TEXEMS: Texture Exemplars for Defect Detection on Random Textured Surfaces

Xianghua Xie, *Member*, *IEEE*, and Majid Mirmehdi, *Member*, *IEEE*

Abstract—We present an approach to detecting and localizing defects in random color textures which requires only a few defect free samples for unsupervised training. It is assumed that each image is generated by a superposition of various-size image patches with added variations at each pixel position. These image patches and their corresponding variances are referred to here as textural exemplars or *texems*. Mixture models are applied to obtain the texems using multiscale analysis to reduce the computational costs. Novelty detection on color texture surfaces is performed by examining the same-source similarity based on the data likelihood in multiscale, followed by logical processes to combine the defect candidates to localize defects. The proposed method is compared against a Gabor filter bank-based novelty detection method. Also, we compare different texem generalization schemes for defect detection in terms of accuracy and efficiency.

Index Terms—Defect detection, texture analysis, texem model, mixture model, EM algorithm.

1 Introduction

Visual surface inspection tasks are concerned with identifying regions that deviate from defect-free samples according to certain criteria, e.g., pattern regularity or color. Machine vision techniques are now regularly used in detecting such defects or imperfections on a variety of surfaces, such as textile, ceramics tiles, wood, steel, silicon wafers, paper, meat, leather, and even curved surfaces, e.g., [1], [2], [3], [4], [5]. Generally, this detection process should be viewed as different to texture segmentation, which is concerned with splitting an image into homogeneous regions. Neither the defect-free regions nor the defective regions have to be texturally uniform. For example, a surface may contain totally different types of defects which are likely to have different textural properties. On the other hand, a defect-free sample should be processed without the need to perform "segmentation," no matter how irregular and unstationary the texture is. Defect localization should also be viewed differently from image classification. It involves classifying the surfaces, not just globally, but going further to localize the defective regions. Moreover, in visual inspection, all the defective surfaces are usually considered as a single class, defect class, as it is a positive verification process. However, those defective samples are likely to form multiple classes in terms of surface properties and would need to be further categorized in a conventional classification approach.

Some materials display complex patterns but appear visually regular on a larger scale, e.g., textile, steel, or wafer. Methods such as co-occurrence matrices, template-based methods, and Fourier-domain analysis have proven useful in detecting defects on materials that exhibit a high degree of regularity and periodicity. For example, filter bank-based approaches, particularly Gabor filters [3], [4], have been applied due to their ability to analyze texture by achieving optimal joint localization in the spatial and frequency domains. Randen and Husøy [6] present a thorough comparative review of texture analysis using filtering techniques. For materials that display complex, random appearance patterns, such as marble slabs or printed ceramic tiles, detecting subtle local defects turns out to be rather difficult [7].

However, the supremacy of filter bank-based methods for texture analysis have been challenged by several authors. For instance, in [8], Varma and Zisserman argued that a large variety of signals (e.g., textures) can be analyzed by just looking at small neighborhoods. They used 7×7 patches to generate a texton-based representation and achieved better performance than the filtering-based methods they compared against. Their results demonstrated that textures with global structures can be discriminated by examining the distribution of local measurements. This is a key factor in the approach in this paper. In [9], Ojala et al. also advocated the use of local neighborhood processing in the shape of local binary patterns (LBP) as texture descriptors. These generate a binary code that characterizes the local texture pattern by thresholding a neighborhood by the gray value of its center. Other works based on local pixel neighborhoods are those which apply Markov Random Field models. For example, in [1], Cohen et al. used Gaussian Markov Random Fields (GMRF) to model defect free textile web. The inspection process was treated as a hypothesis testing problem on the statistics derived from the GMRF model. The testing image was partitioned into nonoverlapping subblocks, where each window was then classified as defective or nondefective.

Recently, in [10], Jojic et al. defined the epitome as a miniature, condensed version of an image containing the constitutive elements of its shape and textural properties needed to reconstruct the image. The epitome also relies on "raw" pixel values to characterize textural and color properties rather than popular filtering responses. An

- *X. Xie is with the Department of Computer Science, University of Bristol, MVB 2.08, Woodland Road, Bristol BS8 1UB, UK. E-mail: xie@cs.bris.ac.uk.*
- *M. Mirmehdi is with Department of Computer Science, University of Bristol, MVB 3.16, Woodland Road, Bristol BS8 1UB, U.K. E-mail: majid@cs.bris.ac.uk.*

Manuscript received 26 Aug. 2005; revised 15 Mar. 2006; accepted 9 Oct. 2006; published online 18 Jan. 2007.
Recommended for acceptance by G. Healey.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0468-0805.
Digital Object Identifier no. 10.1109/TPAMI.2007.1038.

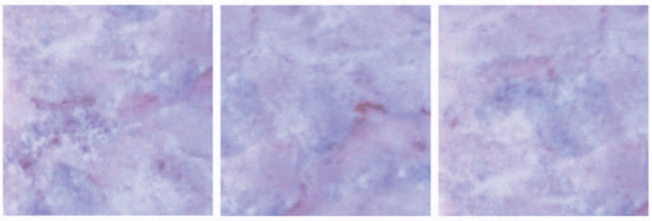

Fig. 1. Example marble tiles from the same family whose patterns are different but visually consistent.

image is defined by its epitome and a smooth, *hidden* mapping from the epitome to its image pixels.

Inspired by the success of nonfiltering local neighborhood approaches and the work in [10], in this paper, we propose a new approach to detecting and localizing defects on random textured surfaces. We generate texture exemplars, referred to as *texems*, by exploring mixture model representations of patches of texture using different formulations, one for gray-level textures and one for color textures.

Some defect types are unpredictable and occur only during production. Due to limited time and resources in a factory environment, it is difficult to collect a wide range of defective samples which is an essential step for a more traditional classification-based approach requiring a significant training stage, such as neural networks. Here, we propose the texem model as a suitable tool in texture defect inspection within a novelty detection framework. To ensure computational efficiency, we also extend the proposed method into a multiscale framework. We evaluate texems using a ceramic tile defect detection application and texture collages created from VisTex textures [11].

In Section 3, the proposed method is presented, including the two different EM-based generations of the texture exemplars, different extensions from gray-level image analysis to color image analysis, the multiscale approach, and the novelty detection stage. Experimental results are given in Section 4. Section 5 concludes the paper.

2 Motivation and Foundations

In an application such as ceramic tile production, the images under inspection may appear different from one surface to another due to the random texture patterns involved. However, the visual impression of the same product line remains consistent. In other words, there exist textural primitives that impose consistency within the product line. Fig. 1 shows three example tile images from the same class (or production run) decorated with a marble texture. Each tile has different features on its surface, but they all still exhibit a consistent visual impression. One may collect enough samples to cover the range of variations and this approach has been widely used in texture classification and defect detection, e.g., for textile defects in [12]. It usually requires a large number of nondefective samples and lengthy training stages; not necessarily practical in a factory environment. Additionally, defects are usually unpredictable.

Instead of the traditional classification approach, we learn, in an unsupervized fashion, textural primitive information from a very small number of training samples. We name the representations *texture exemplars* or *texems*. They occur at various sizes and encapsulate the texture or visual primitives of a given image. Similar to the work in [10], we consider that each surface texture is produced by putting together a certain number of subimage patches of various sizes, possibly overlapped. As the images of the same (tile) product contain the same textural elements, one image can be generated from the patches extracted from other images. Thus, for a few given samples we can easily obtain a large number of patches of various sizes (which can, in turn, generate a large set of new images with the same visual impression). However, it is computationally prohibitive to perform defect detection based on such a large number of patches. Also, the patches themselves contain lots of redundant information. We can reduce the number of patches by learning a relatively small number of primitive representatives, i.e., texems. In this paper, we use mixture models to learn the texems. Fig. 2 illustrates the generalization of texems.

