Subspace learning for Mumford–Shah-model-based texture segmentation through texture patches

Yan Nei Law,¹,* Hwee Kuan Lee,¹ and Andy M. Yip²

¹Bioinformatics Institute, Agency for Science, Technology and Research (A*STAR), 30 Biopolis Street, #07-01 Matrix, Singapore 138671, Singapore
²Department of Mathematics, National University of Singapore, Block S17, 10 Lower Kent Ridge Road, Singapore 119076, Singapore

*Corresponding author: lawyn@bii.a-star.edu.sg

Received 9 December 2010; revised 25 April 2011; accepted 30 April 2011; posted 4 May 2011 (Doc. ID 139392); published 13 July 2011

In this paper, we develop a robust and effective algorithm for texture segmentation and feature selection. The approach is to incorporate a patch-based subspace learning technique into the subspace Mumford–Shah (SMS) model to make the minimization of the SMS model robust and accurate. The proposed method is fully unsupervised in that it removes the need to specify training data, which is required by existing methods for the same model. We further propose a novel (to our knowledge) pairwise dissimilarity measure for pixels. Its novelty lies in the use of the relevance scores of the features of each pixel to improve its discriminating power. Some superior results are obtained compared to existing unsupervised algorithms, which do not use a subspace approach. This confirms the usefulness of the subspace approach and the proposed unsupervised algorithm. © 2011 Optical Society of America

OCIS codes: 100.0100, 100.2000, 100.5010, 100.4994.

1. Introduction

Image segmentation is indispensable in applications that employ imaging systems. It facilitates the extraction of useful information for subsequent high-level image analysis. Texture, usually considered the variation of data at scales smaller than the scales of interest [1], can be a useful cue for object recognition and segmentation. However, it can also be a nuisance because it may create many extra lines in the edge map. Moreover, some complications arise when dealing with features extracted from textured images. First, textured images and their transforms often have no sharp boundary. Second, the algorithm has to be robust to noise while still being able to distinguish noise from small scale textures. Last, as different textures are better captured by different features, one has to face the feature selection problem. The presence of nondiscriminative or redundant features often deteriorates segmentation results.

Recently, an optimization-based texture segmentation model called the subspace Mumford–Shah (SMS) model [2] has been developed. The model is a combination of the Mumford–Shah (MS) image segmentation model [3] and the k-means subspace clustering model [4]. It inherits the essential properties of the two models: it enables clusterwise feature selection, and it regularizes the geometry of the segments in the spatial domain to avoid overfitting. Unlike most standard segmentation algorithms, which assume the provision of a well-chosen set of features, the main advantage of the SMS model is that features are automatically selected so that the model is more flexible and works well with very generic features.

The main idea of SMS is to let the weights of the features be part of the objective function so that image segmentation and feature selection are performed simultaneously. Given the features \{f_j(x)\}_{j=1,...,m} at each pixel x, the basic model seeks to minimize the following objective:

\[ F_{\text{SMS}}(\{\Omega_i\}, \{c_{ij}\}, \{\lambda_{ij}\}) = \mu \cdot \text{Length}(C) + \sum_{i=1}^{n} \sum_{x \in \Omega_i} \sum_{j=1}^{m} \lambda_{ij} |f_j(x) - c_{ij}|^2 + \gamma \sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_{ij} \log \lambda_{ij}, \quad (1) \]
subject to
\[ \sum_{j=1}^{m} \lambda_{ij} = 1, \text{ for } i = 1, 2, \ldots, n, \]  
(2)

\[ 0 \leq \lambda_{ij} \leq 1, \text{ for } i = 1, 2, \ldots, n; \quad j = 1, 2, \ldots, m. \]  
(3)

Here \( \Omega_i \) for \( i = 1, \ldots, n \) are \( n \) mutually exclusive segments that form a partition of the image domain \( \Omega \), and \( C \) is the set of boundary curves of \( \{ \Omega_i \} \). The idea is to partition the image so that each feature \( f_j \) in each segment \( \Omega_i \) is well approximated by a constant \( c_{ij} \). The curves \( C \) are required to be as short as possible to increase the robustness to noise and to avoid spurious segments. The goodness of fit is measured by the fitting term in the second term in Eq. (1). The parameter \( \mu > 0 \) controls the trade-off between the goodness of fit and the total length of \( C \). It can be easily shown that for each fixed \( C \) and \( \{ \lambda_{ij} \} \), the optimal constant \( c_{ij} \) is given by the average of \( f_j \) over \( \Omega_i \).

A key feature of the model is that each segment can use a different subspace in computing the within-segment dispersion. The scalar \( \lambda_{ij} \) is the weight of the \( j \)th feature in the \( i \)th segment. If \( \lambda_{ij} \) is close to zero, then the determination of the \( i \)th segment is insensitive to the \( j \)th feature. Therefore, it opens up the possibility of using an optimal subset of features for each segment. The parameter \( \gamma > 0 \) controls the entropy of the feature weights. Setting \( \gamma = 0 \) results in a trivial solution, where the feature with the minimal sum of squares in each segment is given a full weight of 1. In contrast, setting \( \gamma = \infty \) reduces to the original vector-valued MS model, where all features are equally weighted.

