which computes S_{str} guided by an objective function to be derived in the following subsection. The sketch pursuit is further divided into three phases, where the objective function is used in the last phase. The second part of the algorithm is texture clustering, which computes S_{tex} by a simple clustering method after S_{str} is computed. There is currently no proof that the algorithm will lead to a local mode of (11). This issue is to be investigated in further work.

4.1. Objective function for computing S_{str}

To derive the objective function, let's first consider the following simple scenario first. Suppose one wants to detect whether a coding function $B(x,y|\theta)$ is present in image $\mathbf{I}$ by solving the following hypothesis testing problem:

$$H_0 : \mathbf{I}(x,y) = \mu + N(0, \sigma^2)$$
$$H_1 : \mathbf{I}(x,y) = B(x,y|\theta) + N(0, \sigma^2), \ (x,y) \in A_1,$$

where A_1 is the set of pixels covered by $B(x,y|\theta)$. μ in the null hypothesis H_0 is estimated by the average of the pixel values in A_1. Define

$$\Delta L(B) = \sum_{(x,y)\in A_1} [(\mathbf{I}(x,y) - \mu)^2 - (\mathbf{I}(x,y) - B(x,y \mid \theta))^2]. \qquad (12)$$

Then the log likelihood ratio score for the above hypothesis testing is $\Delta L(B)/2\sigma^2$.

In order to infer the sketch graph $S_{\text{str}} = \{B_i, i = 1, \ldots, n\}$ from the image $\mathbf{I}$, the following objective function is used,

$$
\begin{aligned}
L(S_{\text{str}}) &= \frac{1}{2\sigma^2} \sum_{i=1}^{n} \Delta L(B_i) - \gamma_{\text{str}}(S_{\text{str}}) \\
&= \frac{1}{2\sigma^2} \left[\sum_{i=1}^{n} \Delta L(B_i) - \tilde{\gamma}_{\text{str}}(S_{\text{str}}) \right],
\end{aligned}
\qquad (13)
$$

where $\tilde{\gamma}_{\text{str}}(S_{\text{str}}) = 2\sigma^2 \gamma_{\text{str}}(S_{\text{str}})$. In practice, only $\tilde{\gamma}_{\text{str}}(S_{\text{str}})$ needs to be specified, without σ^2 being explicitly specified, for maximizing $L(S_{\text{str}})$.

The objective function (13) can be justified as the regularized log-likelihood of the following working model:

$$\mathbf{I}(x,y) = B_i(x,y \mid \theta_i) + N(0, \sigma^2), (x,y) \in A_{\text{str},i}, \quad i = 1, \ldots, n,$$
$$\mathbf{I}(x,y) = \mu(x,y) + N(0, \sigma^2), (x,y) \in A_{\text{tex}}. \qquad (14)$$

This working model is a simplified version of the integrated model presented in the previous section, where the texture part A_{tex} is modeled by Gaussian distribution with slowly varying mean $\mu(x,y)$. Specifically, let l_0 be the log-likelihood of the following background model:

$$\mathbf{I}(x,y) = \mu(x,y) + N(0, \sigma^2), (x,y) \in A,$$

then the log-likelihood of the working model (14) is $l_0 + \sum_{i=1}^{n} \Delta L(B_i)/(2\sigma^2)$, assuming that for $(x,y) \in A_{\text{str},i}$, i.e., pixels covered by B_i, $\mu(x,y)$ in the background model are all equal to the average of the pixel values in $A_{\text{str},i}$. Therefore $L(S_{\text{str}})$ in (13) is the regularized log-likelihood of the working model (14). Similar objective functions in the form of likelihood ratios have been previously constructed by Pece [20] and references therein.

The reason that the working model (14) is used is mainly due to its computational simplicity. It also fits the logic that texture statistics arise by pooling the filter responses where no edges or ridges are detected.

The algorithm for computing S_{str} is called the sketch pursuit algorithm. It consists of the following three phases. Phase 0: an edge and ridge detector based on linear filters is run to give an initialization for the sketch graph. Phase 1: a greedy algorithm is used to determine the sketch graph but without using the spatial prior model. Phase 2: a greedy algorithm based on a set of graph operators is used to edit the sketch graph to achieve good spatial organization by achieving a local mode of the objective function.