The texem model is similar to the well-known texton model only in the sense that both try to characterize textural

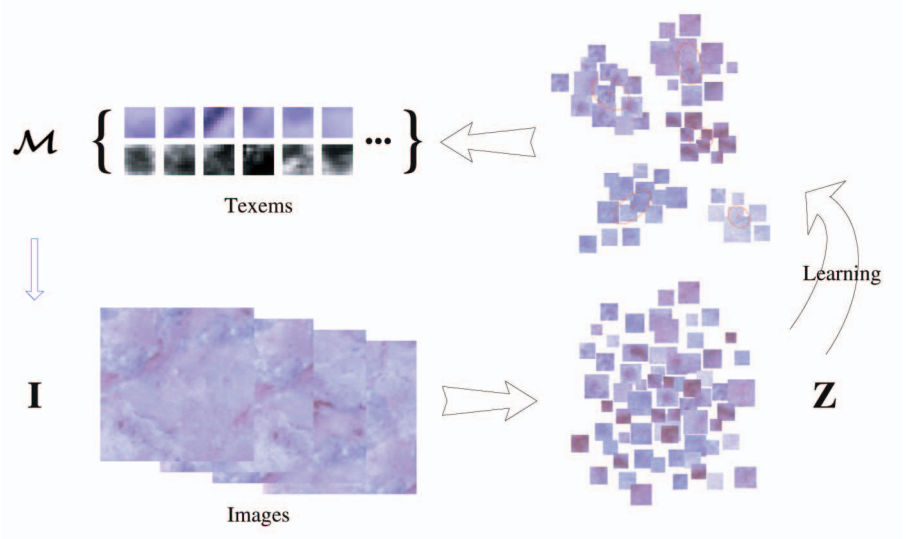

Fig. 2. An illustration of the two-layer structure of the texem model and its bottom-up learning procedure. Images $\mathbf{I}$ are split into patches $\mathbf{Z}$, from which texems $\mathcal{M}$ are learned. The images can, in turn, be generated from superposition of a selection of texems from $\mathcal{M}$.

images by using microstructures. Textons were first formally introduced by Julesz in [13] as fundamental image structures, such as elongated blobs, bars, crosses, and terminators, and were considered as atoms of preattentive human visual perception. Zhu et al. [14] used a three-layer *generative* model to describe textons. An image, as the first layer, was considered as a superposition of a number of image bases in the second layer, such as Laplacian of Gaussians and Gabors, selected from an overcomplete dictionary. These image bases were generated by a smaller number of texton elements, selected from a dictionary of textons, as the third layer in the model. Although also in a three layer structure, Leung and Malik [15] adopted a *discriminative* model, where an image was deterministically transformed by filtering through a bank of Gaussian filters at different orientations, scales, and phases. The central clusters, via K-means clustering, of these filter response vectors from every pixel position were considered as textons. Our proposed method is significantly different from the texton model in that 1) it relies directly on subimage patches instead of using base functions, and 2) it is an implicit, rather than an explicit, representation of primitives. Selection of the base functions is important in the texton model in order to obtain meaningful textons. However, the design of a bank of base functions is nontrivial and likely to be application dependent. Enormous efforts have been carried out in explicitly extracting visual primitives (textons), such as blobs. They generally face the difficulty of finding optimum representation, e.g., the window size (for example see [8]). In the proposed model, each texem is an encapsulation of texture primitive(s). Texems are implicit representations, which makes them more flexible as they come at different sizes and may contain multiple texture primitives, while textons are explicit representations each of which tries to describe a fundamental visual or textural structure. In other words, a texem may contain partial texton or multiple textons. For example, if the texem size reduces to a single pixel, it becomes histogram analysis. If it is the same size as the input images, then the problem turns into image template analysis. In general, each of our texems may contain multiple textural primitives which describes a group of image patches. This implicit representation at various sizes avoids the difficulties of explicitly finding the best primitive representation. Not using base functions also makes our work more convenient to deal with multispectral images, e.g., color images.

Notably, there are a few texton-based works which directly use "raw" pixel values to extract textural primitives, e.g., K-means clustering as used in [8], or Transformed Component Analysis in a vectorized patch space [14]. However, the proposed two-layer generative texem model, utilizing Gaussian mixture modeling, is more advantageous in quantitatively measuring the data similarity. It is also worth noting the importance of using multiscale or various size of patches in learning textural primitives, besides its simplicity. As shown by [8], patch size selection played an important role in their image classification. We advocate multiscale analysis to alleviate the difficulties associated with scale selection.

The epitome can be used to generalize multiple images, while preserving textural details against oversmoothing [10]. We could condense the large number of patches extracted from tile images into an epitomic representation of the whole family. However, as the mappings between the epitome and the training images are hidden and the mappings from the testing images cannot be possibly obtained as a priori knowledge, we would need to consider all possible mappings from the epitome at both the training and testing stages, which involves the examination of different patch sizes at each pixel location in the epitome. Although the epitome is much smaller than the original image, it is still much larger than the patches themselves. Hence, it will be computationally very expensive to perform defect detection directly based on the epitome. Thus, rather than forcing the textural properties to be condensed into a single epitome, we learn multiple epitomic-like representations that generalize the family of texture images without using any implicit mappings. The large number of patches at various sizes can then be described using a very small number of representatives with explicit pixel position correspondence.

3 The Texem Model

In this section, we propose a two-layer generative model (see Fig. 2), in which an image as the first layer is assumed to be generated by superposition of a small number of image patches of various sizes from the second layer with added Gaussian noise at each pixel position. Here, we formally define each texem as a mean image patch associated with a corresponding variance which controls the variation of this particular texem. The form of the variance can vary according to learning schemes. The generation process can be naturally modeled by mixture models with a bottom-up procedure.

Next, we detail the process of extracting texems from a single sample image with each texem containing some of the overall textural primitive information. We shall use two different mixture models. The first is for gray-level images in which we vectorize the image patches and apply a Gaussian mixture model to obtain the texems. In the second, color textures are represented by texems using a mixture model learned based on joint Gaussian distributions within local neighborhoods. This extension of texem to color analysis is examined against other alternatives based on channel separation. We also introduce a multiscale texem representation to drastically reduce the overall computational effort. Finally, we show how to automatically set data likelihood bounds on defect-free samples and perform localized defect detection using texems.

3.1 Gray-Level Texems

For gray-level images, we use a Gaussian mixture model to obtain the texems in a simple and efficient manner [16]. The original image $\mathbf{I}$ is broken down into a set of P patches $\mathbf{Z} = \{\mathbf{Z}_i\}_{i=1}^{P}$, each containing pixels from a subset of image coordinates. The shape of the patches can be arbitrary, but in this study we used square patches of size $d = N \times N$. The patches may overlap and can be of various sizes, e.g., as small as 5×5 to as large as required (however, for large window sizes, one should ensure there are enough samples to populate the feature space). We group the patches of sample images into clusters, depending on the patch size, and describe the clusters using the Gaussian mixture model. Here, each texem, denoted as $\mathbf{m}$, is defined by a mean, μ, and a corresponding covariance matrix, ω, i.e., $\mathbf{m} = \{\mu, \omega\}$. We assume that there exist K texems, $\mathcal{M} = \{\mathbf{m}_k\}_{k=1}^{K}$, $K \ll P$, for image $\mathbf{I}$ such that each patch in $\mathbf{Z}$ can be generated from a texem $\mathbf{m}$.

To learn these texems, the P patches are projected into a set of high dimensionality spaces. The number of these spaces is determined by the number of different patch sizes and their dimensions are defined by the corresponding value of d. Each pixel position contributes one coordinate of a space. Each point in a space corresponds to a patch in $\mathbf{Z}$. Then, each texem represents a class of patches in the corresponding space. We assume that each class is a multivariate Gaussian distribution with mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\omega}_k$, which corresponds to $\mathbf{m}_k$ in the patch domain. Thus, given the kth texem the probability of patch $\mathbf{Z}_i$ is computed as

$$p(\mathbf{Z}_i|\mathbf{m}_k, \psi) = \mathcal{N}(\mathbf{Z}_i; \boldsymbol{\mu}_k, \boldsymbol{\omega}_k), \tag{1}$$

where $\psi = \{\boldsymbol{\alpha}_k, \boldsymbol{\mu}_k, \boldsymbol{\omega}_k\}_{k=1}^{K}$ is the parameter set containing $\boldsymbol{\alpha}_k$, which is the *prior* probability of kth texem constrained by $\sum_{k=1}^{K} \boldsymbol{\alpha}_k = 1$, the mean $\boldsymbol{\mu}_k$, and the covariance matrix $\boldsymbol{\omega}_k$. Since all the texems $\mathbf{m}_k$ are unknown, we need to compute the density function of $\mathbf{Z}$ given the parameter set ψ by applying the definition of conditional probability and summing over k for $\mathbf{Z}_i$,

$$p(\mathbf{Z}_i|\psi) = \sum_{k=1}^{K} p(\mathbf{Z}_i|\mathbf{m}_k, \psi)\boldsymbol{\alpha}_k, \tag{2}$$

and then optimizing the data log-likelihood expression of the entire set $\mathbf{Z}$, given by

$$\log p(\mathbf{Z}|K, \psi) = \sum_{i=1}^{P} \log\left(\sum_{k=1}^{K} p(\mathbf{Z}_i|\mathbf{m}_k, \psi)\boldsymbol{\alpha}_k\right). \tag{3}$$