Unfortunately, this objective function is highly nonconvex, which is difficult to optimize. There can exist many spurious local minima in which a straightforward algorithm will be trapped. For instance, a direct application of the alternating minimization approach often leads to disappointing performance. The results often depend very much on the initial guess, making the model difficult to use in practice. To resolve this problem, some interactive supervised approaches have been proposed [2,5]. These methods require the user to specify some image patches as training data from which the weights \( \{ \lambda_{ij} \} \) can be learned robustly. These algorithms are very robust and give impressive results. However, the main drawback is that it assumes the availability of some well-chosen training data.

A. Objective

Our aim is to develop a fully automatic, robust, and effective algorithm for texture segmentation and feature selection. The approach is to incorporate a patch-based subspace learning technique into the SMS model, to make the minimization of the SMS model robust and accurate. Our first contribution is a fully automatic algorithm, called the auto-subspace MS algorithm (auto-SMS). Unlike the previous methods in [2,5] that require the specification of some training data to facilitate an accurate estimation of the weights, the proposed model can achieve the same goal effectively while still being fully automatic.

Our second contribution is a novel dissimilarity measure defined for any two pixels. Its novelty lies in the use of the relevance scores of the features of each pixel to improve its discriminating power. We refer to this measure as the entropy regularization dissimilarity measure. In the auto-SMS algorithm, this measure is used to cluster some suitably selected texture patches to obtain representatives of each segment.

2. Algorithm

The major steps of the auto-SMS algorithm are shown in Algorithm 1. The details are described in the following sections.

Algorithm 1: Auto Subspace Mumford–Shah

1: \textbf{Input:} image \( u \)
2: Extract features \( \{ f_j \}_{j=1}^{m} \) from \( u \)
3: Select \( k \) random pixels \( S = \{ x_1, \ldots, x_k \} \)
4: For each \( x \in S \), determine the size \( r \) of the local patch \( R_x \) around \( x \)
5: Cluster the patches \( \{ R_x \}_{x \in S} \) into \( n \) groups \( \{ S_i \}_{i=1}^{n} \)
6: Optimize the objective \( P_{\text{Ent}} \) using Algorithm 2
7: \textbf{Output:} segmentation \( \{ \Omega_i \} \)

A. Gabor Feature Extraction

Gabor filters have been widely used in extracting features from textured images [3,6–10], for they are designed to capture local patterns in different scales and orientations. Starting with the two-dimensional Gabor function [7]

\[ G(x, y) = \frac{1}{2\pi \sigma_x \sigma_y} \exp \left[ -\frac{1}{2} \left( \frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2} \right) + 2\pi W x \right], \]

where \( W \) is the modulation frequency and \( i = \sqrt{-1} \), a self-similar filter dictionary can be obtained by dilations and rotations:

\[ G_{pq}(x, y) = a^{-p} G(x', y'), \]

\[ x' = a^{-p} (x \cos \theta + y \sin \theta), \]

\[ y' = a^{-p} (x \sin \theta + y \cos \theta), \]

where \( a > 1 \), \( p \in \mathbb{Z} \), \( q \in \{ 1, 2, \ldots, K \} \), \( \theta = q \pi / K \), and \( K \) is the total number of orientations. By varying the scale \( p \) and the orientation \( q \) while keeping other parameters \( (\sigma_x, \sigma_y, W, a, K) \) fixed, we obtain a set of Gabor filters \( \{ G_{pq} \} \). Then, the Gabor features of an image \( u \) at each location \( (x, y) \) are obtained as the modulus of the Gabor transforms.
\[ f_j(x, y) := \left| \int G_{\rho, \phi}(x - \bar{x}, y - \bar{y}) dx dy \right|. \tag{4} \]

Here, \( G \) denotes the complex conjugate of \( G \).

Gabor transforms contain much redundant information because the supports of their filter responses in the frequency domain are highly overlapped. To reduce the redundancy, Manjunath and Ma \[7\] designed a scheme to select an appropriate subset of filters, such that the half-peak magnitude support of the filter responses in the frequency spectrum touch each other. We apply this scheme in our experiments to determine the parameters \( (\sigma_x, \sigma_y, W, a, K) \).

B. Identifying Patches of Optimal Size

Our approach to the minimization of the SMS model is to utilize some patches that represent each segment to initialize the weights \( \{\lambda_j\} \). Along this way, we develop a texture detection method to find some representable patches of each region. The method has two steps: (1) identify patch candidates, and (2) cluster the selected patches. In this section and the next, we describe methods for the two steps respectively.

