Unsupervised image segmentation with the self-organizing map and statistical methods

Jukka Iivarinen[a] and Ari Visa[b]

[a]Helsinki University of Technology, Laboratory of Computer and Information Science
P.O. Box 2200, FIN-02015 HUT, Finland

[b]Lappeenranta University of Technology, Department of Information Technology
P.O. Box 20, FIN-53851 Lappeenranta, Finland

ABSTRACT

In this paper a special type of image segmentation, a two-class segmentation, is considered. Defect detection in quality control applications is a typical two-class problem. The main idea in this paper is to train the two-class classifier with fault-free samples that is an unexpected approach. The reason is that defects are rare and expensive. The proposed defect detection is based on the following idea: an unknown sample is classified as a defect if it differs enough from the estimated prototypes of fault-free samples. The self-organizing map is used to estimate these prototypes. Surface images are used to demonstrate the proposed image segmentation procedure.

Keywords: image segmentation, unsupervised segmentation, self-organizing map, statistical self-organizing map, density estimation

1. INTRODUCTION

Image segmentation is a fundamental problem in image analysis. The goal of image segmentation is to partition an image into a set of regions which are uniform and homogeneous. In the best case these regions will have a strong correlation with objects in the image. For example, in industrial inspection the goal of segmentation is often to separate objects from the background. In remote sensing applications we may wish to segment a satellite image into different type of regions (sea, land, clouds, etc.).

Clustering is an unsupervised classification technique which is widely used in pattern recognition and in image segmentation.[1–3] In pattern recognition clustering refers to the process which partitions a set of patterns into subsets called clusters. In image segmentation a set of features, a feature vector, is extracted for each image pixel. The extracted feature vectors are partitioned into clusters in feature space. Each cluster can be identified to represent a specific class of image features.

Since late 80's neural network techniques have been proposed for clustering. The most popular one has been the self-organizing map (SOM).[4] In image segmentation the SOM is used to cluster unknown image data (or feature vectors). The map is trained with a large number of unclassified feature vectors which are extracted from images. The types of image regions or even the number of image regions need not to be known in advance. These properties are important in many practical applications. The SOM has been applied e.g. in speech recognition, image segmentation, texture analysis, remote sensing, process control, financial analysis, robot control, and document analysis. Several authors have proposed the SOM to image segmentation. Lampinen and Oja used the SOM to learn the parameters of an AR model for each texture in an image.[5] Visa[6] developed a surface classification method based on a SOM, a learning vector quantization, and the co-occurrence matrix. The same idea was extended to textured image segmentation.[7] The SOM and the co-occurrence matrices have also been used in segmentation and classification of satellite images,[8–10] and in surface inspection.[11] Other SOM-based unsupervised segmentation schemes can be found in Refs. 12–21.

In this paper a special type of image segmentation, a two-class segmentation, is considered. A typical two-class problem is defect detection of surfaces (see a review on texture defect detection in Ref. 22) where a classifier is

Further author information–
J.I.(correspondence): E-mail: Jukka.Iivarinen@hut.fi
A.V.: E-mail: Ari.Visa@lut.fi

typically trained with samples taken only from a fault-free surface.[23–25] This results in many benefits. A tedious collection of defects is not needed and appearance of new defects do not result in retraining of a classifier. Features extracted from fault-free samples are usually assumed to follow a multidimensional Gaussian distribution so that the decision function can be formulated. The proposed segmentation scheme does not have this restriction; the distribution can be of any shape. The self-organizing map is proposed to be used in estimating the distribution of features. Then a new decision function is formulated for each SOM unit. The resulting network structure is called the statistical self-organizing map and it is described in section 2. In section 3 a general framework for unsupervised image segmentation is formulated. Defect detection of a textured surface is used as a case study. Some final comments and concluding remarks are given in section 4.