Hence, the objective is to estimate the parameter set ψ for a given number of texems. EM can be used to find the maximum-likelihood estimate of our mixture density parameters from the given data set $\mathbf{Z}$. That is to find $\hat{\psi}$, where

$$\hat{\psi} = \arg\max \log(\mathcal{L}(\psi|\mathbf{Z})) = \arg\max_{\psi} \log p(\mathbf{Z}|K, \psi). \tag{4}$$

Then, the two steps of the EM stage are as follows: The E-step involves a soft assignment of each patch $\mathbf{Z}_i$ to texems, $\mathcal{M}$, with an initial guess of the true parameters, ψ. This initialization can be set randomly (although we use K-means to compute a simple estimate with K set as the number of texems to be learned). We denote the intermediate parameters as $\psi^{(t)}$, where t is the iteration step. The likelihood of kth texem given the patch $\mathbf{Z}_i$ may then be computed using Bayes' rule

$$p(\mathbf{m}_k|\mathbf{Z}_i, \psi^{(t)}) = \frac{p(\mathbf{Z}_i|\mathbf{m}_k, \psi^{(t)})\boldsymbol{\alpha}_k}{\sum_{k=1}^{K} p(\mathbf{Z}_i|\mathbf{m}_k, \psi^{(t)})\boldsymbol{\alpha}_k}. \tag{5}$$

The M-step then updates the parameters by maximizing the log-likelihood, resulting in new estimates

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}_k &= \frac{1}{P}\sum_{i=1}^{P} p(\mathbf{m}_k|\mathbf{Z}_i, \psi^{(t)}),\\
\hat{\boldsymbol{\mu}}_k &= \frac{\sum_{i=1}^{P} \mathbf{Z}_i p(\mathbf{m}_k|\mathbf{Z}_i, \psi^{(t)})}{\sum_{i=1}^{P} p(\mathbf{m}_k|\mathbf{Z}_i, \psi^{(t)})},\\
\hat{\boldsymbol{\omega}}_k &= \frac{\sum_{i=1}^{P} (\mathbf{Z}_i - \hat{\boldsymbol{\mu}}_k)(\mathbf{Z}_i - \hat{\boldsymbol{\mu}}_k)^T p(\mathbf{m}_k|\mathbf{Z}_i, \psi^{(t)})}{\sum_{i=1}^{P} p(\mathbf{m}_k|\mathbf{Z}_i, \psi^{(t)})}.
\end{aligned} \tag{6}$$

The E-step and M-step are iterated until the estimations stabilize. Then, the texems can be easily obtained by projecting the learned means and covariance matrices back to the patch representation space.

Later, in Sections 3.3 to 3.5, we describe how texems are implemented in a multiscale framework and the results from the different multiscale levels are consolidated into a single defect map.

3.2 Color Texems

Surface inspection tasks are often adequately dealt with using gray-level images. However, due to increasing processing power and the availability of relatively inexpensive color cameras, there are good prospects for more accurate visual inspection using color when appropriate. Furthermore, some defects are chromatic defects by nature, so the use of color then becomes of paramount importance.

Visual inspection using random color texture analysis is nevertheless still largely underdeveloped in the literature and only a limited number of works have been reported so far. In [7] and [17], the authors performed color clustering, followed by binarized spatial pixel distribution analysis to identify textural defects in color ceramic tile images. The color clustering and binarization in the spatial domain partially took into account both spatial and spectral interactions. Mäenpää et al. [18] measured color percentiles based on the accumulated histogram in each RGB channel as chromatic features and co-occurrence matrices and LBP features as textural features to inspect wood surfaces. In [5], the authors used similar features as [18], but performed self-organizing map-based clustering for wood surface inspection. Recently, Tsai et al. [19] transformed color images into the $L^*a^*b^*$ space to derive hue and chroma channels in each of which they performed Gabor filtering. The authors argued that their application processing images in these two chromatic channels only could be resilient to illumination changes assuming that defects are chromatically differentiable.

In this section, we explore different schemes to extend gray-level texems to color images with differing computational complexity and rate of accuracy.

3.2.1 Texem Analysis in Separate Channels

More often than not, color texture analysis is treated as a simple dimensional extension of techniques designed for gray-level images and, so, color images are decomposed into separate channels to perform the same processes. For example, in [20], Lumbreras et al. decomposed each RGB channel using wavelet analysis to sort ceramic tiles based on color and texture characteristics.

Thus, if we are willing to sacrifice some information, we can learn gray-level texems in each RGB channel of an image. However, this gives rise to difficulties in capturing both the interchannel and spatial properties of the texture. Special care is usually necessary, e.g., in [21], opponent features that capture the interaction between color channels are enhanced with spatial features from individual channels to represent color textures. Alternatively, we can decorrelate the image channels using Principal Component Analysis (PCA) and then perform texems analysis in each independent channel separately. We prefer this approach and use it to compare against our full color texem model introduced later.

As the patterns on each image within the same texture family can still be different, the individually derived principal components can also differ from one image to another. Furthermore, defective regions can affect the principal components resulting in different eigenspace responses from

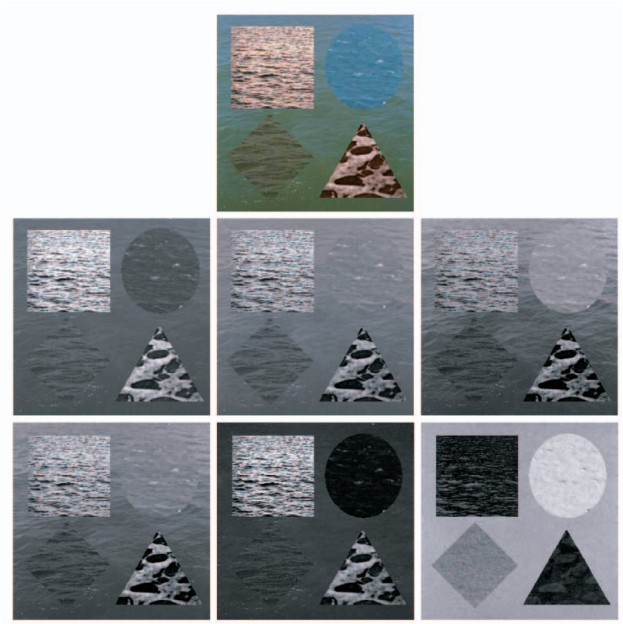

Fig. 3. Channel separation—first row: Original collage image; second row: individual RGB channels; third row: eigenchannel images.

different training samples. Thus, instead of performing PCA on each training image separately, a single eigenspace is generated from several training images. This results in a reference eigenspace in which defect-free samples are represented. Then, all new, previously unseen images under inspection will be projected onto this eigenspace such that the transformed channels share the same principal components.

Let $\mathbf{c}_i = [r_i, g_i, b_i]^T$ be a color pixel, $\mathbf{C} = \{\mathbf{c}_i \in \mathcal{R}^3, i = 1, 2, \ldots, q\}$ be the set of q three-dimensional vectors made up of the pixels from several defect-free samples, and $\bar{\mathbf{c}} = \frac{1}{q}\sum_{\mathbf{c}\in\mathbf{C}} \mathbf{c}$ be the mean vector of $\mathbf{C}$. Then, PCA is performed on the mean-centered color feature matrix $\mathbf{C}$ to obtain the eigenvectors $E = [e_1, e_2, e_3]$, $e_j \in \mathcal{R}^3$. Singular Value Decomposition can be used to obtain these principal components. The color feature space determined by these eigenvectors is referred to as the reference eigenspace $\Upsilon_{\bar{\mathbf{c}},E}$, where the color features are well represented. The tile images are then projected onto this reference eigenspace

$$\mathbf{C}' = \overrightarrow{PCA}(\mathbf{C}, \Upsilon_{\bar{\mathbf{c}},E}) = E^T(\mathbf{C} - \bar{\mathbf{c}}J_{1,q}), \tag{7}$$

where $J_{1,q}$ is a $1 \times q$ unit matrix consisting of all 1s. Fig. 3 shows a comparison of direct RGB channel separation and PCA-based channel separation. The eigenchannels clearly are more differentiating.