First, a set of \( k \) (say 20) pixels \( S = \{x_1, \ldots, x_k\} \) are randomly selected. Then, a square patch

\[ R_{x,r} := \{z \in \Omega | \|x - z\|_{\infty} \leq r\} \tag{5} \]

of half-size \( r \) around each selected pixel \( x \in S \) is sought. Here, \( \|x\|_{\infty} = \max \{|x_i|\} \) is the maximum norm. An ideal choice of \( r \) would be the smallest one \( r^* \) such that the texture is still captured in \( R_{x,r^*} \). A promising scale descriptor based on this principle has been proposed in \[11\] for determining the optimal patch size \( r^* \). The idea of the model is to find the smallest patch such that the difference between the intensity distributions in the patch and in a neighborhood around the patch is minimized.

The image \( u \) is first converted to grayscale and normalized to the range of \([0,1]\). For each local patch \( R_{x,r} \) as defined in Eq. (5), the neighboring patch around it is defined as the set of pixels within a patch of half-size \( 3r \) but not in \( R_{x,r} \):

\[ N_{x,r} := R_{x,3r} \setminus R_{x,r} = \{z \in \Omega | r < \|x - z\|_{\infty} \leq 3r\}. \]

The probability density function, \( P_X = P_X(y) \), on a set of pixels \( X \) is the proportion of pixels in \( X \) having an intensity value \( y \), i.e.,

\[ P_X(y) := \frac{1}{|X|} \sum_{u \in X} \delta_{y,u(x)}, \]

where \( \delta_{y,u(x)} = 1 \) if \( u(x) = y \) and 0 otherwise and \(|X|\) is the number of pixels in \( X \).

The cumulative distribution function, \( F_X = F_X(y) \), on a set of pixels \( X \) is the proportion of pixels having an intensity value less than or equal to \( y \), i.e.,

\[ F_X(y) := \int_0^y P_X(x) dx. \]

The Wasserstein distance between two probability density functions \( P_1 \) and \( P_2 \) is

\[ D_W(P_1, P_2) := \int |F_1(y) - F_2(y)| dy, \]

where \( F_1 \) and \( F_2 \) are the corresponding cumulative distribution functions. The negative entropy \( H \) of a probability density function \( P \) is

\[ H(P) := \int P(y) \log P(y) dy. \]

Finally, the scale descriptor \( r^* \) is defined as the minimizer of the following energy functional:

\[ r^* := \arg\min_r \left\{ D_W(P_{R_{x,r}}, P_{N_{x,r}}) + \nu H(P_{R_{x,r}}) + \eta r \right\}, \tag{6} \]

where \( \nu \) and \( \eta \) are positive design parameters.

The first term is used to realize our preference that the difference between the intensity distributions in \( R_{x,r} \) and in \( N_{x,r} \) be as small as possible. The second term is used to avoid selecting homogeneous patches. The last term is used to make the patch as small as possible. To handle pixels near the texture boundary, \( N_{x,r} \) is divided into eight nonoverlapping squares of half-size \( r \). Then, the one with minimal Wasserstein distance to \( R_{x,r} \) is used in place of \( N_{x,r} \). Once the optimal half-size \( r^* \) (for each \( x \in S \)) is found, we can form the optimal patch \( R_{x,r} \). To simplify notation, we suppress the subscript \( r^* \) and simply write \( R_{x,r} \) as \( R_x \).

C. Clustering Texture Patches

The next step of the auto-SMS algorithm is to cluster the selected patches into groups. Each group is then considered as some sample pixels of each segment. In this section, we propose a new clustering objective function, from which a new implied distance measure is obtained. In the experiments, we illustrate the application of the derived distance measure to texture retrieval and compare the proposed distance to existing distances in terms of the performance in discriminating textures. The detected texture patches can also be used as inputs to segmentation algorithms that require specification of training sets \[12\].

1. Objective Function

The objective function that we propose is a variant of the subspace \( k \)-means model in \[4\], which is a generic clustering model. We found that when applying this model to image data, the estimation of the feature weights can be made simpler based on the spatial correlations existing in most natural images. The original subspace \( k \)-means objective requires updating the weights and the clustering alternatively until
convergent. In our new scheme, the weights can be computed with good quality in one pass. This makes the final results less sensitive to the initial guess of the weights and clustering.