4.2. Sketch pursuit Phase 0: detect edges, ridges, and blobs

The detection method used here is essentially an extension of Canny edge detector [3]. At each pixel, the sum of the squares of D^1G and D^2G responses is computed for each scale and orientation, where D^1G and D^2G denote the Difference of Gaussian filters for computing the first derivatives and second derivatives, respectively. The D^1G and D^2G filters can also be replaced by Gabor sine and Gabor cosine filters, respectively. The kernel functions of the filters are normalized to have mean 0 and L_1 norm 1. In the current implementation, 3–5 scales and 18 orientations are used. The maximum of the combined responses over these scales and orientations is considered the edge–ridge strength at that pixel, and the orientation of the corresponding filters is considered the orientation of this pixel. See Fig. 10 for an example: (a) is the observed image. (b) is the map of the edge–ridge strengths. Using the non-maxima suppression method in Canny edge detector, a pixel is considered a detected edge–ridge point if its edge–ridge strength is above a threshold, and achieves local maximum among the pixels on the norm direction of the edge–ridge orientation of this pixel. Fig. 10c is a binary map that displays the detected edge–ridge points.

The blob strength at a pixel is measured by the maximum response of the isotropic filters over a number of scales. Thresholding and non-maxima suppression are used to get the detected blobs, as shown in Fig. 10d.

4.3. Sketch pursuit Phase 1: sequentially add coding functions

Phase 1 follows Phase 0. The image is assumed to have already been normalized to have mean 0 and variance 1. From the edge–ridge map, the algorithm finds the edge–ridge point of the maximum strength over the whole image. From this point, the algorithm connects the edge–ridge points along the orientation of this current point to form a line segment, until the line segment cannot be extended any further. This procedure is similar to the maximal alignment of Moisan et al. [16].

After that, the profile of this line segment is obtained by averaging the image intensities on the cross-sections of the line segment. Then the extrema of the second derivative of the profile (see Fig. 8 for the illustration of the extrema of the second derivative) are located. The width of the profile is determined so that there are two pixels beyond the extrema on the two sides of the profile. The algorithm starts from the profile of 7 pixel width, and then increases the width until the above criterion is met. By doing so, only the intensity transition across the edge or ridge is modeled by the coding function, while the flat regions on the two sides are left to be captured by texture models. Then the algorithm fits an edge or ridge function B_1, and compute $\Delta L(B_1)$. If $\Delta L(B_1) > \epsilon$, then B_1 is accepted. $\epsilon = 5$ (for image normalized to mean 0 and variance 1) in the current implementation.

Observed image Edge-ridge strength

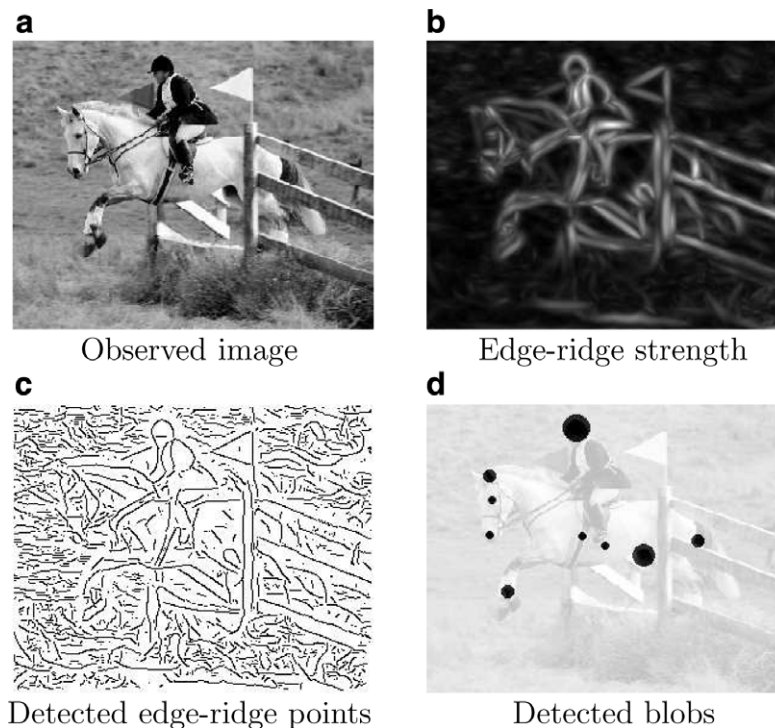Detected edge-ridge points Detected blobs

