Unsupervised Self-Organizing Texture Descriptor

Marco Vanetti & Ignazio Gallo & Angelo Nodari
Dipartimento di Scienze Teoriche e Applicate, Universita' degli studi dell'Insubria

We propose a local texture descriptor based on a pyramidal composition of Self Organizing Map (SOM). As with the SOM model, our visual descriptor presents two operational steps: a first unsupervised learning phase and a second mapping phase involving a dimensionality reduction of the input data. During the first step a large number of image patches, including different classes of textures, are presented to the model. At the end of the learning process the neural weights on each layer of the SOM pyramid will contain good prototypes of the patches used in training at different level of detail. During the mapping phase a new texture patch is presented to the model and, by using a winner take all principle, a winner neuron is selected and its 2D spatial location is used to describe the input patch. Exploiting the topological order of the SOM, two different texture descriptions can be compared using the common Euclidean distance. In the experimental section we show that a simple clustering algorithm like K-means, applied to the local descriptor responses, is able to segment complex texture mosaics with very good results, even in difficult areas like boundaries which separate two different textures.

1 INTRODUCTION

In order to automatically produce a description of a natural image, a fundamental role is played by texture descriptors. Images representing real objects often do not exhibit regions with uniform intensities but, due to the physical properties of real surfaces, they contain frequent variations of brightness which form certain repeated patterns called visual texture or more simply: texture.

Over the years, many problems involving texture analysis have been proposed, the main ones are listed below. *Texture classification* aims to produce a classification map of an image where each uniform textured region is identified by a particular texture class which belong to. *Texture segmentation* is focused on finding texture boundaries even if it is not possible to classify each region. Figure 1 shows an example of unsupervised texture segmentation obtained applying a K-means clustering to the local descriptor proposed in this paper. *Texture synthesis* is used for image compression applications and in computer graphics, with the aim of rendering object surfaces which need to be as realistic as possible. Finally, with *shape from texture*, we aim to extract the three-dimensional shape of objects in a scene using texture information, distorted by imaging process and the perceptive projection (Tuceryan and Jain 1998). Despite the final purpose is quite different, each of the problems listed above requires a texture descriptor, which becomes an essential tool in many applications.

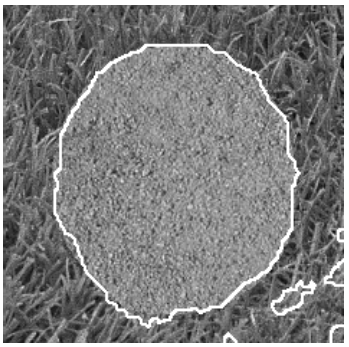


Figure 1: Segmentation between two areas with different textures obtained using the proposed descriptor. Segmentation border is depicted with a white line.

A common denominator for most successful texture descriptors is that the textured image is submitted to a linear transform, filter or filter bank. Methods using this common scheme are called *filtering approaches*, and received an extensive survey in (Randen and Husy 1999), a comparative study where various filtering approaches have been evaluated within a texture classification framework.

An important issue that characterizes most of the filtering approaches is the selection of an appropriate filter bank. The most commonly are the Gabor filters, inspired by experiments with animal visual systems (Daugman 1980), and signal-processing based filters, designed with desirable band-pass properties in the

Fourier domain (Bovik 1991). However, the optimal choice of a filter bank is often influenced by the particular application and may require a lot of experimentation.

A simple and promising strategy to combine multiple filters, resulting in a compact description of the texture, is the spectral histogram, first suggested in psychophysical studies on texture modeling (Bergen and Adelson 1988) and later used for texture analysis and synthesis (Heeger and Bergen 1995) (Zhu, Wu, and Mumford 1997). Spectral histogram is based on the assumption that all of the spatial information characterizing a texture image can be captured in the first order statistics of an appropriately chosen set of linear filter outputs. Spectral histogram can also be used as a local descriptor, using and appropriately sized integration window, in this case the descriptor is often called Local Spectral Histogram (LSH).

LSH is a powerful local texture descriptor, able to seize general aspects of texture as well as non-texture regions. In (Liu and Wang 2006) a LSH based on a filter bank based composed of eight filter (pixel intensity, two gradient filters, two-scales Laplacian of Gaussian and three Gabor filters) has been used for texture segmentation, attaining the state of the art in the field of unsupervised texture segmentation methods based on filter bank.