We shall derive an objective function for clustering patches as an approximation to that for clustering the entire image. Consider the clustering of all pixels in $\Omega$ via the subspace $k$-means clustering model:

$$F(\{\Omega_i\}, \{c_{ij}\}, \{\lambda_{ij}\}) := \sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_{ij} |f_j(x) - c_{ij}|^2$$

$$+ \gamma H(\{\lambda_{ij}\}), \quad (7)$$

subject to the constraints [Eqs. (2) and (3)]. Here,

$$H(\{\lambda_{ij}\}) := \sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_{ij} \log \lambda_{ij}$$

is the total of the negative entropies of the weight distributions $\{\lambda_{ij}, \lambda_{2j}, \ldots, \lambda_{m_j}\}$ for each $j$. For each fixed segmentation $\Omega_i$, the corresponding optimal fitting constants $\{c_{ij}\}$ and the optimal weights $\{\lambda_{ij}\}$ are given by

$$c_{ij} = \hat{f}_j(\Omega_i) := \frac{1}{|\Omega_i|} \sum_{x \in \Omega_i} f_j(x), \quad (8)$$

$$\lambda_{ij} = \frac{\exp(-D_{ij})}{\sum_{k=1}^{m} \exp(-D_{ik})}, \quad (9)$$

where

$$D_{ij} := \sum_{x \in \Omega_i} |f_j(x) - \hat{f}_j(\Omega_i)|^2.$$

Because of the dependence of the optimal weights (9) on the clustering, they have to be updated iteratively when no additional prior knowledge is available. This can sometimes lead to severe local convergence problems. However, in the case of image data, it is reasonable to approximate the intensity variance within a segment by that within a patch. This is because the patches are selected such that the intensity distributions in the local patch and in the neighboring patch have a minimal difference. Therefore, we postulate the approximation

$$\frac{1}{|\Omega_i|} \sum_{x \in \Omega_i} |f_j(x) - \hat{f}_j(\Omega_i)|^2 \approx \frac{1}{|R_x|} \sum_{x \in R_x} |f_j(x) - \hat{f}_j(R_x)|^2$$

for each $x \in S \cap \Omega_i$, where $\hat{f}_j(R_x)$ is the average of $f_j(x)$ over $R_x$. This naturally suggests the following approximation of the optimal weights:

$$\lambda_{ij} \approx \hat{\lambda}_{x_j} := \frac{\exp(-D_{x_j})}{\sum_{k=1}^{m} \exp(-D_{x_k})}, \quad (10)$$

where

$$D_{x_j} := \sum_{x \in R_x} |f_j(x) - \hat{f}_j(R_x)|^2.$$

The factor $|\Omega_i|/|R_x|$ is absorbed into the parameter $\gamma$, and the same $\gamma$ is used for all patches for simplicity. Using Eq. (10), the weights are computed solely based on the detected patches.

We note that $\lambda_{ij}$ is defined for each cluster and each feature. But $\hat{\lambda}_{x_j}$ is defined for each pixel and each feature. Therefore, we have extended the notion of clusterwise subspace in the literature to pixelwise subspace. This leads to the following definition:

**Definition 1** The pixelwise entropy regularization (ER) feature relevance score for the pixel $x$ in the $j$ th dimension is defined by $\hat{\lambda}_{x_j}$ in Eq. (10).

Once the weights are determined, we apply them to construct a new clustering objective function for clustering the patches $\{R_x\}_{x \in S}$ (identified by their centers). We denote by $\{S_i\}$ a clustering of $S$. First, we notice that by substituting the optimal constants $c_{ij} = \hat{f}_j(S_i)$ into the objective [Eq. (7)] and replacing $\Omega_i$ with $S_i$, the objective can be expressed based on pairwise distances within the clusters:

$$F(\{S_i\}, \{\lambda_{ij}\}) = \sum_{i=1}^{n} \sum_{x \in S_i} \sum_{m} \lambda_{ij} |f_j(x) - f_j(x')|^2$$

$$+ \gamma H(\{\lambda_{ij}\}). \quad (11)$$

Next, we fix the weights to $\hat{\lambda}_{x_j}$ and ignore the entropy term, which becomes a constant factor, to arrive at the proposed objective function:

$$F(\{S_i\}) = \sum_{i=1}^{n} \sum_{x \in S_i} \sum_{x' \in S_i} \sum_{j=1}^{m} \lambda_{x_j} |f_j(x) - f_j(x')|^2. \quad (12)$$

The objective [Eq. (12)] depends directly on the features $f_j(x)$ at the patch centers $x \in S$. The features at other locations within the patches are not directly expressed in the objective, but they affect the value of $\lambda_{x_j}$. Moreover, if $f_j$ varies a lot within a patch $R_x$, then according to (10), the weight $\hat{\lambda}_{x_j}$ will be small. Thus, the objective selects features that are uniformly in the patch, so that comparing two patches solely based on their centers is feasible.

The term

$$d_{x}(x') := \sum_{j=1}^{m} \lambda_{x_j} |f_j(x) - f_j(x')|^2$$

in Eq. (12) can be viewed as a (nonsymmetric) distance from $x$ to $x'$, calculated using the subspace of $x$. By rearranging the sums, the objective [Eq. (12)] can be written as

$$F(\{S_i\}) = \sum_{i=1}^{n} \frac{1}{|S_i|} \sum_{x \in S_i} \sum_{x' \in S_i} d(x, x'). \quad (13)$$

where

$$\hat{D}_x := \sum_{x \in R_x} |f_j(x) - \hat{f}_j(R_x)|^2.$$
Thus, the objective [Eq. (12)] gives rise to an implied distance measure, which is the average of $d_s(x')$ and $d_v(x)$.