2. THE STATISTICAL SELF-ORGANIZING MAP

A new extension to the original SOM algorithm, the statistical SOM, is introduced next (some earlier work can be found in Ref. 11,26) with a short description of the original SOM algorithm. The motivation behind the statistical SOM was to develop a simple and fast algorithm that allows us to determine if a given unknown sample belongs to a distribution of previously encountered samples. No Gaussian assumptions of the shape of the distribution have to be made; instead, the distribution can be of any shape. The SOM is used to estimate this distribution, and then a new decision function is formulated to each SOM unit.

2.1. The Original SOM Algorithm

The self-organizing map (SOM)[4] is an unsupervised, self-organizing neural network. It makes a topology-preserving mapping from a high-dimensional feature space onto a two-dimensional lattice. The mapping is nonlinear. The topology preservation means that the spatial organization of input data is preserved in the mapping. Thus the SOM is a practical tool for e.g. clustering, interpretation, and visualization of high-dimensional data.

The SOM consists of a set of map units that are topologically arranged to a (usually) two-dimensional grid. The SOM is a competitive network. For each feature vector $\mathbf{f}(t)$ the best matching map unit is presented as an output. The best matching unit c is determined by comparing the input vector $\mathbf{f}(t)$ and the weight vectors $\mathbf{m}_i$ of each map unit,

$$||\mathbf{f}(t) - \mathbf{m}_c|| = \min_i \{||\mathbf{f}(t) - \mathbf{m}_i||\}. \tag{1}$$

The usual distance measure is the Euclidean one,

$$||\mathbf{f}(t) - \mathbf{m}_i|| = \sqrt{\sum_j (f_j(t) - m_{ij})^2}. \tag{2}$$

The map units partition feature space into small regions called the Voronoi regions (Figure 1). The resulting partition is called the Voronoi tessellation. For each map unit i the Voronoi set F_i is defined to include those input feature vectors $\mathbf{f}(t)$ which belong to the Voronoi region of the unit i (Figure 1), i.e., the Voronoi set F_i consists of those input vectors $\mathbf{f}(t)$ for which the unit i is the best-matching unit,

$$F_i = \{\mathbf{f}(t) \mid i = \arg\min_k \{||\mathbf{f}(t) - \mathbf{m}_k||\} \ \forall\, t\}. \tag{3}$$

Training the SOM The SOM is trained with a large number of unclassified feature vectors. During the training phase the weight vectors of the best matching unit and its neighboring units are updated. The neighborhood $N_c(t)$ is defined to include all the units inside a certain distance from the best matching unit. The training process can then be expressed as

$$\mathbf{m}_i(t+1) = \begin{cases} \mathbf{m}_i(t) + \alpha(t)\big[\mathbf{f}(t) - \mathbf{m}_i(t)\big] & \text{if } i \in N_c(t) \\ \mathbf{m}_i(t) & \text{if } i \notin N_c(t), \end{cases} \tag{4}$$

where $\alpha(t)$ is the learning rate function ($0 < \alpha(t) < 1$) and $N_c(t)$ is the neighborhood function. $\alpha(t)$ decreases monotonically to zero during training. The radius of $N_c(t)$ also decreases monotonically during training. In the beginning of training $N_c(t)$ is quite wide (even more than half the diameter of the map) and it shrinks with time until it contains only the nearest neighbors of the best matching unit c. The classical K-means clustering algorithm can be obtained from the learning algorithm of the SOM just by setting $N_c(t) = \{c\}$ for all t.

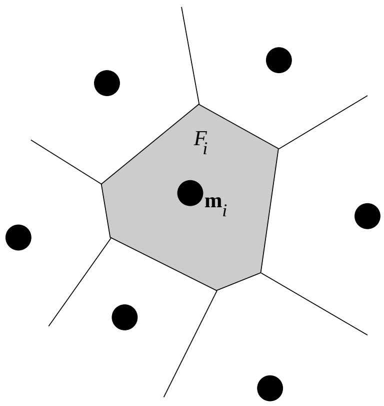


Figure 1. The Voronoi region of the map unit i (in a two-dimensional case).