The main drawback of LSH is that it requires large integration windows to extract meaningful texture features from the image, this results in a poor reliability of the description along texture boundaries. A solution to the aforementioned problem has been proposed in (Liu and Wang 2006) by using asymmetric windows and a refined probability model based on seed points automatically extracted from the segmented regions.

Some work tried to generalize the methods based on multichannel filtering by training: in a supervised fashion, a neural network in order to find a minimal set of specific filters. These methods may delegate to the neural network the dual task of extracting features and classifying textures (Jain and Karu 1996) (Kim, Jung, Park, and Kim 2002), or perform separately the second phase using a most powerful classifiers such as Support Vector Machines (Melendez, Gironés, and Puig 2011).

In this paper we propose an innovative texture descriptor, based on a pyramidal composition of Self Organizing Map (SOM) (Kohonen 1990), that is capable of extract a powerful local texture feature from an image without requiring any supervision or handcrafted filter bank. The pyramidal nature of the approach perform an image analysis using pixel contexts which become progressively larger. At each layer of the pyramid, only the most relevant feature for the particular context will be extracted by the SOM and the image

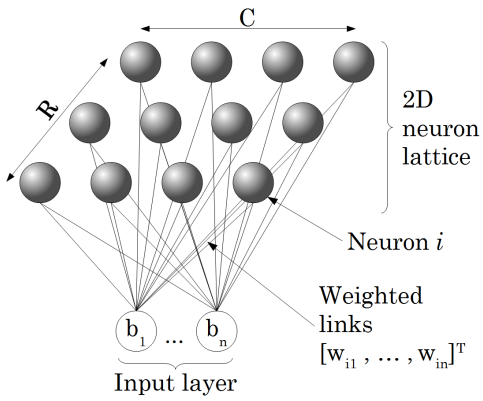


Figure 2: Two dimensional SOM.

will be "redrawn", for the upper layer of the pyramid, deprived of redundant information.

Considering the complexity of the non-linear dimensionality reduction introduced by each SOM, the validity of the proposed approach is difficult to prove analytically. However, to evaluate the method, we used a very simple unsupervised texture segmentation strategy, based on a K-means clustering algorithm. In this way we highlight the goodness of the proposed descriptor, excluding contributions attributable to a supervised machine learning method or a post-processing/refinement phase.

This work is organized as follow. In Section 2 is described the proposed texture descriptor, based on the SOM unsupervised learning method. In Section 3 are shown and discussed experimental results, using the K-means clustering algorithm for texture segmentation. Finally, Section 4 gives the conclusions.

2 PROPOSED DESCRIPTOR

As discussed in Section 1, the proposed descriptor is based on a pyramidal composition of SOM. SOM is an artificial neural network first proposed by Teuvo Kohonen in early 1981, able to produce, without supervision, a spatially organized internal representation of various features of input signals (Kohonen 1990). As depicted in Figure 2 we employ a two dimensional SOM, composed of a 2D lattice of neurons, each of which is fully connected to the input layer through a series of weighted links $\mathbf{w_i} = [w_{i1}, w_{i2}, \ldots, w_{in}]^T$ where $0 \leq w_{ij} \leq 1$, i is the index of a single neuron and n is the dimension of the input data.

The proposed method involves an initial training phase, where a large number of training vectors are presented to the network and the neural weights are updated according to a particular rule. Training vectors are extracted from the input image using an overlapping sliding window approach, the window shall henceforth be called *context window*.

A training vector is composed of the intensity/brightness values of pixels within the context window and denoted by $\mathbf{b} = [b_1, b_2, \ldots, b_n]^T$ where

$0 \leq b_j \leq 1$ and n is the total number of pixels. The input image will be properly border-padded[1] so that, in total, $H \cdot W$ training vectors will be extracted, where H is the height and W is the width in pixels of the image.