Fig. 10. Sketch pursuit Phase 0: detecting edges, ridges, and blobs using derivative filters. For the observed image (a), a set of derivative filters are applied to get the edge–ridge strength (b). The detected edge–ridge points (c) are computed by non-maxima suppression. (d) Shows the detected blobs.
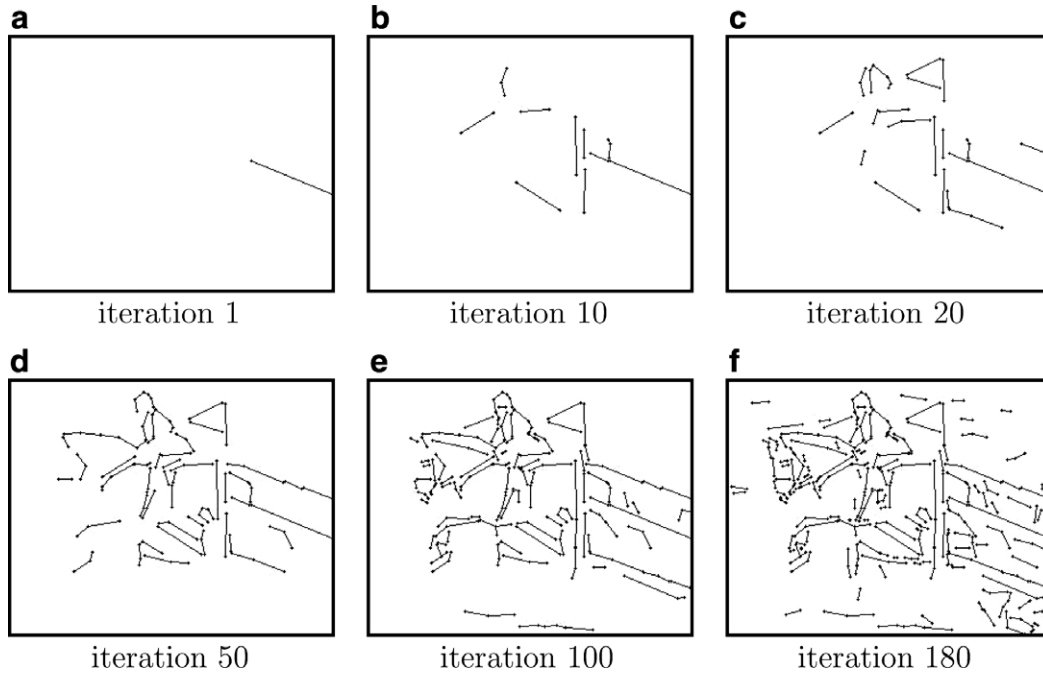
Fig. 11. Sketch pursuit Phase 1: sequentially add coding functions. The sketch graph is shown after iterations $1, 10, 20, 50, 100,$ and 180.

operators	graph change	illustration
G_1, G_1'	create/delete	
G_2, G_2'	grow/shrink	
G_3, G_3'	connect/disconnect	
G_4, G_4'	extend to touch/remove to disconnect	
G_5, G_5'	extend to join/remove to disconnect	
G_6, G_6'	combine/break	
G_7, G_7'	combine/split	
G_8, G_8'	merge/split	
G_9, G_9'	create/remove a blob	
G_{10}, G_{10}'	switch between stroke(s) and a blob	

