Texture Classification Using an Invariant Texture Representation and a Tree Matching Kernel

Somkid Soottitantawat[1] and Surapong Auwatanamongkol[2]

**[1] School of Applied Statistics,
National Institute of Development Administration, Bangkok, Thailand, 10240**

**[2] School of Applied Statistics,
National Institute of Development Administration, Bangkok, Thailand, 10240**

Abstract
In this paper, an alternative approach for texture classification using an invariant texture representation and a tree matching kernel is proposed. The approach identifies regions of a given texture image using a Speed-Up Robust Feature or SURF descriptor. The regions of all training texture images are then clustered into a tree of non-uniformly shaped regions based on the distribution of them using a hierarchical k-means algorithm. The tree structure forms a tree of keypoints to be used for determining similarities between two texture images. The similarity is computed based on an approximate matching kernel called a tree matching kernel. Finally, Support Vector Machines (SVMs) with the tree matching kernels are constructed to classify textures. The performances of the proposed method are evaluated through experiments performed on textures from the Brodatz and UIUCTex datasets. The experiment results demonstrate that the proposed approach is quite robust to scale, rotation, deformation and viewpoint changes and achieves higher classification rates than some other well known methods.
Keywords: *Texture Classification, Tree of Keypoints, SURF Descriptor, Support Vector Machines, Tree Matching Kernel, Hierarchical K-means Clustering.*

1. Introduction

In the visual world, textures can be regarded as the visual appearances of surfaces and may be perceived as being directional or non-directional, smooth or rough, coarse or fine, regular or irregular, etc. Several textures are observed on both artificial and natural objects and scenes. The surface characteristics of textures can be used to recognize objects in an image, to segment an image and to understand an image [27]. So, textures play an important role in many image analyses, computer vision and pattern recognition tasks. However, environment and illumination conditions can affect the appearance of textures, and so complicate the tasks. Textures in real images can vary in scale, brightness, and rotation as imaging conditions change. Therefore, to enable texture analysis in real images, texture representation should be invariant to imaging conditions such as non-rigid deformation, viewpoint, scaling and lighting. A brief review of the invariant texture analysis methods is presented in [17].

The goal in this research is to perform texture classification that is robust to the mentioned environment and illumination conditions. A texture representation, which is invariant to the conditions, along with the new classification method is proposed. Our approach consists of the following steps: (1) Constructing an invariant texture representation step that consists of feature detection and then extraction of texture regions, which are invariant to the conditions due to both geometric and photometric transformations. We propose that Speeded Up Robust Features (SURF) is to be used as local invariant descriptors for the texture regions. (2) Building a tree of keypoints (regions) step performed hierarchical k-means clustering on all regions of all textures in the training set. A tree of keypoints is then constructed from the hierarchical k-means clustering. (3) Texture modeling step builds multi-SVM classifiers with a one-against-all tournament approach to classifying texture images. The tree matching kernel is used in the SVMs and utilizes the tree of keypoints to determine the similarity between two textures. Figure 1 shows a sketch of the process for the proposed approach.

The organization of this paper is as follows. Related works are reviewed in section 2; the proposed approach is described in section 3; experiments and results, designed to evaluate the effectiveness of the proposed approach, are presented in section 4 and finally conclusions are given in section 5.

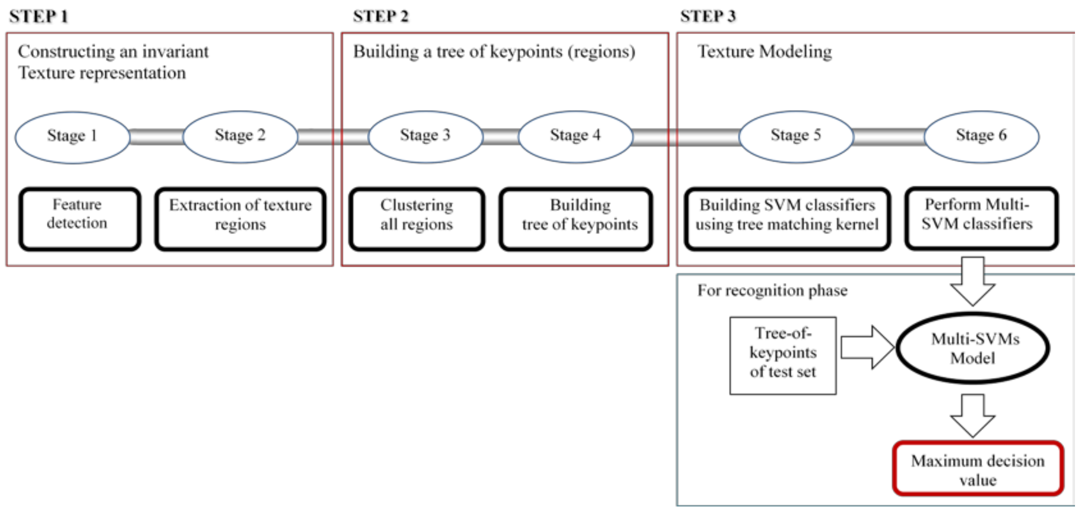
STEP 1
Constructing an invariant Texture representation
Stage 1
Stage 2
Feature detection
Extraction of texture regions
STEP 2
Building a tree of keypoints (regions)
Stage 3
Stage 4
Clustering all regions
Building tree of keypoints
STEP 3
Texture Modeling
Stage 5
Stage 6
Building SVM classifiers using tree matching kernel
Perform Multi-SVM classifiers
For recognition phase
Tree-of-keypoints of test set
Multi-SVMs Model
Maximum decision value