2.2. The Statistical SOM Algorithm

The basic idea in the statistical SOM is to fit a one-dimensional density function to each component plane of the Voronoi set of each map unit, and then determine a confidence interval to each one-dimensional density function. The simplest density function is a uniform density function, which is actually a good choice. Other possibilities are the Gaussian density function and the gamma function. Experiments with these three density functions showed the superiority of the uniform density function. However, if more accuracy is needed, the best choice is to approximate the densities with the Parzen estimation procedure.[27] It is a tedious procedure but fortunately it has to be made only once after the map is trained. In the following it is first shown how the confidence intervals are determined when assuming a uniform distribution, and then this is extended to the Parzen approximations of densities.

Let us assume a SOM with weight vectors $\mathbf{m}_i$ that is trained with a large number of feature vectors $\mathbf{f}(t)$. All feature vectors should originate from the same distribution. Each map unit $\mathbf{m}_i$ has a Voronoi region and a Voronoi set F_i (Eq. (3)). The weight vector $\mathbf{m}_i$ is replaced with a new weight vector $\mathbf{m'}_i$ and a confidence interval vector $\mathbf{d'}_i$. These determine the dimensions ($\mathbf{d'}_i$) and the center ($\mathbf{m'}_i$) of a hypercube that is placed to a map unit i. All points inside these hypercubes belong to the estimated distribution. Using these notations the confidence interval for the jth component plane of the Voronoi set F_i is then

$$\left[m'_{ij} - \frac{d'_{ij}}{2}\ ,\ m'_{ij} + \frac{d'_{ij}}{2}\right]. \tag{5}$$

The confidence interval vector $\mathbf{d'}_i$ thus defines the widths of the confidence intervals in the unit i.

2.2.1. A uniform density function

If a uniform distribution is assumed for the jth component plane of the Voronoi set F_i, then the confidence interval vector $\mathbf{d'}_i$ is given as

$$d'_{ij} = d\big(\max\{f_j(t)\} - \min\{f_j(t)\}\big),\ \forall\, \mathbf{f}(t) \in F_i, \tag{6}$$

where d is the confidence level (typically between 90% and 100%). The new weight vector $\mathbf{m'}_i$ is taken to be the center of the confidence interval,

$$m'_{ij} = \frac{\min\{f_j(t)\} + \max\{f_j(t)\}}{2},\ \ \forall\, \mathbf{f}(t) \in F_i. \tag{7}$$

2.2.2. A Parzen estimated density function

The density of $f_j(t)$, the jth component of $\mathbf{f}(t) \in F_i$, can be approximated by a histogram which is formed from samples. The number of histogram bins depends on the desired accuracy of the estimation. It is important to have a sufficient number of samples in each Voronoi set. Otherwise the density estimates are not accurate. When estimating the density with the Parzen estimation method, there are two important factors to consider: the choice of a kernel function and the number of bins. The simplest kernel function γ is the rectangular function,

$$\gamma(z) = \begin{cases} \frac{1}{2h} & |z| \leq h \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where $2h$ is the width of the rectangular kernel. The rectangular kernel produces a discontinuous density estimate. More continuous density estimates can be produced by continuous kernel functions, for example, by the Gaussian kernel. The rectangular kernel function, however, is sufficient to our purposes.