Once we obtain the reference eigenspace, $\Upsilon_{\bar{\mathbf{c}},E}$, defect detection and localization are then performed in each of the three corresponding channels by examining the local context using the gray-level texem model described earlier in Section 3.1 (more details can be found in [22]).

3.2.2 Full Color Model

By decomposing the color image and analyzing image channels individually, the interchannel and intrachannel spatial interactions are not taken into account. To facilitate such interactions, we use a different formulation for texem representation and consequently change the inference procedure so that no vectorization of image patches is required and color images do not need to be transformed into separate channels. Contrary to the way gray-level texems were developed, where each texem was represented by a single multivariate Gaussian function, for color texems we assume that pixels are statistically independent in each texem with Gaussian distribution at each pixel position in the texem. This is similar to the way the image epitome is generated by Jojic et al. [10]. Thus, the probability of patch $\mathbf{Z}_i$ given the kth texem can be formulated as a joint probability at each pixel position, i.e.,

$$p(\mathbf{Z}_i|\theta_k) = p(\mathbf{Z}_i|\boldsymbol{\mu}_k, \boldsymbol{\omega}_k) = \prod_{j\in S} \mathcal{N}(\mathbf{Z}_{j,i}; \boldsymbol{\mu}_{j,k}, \boldsymbol{\omega}_{j,k}), \tag{8}$$

where θ_k denotes the kth texem's parameters with mean $\boldsymbol{\mu}_k$ and variance $\boldsymbol{\omega}_k$, S is the pixel patch grid, $\mathcal{N}(\mathbf{Z}_{j,i}; \boldsymbol{\mu}_{j,k}, \boldsymbol{\omega}_{j,k})$ is a Gaussian distribution over $Z_{j,i}$ and $\boldsymbol{\mu}_{j,k}$ and $\boldsymbol{\omega}_{j,k}$ denote mean and covariance matrix at the jth pixel position in the kth texem. For our mixture model, similar to (2) but using the component probability function as given in (8), we assume the following probabilistic model

$$p(\mathbf{Z}_i|\Theta) = \sum_{k=1}^{K} p(\mathbf{Z}_i|\theta_k)\boldsymbol{\alpha}_k, \tag{9}$$

where the parameters are $\Theta = \{\boldsymbol{\alpha}_k, \theta_k\}_{k=1}^K$ and can be determined by optimizing the data log-likelihood given by

$$\log p(\mathbf{Z}|K, \Theta) = \sum_{i=1}^{P} \log\left(\sum_{k=1}^{K} p(\mathbf{Z}_i|\mathbf{m}_k, \theta_k)\boldsymbol{\alpha}_k\right). \tag{10}$$

The EM technique can be used again to find the maximum-likelihood estimate

$$\hat{\Theta} = \arg\max \log(\mathcal{L}(\Theta|\mathbf{Z})) = \arg\max_{\Theta} \log p(\mathbf{Z}|K, \Theta). \tag{11}$$

The new estimates, denoted by $\hat{\boldsymbol{\alpha}}_k$, $\hat{\boldsymbol{\mu}}_k$, and $\hat{\boldsymbol{\omega}}_k$, are updated during the EM iterations

$$\begin{aligned}
\hat{\boldsymbol{\alpha}}_k &= \frac{1}{P}\sum_{i=1}^{P} p(\mathbf{m}_k|\mathbf{Z}_i, \Theta^{(t)}),\\
\hat{\boldsymbol{\mu}}_k &= \{\hat{\boldsymbol{\mu}}_{j,k}\}_{j\in S},\\
\hat{\boldsymbol{\omega}}_k &= \{\hat{\boldsymbol{\omega}}_{j,k}\}_{j\in S},\\
\hat{\boldsymbol{\mu}}_{j,k} &= \frac{\sum_{i=1}^{P} \mathbf{Z}_{j,i} p(\mathbf{m}_k|\mathbf{Z}_i, \Theta^{(t)})}{\sum_{i=1}^{P} p(\mathbf{m}_k|\mathbf{Z}_i, \Theta^{(t)})},\\
\hat{\boldsymbol{\omega}}_{j,k} &= \frac{\sum_{i=1}^{P} (\mathbf{Z}_{j,i} - \hat{\boldsymbol{\mu}}_{j,k})(\mathbf{Z}_{j,i} - \hat{\boldsymbol{\mu}}_{j,k})^T p(\mathbf{m}_k|\mathbf{Z}_i, \Theta^{(t)})}{\sum_{i=1}^{P} p(\mathbf{m}_k|\mathbf{Z}_i, \Theta^{(t)})},
\end{aligned} \tag{12}$$

where

$$p(\mathbf{m}_k|\mathbf{Z}_i, \Theta^{(t)}) = \frac{p(\mathbf{Z}_i|\mathbf{m}_k, \Theta^{(t)})\boldsymbol{\alpha}_k}{\sum_{k=1}^{K} p(\mathbf{Z}_i|\mathbf{m}_k, \Theta^{(t)})\boldsymbol{\alpha}_k}. \tag{13}$$

The iteration continues till the values stabilize. Various sizes of texems can be used and they can overlap to ensure they capture sufficient textural characteristics. We can see that when the texem reduces to a single pixel size, (12) becomes Gaussian mixture modeling based on pixel colors.

Fig. 4 illustrates six 9×9 texems extracted from one of the images shown in Fig. 1. They are arranged according to their descending order of *priors* $\boldsymbol{\alpha}_k$. We may treat each prior,

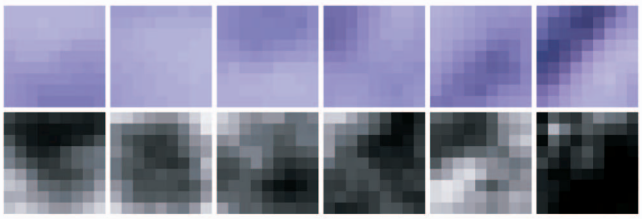

Fig. 4. Six 9×9 texems extracted from the images shown in Fig. 1. Top row: The six means. Bottom row: Their corresponding variance images (Images are enhanced for viewing purposes). Note each element in a texem mean is a 3×1 color vector, and each element in a texem variance is a 3×3 matrix characterizing the covariance in the color space. Thus, for visualization purposes, each element in the variance images is represented using total variance.

α_k, as a measurement of the contribution from each texem. The image then can be viewed as a superposition of various sizes of image patches taken from the means of the texems, a linear combination, with added variations at each pixel position governed by the corresponding variances.

3.3 Multiscale Texems

In order to capture sufficient textural properties, texems can be from as small as 3×3 to larger sizes such as 20×20. However, the dimension of the space patches $\mathbf{Z}$ are transformed into will increase dramatically as the dimension of the patch size d increases. This means that a very large number of samples and high-computational costs are needed in order to accurately estimate the probability density functions in very high-dimensional spaces, [23], forcing the procurement of a large number of training samples.

Instead of generating variable-size texems, fixed-size texems can be learned in multiscale. This will result in (multiscale) texems with a small size, e.g., 5×5. Besides computational efficiency, exploiting information at multiscale offers other advantages over single-scale approaches. Characterizing a pixel based on local neighborhood pixels can be more effectively achieved by examining various neighborhood relationships. The corresponding neighborhood at coarser scale obviously offers larger spatial interactions. Also, processing at multiscale ensures the capture of the optimal resolution, which is often data dependent. A simple multiscale approach by using a Gaussian pyramid is found sufficient.

Let us denote $\mathbf{I}^{(n)}$ as the nth level image of the pyramid, $\mathbf{Z}^{(n)}$ as all the image patches extracted from $\mathbf{I}^{(n)}$, l as the total number of levels, and $S^{\downarrow}$ as the down-sampling operator. We then have

$$\mathbf{I}^{(n+1)} = S^{\downarrow} G_{\sigma}(\mathbf{I}^{(n)}), \qquad \forall n, n = 1, 2, \ldots, l-1, \tag{14}$$

where G_σ denotes the Gaussian convolution. The finest scale layer is the original image, $\mathbf{I}^{(1)} = \mathbf{I}$. We then extract multiscale texems from the image pyramid using the method presented in the previous section. Similarly, let $\mathbf{m}^{(n)}$ denote the nth level of multiscale texems and $\Theta^{(n)}$ the parameters associated at the same level, which will then be used for novelty detection at the corresponding level of the pyramid.