Let us describe now how the unsupervised learning happens. By presenting a new input vector to the SOM, a single neuron k will be activated in a particular location of the network, we call this neuron "winner". The winner selection occurs by satisfying the following identity:

$$\|\mathbf{b} - \mathbf{w_k}\| = \min_i \{\|\mathbf{b} - \mathbf{w_i}\|\} \quad (1)$$

The step just described is followed by the update of the weights in the neighborhood of the winner neuron. The update is described by the following equation:

$$\mathbf{w_i}(t+1) = \mathbf{w_i}(t) + \alpha h_{ik}[\mathbf{b}(t) - \mathbf{w_i}(t)] \quad (2)$$

Referring to the equation 2, α is a scalar constant called *adaptation gain* or *learning rate*, $0 < \alpha < 1$, and the function h_{ik} is a scalar "bell curve" kernel function defined as:

$$h_{ik} = \exp\left(-\frac{\|\mathbf{q_i} - \mathbf{q_k}\|^2}{2\sigma^2}\right) \quad (3)$$

where the vectors $\mathbf{q_k} = [q_{kr}, q_{kc}]^T$ and $\mathbf{q_i} = [q_{ir}, q_{ic}]^T$ denote the coordinates of the winning neuron k and the neuron to be updated i, and the r in subscript is in reference to the row number inside the neuron lattice and c refers to the column number. Scalars σ and α can be chosen as time-variable functions, monotonically decreasing with iterations.

At the end of the training phase, the spatial location, represented by the coordinates of each neuron in the network, corresponds to a particular domain or feature of input signal patterns (Kohonen 1990) and the weights of each neuron contain a good prototype of the input patches (Gersho and Gray 1992). By using a small window of local context around each pixel, the proposed method tries to discover local salient features from the image.

Once the SOM is trained, its neural weights w can be treated as constant values, and employing the same sliding window approach used during the previous training phase, we can map each pixel of the input image in the two dimensional Euclidean space of the activated neurons within the SOM lattice. Using again Equation 1, we thus generate a new image with two channels, the first dependent on the row number of the winner neurons and the second on the column number. The new image, that we call *remapped image*, can be formally calculated from the input image $I_0(x, y)$ using:

$$I_1(x, y) = \left[\frac{q_{kr}}{R}, \frac{q_{kc}}{C}\right]^T \quad (4)$$

where, for each pattern $\mathbf{b}$ centered on the pixel (x, y) of the input image I_0, the winner neuron k is found using Equation 1. R and C refer to the size, in rows and columns, of the neural lattice. Note that I_1 is a two-channels image, therefore each pixel contains two intensity values.

Since each pattern is extracted by simply concatenating the values of the pixels within the context window, the input image can have an arbitrary number of channels. In this way the learning process and the subsequent remapping can be performed iteratively on more layers, following a pyramidal approach. For each layer of the pyramid, the parameters involved are the context window size, the size of the SOM and the learning parameters σ and α. Figure 3 graphically explains the remapping strategy just described.

In the following section we show a sample configuration based on three layer and applied to some real and synthetic images.

Table 1: Parameter configuration of the model proposed in the experiments.

	Context Window size (pixels)	SOM size (neurons)	σ	α
Layer 1	2×2	10×10	1	0.1
Layer 2	4×4	15×15	3	0.01
Layer 3	8×8	15×15	3	0.01

3 EXPERIMENTS

To test the proposed method, we used a configuration based on a three-layers pyramid, with a context window of 2×2 pixels for the layer that operates on the input image. The second layer involves a 4×4 context window and the third layer further doubles the window size. Table 1 collects the parameters used for experiments. The SOM parameters were chosen primarily taking into account the size of the input pattern. Note that a 15×15 sized SOM contains about twice the neurons of a 10×10 sized SOM.

Using the architecture described above, the overall training/mapping pipeline can be schematized as follows: *(1)* the first layer is trained using the input image I_0, *(2)* the first layer performs the remapping, *(3)* the second layer is trained using the remapped image provided by the first layer I_1, *(4)* the second layer performs the remapping, *(5)* the third layer is trained using the remapped image provided by the second layer I_2, *(6)* the third layer performs the final remapping providing the output image I_3.

To evaluate the proposed texture descriptor, we em-

[1] We used a "mirror" border padding strategy, as explained in (Szeliski 2010).