In order to simplify the following formulation, the subscripts i and j, which refer to the map unit i and the component plane j, are dropped. Let us assume a one-dimensional signal $f(t)$, $t = 0, 1, ..., N-1$, of a random variable f, and let $f^{\min}$ be the minimum and $f^{\max}$ the maximum value of f. If the number of bins in the histogram is K then the width of the rectangular kernel γ can be obtained,

$$2h = \frac{f^{\max} - f^{\min}}{K}. \tag{9}$$

The centers of the first bin and the last bin of the histogram are thus

$$b_1 = f^{\min} + h \quad \text{and} \quad b_K = f^{\max} - h. \tag{10}$$

Each sample $f(t)$ is then assigned to the nearest bin in the histogram, i.e., to the bin k that satisfies the condition

$$|b_k - f(t)| \leq h. \tag{11}$$

Let s_k be the number of samples in the kth bin of the histogram. The confidence interval with a confidence level d can be obtained by solving the indices k^f and k^l of the first bin and the last bin that belong to the confidence interval from the following equations,

$$\sum_{k=1}^{k^f-1} \frac{s_k}{N} = \frac{1-d}{2} \quad \text{and} \quad \sum_{k=k^l+1}^{K} \frac{s_k}{N} = \frac{1-d}{2}, \tag{12}$$

where N is the total number of samples $f(t)$. For example, if the desired confidence level is 95%, then $d = 0.95$, and the above formulas define the bins in the beginning and in the end of the histogram whose areas are $\frac{1-0.95}{2} = 0.025$ each. Notice that in practice the equalities are not met exactly. The first and the last histogram bins that belong to the confidence interval are b_{k^f} and b_{k^l}.

When the above procedure is applied to all component planes j of each Voronoi set F_i, the parameters b_{ijk^f} and b_{ijk^l} are obtained for all combinations of i and j. The confidence interval vector $\mathbf{d'}_i$ can now be calculated,

$$d'_{ij} = b_{ijk^l} - b_{ijk^f} + 2h. \tag{13}$$

The new weight vector $\mathbf{m'}_i$ is taken to be the center of the confidence interval,

$$m'_{ij} = \frac{b_{ijk^f} + b_{ijk^l}}{2}. \tag{14}$$

2.2.3. Distance functions for the statistical SOM

The matching error e_i for a feature vector $\mathbf{f}(t)$ and a map unit i is defined similarly to the Hamming distance as

$$e_i = \sum_j e_{ij}, \text{ where } e_{ij} = \begin{cases} 0 & \text{if } (f_j(t) - m'_{ij})^2 \leq \left(\frac{d'_{ij}}{2}\right)^2 \\ 1 & \text{otherwise,} \end{cases} \tag{15}$$

where m'_{ij} is the jth component of a new weight vector $\mathbf{m}'_i$, and d'_{ij} is the jth component of a confidence interval vector $\mathbf{d'}_i$ of a map unit i. A more efficient way to do the comparison is to simply test if a feature vector $\mathbf{f}(t)$ is inside the confidence interval,

$$e_i = \sum_{j=1}^{n} e_{ij}, \text{ where } e_{ij} = \begin{cases} 0 & \text{if } m'_{ij} - \frac{d'_{ij}}{2} \leq f_j(t) \leq m'_{ij} + \frac{d'_{ij}}{2}. \\ 1 & \text{otherwise} \end{cases} \tag{16}$$

This formula has been used successfully in the hardware implementation of the statistical SOM.[28] The best-matching map unit c is now chosen to be the unit with the minimum error e_c,

$$e_c = \min_i \{e_i\}. \tag{17}$$

If there are several map units with the same minimum error e_c, one of them can be selected as the best-matching unit. We are not interest in the actual index of the best-matching unit, only the minimum error e_c must be known.

2.3. Comments on the Statistical SOM

The size of the map determines the accuracy of distribution estimation. More map units means more accuracy. But if the number of training samples is not increased when the map size is increased, then the number of samples in each Voronoi set is decreased and thus the accuracy of density estimation is decreased. Especially when using the Parzen estimation procedure in confidence interval calculation, it is important to have a sufficient number of samples in each Voronoi set. Choice of the confidence level d plays a fundamental role. If the training set consists of plain fault-free samples, then the confidence level should be set very near to 100% so that the mean of the minimum error e_c would be zero. If it can be assumed that there are some outliers (i.e., samples originating from different distribution) in a training set, then d must be set to a lower value so that the effect of outliers can be eliminated. With the uniform density function, d takes usually values within 95% and 100%, and with the Parzen estimated densities within 99% and 100%. d can be set to 100% if training data is totally free of outliers. If there are outliers, the distributions tend to stretch, and the uniform density function is not a very good choice. With the Parzen estimation procedure, these stretched densities can be accurately approximated, and the confidence intervals can be set much tighter.