During the EM process, the stabilized estimation of a coarser level is used as the initial estimation for the finer level, i.e.,

$$\hat{\Theta}^{(n,t=0)} = \Theta^{(n+1)}, \tag{15}$$

which helps speed up the convergence and achieve a more accurate estimation.

3.4 Unsupervized Training

Novelty detection frees the application from having to provide a portfolio of defects within a supervized training stage. In fact, for our example, application of randomly textured color ceramic tiles, texems lend themselves well to performing unsupervized training and testing for novelty detection. This is achieved by automatically determining the threshold of statistical texture variation of defect-free samples at each resolution level.

For training, a small number of defect free samples (e.g., 4 or 5 only) are arranged within the multiscale framework, i.e., each image is presented as a pyramid in the multiscale fashion as denoted in (14) and patches with the same texem size are extracted. The probability of a patch $\mathbf{Z}_i^{(n)}$ belonging to texems in the corresponding nth scale is

$$p(\mathbf{Z}_i^{(n)}|\Theta^{(n)}) = \sum_{k=1}^{K^{(n)}} p(\mathbf{Z}_i^{(n)}|\mathbf{m}_k^{(n)}, \Theta^{(n)})\boldsymbol{\alpha}_k^{(n)}, \tag{16}$$

where $\Theta^{(n)}$ represents the parameter set for level n, $\mathbf{m}_k^{(n)}$ is the kth texem at the nth image pyramid level and $p(\mathbf{Z}_i^{(n)}|\mathbf{m}_k^{(n)}, \Theta^{(n)})$ is a product of Gaussian distributions shown in (9) with parameters associated to texem set $\mathcal{M}$. Based on this probability function, we then define a novelty score function as the negative log-likelihood

$$\mathcal{V}(\mathbf{Z}_i^{(n)}|\Theta^{(n)}) = -\log p(\mathbf{Z}_i^{(n)}|\Theta^{(n)}). \tag{17}$$

The lower the novelty score, the more likely the patch belongs to the same family and vice versa. Thus, it can be viewed as a same source similarity measurement. The distribution of the scores for all the patches $\mathbf{Z}^{(n)}$ at level n of the pyramid forms a 1D novelty score space which is not necessarily a simple Gaussian distribution. In order to find the upper bound of the novelty score space of defect-free patches (or the lower bound of data likelihood), K-means clustering is performed in this space to approximately model the space. The cluster with the maximum mean is the component of the novelty score distribution at the boundary between good and defective textures. This component is characterized by mean $u^{(n)}$ and standard deviation $\sigma^{(n)}$. This K-means scheme replaces the single Gaussian distribution assumption in the novelty score space, which is commonly adopted in a parametric classifier in novelty detection, e.g., [24] and for which the correct parameter selection is critical. Instead, dividing the novelty score space and finding the critical component, here called the boundary component, can effectively lower the parameter sensitivity. The value of K should be generally small (we empirically fixed it at 5). It is also notable that a single Gaussian classifier is a special case of the above scheme, i.e., when $K = 1$. The maximum novelty score (or the minimum data likelihood), $\Lambda^{(n)}$ of a patch $\mathbf{Z}_i^{(n)}$ at level n across the training images is then established as

$$\Lambda^{(n)} = u^{(n)} + \lambda\sigma^{(n)}, \tag{18}$$

where λ is a simple constant ($\lambda = 2$ in the experiments in Section 4). This completes the training stage in which, with only a few defect-free images, we determine the texems and an automatic threshold for marking new image patches as good or defective.

3.5 Novelty Detection and Defect Localization

In the testing stage, the image under inspection is again layered into a multiscale framework and patches at each pixel position $\mathbf{x}$ at each level n are examined against the learned texems. The probability for each patch and its novelty score are then computed using (16) and (17) and compared to the maximum novelty score, determined by $\Lambda^{(n)}$, at the corresponding level. Let $Q^{(n)}(\mathbf{x})$ be the novelty score map at the nth resolution level. Then, the potential defect map, $\mathcal{D}^{(n)}(\mathbf{x})$, at level n is

$$\mathcal{D}^{(n)}(\mathbf{x}) = \begin{cases} 0 & \text{if } Q^{(n)}(\mathbf{x}) \leq \Lambda^{(n)} \\ Q^{(n)}(\mathbf{x}) - \Lambda^{(n)} & \text{otherwise,} \end{cases} \tag{19}$$

$\mathcal{D}^{(n)}(\mathbf{x})$ indicates the probability of there being a defect. Next, the information coming from all the resolution levels must be consolidated to build the certainty of the defect at position $\mathbf{x}$. We follow a method described in [3] which combines information from different levels of a multiscale pyramid and reduces false alarms. It assumes that a defect must appear in at least two adjacent resolution levels for it to be certified as such. Using a logical AND, implemented through the geometric mean of every pair of adjacent levels, we initially obtain a set of combined maps as

$$\mathcal{D}^{(n,n+1)}(\mathbf{x}) = [\mathcal{D}^{(n)}(\mathbf{x})\mathcal{D}^{(n+1)}(\mathbf{x})]^{1/2}. \tag{20}$$

Note each $\mathcal{D}^{(n+1)}(\mathbf{x})$ is scaled up to be the same size as $\mathcal{D}^{(n)}(\mathbf{x})$. This operation reduces false alarms and yet preserves most of the defective areas. Next, the resulting $\mathcal{D}^{(1,2)}(\mathbf{x}), \mathcal{D}^{(2,3)}(\mathbf{x}), \ldots, \mathcal{D}^{(l-1,l)}(\mathbf{x})$ maps are combined in a logical OR, as the arithmetic mean, to provide

$$\mathcal{D}(\mathbf{x}) = \frac{1}{l-1}\sum_{n=1}^{l-1} \mathcal{D}^{(n,n+1)}(\mathbf{x}), \tag{21}$$

where $\mathcal{D}(\mathbf{x})$ is the final consolidated map of (the joint contribution of) all the defects across all resolution scales of the test image.

The multiscale, unsupervized training, and novelty detection stages are applied in a similar fashion as described above in the cases of gray-level and the full color model texem methods. In the separate channel color approaches (i.e., before and after decorrelation), the final defective maps from each channel are ultimately combined.

4 Experimental Results

In this section, several sets of experiments are presented. Initially, the results of applying the proposed method to detecting defects on ceramic tiles are given. We can not compare and evaluate these localized defects against a groundtruth since the defects in our data set are difficult to localize by hand. However, whole tile classification rates, based on overall "defective" and "defect-free" labeling by factory-floor experts is presented. Moreover, in order to evaluate the proposed method, we outline the result of our experiments on texture collages made from textures in the MIT VisTex texture database [11]. A comparative study of three different approaches to texem analysis on color images and a Gabor filter bank-based method is given.

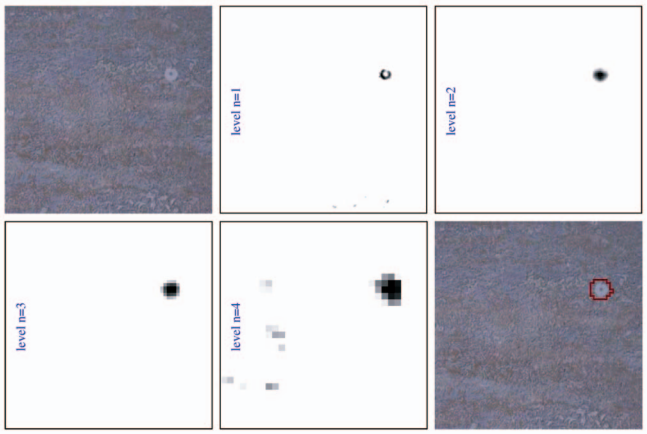

Fig. 5. Localizing textural defects—from top left to bottom right: original defective tile image, detected defective regions at different levels $n = 1, 2, \ldots, 4$, and the final defective region superimposed on the original image.

4.1 Ceramic Tile Application

We applied the proposed *full color texem model* to a variety of randomly textured tile data sets with different types of defects including physical damage, pin holes, textural imperfections, and many more. The 256×256 test samples were preprocessed to assure homogeneous luminance, spatially and temporally.[1] In the experiments, only five defect-free samples were used to extract the texems and to determine the upper bound of the novelty scores $\Lambda^{(n)}$. The number of texems at each resolution level were empirically set to 12 and the size of each texem was set to 5×5 pixels. The number of multiscale levels was $l = 4$. These parameters were fixed throughout our experiments on varieties of random texture tile prints.