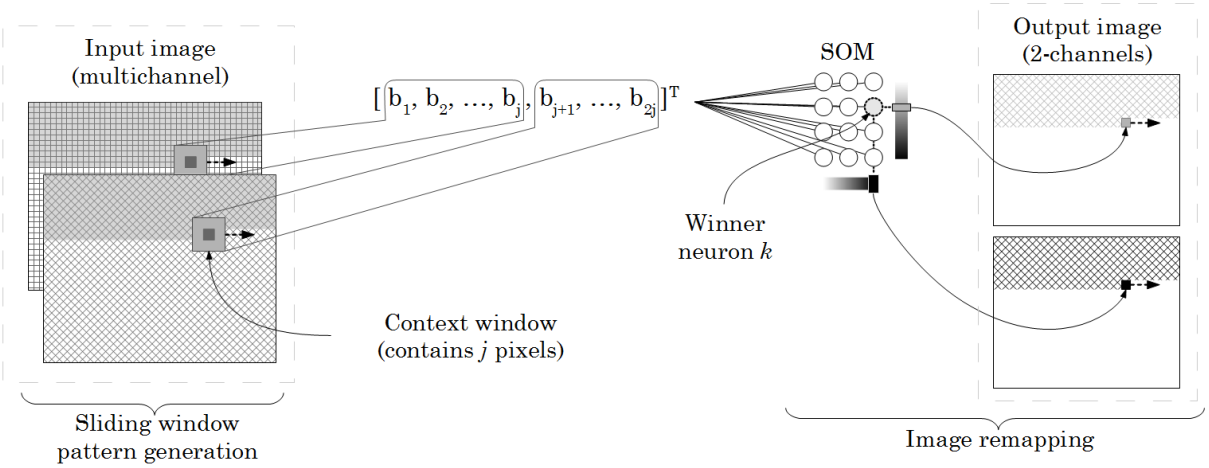


Figure 3: Schema representing the image remapping strategy, performed exploiting the SOM competitive behavior.

ployed a simple segmentation strategy based on the K-means algorithm. Pattern set was created by concatenating pixel intensities from the last layer to their normalized coordinates, in order to create a raw topological constraint. Formally:

$$P = \bigcup_{x=0}^{W} \bigcup_{y=0}^{H} \left(I_3(x,y) || \left[\frac{x}{W}, \frac{y}{H} \right]^T \right) \quad (5)$$

Figure 4(a) depict a 5-texture mosaic used in (Liu and Wang 2006) to test an unsupervised segmentation method. The authors have obtained a 3.90% error using a LSH texture descriptor with a 19×19 pixels integration window and a filter bank composed of one intensity filter, two gradient filters, two-scales Laplacian of Gaussian and three Gabor filters. By applying a refined probability model to localize the region boundaries, they have reduced the error to 0.95%. The proposed method performs with an error of $1,83\%$, Figure 4(e) shows the resulting segmentation, the ground truth segmentation and a map that highlights wrong segmented pixels. The reported errors refer to the percentage of pixels incorrectly segmented.

As can be seen in Figure 4(d), the local texture description is smoothed near the boundary between two different textures, this is due to the context window size. Despite this fact, the local texture description is still reliable, since the K-means clustering, using the common Euclidean distance as a metric of distance, is able to recognize and separate with a good precision the two textures along the boundary. Considering that we do not use any "handcrafted" feaure/filter and our method does not rely on a specific border localization technique, the result obtained is very challenging.

Figure 5(a) is another 5-texture mosaic used in (Karoui, Fablet, Boucher, Pieczynski, and Augustin 2008) to test a supervised approach based on empirical marginal distributions of local texture features like co-occurrence distributions, Gabor magnitude distributions, etc. They have obtained a 3.1% error while our error is 5.14%. The two results are comparable, but the problem studied here is essentially more difficult, given the unsupervised nature of the feature extraction process and of the image segmentation.

Table 2: Segmentation results obtained using different subsets of the proposed 3-layers architecture.