3. CASE STUDY: UNSUPERVISED IMAGE SEGMENTATION

Defect detection of a textured surface is used as a case study. It is a typical two-class segmentation problem to which the statistical SOM can be applied. Defect detection is based on the following idea: an unknown sample is classified as a defect if it differs enough from the estimated prototypes of the distribution of features extracted from fault-free samples. The statistical self-organizing map is used to estimate these prototypes as described in the previous section.

3.1. The Proposed Segmentation Procedure

The proposed segmentation procedure is done as follows. An image is scanned by a gliding window. A feature vector is extracted from the gliding window for a center image pixel. Each image pixel is then classified as a defect or as a fault-free pixel. Three different steps are thus recognized (Figure 2):

1. Extract a feature vector **f** for an image pixel.
2. Compare **f** with the prototype vectors and find the best-matching unit c (Eqs. (15) - (17)). Show the minimum error e_c as the output.
3. If e_c is larger than the specified limit T, the pixel is classified as a defect. Otherwise the pixel is fault-free.
4. Repeat steps 1-3 for each image pixel.

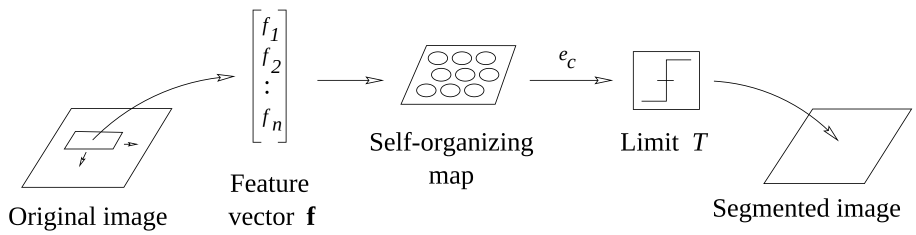


Figure 2. The proposed segmentation scheme.

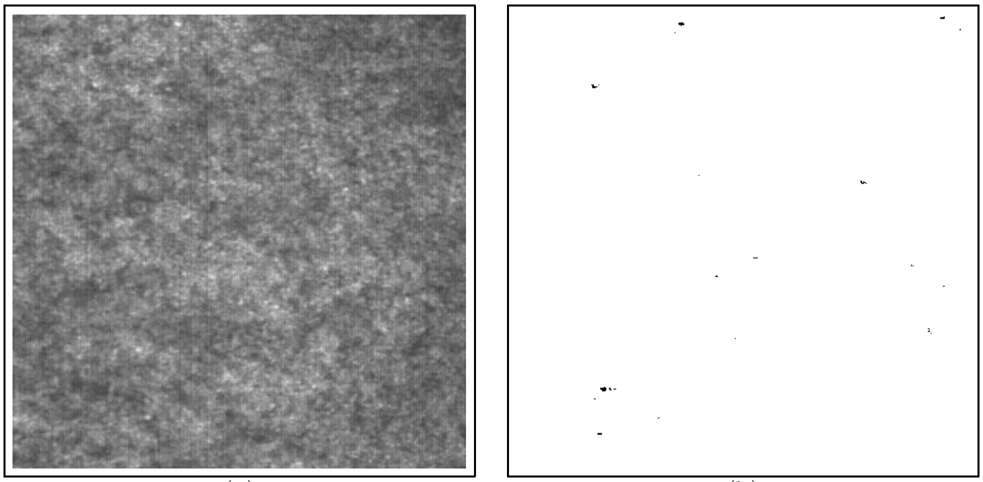
The limit T is given to the process in advance. Determination of the limit T is quite straightforward. Firstly, the T is an integer and it takes values between zero and n (n is the dimension of a feature vector). Secondly, the value of T depends on the desired accuracy of segmentation. For large values of T noise is eliminated and only the most evident (or easily detectable) defects are found. On the other hand, for small values of T noise is increased but also the weak (or difficult) defects are found. So the value of T is a compromise between accuracy and noise.