**Definition 2** The ER distance measure between two pixels is defined by $d(x, x')$ in Eq. (14).

The ER distance is symmetric but does not satisfy the triangle inequality. The lack of the triangular inequality is actually desirable because it can happen that pixel $A$ is similar to pixel $B$ in one subspace and pixel $B$ is similar to pixel $C$ in another subspace, but pixels $A$ and $C$ are not similar in any subspace.

2. **Hierarchical Clustering**

The objective function [Eq. (13)] can be naturally optimized using an agglomerative hierarchical clustering method. Initially, each pixel is considered as a cluster. At each step, two clusters are selected to merge, until a predefined number of clusters ($m$) remain. To determine the pair of clusters to join, we use Ward’s criterion [13], which selects the pair $(S_i, S_j)$ that gives the minimal increase of the objective. Let $\Delta(S_1, S_2)$ be the increase in Eq. (13) by joining $S_1$ and $S_2$. Then,

$$(S_i, S_j) = \arg\min_{S_1 \cup S_2} \Delta(S_1, S_2).$$

The energy change is given by

$$\Delta(S_1, S_2) = \frac{1}{2(|S_1| + |S_2|)} \sum_{x \in S_1} \sum_{x' \in S_2} d(x, x') - \frac{1}{2|S_1|} \sum_{x \in S_1} \sum_{x' \in S_2} d(x, x') - \frac{1}{2|S_2|} \sum_{x \in S_1} \sum_{x' \in S_2} d(x, x').$$

D. **Optimization of the Subspace MS Model**

The fundamental approach we used for segmentation is to optimize the subspace MS objective $F_{SMS}$ in Eq. (1). However, as mentioned above, a direct optimization often gets stuck in poor local minima. In this section, we describe an algorithm that makes use of the clustering of the patches $\{S_j\}$ to guide the estimation of the optimal weights to obtain a good solution of Eq. (1). The method is fully unsupervised. Some supervised versions have been proposed in [2,5].

Our basic idea is that the weights estimated from the clusters of selected patches should be more reliable than those from some initial segments, which can contain pixels from various true segments. Let $R_i = \bigcup_{x \in S_i} R_x$ be the union of the patches that are grouped together in the $i$th group. First, we define a patch-based objective function from which the weights can be estimated reliably:

$$F_{Patch}((\lambda_{ij})) := \frac{1}{|\Omega|} \sum_{j=1}^{n} \sum_{x \in R_j} \lambda_{ij} \left( f_j(x) - \bar{f}_j(R_i) \right)^2 + \gamma H((\lambda_{ij})),$$

where $\bar{f}_j(R_i)$ is the average of $f_j$ over $R_i$. The constant $|\Omega|/|\bigcup_{x \in S} R_x|$ is introduced to normalize the difference between the size of the patches and $\Omega$.

Next, we consider the convex combination of the objective $F_{SMS}$ and $F_{Patch}$ with a mixing parameter $\beta \in [0, 1]$: $F_{Auto}(\{\Omega_i\}, \{c_j\}, \{\lambda_{ij}\}) := \beta F_{Patch}(\{\lambda_{ij}\}) + (1 - \beta) F_{SMS}(\{\Omega_i\}, \{c_j\}, \{\lambda_{ij}\})$.

The parameter $\beta$ controls the degree of involvement of the detected patches. When optimizing the objective function, the parameter $\beta$ is gradually decreased from 1 toward 0 so that the original objective $F_{SMS}$ is eventually optimized. This can make the final weights depend on the image globally and insensitive to any bias introduced by the patches.

One may think of our approach as using the detected patches to specify some initial values of the weights for the optimization of $F_{SMS}$. Once the weights are in place, the problem essentially reduces to the original MS problem.

To optimize the objective $F_{SMS}$, we use the alternating minimization approach in Algorithm 2. The parameter $\beta$ is reduced every other $K$ steps. In the first pass, $F_{Auto} = F_{Patch}$ so that the weights are computed from the detected patches. Details of the individual steps are provided in Appendix A.

**Algorithm 2: Alternating Minimization of $F_{Auto}$**

1: Input: features $\{f_j\}$, a clustering of patches $\{S_j\}$
2: Initialize $\beta^0 = 1$
3: Initialize $(\Omega^0_j)$ using e.g., k-means
4: Initialize $k = 0$
5: repeat
6: $\{c_j^{k-1}\} = \arg\min_{c_j} F_{Auto}(\{\Omega^k_j\}, \{c_j\}, \{c_j^{k-1}\})$
7: $\{\lambda_{ij}^{k-1}\} = \arg\min_{\lambda_{ij}} F_{Auto}(\{\Omega^k_j\}, \{c_j^{k-1}\}, \{\lambda_{ij}^{k-1}\})$
8: $\{\Omega^{k+1}_j\} = \arg\min_{\Omega_j} F_{Auto}(\{\Omega^{k+1}_j\}, \{c_j^{k+1}\}, \{\lambda_{ij}^{k+1}\})$