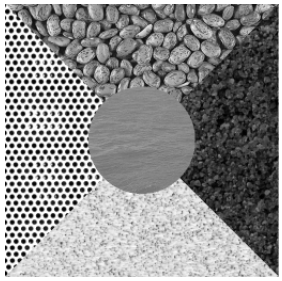
	Figure 4(a) error (%)	Figure 5(a) error (%)
Only Intensity	29.45	52.45
Only Layer 1	17.39	19.98
Only Layer 2	28.97	23.77
Only Layer 3	29.17	35.95
Layer 1 + 2	10.14	9.22
Layer 1 + 3	8.68	4.86
All layers	1.83	5.14

(Awate, Tasdizen, and Whitaker 2006) proposed the 2-class mosaic in Figure 1 as a challenging image since it show two textures that are both irregular and have similar means and gradient-magnitudes. No numerical result is available in their paper, but the results that we obtained is qualitatively comparable with that shown in (Awate, Tasdizen, and Whitaker 2006), obtained using an unsupervised approach that minimizes the entropy-based metric on the probability density functions of image neighborhoods.

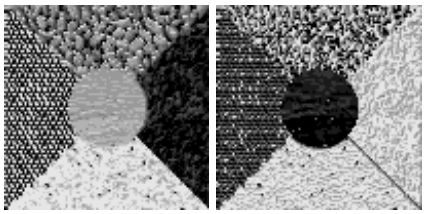
All the proposed texture mosaics are composed of textures taken from the Brodatz album (Brodatz 1966) and the *Vision Texture Dataset*[2].

To investigate the contribution of each layer in the overall process, we evaluated the method by excluding different subsets of layers. The worst result is obtained by applying the K-means clustering directly on

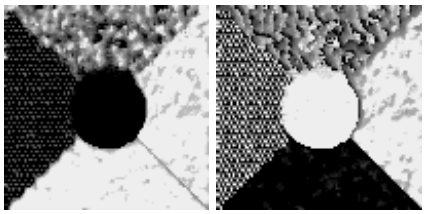
[2]The Vision Texture Dataset is provided by the MIT Vision and Modeling Group, `http://vismod.media.mit.edu/`



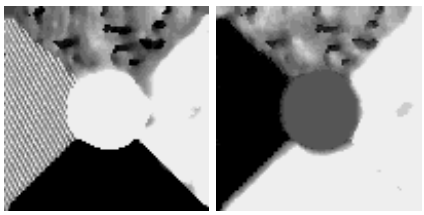
(a)



(b)



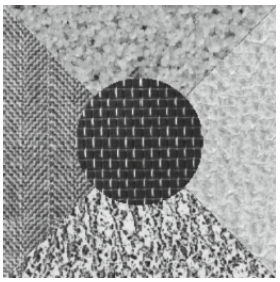
(c)



(d)

(e)

Figure 4: (a) Input image, a mosaic composed of 5 textures. (b)(c)(d) Two channels images remapped by the first (b), second (c) and third (d) layers of the SOM pyramid. (e) From the left, the final segmentation, the ground truth segmentation and the segmentation error map. Wrong pixels are shown in black.



(a)

(b)

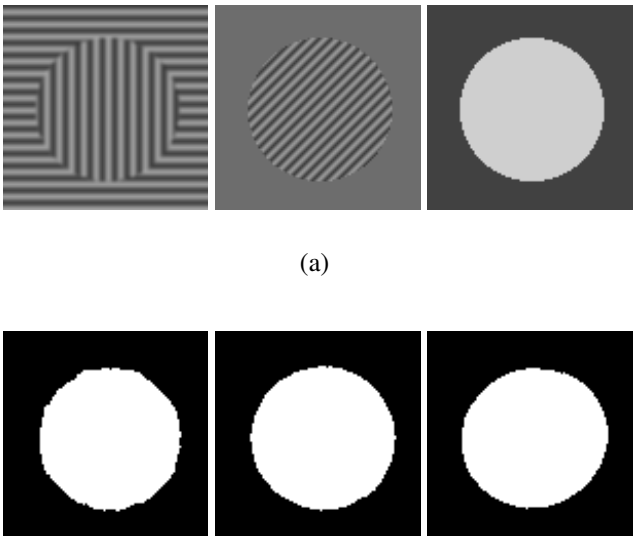
Figure 5: (a) Texture mosaic composed of 5 textures. (b) From the left, the final segmentation, the ground truth segmentation and the error map. Wrong pixels are shown in black.

the intensity levels of the input image, while the best configuration involves all the three levels. Results in Table 2 show that the strength of the descriptor lies primarily in the pyramidal approach and, using a shallow architecture, the segmentation accuracy suddenly decreases.