3.2. Feature Extraction

Images of many surfaces can be considered as stochastic textures, hence the co-occurrence matrices,[29] also known as the spatial gray level dependence matrices, are used for texture description. The co-occurrence matrix method has been found to be good in classifying stochastic textures.[30,31] The co-occurrence matrix is a second order statistical measure of gray level variation. It indicates the joint probability of gray level occurrence of two pixels at a certain displacement in an image. The co-occurrence matrix can be reduced to make calculation time and memory requirements smaller. One approach is to concatenate the rows and columns into two vectors. Another approach is to calculate a set of features. Haralick et al.[29] suggest 14 textural features, and Conners et al.[32] add two features more. In this study three features calculated from the co-occurrence matrix are used. These are energy, contrast, and mean.

3.3. Experimental Setup and Results

The size of surface images was 512x512 and they were quantized to 32 gray levels. The feature vector consisted of six features, namely energy, contrast, and mean, which were calculated from two co-occurrence matrices formed at different displacements. The different displacements were used to avoid the problem concerning improper spatial resolution.[6] The window size used in the co-occurrence matrix calculation was 15x5. The size of the training set was 100000 feature vectors extracted from fault-free samples taken randomly from the image in Figure 3(a). The size of the map was 9 units. The Parzen estimated densities were used. The proposed method is not sensitive to the size of the map. In our case smaller maps do not yield an approximation of the distribution of feature vectors which is accurate enough. On the other hand, only slight improvements can be achieved with bigger maps, only more training data is needed. Especially when using the Parzen estimation procedure in confidence interval calculation, it is important to have a sufficient number of samples in each Voronoi set. Otherwise the density estimation cannot be done accurately. The number of histogram bins H in Parzen estimated densities was 100, so there were approximately 100 samples in each bin (10000 samples in each Voronoi set).



(a) (b)

Figure 3. (a) Surface image used for training and (b) its segmentation with the Parzen estimated densities with $d = 99.9\%$, $T = 1$. Defects are marked with black color.

In the surface image in Figure 4(a) a dark spot with small cracks and holes is clearly visible. Also several small spots (dark and light) can be seen. Segmentation of such an image is not trivial. Gray level thresholding is not a very applicable technique since the considered textures are stochastic ones. In Figure 4(c) the iterative optimal