Fig. 5 shows a random texture example with a defect in the upper right region introduced by a printing problem. The potentially defective regions detected at each resolution level n, $n = 1, \ldots, 4$, are marked on the corresponding images in Fig. 5. It can be seen that the texems show good sensitivity to the defective region at different scales. As the resolution progresses from coarse to fine, additional evidence for the defective region is gathered. The final image shows the defect superimposed on the original image. As mentioned earlier, the defect fusion process can eliminate false alarms, e.g., see the extraneous false alarm regions in levels $n = 1$ and $n = 4$ which disappear after the operations in (20) and (21).

More examples of different random textures are shown in Figs. 6 and 7. In each family of patterns, the textures are varying but have the same visual impression. In each case, the proposed method could find structural and chromatic defects of various shapes and sizes.

Fig. 8 shows three examples when using *gray-level* texems. Various defects, such as pin holes, bumps, dips, and print errors, are successfully detected. Gray-level texems were found adequate for most defect detection tasks where defects

1. Template profile-based luminance correction was used to effectively and efficiently compensate spatial and temporal luminance inhomogeneity due to cosine-fourth fall-off, vignetting effect, dust interference, and other nonlinear interferences in the image acquisition chain. The template profile was obtained from a standard white surface with uniform reflectance. In case of any abrupt changes in the imaging system, the profile was updated after a certain time period, similar to [2].

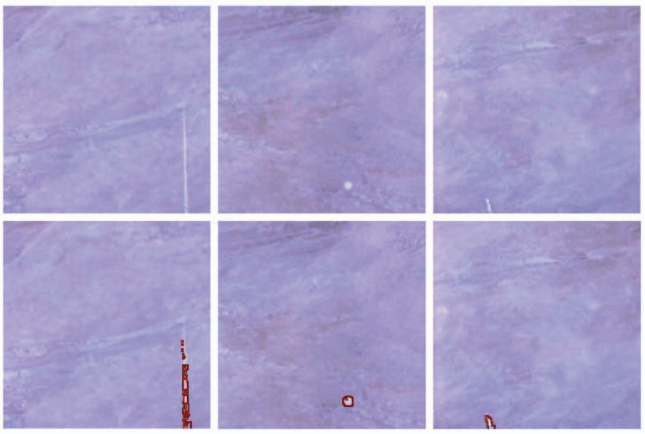

Fig. 6. Defect localization (same texture family)—first row: original images; second row: superimposed defective regions on original images, from left—print error, surface defect, and small bump.

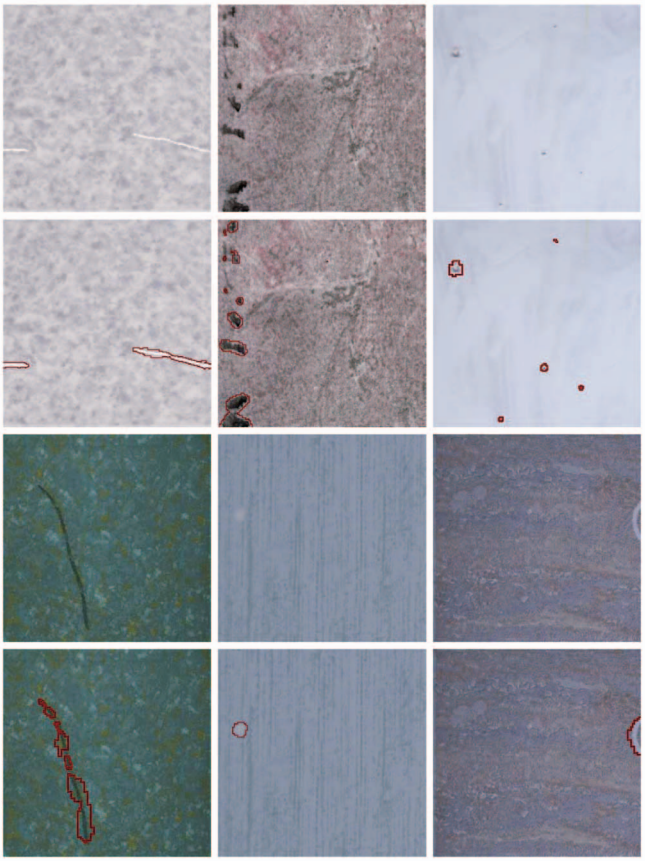

Fig. 7. Defect localization (different textures)—The first row shows example images from three different tile families with different chromato-textural properties. Defects shown in the next row, from left to right, include cracks, print error, and surface bumps. The third row shows another three images from three different tile families, two in color and one in gray-level. Defects shown in the last row, from left to right, include print errors and missing print.

were still reasonably visible after converting from color to gray scale. However, color texems were found to be more powerful in localizing defects and better discriminants in cases involving chromatic defects. Two examples are compared in Fig. 9. The first shows a tile image with a defective region, which is not only slightly brighter but also less saturated in blue. The color texem model achieved better

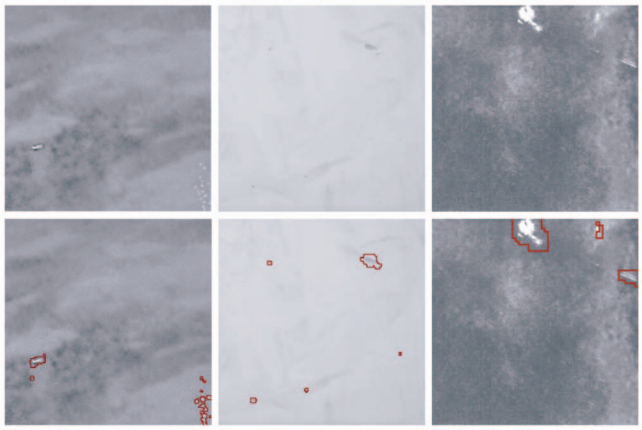

Fig. 8. Defect localization using gray-level texems: bump and pin holes, dimps, and print error

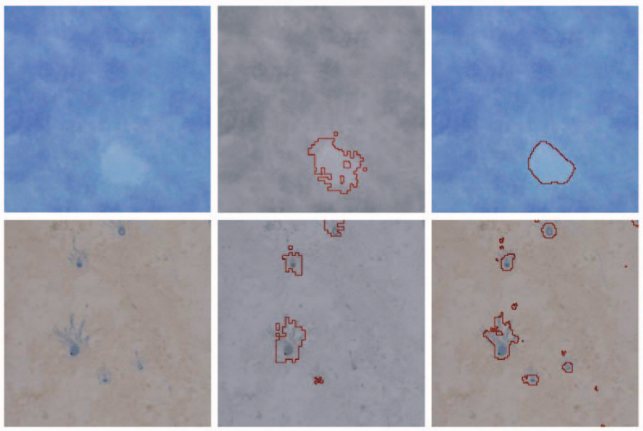

Fig. 9. Defect localization comparison: left column—original texture with print errors, left column—results using gray-level texems, right column—results using color texems.

results in localizing the defect than the gray-level one. The second row in Fig. 9 demonstrates a different type of defect which clearly possesses a different hue from the background texture. The color texems found more affected regions, more accurately.

The full color texem model was tested on 1,018 tile samples from 10 different families of tiles consisting of 561 defect-free samples and 457 defective samples. It obtained a defect detection accuracy rate of 91.1 percent, with sensitivity at 92.6 percent and specificity at 89.8 percent. The gray-level texem method was tested on 1,512 gray-level tile images from eight different families of tiles consisting of 453 defect-free samples and 1,059 defective samples. It obtained an overall accuracy rate of 92.7 percent, with sensitivity at 95.9 percent specificity at 89.5 percent. We compare the performance of gray-level and color texem models on the same dataset in the next set of experiments.