Fig. 12. The 10 pairs of reversible graph operators used in the sketch pursuit process.
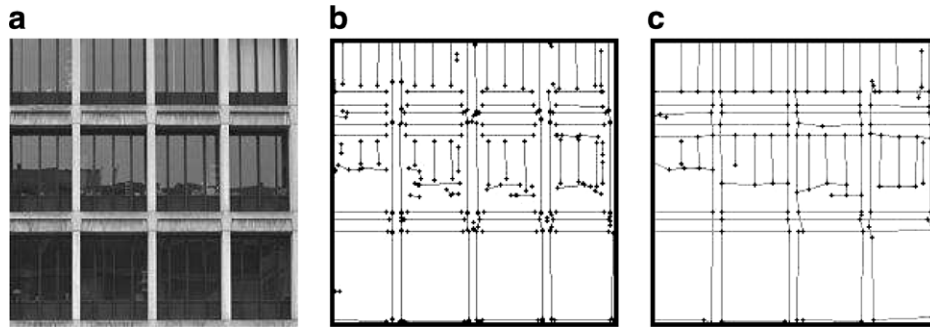
Fig. 13. Sketch pursuit. (a) Observed image. (b) Sketch graph after Phase 1. (c) Sketch graph after Phase 2.
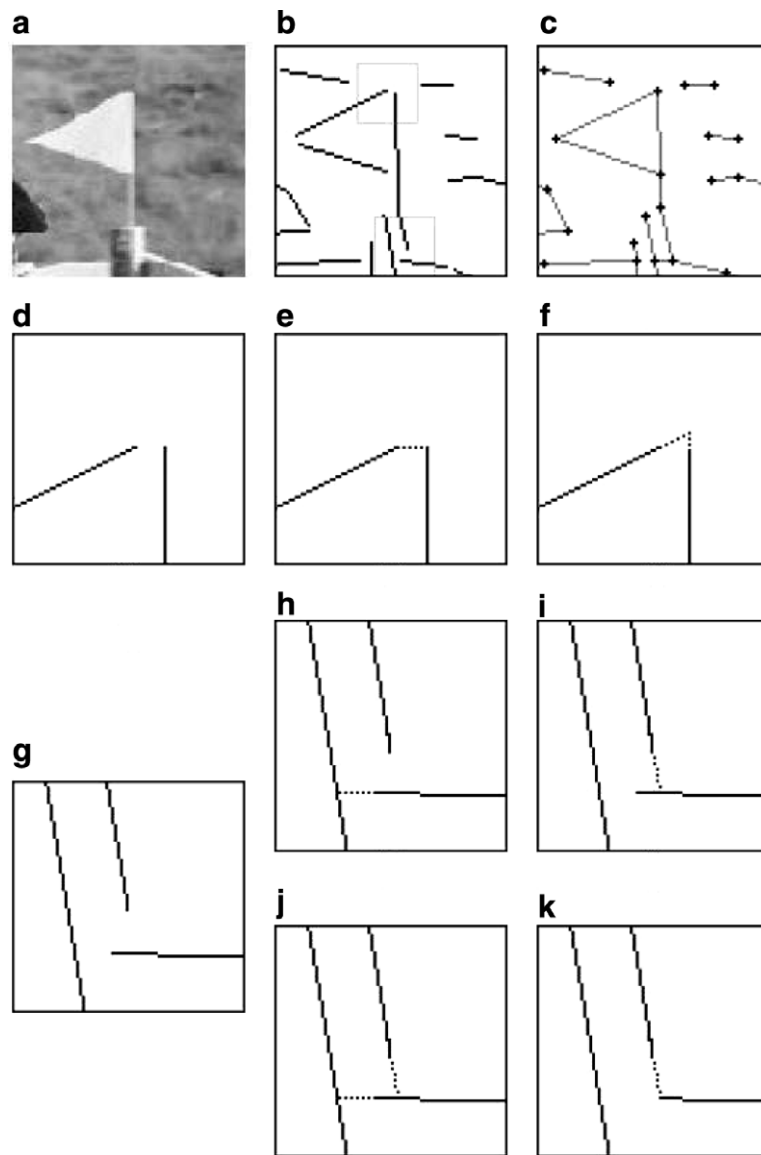

Fig. 14. An example of applying graph operators. (a) A local image patch from the horse-riding image. (b) Sketch graph after sketch pursuit Phase 1. (c) Sketch graph after sketch pursuit Phase 2. (d) Zoom-in view of the upper rectangle in (b). (e) Applying graph operator G_3—connecting two vertices in (d). (f) Applying graph operator G_5—extending two strokes to join each other. (g) Zoom-in view of the lower rectangle in (b). (h) Applying graph operator G_4—extending one stroke to touch another stroke. (i) Applying graph operator G_4 to another stroke. (j) Combining (h) and (i). (k) Applying graph operator G_4—extending one stroke, and applying G_1'—removing one stroke.

The procedure is continued as follows. From each end of the accepted coding function B_1, the algorithm searches the connected and aligned points in the edge–ridge map to form a line segment. It then fits the coding function B_2, and computes $\Delta L(B_2)$ to decide whether to accept it or not, by comparing it to ϵ. By repeating the above procedure, a chain of connected coding functions is obtained to form a curve.

If no more coding functions can be added to the ends of this curve, the algorithm then chooses the edge–ridge point with the maximum strength among all the edge–ridge points that are not covered by existing coding functions, and start a new curve. This process is repeated until no more curve can be started.

Fig. 11 shows the results of sketch pursuit Phase 1 on the horse-riding image. The sketch graphs are obtained after $1, 10, 20, 50, 100$, and 180 steps.

4.4. Sketch pursuit Phase 2: fix corners and junctions by graph operators

After getting the initial sketch graph in Phase 1, a collection of graph operators is used to edit the sketch graph to fix the corners and junctions. This phase is accomplished by a greedy algorithm that seeks to find a local mode of the objective function

$$L(S_{\mathrm{str}}) = \sum_{i=1}^{n} \Delta L(B_i) - \sum_{d=0}^{4} \lambda_d |V_d|, \qquad (15)$$

where $|V_d|$ is the number of nodes with d arms in the sketch graph S_{str}. In the current implementation, $\lambda_0 = 1$, $\lambda_1 = 5$, $\lambda_2 = 2$, $\lambda_3 = 3$, $\lambda_4 = 4$.