(by Algorithm 3 in Appendix A)
9: if $k$ is a multiple of $K$ then $\beta^{k+1} = 0.9 \beta^k$
else $\beta^{k+1} = \beta^k$ endif
10: $k = k + 1$
11: until $(\Omega^k_j)$ stabilizes
12: Output: segmentation $\{\Omega^k_j\}$

3. **Experimental Results**

In this section, we evaluate the effectiveness of the proposed ER distance and the proposed auto-SMS algorithm empirically. All algorithms are implemented in MATLAB 7 with a Pentium D 3 GHz machine.
A. Discriminating Power of Entropy Regularization Distance Measure

The aim of this test is to study the effectiveness of the ER distance in discriminating textures. We do this by comparing the ER distance with the weighted-mean-variance (WMV) distance [7] in retrieving images. WMV is a very popular distance measure based on mean and variance of Gabor features.

Images. The simulated texture database contains 32 images of different texture classes from the Brodatz album [14]. Five hundred twelve query patterns are constructed by dividing each of the 32 images into 16 nonoverlapping images with size 128 × 128.

Features. Each database pattern and query pattern is considered as a texture patch and is represented by a single feature vector \( F \). For each of these patches, 24 Gabor features \( \{ f_j(x) \}_{j=1}^{24} \) of four scales and six orientations are first computed at each pixel \( x \in \Omega \), as in Eq. (4). Then, the averages \( \{ f_j \} \) over the patch are computed. For the ER distance, the feature vector is \( F = \{ f_1, \ldots, f_{24} \} \). For the WMV distance, the standard deviations \( \{ \sigma_j \} \) over the patch are also computed to form a feature vector \( F = \{ f_1, \sigma_1, \ldots, f_{24}, \sigma_{24} \} \).

Method. For each query pattern, the ER and WMV distances to each database pattern are computed. For the ER distance, the weights and distance are computed using Eqs. (10) and (14) with feature vector \( F \) respectively. To compensate the difference in image size, \( \gamma = 1/2 \) and \( \gamma = 1/32 \) are used when computing the weights of the database patterns and query patterns respectively. For the WMV distance, it is defined by

\[
\sum_{j=1}^{24} \frac{f_j - \overline{f_j}}{\text{std}(f_j)} + \frac{\sigma_j - \overline{\sigma_j}}{\text{std}(\sigma_j)},
\]

where the \( \text{std}(\cdot) \) is the standard deviation of the respective feature over the database.

Results. Table 1 shows the frequency of the rank of the ideal retrieval for both distance measures. The proposed distance has a much higher accuracy than the WMV distance, 88.87% versus 73.63%, respectively. Here, the accuracy is computed as the percentage of times the ideal image is ranked first. Even in the cases where the ideal pattern is not ranked first, its rank under the proposed measure is still higher than that under the WMV distance. The average rank under the ER distance is 1.23, and that under the WMV distance is 1.89. This shows that the proposed measure is more effective in differentiating different textures.

In Fig. 1, two examples of incorrect retrievals are shown. For each distance, the five database patterns closest to the query pattern are shown. We observe that the ER distance gives patterns that are visually closer to the query than the WMV distance.

B. Texture Segmentation

The next test compares the auto-SMS segmentation algorithm and several other existing methods.

Images. In the first set of experiments, we focus on the problem of foreground–background (i.e., two-class) segmentation. One synthetic image and three real-life images are used for evaluation. The synthetic image contains a square over the background in the red and green channels. Gaussian noise is added to the inner region, the outer region, and the entire image in the red, green, and blue channels, respectively. In this way, the inner and outer regions are best described by the green and red channels, respectively. The endometrial tissue image is taken from [15]. The stem cell image is obtained within the first and second authors’ institute. The cuttlefish image is obtained from the World Wide Web. In the second set of experiments, we demonstrate the performance of the proposed method on segmenting images with multiple classes. We use a cross section of a human retina in [16]. The image consists of four retinal layers (from left to right): ganglion cell layer (GCL), inner and outer nuclear layer (INL), outer nuclear layer (ONL), and rod outer segment (OS). These real images are difficult to segment, because the textures in different segments do not exhibit significant difference.

Features. A relatively large set of features including RGB channels, Gabor features, Laws’ mask features [17], and Brodatz texture similarity measures [18] are used. The data therefore contain many less informative and less discriminative features. In this way, we can compare the results with and without the use of subspaces. It is expected that as the number of irrelevant features increases, the performance of MS models without subspace selection can become worse, because distances computed in high-dimensional spaces are less reliable. However, the proposed auto-SMS model is much less sensitive to irrelevant features.