As patches are extracted from each pixel position at each resolution level, a typical *training stage* involves examining a very large number of patches. For the gray-level texem model, this takes around 7 minutes, on a 2.8GHz Pentium 4 Processor running Linux with 1GB RAM, to learn the texems in multiscale and to determine the thresholds for novelty detection. The testing stage then requires around 12 seconds to inspect one tile image. The full color texem model is computationally more expensive and can be more than

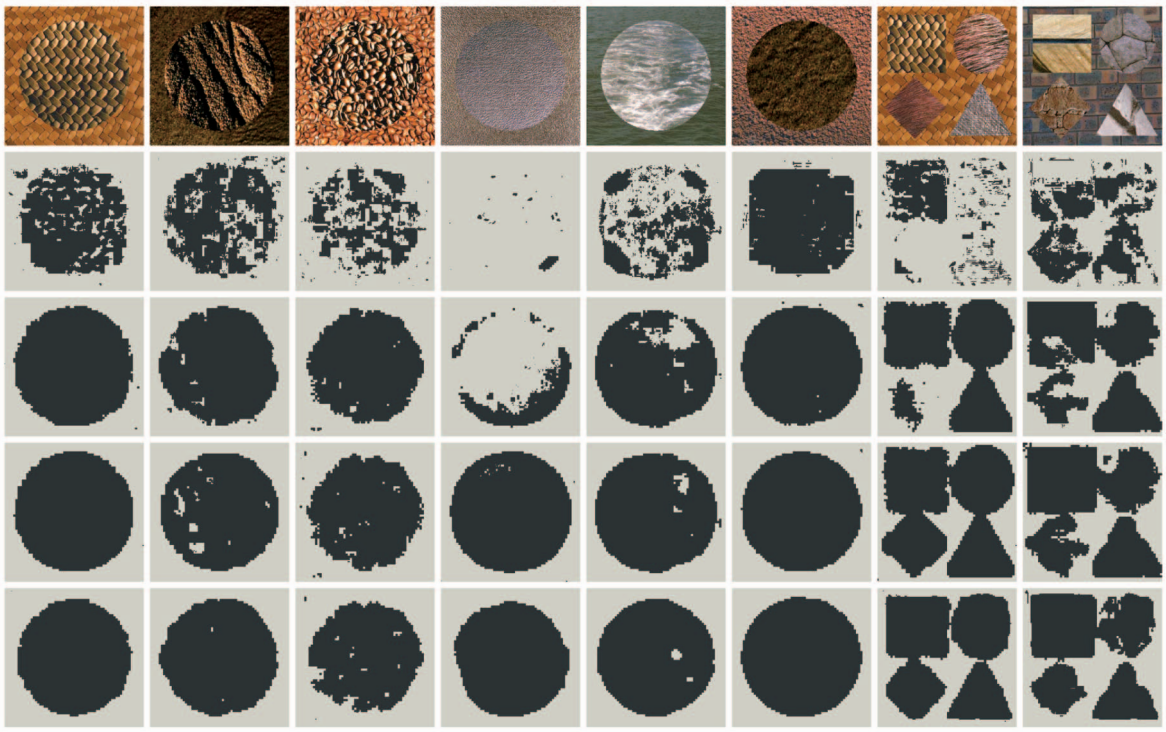

Fig. 10. Collage samples made up of materials such as foods, fabric, sand, metal, water, and novelty detection results (without any postprocessing). Rows from top: original collage images, Escofet et al.'s method [3], gray-level texems directly in RGB channels, gray-level texems in PCA decorrelated RGB eigenchannels, and full color texem model.

10 times slower. However, this can be reduced to the same order as the gray-level version by performing window-based, rather than pixel-based, examination at the training and testing stages.

4.2 Evaluation Using VisTex Collages

To evaluate the accuracy of the proposed method, we generated a set of 28 image collages (some are shown in Fig. 10) from textures in the MIT VisTex database [11]. In each case the background is the learned texture for which color texems are generated and the foreground (disk, square, triangle, and rhombus) is treated as the novelty to be detected. Note this is not a texture segmentation exercise, but rather defect segmentation. It should also be noted that the textures used were selected to be particularly similar in nature between the foreground and the background, e.g., see the collages in the first or third columns of Fig. 10. The testing results were quantified using *specificity* to show how accurately defect-free samples were classified, *sensitivity* to show how accurately defective samples were classified, and *accuracy* as the correct classification rate of all samples. They are defined as

$$\begin{cases} spec. = \frac{N_t \cap N_g}{N_g} \times 100\% \\ sens. = \frac{P_t \cap P_g}{P_g} \times 100\% \\ accu. = \frac{N_t \cap N_g + P_t \cap P_g}{N_g + P_g} \times 100\%, \end{cases} \tag{22}$$

where P is the number of defective samples, N is the number of defect-free samples, and the subscripts t and g denote the results by testing and groundtruth, respectively. The foreground is set to occupy 50 percent of the whole image to allow the sensitivity and specificity measures have equal weights.

We compared channel separation-based gray-level texems for novelty detection on these color collage images. The PCA-based decomposition showed a significant improvement over correlated RGB channels with an overall accuracy of 84.7 percent compared to 79.1 percent (see Table 1). Gray-level texem analysis in image eigenchannels appear to be a plausible approach to perform color analysis with relatively economic computational complexity. However, the full color texem model, which modeled interchannel and intrachannel interactions simultaneously, further improved the performance with an overall detection accuracy of 90.9 percent, with 91.2 percent sensitivity and 90.6 percent specificity. Example segmentations (without any postprocessing) using gray-level texems in RGB channels, in eigenchannels, and color texems are shown in the third, fourth, and last row of Fig. 10, respectively.

We also compared the proposed method against a Gabor filtering-based novelty detection method [3] and a nonfiltering method using LBPs [9]. The LBP coefficients were extracted from each RGB color band. The estimation of the range of coefficient distributions for defect-free samples and the novelty detection procedures were the same as that described in Section 3.5. We found that LBP performs very poorly, but a more sophisticated classifier may improve the performance. Gabor filters have been widely used in defect detection, see [3], [4] as typical examples. The work by Escofet et al. [3], referred to here as Escofet's method, is the most comparable to ours, as it is 1) performed in a novelty detection framework and 2) uses the same defect fusion scheme across the scales. Thus, following Escofet's method to perform novelty detection on the synthetic image collages, the images

TABLE 1
Novelty Detection Comparison

Test No.	Escofet's Method			RGB Channel			Eigenchannel			Color Texem Method		
	spec.	**sens.**	**accu.**	**spec.**	**sens.**	**accu.**	**spec.**	**sens.**	**accu.**	**spec.**	**sens.**	**accu.**
1	95.6	82.7	89.2	81.7	100	90.7	82.0	100	90.9	91.9	99.9	95.9
2	96.9	83.7	90.3	80.7	100	90.2	80.8	100	90.3	84.4	100	92.1
3	96.1	61.5	79.0	87.6	99.9	93.7	82.4	100	91.1	91.1	99.8	95.4
4	98.0	53.1	75.8	94.3	97.2	95.7	93.9	95.7	94.8	97.0	92.9	95.0
5	98.8	1.5	50.7	87.3	30.7	59.3	77.9	99.6	88.6	92.1	98.8	95.4
6	96.6	70.0	83.4	76.6	100	88.2	77.8	100	88.8	96.3	98.6	97.4
7	98.9	26.8	63.2	96.0	93.4	94.7	90.1	98.6	94.3	98.6	79.4	89.0
8	91.4	74.4	83.0	87.8	97.7	92.7	85.6	95.3	90.4	89.6	99.8	94.7
9	90.8	49.0	70.1	85.5	52.0	68.9	76.1	100	87.9	86.4	100	93.1
10	94.3	7.2	51.2	92.2	25.2	59.1	77.8	99.2	88.4	92.8	99.6	96.2
11	94.6	8.6	52.1	89.1	33.6	61.6	80.3	97.2	88.6	96.3	90.8	93.6
12	86.9	44.0	65.7	82.5	88.4	85.4	79.5	97.7	88.5	88.4	98.8	93.5
13	96.8	71.0	84.0	93.5	47.8	70.9	93.0	49.0	71.2	91.0	91.9	91.5
14	90.7	95.2	93.0	80.9	99.9	90.3	81.1	100	90.5	82.5	100	91.1
15	98.4	27.2	63.2	98.7	55.3	77.2	98.3	74.8	86.7	96.5	76.3	86.5
16	95.5	43.0	69.3	84.5	78.1	81.3	86.5	92.7	89.6	96.3	71.2	83.8
17	80.0	56.5	68.2	75.1	60.8	67.9	62.3	87.9	73.8	83.5	98.7	91.1
18	73.9	60.4	67.2	64.9	69.5	67.2	60.9	91.9	74.8	83.9	96.5	90.2
19	84.9	52.0	68.4	75.1	60.0	67.5	57.0	87.4	72.2	90.4	71.3	80.9
20	94.4	52.0	73.2	83.9	91.8	87.8	85.4	90.0	87.7	95.1	88.8	91.9
21	94.0	48.9	71.6	78.6	97.3	87.8	88.4	98.4	93.4	95.8	75.9	85.9
22	95.8	23.4	60.0	88.5	49.8	69.4	79.5	76.3	77.9	92.2	72.0	82.2
23	97.1	35.1	66.5	98.2	44.5	71.6	96.6	34.8	66.0	93.6	67.8	80.9
24	89.4	46.4	67.9	60.6	69.8	65.2	64.5	86.8	75.7	81.6	98.1	89.8
25	82.6	92.9	87.7	58.7	100	79.4	64.8	99.9	82.3	88.3	100	93.9
26	94.5	55.3	74.9	84.1	91.6	87.9	76.5	94.2	85.3	94.3	92.2	93.2
27	93.9	36.5	65.2	73.2	87.8	80.5	64.7	99.9	82.3	85.9	98.9	92.4
28	81.2	55.3	68.3	74.5	88.3	81.4	65.7	94.6	80.1	82.0	95.2	88.6
Overall	92.2	50.5	**71.5**	82.7	75.4	**79.1**	78.9	90.8	**84.7**	90.6	91.2	**90.9**