The reversible graph operators that are used to edit the sketch graph are summarized and explained in Fig. 12.
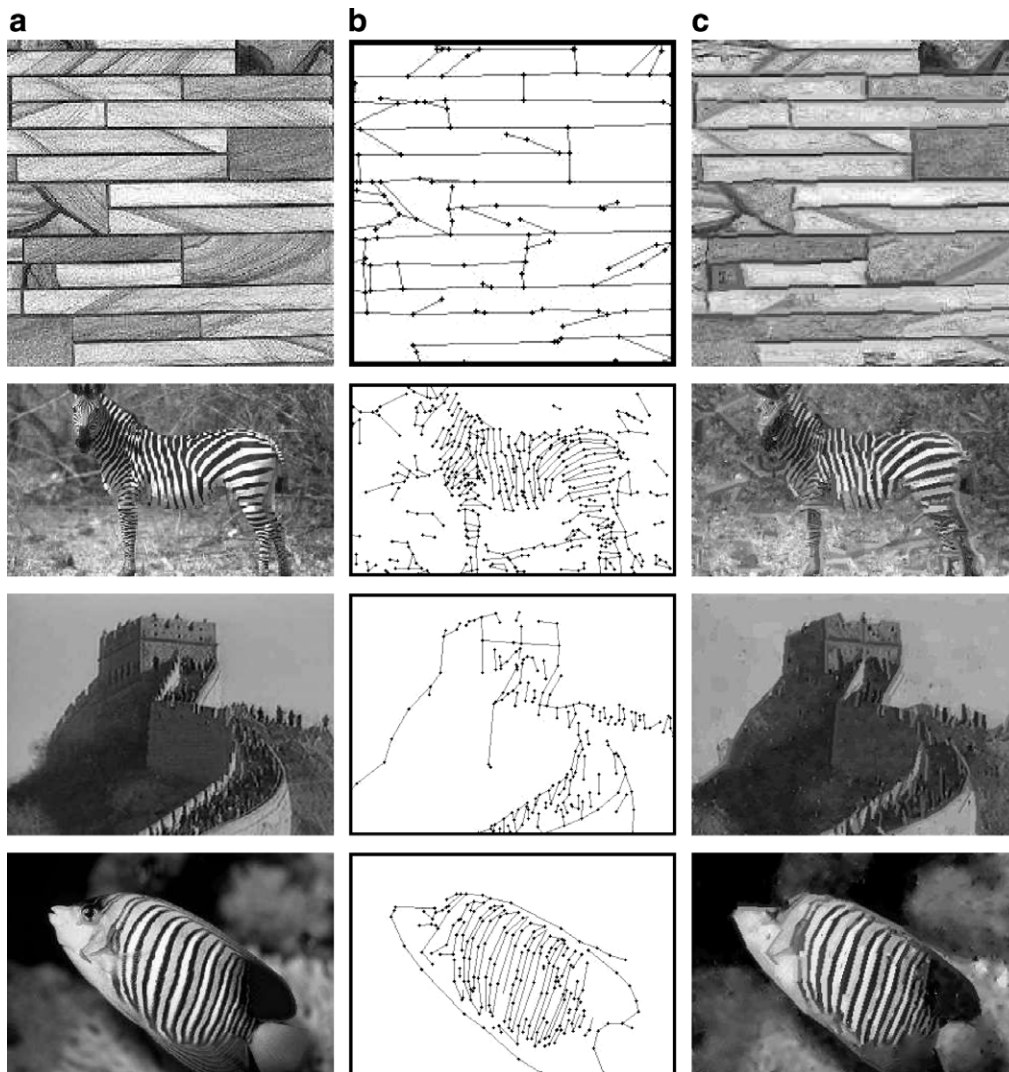

Fig. 15. More results of the primal sketch model. (a) Observed image. (b) Sketch graph. (c) Synthesized image from the fitted primal sketch model.

Fig. 13 shows an example of the sketch pursuit process. (a) is an input image. (b) is the sketch graph after Phase 1. (c) is the sketch graph after Phase 2. Comparing the sketch graphs in (b) and (c), one can see that some of the corners and junctions are missed in Phase 1, but they are recovered in Phase 2.

An example of applying graph operators is show in Fig. 14. The 10 pairs of reversible graph operators change the topological property of the sketch graph.

The computational strategy in this phase is as follows. As illustrated in Fig. 14, from a current sketch graph, the algorithm randomly chooses a local subgraph. All of the 10 pairs of graph operators are attempted for editing that local subgraph. All the new subgraphs that can be produced after 3–5 steps of graph operations (usually around 5–20 new subgraphs) are examined. For each new subgraph, the increase in $L(S_{str})$ in (15) is computed if the current subgraph is changed to this new subgraph. Currently a greedy method is adopted, where the new subgraph that gives the maximum increase in $L(S_{str})$ is chosen. This process is repeated until no modification can be accepted.