Parameter Setting. The parameters for the Gabor filters are set as in [7], i.e., \( (\sigma_z, \sigma_y, W, a, K) = (1.4054, 1.8544, 0.4, 2, 6) \). For the

<table>
<thead>
<tr>
<th>Rank</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
<th>21</th>
<th>22–32</th>
</tr>
</thead>
<tbody>
<tr>
<td>ER</td>
<td>455</td>
<td>30</td>
<td>12</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>WMV</td>
<td>377</td>
<td>65</td>
<td>23</td>
<td>10</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
The ratios of the distances of MS with auto-SMS as the baseline are reported in Fig. 4. A ratio greater than one indicates that auto-SMS gives a segment that is closer to the corresponding segment of the manual segmentation than the MS model. Let $g_{X,i}(i)$ and $G_{X,i}(i)$ be, respectively, the marginal probability density function (pdf) and the associated cumulative distribution function (cdf) of the $j$th feature in the segment $X$. Let $X$ and $Y$ be two segments. The four distances used are given as follows [20,21]:

1. Minkowski: $\sum_i \sum_j |g_{X,i}(i) - g_{Y,j}(i)|$
2. Kolmogorov–Smirnov: $\sum_i \max_j |G_{X,i}(i) - G_{Y,j}(i)|$
3. $\chi^2$ statistic: $\sum_i \sum_j \frac{(g_{X,i}(i) - g_{Y,j}(i))^2}{g_{X,i}(i) + g_{Y,j}(i)}$
4. Wasserstein: $\sum_i \sum_j |G_{X,i}(i) - G_{Y,j}(i)|$

**Results on Two-Class Images.** In each part of Fig. 3, the first row shows the detected patches of each segment (foreground and background) using the proposed texture detection method and the manual segmentation for comparison. The second row shows (from left to right) the segmentation results using the $k$-means, MS, and auto-SMS models. The percentage segmentation error values are reported in Table 2. Four other distance measures between manual segmentation and computed segmentation are given in Fig. 4. Overall, the auto-SMS outperforms all other three models and is able to obtain clear boundaries. The results of $k$-means on all the images are poor. This shows that the feature selection mechanism in the auto-SMS model is crucial to obtain good results.

<table>
<thead>
<tr>
<th>Image</th>
<th>Auto-SMS</th>
<th>MS</th>
<th>MSNLST</th>
<th>$k$-means</th>
</tr>
</thead>
<tbody>
<tr>
<td>RGB</td>
<td>0.4%</td>
<td>22.9%</td>
<td>40.4%</td>
<td>38.1%</td>
</tr>
<tr>
<td>Tissue</td>
<td>9.8%</td>
<td>13.8%</td>
<td>15.8%</td>
<td>14.4%</td>
</tr>
<tr>
<td>Stem cell</td>
<td>6.5%</td>
<td>6.8%</td>
<td>35.9%</td>
<td>16.6%</td>
</tr>
<tr>
<td>Cuttlefish</td>
<td>5.3%</td>
<td>21.7%</td>
<td>14.1%</td>
<td>20.3%</td>
</tr>
</tbody>
</table>

![Table 2. Percentage Error (Percentage Symmetric Difference) with Respect to Manual Segmentation](image)

![Fig. 2. The figure demonstrates how to compare the true segmentation and the computed segmentation based on distribution of feature values. In each segment, the marginal pdf and the associated cdf of the $j$th feature of manual segmentation and computed segmentation are computed. Then, four kinds of distance between the pdf or cdf of the manual and computed segmentations are computed.](image)
Figure 5 shows the percentage segmentation error resulting from the auto-SMS model with various $\gamma$ values between 0.05 and 10. We observe that the accuracy does not change much for all $\gamma < 1$. This shows that the optimal weights are robust to $\gamma$. However, when $\gamma$ is so large ($\gamma > 1$) that the weights are forced to be uniform, the performance deteriorates. For the synthetic image, a sharp increase in error occurs when $\gamma = 10$, because the entire inner square is merged with the background due to the discrete nature of the image. This demonstrates the importance of introducing feature selection. For other images, as $\gamma$ becomes too large, the error increases in a steady manner.
Figure 6 shows the percentage segmentation error with various $\mu$ values. We observe the typical characteristic error plot of regularization models, where the error smoothly varies and a minimum point exists.

Results on Multiple-Class Images. Figure 7 shows the detected patches of each region using the proposed texture detection method. Figure 8 shows the manual segmentation and the segmentation results using $k$-means, MS, and auto-SMS models. The percentage segmentation error values are also reported. We observe that both MS and $k$-means trend to combine the INL and ONL regions. The boundary between the INL and ONL regions in the result of MS was significantly shifted. Only the GCL and OS regions, which have a high contrast to other regions, are extracted properly. In contrast, the segmentation result obtained by the proposed method visually looks closer to the ground truth provided by the image source [16] and has the lowest percentage segmentation error.