Fig 1. The framework for the proposed approach.

2. Related Works

In this section, some related works on texture classification are reviewed. Lazebnik et al. [29] proposed a sparse texture representation using local affine regions for recognizing textured surfaces under a wide range of transformations, including viewpoint changes and image nonrigid deformations. Features descriptors of each texture are clustered using standard k-means to form the texture signature $\{(m_1, u_1), (m_2, u_2),\ldots(m_k,u_k)\}$, where k is the number of clusters, m_i is the center of the i^{th} cluster, and u_i is the weight of the cluster. They used Earth Mover's Distance (EMD) to measure the similarity between two signatures. During the classification stage, nearest-neighbor classification with EMD was used to classify texture images. Mellor et al. [23] described a method based on invariant combinations of linear filters. Unlike Lazebnik's methods, they proposed a novel family of filters, which provides scale invariance, resulting in a texture description invariant to local changes in orientation, contrast and scale and robust to local skew. The χ^2 similarity measure is used on histograms derived from their filter responses. Recently, Qin et al. [22] presented a novel approach to classify texture collections using an unsupervised approach. Given image database, they extracted a set of invariant descriptors from each image and the descriptors of all the images were clustered to form keypoints. A texture image can be represented by a bag-of-keypoints. Probabilistic Latent Semantic Indexing (PLSI) and Non-negative Matrix Factorization (NMF) were used for the unsupervised texture classification.

Several local features for texture representation were proposed in [9], [14], [16], [20], [30]. The local features are distinctive, invariant to many kinds of geometric and photometric transformation. They are suitable for image classification and have been used in many applications, e.g. object recognition [1], [3], scene recognition [31], robot localization [2], and texture classification [25], [26], [29], [30]. A review of other local features can be seen in [16].

The main drawbacks of the local feature methods are that different local feature methods can produce different numbers of feature vectors and generate no obvious structural information about the vectors, for example, no ordering among the vectors is produced. Thus, to overcome these problems, similar feature vectors can be clustered to create a designated number of representative feature vectors. For example, Boughorbel et al. [28] proposed that the centroids of the computed clusters represented virtual features for images. Csurka et al. [12] used a similar clustering technique to construct bags of keypoints. Each bag is represented by a bin of a histogram. Hence, features of two images can be compared through the matching of their feature histograms.

3. Proposed Approach

3.1 Region Formation

For texture analysis, region formation is the first essential task. To form regions in a texture, we propose the recently developed Speeded-Up Robust Features (SURF) [14], [10], to use as the local descriptor. SURF has already been used in a few real world applications [2], [13], [15]. SURF is very well suited for tasks in object detection, object recognition and image retrieval [13]. SURF possesses more discriminative power than other features such as SIFT [9] and it can be computed more efficiently and yields a lower dimensional feature descriptor resulting in faster matching [2]. SURF can be computed as follows.

3.1.1 Interesting Point Detection

To compute SURF, interesting points must be detected. The detection is performed using the Hessian-matrix approximation. Given a point x = (x, y) in an image I, the Hessian matrix H(x, σ) in x at scale σ is defined as follows:

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix}, \qquad (1)$$

Where $L_{xx}(x,\sigma)$ is the convolution of the Gaussian second order derivative $\frac{\partial^2}{\partial x^2} g(\sigma)$ with the image I in point x, and similarly for $L_{xy}(x,\sigma)$ and $L_{yy}(x,\sigma)$.