4.5. Texture clustering and synthesis

After Phase 2 of sketch pursuit is finished, the lattice Λ_{tex} for textures is available. The k-mean clustering method is employed to divide Λ_{tex} into homogeneous texture regions. First the histograms of the filters within a local window (e.g. 7×7 pixels) are computed. For example, if 7 filters are used and there are 7 bins for each histogram,

then there is a 49-dimensional feature vector for each pixel. These feature vectors are then clustered into different regions. In the current implementation, a small number (5–7) of small derivative filters are used for characterizing textures.

After segmenting Λ_{tex} into texture regions, the textures can be re-synthesized in these regions by matching the corresponding histograms of filter responses. Texture synthesis is optional and is mainly for the purpose of model checking, to see if the synthesized image is visually similar to the observed image.

5. Experiments

5.1. Primal sketch representation of real images

Figs. 15–17 show more experiments. It appears that the primal sketch representation captures both the structural and textural aspects of the image.

There are a number of limitations in our model. First, the relationship between the sketch graph and the segmentation of the texture part is not modeled. Second, the model does not include structures caused by lighting and shading. Third, the model only assumes piecewise homogeneous textures. It cannot account for textures on slanted surfaces.

5.2. Lossy image coding

The primal sketch model provides a lossy image coding scheme, in the sense that the texture regions are coded by
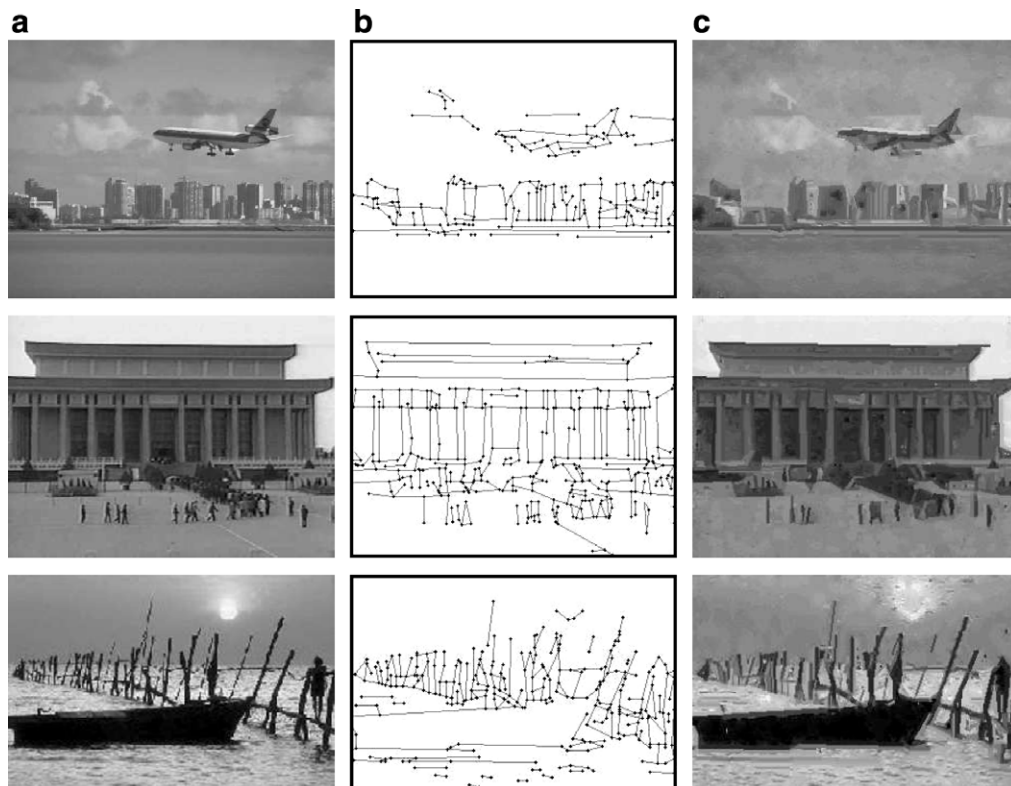