4. Conclusion

In this paper, we develop a robust and effective algorithm for texture segmentation and feature selection. We show that with the help of the detected patches, the proposed auto-SMS algorithm can effectively overcome the nonrobustness of the plain SMS while still being fully automatic. In our experiments, we show that even when the images have a very low contrast in textures, the algorithm can give much better results than other, nonsubspace methods. This is due to the ability of the model to filter out irrelevant features, so that the contrast between different segments is increased. Along the way, we also propose a new dissimilarity measure for clustering texture patches and demonstrate its superiority in discriminating textures. While the detected patches are used to aid the optimization of the SMS objective, they can in fact be used to “automate” other methods that require the user’s specification of sample segments, e.g., [22].

Appendix A: Implementation Details of Algorithm 2

In Algorithm 2, there are three minimization problems that need to be solved alternatively. A closed form solution to the minimization with respect to $\{c_{ij}\}$ is given by

$$\lambda_{ij} = \frac{\exp\left(-\frac{D_{ij}}{\tau}\right)}{\sum_{k=1}^{M} \exp\left(-\frac{D_{ik}}{\tau}\right)},$$

where

$$D_{ij} := (1 - \beta) \sum_{x \in \Omega} |f_j(x) - c_{ij}|^2 + \frac{\beta |\Omega|}{|\Omega x \in S R_x|} \cdot \sum_{x \in \Omega j} |f_j(x) - \tilde{f_j}(R_i)|^2.$$

A closed form solution to the minimization with respect to $\{\Omega_i\}$, a closed form solution does not exist. The objective function $P_{SMS}$ is nonconvex and therefore may possess many local minima (even for fixed fitting constants $\{c_{ij}\}$ and fixed weights $\{\lambda_{ij}\}$). Here we present a fast two-phase algorithm based on the results of Chan et al. [23], Chambolle [24], Aujol et al. [25], and Bresson et al. [26]. An advantage of the method is that a global minimum for fixed fitting constants and weights is obtained. Moreover, using a dual formulation, a very fast iterative method is devised. For images with multiple classes, we use the Stochastic level set method proposed in [27] to globally minimize the objective $P_{SMS}$. Details could be found in [27].
The segmentation is encoded into the binary variable $\chi(x)$ for each pixel $x \in \Omega$. For other notations, $u \in [0, 1]$ is a relaxation of $\chi$ such that thresholding $u$ gives $\chi$, $\theta$ is a parameter controlling the closeness between $u$ and an auxiliary variable $v$, $\delta = 1/8$ is a step size to ensure convergence, div is the discrete divergence operator, and $\nabla$ is the discrete gradient operator.

In practice, we can often speed up the convergence of the outer alternating minimization in Algorithm 2 by iterating only once in the repeat-until loop in Algorithm 3.

### Algorithm 3: Minimization of $f^{\text{pos}}$ with Respect to $\{\Omega_i\}$

1. **Input:** features $\{f_j\}$, weights $\{\lambda_j\}$, fitting constants $\{c_{ij}\}$
2. Initialize $u^0$ by using k-means or the $u^k$ from the last run of Algorithm 3 or a random assignment
3. Initialize $p^0(x) = (0, 0)$ for each pixel $x$
4. Initialize $k = 0$
5. Set $r(x) = \sum_{j=1}^{m} \{\lambda_j |c_{ij} - f_j(x)|^2 - \lambda_j |c_j - f_j(x)|^2\}$ for each $x \in \Omega$
6. **repeat**
7. for each pixel $x \in \Omega$ do
8.  $\chi^{k+1}(x) = \begin{cases} 1 & \text{if } u^k(x) > 0.5, \\ 0 & \text{otherwise} \end{cases}$
9.  $u^{k+1}(x) = \min\{\max\{u^k(x) - \theta r(x), 0\}, 1\}$
10. $p^{k+1}(x) = \frac{p^k(x) + \delta \left[ \nabla \left( \text{div} p^k - \nabla u^{k+1} \right) \right](x)}{1 + \delta \left[ \nabla \left( \text{div} p^k - \nabla u^{k+1} \right) \right](x)}$
11. $u^{k+1}(x) = u^{k+1}(x) - \mu \theta (\text{div} u^{k+1})(x)$
12. **end for**
13. $k = k + 1$
14. **until** $u^k$ converges
15. **Output:** segmentation $\chi^k$

This work was supported in part by the Biomedical Research Council of the Agency for Science, Technology and Research (A*STAR), Singapore, and by the National University of Singapore under Grant R-146-000-116-112.

### References and Notes

18. The Brodatz texture similarity measures are defined as follows. Let $u$ be a given image and $\{v_k\}$ be the set of all images in the Brodatz database. The $k$th feature value $f_k(x, y)$ at pixel $(x, y) \in \Omega$ is defined by $f_k(x, y) = d(u(\cdot; \cdot; x, y), v_k)$, where $d(\cdot; \cdot)$ is defined by Eq. (15) and $u(\cdot; \cdot; x, y)$ is a local patch in $u$ around $(x, y)$.