were filtered through a set of 16 Gabor filters, comprising four orientations and four scales. The texture features were extracted from filtering responses. Feature distributions of defect-free samples were then used for novelty detection. The same logical process was used to combine defect candidates across the scales. An overall detection accuracy of 71.5 percent was obtained by Escofet's method; a result significantly lower than the proposed method (see Table 1). Example results are shown in the second row of Fig. 10. Again, a more complicated and traditional classification scheme and more finely tuned parameters may improve Escofet's Gabor filter-based performance; however, the training stage and the detection process will inevitably become lengthier and more difficult to control. This demonstrates the good generalization and yet good discriminant abilities of texems.

There are two important parameters in the texem model, the size of texems and the number of the texems. In theory, the size of the texems is arbitrary. Thus, it can easily cover all the necessary spatial frequency range. However, for the sake of computational simplicity, a window size of 5×5 or 7×7 across all scales generally suffices. The number of texems can be automatically determined using model order selection methods, such as MDL, though they are usually computationally expensive. We used 12 texems in each scale for over 1,000 tile images and collages and found reasonable, consistent performance for novelty detection.

5 Conclusions

We presented an automatic defect detection and localization algorithm for randomly textured surfaces. The proposed method only trained on a small number of defect-free samples with the aid of novel texture exemplars, i.e., texems, which contain primitive textural information of a given image or set of images. We demonstrated their derivation for gray-level and color images using two different mixture models and applied them within a novelty detection framework to localize defects on new images. While we present this work with respect to ceramic tiles, the proposed method should be suitable to other flat textured surfaces, such as textiles, where defect detection can be viewed as texture abnormality detection.

In terms of industrial inspection as an application area, the computational needs of the method are somewhat demanding for a real-time factory installation, however, they are not far off. We shall investigate various avenues, including dedicated hardware and software, to achieve a rate of around 1-2 (tile) surfaces per second which is an acceptable tile industry norm.

Acknowledgments

The research work was funded by EC project G1RD-CT-2002-00783 MONOTONE and X. Xie was partly funded by the ORSAS, Universities UK.

References

[1] F. Cohen, Z. Fan, and S. Attali, "Automated Inspection of Textile Fabrics Using Textural Models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 803-809, Aug. 1991.

[2] C. Boukouvalas, J. Kittler, R. Marik, and M. Petrou, "Automatic Color Grading of Ceramic Tiles Using Machine Vision," *IEEE Trans. Industrial Electronics*, vol. 44, no. 1, pp. 132-135, 1997.

[3] J. Escofet, R. Navarro, M. Millán, and J. Pladellorens, "Detection of Local Defects in Textile Webs Using Gabor Filters," *Optical Eng.,* vol. 37, no. 8, pp. 2297-2307, 1998.
[4] A. Kumar and G. Pang, "Defect Detection in Textured Materials Using Gabor Filters," *IEEE Trans. Industrial Applications,* vol. 38, no. 2, pp. 425-440, 2002.
[5] O. Silvén, M. Niskanen, and H. Kauppinen, "Wood Inspection with Nonsupervised Clustering," *Machine Vision and Applications,* vol. 13, pp. 275-285, 2003.
[6] T. Randen and J. Husøy, "Filtering for Texture Classification: A Comparative Study," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 21, no. 4, pp. 291-310, Apr. 1999.
[7] J. Kittler, R. Marik, M. Mirmehdi, M. Petrou, and J. Song, "Detection of Defects in Colour Texture Surfaces," *IAPR Proc. Machine Vision Applications,* pp. 558-567, 1994.
[8] M. Varma and A. Zisserman, "Texture Classification: Are Filter Banks Necessary?" *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 691-698, 2003.
[9] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 7, pp. 971-987, July 2002.
[10] N. Jojic, B. Frey, and A. Kannan, "Epitomic Analysis of Appearance and Shape," *Proc. IEEE Int'l Conf. Computer Vision,* pp. 34-42, 2003.
[11] "VisTex Texture Database," MIT MediaLab, 1995, http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html.
[12] A. Kumar, "Neural Network Based Detection of Local Textile Defects," *Pattern Recognition,* vol. 36, no. 7, pp. 1645-1659, 2003.
[13] B. Julesz, "Textons, the Element of Texture Perception and Their Interactions," *Nature,* vol. 290, pp. 91-97, 1981.
[14] S. Zhu, C. Guo, Y. Wang, and Z. Xu, "What Are Textons?" *Int'l J. Computer Vision,* vol. 62, no. 1-2, pp. 121-143, 2005.
[15] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials Using Three-Dimensional Textons," *Int'l J. Computer Vision,* vol. 43, no. 1, pp. 29-44, 2001.
[16] X. Xie and M. Mirmehdi, "Texture Exemplars for Defect Detection on Random Textures," *Proc. Int'l Conf. Advances in Pattern Recognition,* pp. 404-413, 2005.
[17] K. Song, J. Kittler, and M. Petrou, "Defect Detection in Random Colour Textures," *Image and Vision Computing,* vol. 14, pp. 667-683, 1996.
[18] T. Mäenpää, J. Viertola, and M. Pietikäinen, "Optimizing Color and Texture Features for Real-Time Visual Inspection," *Pattern Analysis and Applications,* vol. 6, no. 3, pp. 169-175, 2003.
[19] D. Tsai, C. Lin, and K. Huang, "Defect Detection in Coloured Texture Surfaces Using Gabor Filters," *Imaging Science J.,* vol. 53, no. 1, pp. 27-37, 2005.
[20] F. Lumbreras, J. Serrat, R. Baldrich, M. Vanrell, and J. Villanueva, "Color Texture Recognition through Multiresolution Features," *Quality Control by Artificial Vision,* vol. 1, pp. 114-121, 2001.
[21] A. Jain and G. Healey, "A Multiscale Representation Inclusing Opponent Color Features for Texture Recognition," *IEEE Trans. Image Processing,* vol. 7, no. 1, pp. 124-128, 1998.
[22] X. Xie and M. Mirmehdi, "Localising Surface Defects in Random Colour Textures Using Multiscale Texem Analysis in Image Eigenchannels," *Proc. IEEE Int'l Conf. Image Processing,* vol. 3, pp. 1124-1127, 2005.
[23] B. Silverman, *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, 1986.
[24] A. Monadjemi, M. Mirmehdi, and B. Thomas, "Restructured Eigenfilter Matching for Novelty Detection in Random Textures," *Proc. BMVA British Machine Vision Conf.,* pp. 637-646, 2004.

Xianghua Xie received the MSc (with commendation) and PhD degrees in computer science in 2002 and 2006, respectively, from the University of Bristol, United Kingdom. He is currently a research assistant in the Department of Computer Science, University of Bristol, United Kingdom. His current research interests are video analysis, texture analysis, image segmentation, surface inspection, deformable models, and historical document analysis. He is a member of the BMVA and the IEEE.

Majid Mirmehdi received the BSc (Hons) and PhD degrees in computer science in 1985 and 1991, respectively, from the City University, London. He has worked both in industry and in academia. He is currently a Reader in the Department of Computer Science at the University of Bristol, United Kingdom. His research interests include texture analysis, color image analysis, medical imaging, and document recognition. He has more than 100 refereed conference and journal publications in these areas. He is an associate editor of the *Pattern Analysis and Applications Journal*. He is a member of the IEE, IEEE, and a member and the chairman of the British Machine Vision Association.