Fig. 16. More results of the primal sketch model. (a) Observed image. (b) Sketch graph. (c) Synthesized image from the primal sketch model.
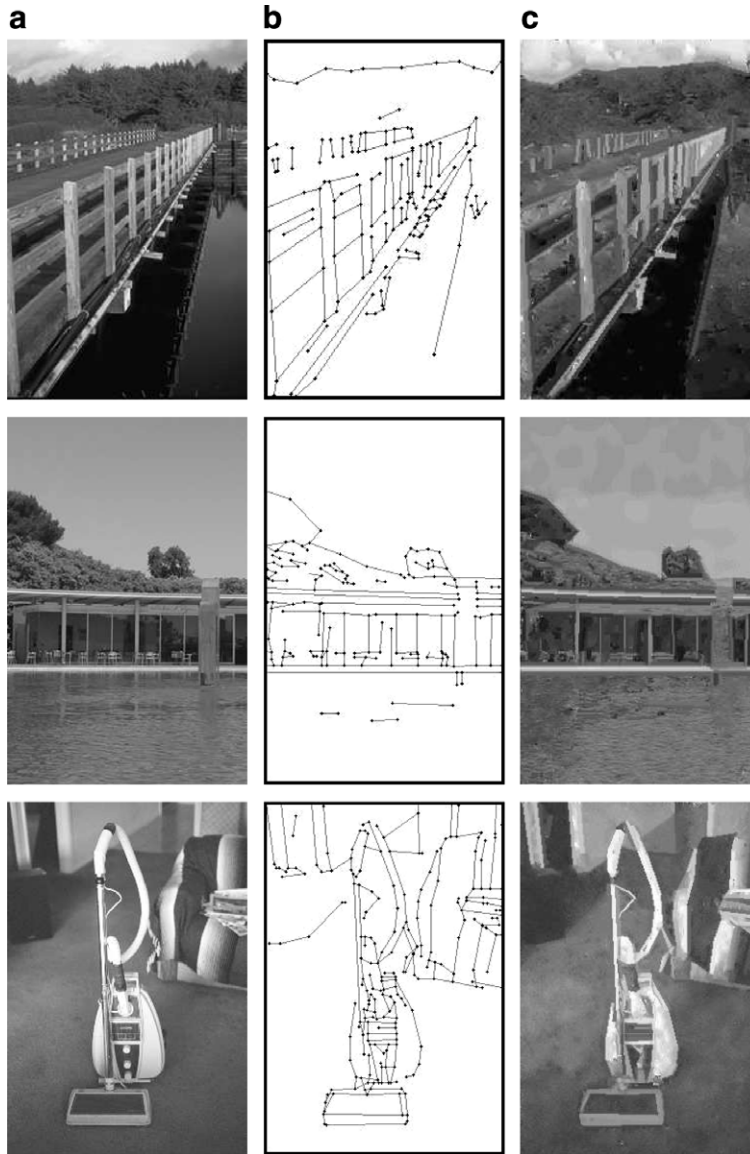
Fig. 17. More results of the primal sketch model. (a) Observed image. (b) Sketch graph. (c) Synthesized image from the primal sketch model.

histograms of filter responses, and the exact pixel values in texture regions are not coded. Table 1 counts the number of parameters for describing the primal sketch model in the experiment shown in Fig. 1.

The observed image has 300×240 pixels, of which 18,185 pixels (around 25%) are considered by the model as belonging to structure domain. The sketch graph has 152 vertices and 275 coding functions that are coded by 1421 bytes. The texture pixels are summarized (instead of being exactly coded) by 455 parameters, with 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, the total coding length for the textures is 1628 bytes. The total coding length for the synthesized

Table 1
The coding length of the primal sketch model in the experiment shown in Fig. 1

	Coding description	Coding length (bits)
Sketch graph	Vertices 152	$152 \times 2 \times 9 = 2736$
	Strokes (275)	$275 \times 2 \times 4.7 = 2585$
Sketch image	Profiles (275)	$275 \times (2.4 \times 8 + 1.4 \times 2) = 6050$
Total for structure domain	18,185 pixels	11,371
Region boundaries	3659 pixels	$3659 \times 3 = 10,977$
Texture regions and histograms		$7 \times 5 \times 13 \times 4.5 = 2048$
Total for texture domain	41,815 pixels	13,025
Total for the whole image	72,000 pixels	3049 bytes, 0.04 byte/pixel

image in Fig. 1f is 3049 bytes or 0.04 byte per pixel. For lossless (error = 0) JPEG 2000 coding, the compression rate is 0.685 byte per pixel. Of course, we are not making a direct comparison here, since our method is lossy coding. For this type of lossy coding, it is hard to quantify the perception error. We shall leave this issue to future study.