SURF approximates the second order Gaussians derivatives with box filters. Image convolutions with these box filters can be computed rapidly by using integral images. The location and the scale of the interesting points are selected by relying on the determinant of the Hessian matrix. Interesting points are localized in scale and image space by applying a non-maximum suppression in a 3x3x3 neighborhood. Then, the local maxima found of the approximated Hessian matrix determinant are interpolated in the scale and image space. For more details, please refer to [14].

3.1.2 Constructing Region Descriptors

This stage consists of two steps. First, SURF constructs a circular region around the detected interesting points in order to assign a unique orientation to the former. The orientation is computed using Haar wavelets responses in both x and y directions. The Haar wavelets can be quickly computed via integral images. The dominant orientation is estimated and included as the interesting point information. Next, SURF descriptors are constructed by extracting square regions around these interesting points. These are oriented in the directions assigned in the previous step. The windows are split up in 4x4 sub-regions in order to retain some spatial information. In each sub-region, Haar wavelets responses in horizontal and vertical directions (d_x and d_y) are summed up over each sub-region. Moreover, the absolute values $|d_x|$ and $|d_y|$ are summed in order to obtain information about the polarity of the image intensity changes. Therefore, the underlying intensity pattern of each sub-region is described by a vector $V = [\sum d_x , \sum d_y, \sum|d_x|, \sum|d_y|]$. The resulting descriptor vector for all 4x4 sub-regions is of length 64, giving the standard SURF descriptor, SURF-64. An important characteristic of SURF is its fast extraction process due to the fast integral process of images and the fast non-maximum suppression algorithm. Figure 2 shows the output of the SURF regions on two sample images.

3.2 Clustering Regions

After the SURF regions of all the training texture images have been identified, the SURF regions are clustered to form a hierarchical tree. This stage is a one-time process performed before building the SVMs. The similar hierarchical tree structures have been proposed to handle descriptor matching, for instance, searching the tree for images [21] kd-tree [9] and metric trees [5], [8].

Many computer vision algorithms, including our proposed algorithm, require searching as the closest data point of a given data on a high-dimensional space [24]. The nearest neighbor search has been used to find the closest point. However, the performance of the search varies depending on properties of the datasets, such as dimensionality, correlation, clustering characteristics, and size. A better approach is to use metric tree based index methods, proposed by [8] for a fast nearest neighbor search in very large databases. This method is based on searching for the closest leaf node in a hierarchical k-means tree starting from the root down to the leaf.

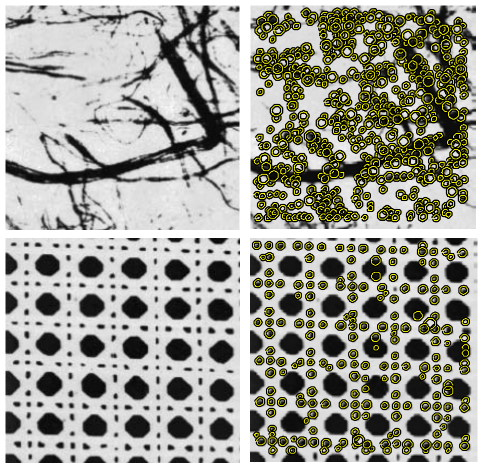

Fig. 2 Outputs of the SURF regions on two sample texture images. Left: original images, right: regions found by SURF descriptors.

The hierarchical k-mean clustering starts with a k-means process (the k-means process is described by Algorithm1) on all SURF regions of all texture images in the training dataset. Next, the same k-means process is recursively applied to each cluster derived from the previous clustering, i.e. recursively splitting each cluster into k subclusters. The recursion is stopped when the number of SURF regions in a cluster is smaller than k. The hierarchical clustering forms a corresponding tree where each leaf corresponds to each of the SURF regions, representing the keypoints of all the texture images in the training dataset. Each node in the tree is represented by the node index, the centroid and the radius (maximum distance from the centroid) of its corresponding cluster. The centroid and the radius of a node will be used mainly to determine whether the nearest leaf node for a given data is located inside the subtree rooted at the node.