As a final experiment we tested the method on three synthetically generated images. The first image in Figure 6(a), is composed by two wave-gradient regions with two different orientations. The mean intensity is constant within the two regions and the only discriminant information is the orientation of the wave pattern. The second image shows two regions, one with a wave-gradient texture and one with a solid color. Also in this case both regions have the same mean intensity. The third image contains two non-textured areas with different intensities. As can be seen in Figure 6(b), in all three cases, the proposed method has been able to distinguish the two regions almost perfectly. This result experimentally prove that the proposed descriptor can handle, at the same time, texture regions as well as non-texture regions.

Before being processed, the input image is scaled to 100×100 pixels. Under these conditions, the computational time required to process an image is about 15 seconds[3], where more than half of the time is spent training the third layer. This is due to the large patches (8×8 pixels) used on the layer, which generate big

[3]Results were obtained using an unoptimized, single C# thread, on an Intel(R) Core(TM) i5 mobile CPU at 2.30Ghz.



(a)

(b)

Figure 6: (a) Three synthetically created texture-non texture mosaics. (b) Segmentation results obtained with the proposed method.

training patterns.

4 CONCLUSIONS

In this paper we have presented a new texture descriptor that is able to characterize textured as well as non-textured regions with high accuracy. The potential of the method lies in its independence from a feature bank and its ability to automatically extract, without supervision, salient information using only small image patches. The descriptor exploits the important topological ordering property of the SOM allowing a smoothed and reliable image description even in areas with strong transitions, such as the boundary between two different textures or two different colors.

Comparison with other state of the art methods shows that our solution gives comparable results even without a directly managing of difficult areas, such as texture boundaries. The provided three-layers configuration offers good results on images of different types. Future research is focused in improving the segmentation accuracy and defining a method to automatically find an optimal parameters setting.

REFERENCES

Awate, S. P., T. Tasdizen, and R. T. Whitaker (2006). Unsupervised texture segmentation with nonparametric neighborhood statistics. In *European Conference on Computer Vision*, pp. 494–507.

Bergen, J. R. and E. H. Adelson (1988). Early vision and texture perception. *Nature 333*, 363–364.

Bovik, A. C. (1991). Analysis of multichannel narrow-band filters for image texture segmentation. *IEEE Transactions on Signal Processing 39*, 2025–2043.

Brodatz, P. (1966). Textures: a photographic album for artists and designers.

Daugman, J. (1980). Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Research 20*, 847–856.

Gersho, A. and R. M. Gray (1992). Vector quantization and signal compression.

Heeger, D. J. and J. R. Bergen (1995). Pyramid-based texture analysis/synthesis. In *Annual Conference on Computer Graphics*, pp. 229–238.

Jain, A. K. and K. Karu (1996). Learning texture discrimination masks. *IEEE Transactions on Pattern Analysis and Machine Intelligence 18*, 195–205.

Karoui, I., R. Fablet, J.-M. Boucher, W. Pieczynski, and J.-M. Augustin (2008). Fusion of textural statistics using a similarity measure: application to texture recognition and segmentation. *Pattern Analysis and Applications 11*, 425–434.

Kim, K. I., K. Jung, S. H. Park, and H. J. Kim (2002). Support vector machines for texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 1542–1550.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE 78*, 1464–1480.

Liu, X. and D. Wang (2006). Image and texture segmentation using local spectral histograms. *IEEE Transactions on Image Processing 15*, 3066–3077.

Melendez, J., X. Gironés, and D. Puig (2011). Supervised texture segmentation through a multilevel pixel-based classifier based on specifically designed filters. In *ICIP*, pp. 2869–2872.

Randen, T. and J. H. Husy (1999). Filtering for texture classification: A comparative study. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*, 291–310.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*.

Tuceryan, M. and A. K. Jain (1998). Texture analysis. *The Handbook of Pattern Recognition and Computer Vision*.

Zhu, S. C., Y. N. Wu, and D. Mumford (1997). Minimax entropy principle and its application to texture modeling. *Neural Computation 9*, 1627–1660.