6. Discussion

The modeling of textures and structures has long been studied in vision. Julesz [11] first proposed a texture theory 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 [12] and identified bars, edges, corners, terminators as textons. Marr [15] summarized Julesz's theories and other experimental results and proposed primal sketch as a symbolic representation of the raw image. In our model, the sparse coding model can be viewed as a mathematical formulation of Julesz's textons or Marr's tokens. The Markov random field model can be viewed as a mathematical formulation of Julesz's first conjecture on feature statistics. Marr argued that the primal sketch representation should be parsimonious and sufficient to reconstruct the original image without much perceivable distortion. This is reflected in our model by the fact that it is generative, and it can be checked by synthesizing images from fitted models to see if the synthesized images are visually similar to the original image.

The filters used in this article are biologically motivated by neurological observations of V1. Our model can be considered as a scheme for processing information after linear filtering, where the strong and highly aligned responses are combined into coding functions with explicit geometric and photometric parameters, and the weak responses that are not aligned are grouped into marginal histograms as texture statistics. The edge and ridge coding functions may also account for the joint distributions of filter responses studied by Portilla and Simoncelli [21].

In comparison with Canny edge detection [3], our representation is generative, and is more complete, in the sense that not only a sketch graph is computed for the structure domain, but texture patterns are also characterized on the texture domain.

References

[1] G. Aubert, P. Kornprobst, Mathematical Problems in Image Processing, Springer, Berlin, 2002.

[2] J. Besag, Spatial interaction and the statistical analysis of lattice systems (with discussion), J. Royal Stat. Soc. series B 36 (1974) 192–236.

[3] J. Canny, A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. 8 (1986) 679–698.

[4] T. Chan, J. Shen, Local inpainting model and TV inpainting, SIAM J. Appl. Math. 62 (3) (2001) 1019–1043.

[5] D. Chandler, Introduction to Modern Statistical Mechanics, Oxford University Press, Oxford, 1987.

[6] J. Daugman, Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters, J. Opt. Soc. Am. 2 (1985) 1160–1169.

[7] J.H. Elder, S.W. Zucker, Local scale control for edge detection and blur estimation, IEEE Trans. PAMI 20 (7) (1998) 699–716.

[8] S. Geman, C. Graffigne, Markov random field image models and their applications to computer vision, Proc. Int. Cong. Math. 1 (2) (1987) 1496–1517.

[9] A. Gersho, R.M. Gray, Vector Quantization and Signal Processing, Kluwer, Boston, MA, 1992.

[10] D.J. Heeger, J.R. Bergen, Pyramid based texture analysis/synthesis, Comp. Graphics Proc. (1995) 229–238.

[11] B. Julesz, Visual pattern discrimination, IRE Trans. Inf. Theory IT-8 (1962) 84–92.

[12] B. Julesz, Textons, the elements of texture perception and their interactions, Nature 290 (1981) 91–97.

[13] S. Mallat, A Wavelet Tour of Signal Processing, Academic Press, NY, 1998.

[14] S. Mallat, Z. Zhang, Matching pursuit in a time-frequency dictionary, IEEE Sig. Proc. 41 (1993) 3397–3415.

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

[16] L. Moisan, A. Desolneux, J.-M. Morel, Meaningful alignments, Int. J. Comp.Vision 40 (1) (2000) 7–23.

[17] D. Mumford, J. Shah, Optimal approximation by piecewise smooth functions and association. Variational prob., Comm. Pure Appl. Math. 42 (5) (1989) 577–685.

[18] B.A. Olshausen, D.J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, Nature 381 (1996) 607–609.

[19] A. Pece, The problem of sparse image coding, J. Math. Imaging Vis. 17 (2) (2002) 89–108.

[20] A. Pece, Contour tracking based on marginalized likelihood ratios, Image Vision Comput. (2005).

[21] J. Portilla, E.P. Simoncelli, A parametric texture model based on joint statistics of complex wavelet coefficients, Int. J. Comput. Vis. 40 (1) (2000) 49–71.

[22] Z. Tu, X. Chen, A. Yuille, S.-C. Zhu, Image parsing: unifying segmentation, detection, and object recognition, Int. J. Comput. Vis. (2005).

[23] G. Winkler, Image Analysis, Random Fields, and Dynamic Monte Carlo Methods, Springer, Berlin, 1995.

[24] Y.N. Wu, S.C. Zhu, X.W. Liu, Equivalence of Julesz Ensemble and FRAME models, Int. J. Comput.Vis. 38 (3) (2000) 245–261.

[25] R.A. Young, The Gaussian derivative model for spatial vision: I. Retinal mechanism, Spatial Vision 2 (4) (1987) 273–293.

[26] S.C. Zhu, Y.N. Wu, D. Mumford, Minimax entropy principle and its applications in texture modeling, Neural Comput. 9 (8) (1997) 1627–1660.