The main advantage of a k-means algorithm is its computational simplicity, which is convenient for large data sets. Its time complexity is O(NkId) when clustering N data points of d dimensions with k centers and I iterations. However, the centers of clusters for k-means algorithms are often initialized randomly, which may result in different clustering solution from run to run. Several methods [5] were proposed to overcome this problem. One simple technique that we adopt to address this problem is to perform k-means for multiple runs and select the solution from the run that yields the minimum sum of squared errors (SSE). The second problem with the k-means algorithm is that empty clusters can be obtained if no SURF regions are allocated to a cluster during the assignment step. If this happens, the empty cluster will be given a new centroid selected randomly from the members of the cluster that has the highest SSE. This will split the cluster and reduce the overall SSE of the clustering.

```
Algorithm1:  k-means process used in this paper.
1: Define the number of clusters k
2: Define the number of runs m
3: For i=1 to m.
4:    Select k SURF regions randomly as initial centroids.
5:    While (centroids do change)
6:         Form k clusters by assigning each SURF region to
           its closest centroid.
7:         Recompute the centroid of each cluster.
8:   End
9:  Record the k cluster centroids for the ith run
10: End
11: Select the set of k clusters of the run with minimum
    SSE.
```

The computational cost for building the tree structure using hierarchical k-means clustering on a given training set is O(LNkId), where L is the number of levels in the tree. The building time can be reduced significantly by limiting the number of iterations in the k-means clustering stage instead of running it until its convergence is reached [24]. Moreover, the branching factor k can affect the precision of finding the closest data in the tree as well as the building time of the tree. A higher branching factor has proven to give better precision but also a higher building time [24]. Therefore, there is a tradeoff between the precision and the building time. Figure 3 shows an example hierarchical k-means tree built from 3 texture images (with total of 19 SURF regions).

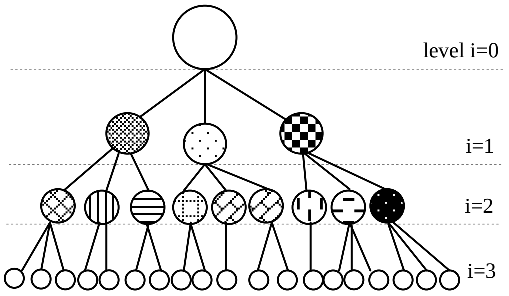

Fig. 3 Three levels of a hierarchical tree with branch factor = 3 populated to represent 3 training images with 19 SURF regions.

3.3 Tree Matching Kernel

After all the SURF regions of all the training texture images have been clustered yielding a tree of keypoints T. A tree of keypoints for any texture image can be constructed by performing hierarchical search (from root to a leaf of T) to identify which clusters of nodes at each level of T that each SURF region of the texture image belongs to i.e. determining the cluster with the closest centroid. All nodes of T, that any SURF region of the image belongs to, form the tree of keypoints of the image. Clearly, the tree of keypoints must be a subgraph of T. Let T_1 and T_2 be the trees-of-keypoints of any two texture images. To determine the similarity between the two images, the tree matching kernel score of T_1 and T_2 will be determined instead. The tree matching kernel score, inspired by Grauman et al. [18], [19], can be defined as the sum of the weighted number of SURF matches found at all common nodes of the two trees. The number of SURF matches found at a common node is defined to be the minimum between the numbers of SURF regions belonging to the two images and located within the cluster of the node minus the numbers of SURF matches already counted by all of their child nodes. The sum of these weighted counts yields the approximate tree matching kernel score.

Let $c_{ij}(T_1)$ be the number of SURF regions in T_1 located inside the cluster of the jth node of level i, $c_{ij}(T_2)$ be the number of SURF regions in T_2 located inside the cluster of the jth node of level i. The tree matching kernel score is computed as follows:

$$Sim(T_1, T_2) = \sum_{i=m-1}^{1} \sum_{j=1}^{k^i} (w_{ij} - p_{ij}) \, min\left(c_{ij}(T_1), c_{ij}(T_2)\right). \tag{2}$$

Where k is the number of sub-clusters for each cluster, m is the highest number of the levels of two trees, w_{ij} is the weight associated with the jth node at level i of T, p_{ij} is the weight associated with the parent node of the jth node at level i of T. The subtraction part weighted by p_{ij} corresponds to the weighted number of SURF matches that need to be subtracted from the count of the node's parent. The weight of the j th node at level i of T are set to be equal to exp (-2*d_{ij}) where d_{ij} is the radius of the cluster of the node. This is according to the fact that the SURF matches found at a lower level (larger i) with a smaller value of radius should contribute more significantly to the similarity score than those of the higher levels with higher radius values. This weighting scheme meets the condition required in [18], [19] and makes the kernel, defined by the equation (2), a Mercer kernel (i.e., a symmetric positive definite kernel), which can be used with kernel based methods such as the Support Vector Machine (SVM). The matching scores are normalized by the number of SURFs of the two images in order not to favor larger images. Note that the sum in the equation (2) starts with the index i = m-1 because there will be no SURF region matches at the leaf level i = m. The time required to compute a tree matching kernel score between two trees-of-keypoints is O(NL). A tree matching kernel score between a testing texture image and a training texture image representing a kernel vector can be determined in the same way as described above.

3.4 Texture Classification with SVMs

Several approaches have been proposed to generalize the binary SVM classifier to solve problems where multi-classes apply. To apply SVM for multi-class classifications, two main approaches have been suggested. The first approach is called "one against one". In this approach, a series of binary classifiers is applied to each pair of classes, and the most commonly computed class is assigned to the given object. This method requires m(m-1)/2 classifiers to be built. The second approach, which requires less classifiers to be built, is called "one against rest". This approach requires m binary classifiers where the i th classifier is built as the samples in the i th class are treated as positive examples and the rest as negative examples. In the recognition phase, a testing texture is presented to all m classifiers and is labeled according to the highest decision value among the m classifiers. Because of simplicity, the one-against-rest approach was adopt to build the multi-class SVM classifier in the experiments [7].

4. Experiments

Experiments were carried out on textures images from Brodatz [35] and UIUCTex [31] database. The experiments are similar to those in [29]. Two performance measures [22] are used to evaluate our approach, as follows.
The confusion matrix (CM):

$$CM_{ij} = \frac{|\{I_a \in Test_j : f(I_a)=i\}|}{|Test_j|} \qquad (3)$$

$Test_j$ is the set of testing images which belong to class j, and $f(I_a)$ is the class label which obtains the highest classifier score from the multi-class classifier for a given image I_a.

The mean classification rate (CR):

$$CR = \frac{\sum_{j=1}^{N} |Test_j| CM_{ij}}{\sum_{j=1}^{N} |Test_j|} \qquad (4)$$

All experiments are carried on a PC Intel Core 2 Quad CPU Q6600, with a clock rate of 2.4 GHz and 4.0 GB of RAM. The classifiers are implemented using Visual C++ 2003 and Matlab (R2008a).

4.1 Datasets

The first dataset from Brodatz is a collection of texture images that features significant inter-class variability, but no geometric transformations between members of the same class. The dataset consists of 111 images. Following the same procedure as [29], we form classes by partitioning each image into nine non-overlapping fragments, for a total of 999 images. Fragment resolution is 215 × 215 pixels. The training set of Brodatz consists of randomly selected 333 images (3 images/class) and the other 666 images (6 images/class) are used for testing.
The second dataset from UIUCTex consists of 40 examples for each of the 25 texture types. The database is publicly available at http://www-cvr.ai.uiuc.edu/ponce_grp. The resolution of each sample is 640×480 pixels. This database includes surfaces whose texture is mainly due to albedo variations (e.g., wood and marble), 3D shape (e.g., gravel and fur), as well as a mixture of both (e.g., carpet and brick). Significant viewpoint changes and scale differences are present within each class, and illumination conditions are uncontrolled [29]. The training set of UIUCTex contains 250 images (10 images/class) and the rest of the 750 images (30 images/class) are used for testing.

Figure 4 and 5 show several example texture images from the two datasets.

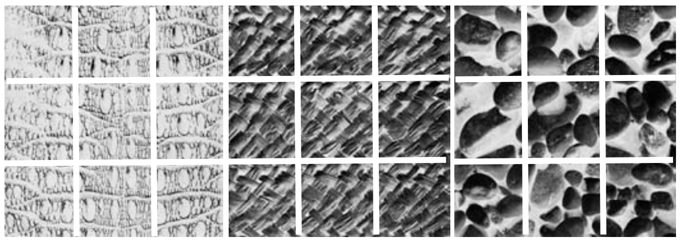

Fig. 4 Examples of three classes of textures from Brodatz dataset.

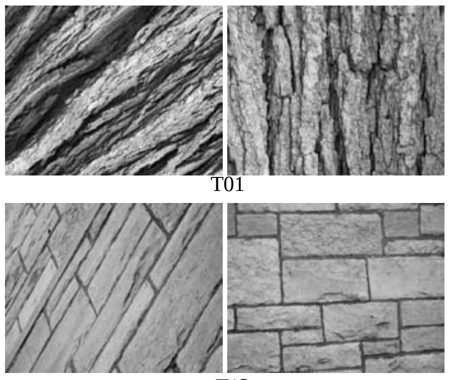

Fig. 5 Examples of three classes of textures from UIUCTex database.

4.2 Region Formation

About 220,493 SURF regions were extracted from the Brodatz training images. The average number of SURF regions extracted per image is about 663. Also, about 268,646 SURF regions were extracted from UIUCTex training images. The average number of SURF regions extracted per image is about 1,075.

4.3 Clustering SURF Regions to build a tree of keypoints

At this stage, a hierarchical tree of 64-dimensional SURF regions for each of the two datasets was built using hierarchical k-mean tree algorithm. The branching factor of 10 is used for the k-mean tree construction. Up to ten iterations of k-means were run and the tree building process can take a few hours. The tree, built from the Brodatz training set, contains 327,567 nodes (220,493 leaf nodes and 107,074 internal nodes) with d = 64, the number of levels = 10 and k = 10, while the tree, built from UIUCTex training set, contains 395,866 nodes (268,646 leaf nodes and 127,220 internal nodes) with d = 64, the number levels = 8 and k = 10.

4.4 SVM Classification

After the tree of keypoints was built, multi-class SVM classifiers were built using one-against-rest scheme. The regularization constant C was set at 1, 10, 100, 1000, and 10000. Each C was evaluated using 5-fold cross validation method on the training set only. The SVM classifiers with the parameter C=10 gave the best classification rates.

Once the training of SVM classifiers had been completed, the test samples were fed into the classifiers and the predicted class IDs of the test samples were compared with the true labels. The results are reported in terms of the mean classification rates.

4.5 Results

In the first experiment, all 111 textures in the Brodatz dataset were used. According to Xu et al. [11], the 111 textures can be grouped into 3 types based on the degree of regularity or type of structure of the textures. The first type consists of six textures which have degrees of regularity more than 5 (highly regular type). The second type consists of fifty five textures which have degrees of regularity between 4 and 5 (regular type). Finally, the third type consists of fifty textures which have a degree of regularity between 3 and 4 (irregular type). The results of the experiments performed on the three types of the textures are summarized in Table 1. It can be seen that the textures of the highly regular type achieve a classification rate of 100 percent, the textures of regular type achieve a mean classification rate of 90.91 percent, and the textures of irregular type achieve a mean classification rate of 87.85 percent. The mean classification rate for all textures is 90.84 percent. The results are better than those of the three previous works, see Table 3 for the comparisons. Figure 6 shows three textures that were classified incorrectly because the training and testing examples are highly non-homogeneous.

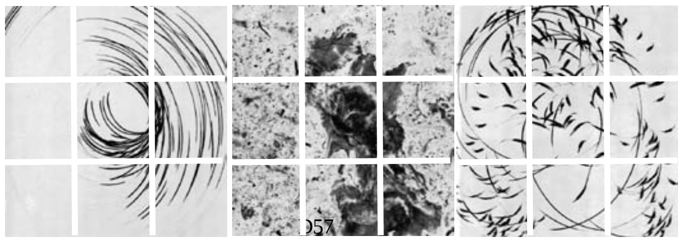
Fig.6 Examples of misclassified textures from Brodatz dataset.

In the second experiment, a set of experiments were conducted on the UIUCTex dataset with an increasing number of classes. The numbers of classes used in the set of experiments are 3, 8 and 25, the same as [22]. The results from the experiments are summarized in Table 2. For the experiments performed on the three classes of T23-T25 and the eight classes of T18-T25, Qin et al. [22] method achieved the best performance. However, as the number of classes increases to 25, the classification task becomes more challenging. The proposed methods achieved the mean classification rate of 93.60 percent for the case of 25 classes, which is better than the previous works of [22], [29].

Table 1 : The mean classification rate results of the Brodatz dataset

Categories	***Proposed Method***
D6, D20, D33, D51, D52, D101 (6 classes - highly regular type)	100.00
D2, D3, D4, D5, D10, D14, D16, D17, D19, D21, D23, D24, D27, D32, D34, D35, D36, D37, D38, D39, D41, D42, D43, D45, D46, D49, D50, D53, D54, D55, D56, D59, D63, D64, D65, D66, D68, D72, D75, D79, D81, D83, D84, D85, D86, D92, D94, D96, D98, D100, D102, D103, D105, D106, D110 (55 classes - regular type)	90.91
D1, D7, D8, D9, D11, D12, D13, D15, D18 D22, D25, D26, D28, D29, D30, D31, D40, D44, D47, D48, D57, D58, D60, D61, D62, D67, D69, D70, D71, D73, D74, D76, D77, D78, D80, D82, D87, D88, D89, D90, D91, D93, D95, D97, D99, D104, D107, D108, D109, D111 (50 classes - irregular type)	87.85
D1 - D111 (111 classes – all types)	90.84

Table 2 : The mean classification rate results of the UIUCTex dataset

Categories	***Lazebnik [29]***	***Qin [22]***	***Proposed***
T23 – T25 (3 classes are fabric texture)	95.89	100	94.44
T18 – T25 (8 classes are fabric, wall paper, fur and two carpets)	93.70	98.40	93.75
T1-T25(25 classes)	92.61	83.00	93.60

The proposed approach has shown to give high classification rates on both datasets comprising of scale, rotation, deformation and viewpoint changes hence it has proven to be robust to these environment and imaging conditions. The experiment results also show that the kernel-based learning method can yield better texture classification rates than those of the nearest-neighbor classification method with EMD [29]. Furthermore, they show that the texture representations based on the distribution of the local features like SURF are more effective than the combinations of linear filters [23] for texture classifications.

Table 3 : The mean classification rates of the four different methods for the two texture databases.

Methods	***Brodatz*** 3 trainings per class	***UIUCTex*** 10 trainings per class
Lazebnik et al. [29]	88.15	92.61
Mellor et al.[23]	89.71	92.84
Qin et al. [22]	64.46	83.00
Proposed	90.84	93.60

5. Conclusions

In this paper, a novel texture classification method is proposed. The method uses SURF descriptors to represent the key points of a texture. All the key points of all training texture images are then clustered by a hierarchical k-means algorithm yielding a tree structure called a tree of keypoints. The tree is built to facilitate the evaluation of the tree matching kernel used by multi-class SVMs to classify a given texture. Results from experiments conducted on textures from two databases, Brodatz and UIUCTex, have shown that SURF descriptors are invariant to many kinds of geometric and photometric transformation such as scale, rotation, deformation and viewpoint changes, and can be used effectively with the tree matching kernel to achieve high classification rates on multi-class SVMs.

Acknowledgments

This research was partially funded by the National Institute of Development Administration (NIDA). We also would like to thank the School of Applied Statistics of NIDA for providing us supports on this research work.

References

[1] A.C. Berg, T.L.Berg, and J. Malik, “Shape matching and object recognition using low distortion correspondence”, in Proceedings of the IEEE International Conference Computer Vision Pattern Recognition, Vol. 1, 2005, pp. 26-33.
[2] A.C. Murillo, J.J. Guerrero and C. Sagues, “SURF features for efficient robot localization with omnidirectional images”,in IEEE International Conference on Robotics and Automation, Rome-Italy, 2007, pp. 3901-3907.
[3] A.E. Johnson and M. Hebert, “Using spin images for efficient object recognition in clustered 3D scenes”, in IEEE Transaction Pattern Analysis Machine Intelligence, Vol.21, No.5, 1999, pp.433-449.
[4] B. Caputo and L. Jie, “A Performance Evaluation of Exact and Approximate Match Kernels for Object Recognition”, in Electronic Letters on Computer Vision and Image Analysis, Vol. 8, No. 3, 2009, pp. 15-26.

[5] B. Leibe, K. Mikolajczyk, and B. Schiele, “Effcient clustering and matching for object class recognition”, in Proceedings of British Machine Vision Conference, 2006.
[6] B. Sch¨olkopf and A. Smola. Learning with Kernels. The MIT Press, Cambridge, MA, 2002.
[7] C.H. Hsu and C.J.Lin, “A Practical Guide to Support Vector Classification”, Department of Computer Science, National Taiwan University, Taipei, Taiwan, 2008.
[8] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary tree", in IEEE Conference on Computer Vision and Pattern Recognition. New York, 2006, pp. 2161-2168.
[9] D.G. Lowe, “Distinctive image features from scale-invariant keypoints”, in International Journal of Computer Vision, Vol. 60, No. 2, 2004, pp.91-110.
[10] E. Christopher, "Notes on the open SURF Library", 2009.
[11] F. Xu and Y.J. Zhang, “Evaluation and comparison of texture descriptors proposed in MPEG-7”, in Journal of Visual Communication and Image Representation, Vol.17 (August), 2006, pp.701-716.
[12] G. Csurka, C. Dance, L. Fan, J. Willamowsky and C. Bray, “Visual categorization with bags of keypoint”, in Proceedings of the ECCV International Workshop on Statistical Learning in Computer Science, 2004.
[13] H. Bay, B. Fasel, and L.V. Gool, “Interactive museum guide: Fast and robust recognition of museum objects”, in The First International workshop on mobile vision, 2006.
[14] H. Bay, T. Tuytelaars and L. V. Gool, “Surf: Speeded up robust features”, in The ninth European Conference on Computer Vision, 2006, http://www.vision.ee.ethz.ch/surf/.
[15] H. Zuo, W. Hu, O. Wu, Y. Chen, and G. Luo, “Detecting Image Spam Using Local Invariant Features and Pyramid Match Kernel”, WWW 2009, April 20–24, 2009, pp. 1187-1188.
[16] J. Li and N.M. Allinson, “A comprehensive review of current local features for computer vision”, in Neurocomputing,Vol. 71, 2008, pp.1771– 1787.
[17] J. Zhang, and T. Tan, “Brief review of invariant texture analysis methods”, in Pattern Recognition. Vol 35, 2002, pp.735–747.
[18] K. Grauman and T. Darrell, "Approximate Correspondences in High Dimensions", in Advances in Neural Information Proceeding System, 2006.
[19] K. Grauman and T.Darrell, "The pyramid match kernel: Discriminative classification with sets of image features", in ICCV, 2005.
[20] K. Mikolajczyk and C. Schmid, “A performance evaluation of local descriptors”, in IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol.27, No 10, 2005, pp.1615–1630.
[21] K. Mikolajczyk and J. Matas, “Improving Descriptors for Fast Tree Matching by Optimal Linear Projection”, in iccv, 2007, pp. 1-8.
[22] L. Qin , Q. Zheng, S. Jiang, Q. Huang and W. Gao, "Unsupervised texture classification: Automatically discover and classify texture patterns", in Image and Vision Computing, Vol.26 , 2008, pp.647-656.
[23] M. Mellor, H. Byung-Woo and M. Brady, "Locally rotation, contrast, and scale invariant descriptors for texture analysis", in IEEE Transactions on pattern analysis and machine intelligence, 2007.
[24] M. Muja and D. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration", 2009.
[25] M. Varma and A. Zisserman, “A statistical approach to texture classification from single images”, in International Journal of Computer Vision, Vol.62, 2005, pp.61-81.
[26] M. Varma and A. Zisserman, “Classifying images of materials: achieving viewpoint and illumination independence”, in Proceeding of the the European Conference on Computer Vision, Lecture notes in Computer Science, Vol.2352, No.3, 2002, pp.255-271.
[27] M.Turtinen, “Learning and Recognizing Texture Characteristics using Local Binary Patterns”, Ph.D. thesis, University of Oulu, 2007.
[28] S. Boughorbel, J.P. Tarel and N. Boujemaa, “The intermediate matching kernel for image local features”, in International Joint Conference onNeural Networks, 2005, pp.889-894.
[29] S. Lazebnik, C. Schmid and J. Ponce, "A Sparse Texture Representation using Local Affine Regions", in IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol27 (August), 2005, pp.1265-1278.
[30] S. Lazebnik, C. Schmid and J. Ponce, "Affine-Invariant Local Descriptors and Neighborhood Statistics for Texture Recognition", in Proceedings of the Ninth IEEE International Conference on Computer Vision, 2003.
[31] S. Lazebnik, “Local, Semi-local and Global Models for Texture, Object and Scene Recognition”, Ph.D. thesis, University of Illinois at Urbana-Champaign, 2006.
[32] S.B. Kotsiantis and P.E. Pintelas, “Recent Advances in Clustering: A Brief Survey”, in WSEAS Transactions on Information Science, 2004.
[33] S.J. Taylor and N. Cristianini, "Kernel Methods for Pattern Analysis. Cambridge University Press, 2004.
[34] V. Vapnik and C.Cortes, “Support-Vector-Networks”, in Machine Leaning, 20, 1995.
[35] http://www.ux.uis.no/~tranden/brodatz.html

Somkid Soottitantawat is a PhD candidate at the School of Applied Statistics, National Institute of Development Administration, Bangkok, Thailand. She received a Master’s Degree in Computer Science from the School of Applied Statistics, National Institute of Development Administration and completed her Bachelor’s Degree in Education in General Science/Computer Education from Chulalongkorn University, Bangkok, Thailand. Her research interests are in Pattern Recognition and Data Mining.

Surapong Auwatanamongkol is an associate professor at the School of Applied Statistics, National Institute of Development Administration, Bangkok, Thailand. He received a Ph.D. degree in Computer Science from Southern Methodist University, USA, a Master’s degree in Computer Science from Georgia Institute of Technology, USA, and a Bachelor’s degree in Electrical Engineering from Chulalongkorn University, Bangkok, Thailand. His fields of research include Pattern Recognition, Data Mining, Evolutionary Computation, Parallel and Distributed Computing. He has published several papers in various international journals and conferences.