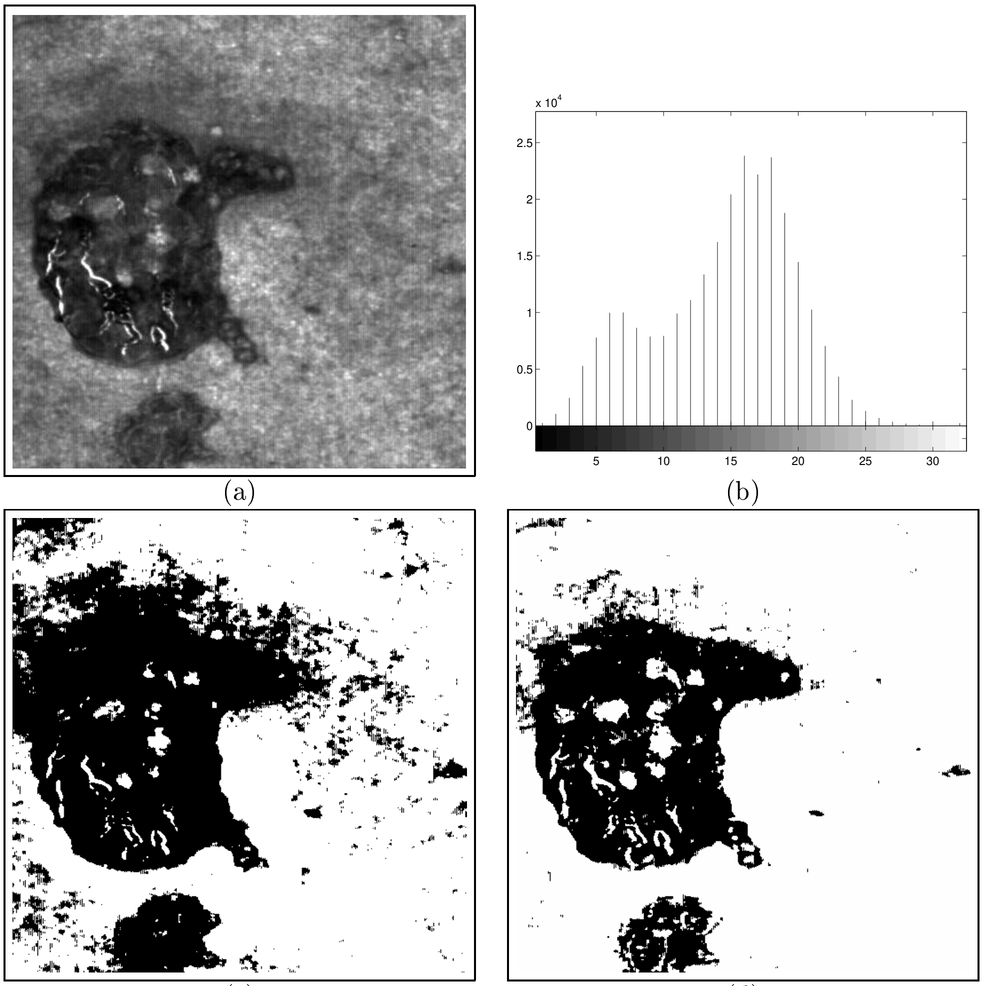


Figure 4. (a) A surface image, (b) its gray level histogram, and the results of gray level thresholding (c) by an iterative algorithm ($g_{th} = 12$) and (d) by hand ($g_{th} = 9$). Defects are marked with black color.

threshold algorithm was used (Ref. 33, pp. 118-119). The obtained threshold value was $g_{th} = 12$. In Figure 4(d) the threshold was selected by hand according to the histogram in Figure 4(b) ($g_{th} = 9$). The pixels having gray level value 32 (holes) were also classified as a defect. Even if the thresholded image is quite good, gray level thresholding is not a suitable technique in this kind of problem. For a discussion of different thresholding techniques and their restrictions, see for example Ref. 34, pp. 14-28.

Segmentation results of the image in Figure 4(a) with Parzen estimated densities are depicted in Figure 5. If using a confidence level $d = 99\%$ or lower, a noisy segmentation is obtained. Such high confidence levels are due to the 'goodness' of training data, i.e., only few defect pixels were included in training data. The best results are obtained when using $T = 1$. If a complete match is demanded for a fault-free sample ($T = 0$), a more noisy segmentation is obtained. In Figure 3 is the surface image which was used in training and its segmentation with $d = 99.9\%$ and $T = 1$. Only few pixels are classified as a defect. More examples are shown in Figure 6. Defects are found quite nicely although some noise is present in all segmentations. Some of it can be eliminated in a classification phase that follows segmentation.[26,35]

4. CONCLUSIONS

The idea of using the self-organizing map to estimate the distribution of fault-free samples and then to classify an unknown sample as a defect if it differs enough from this estimated distribution, is not a new one.[36] The remaining

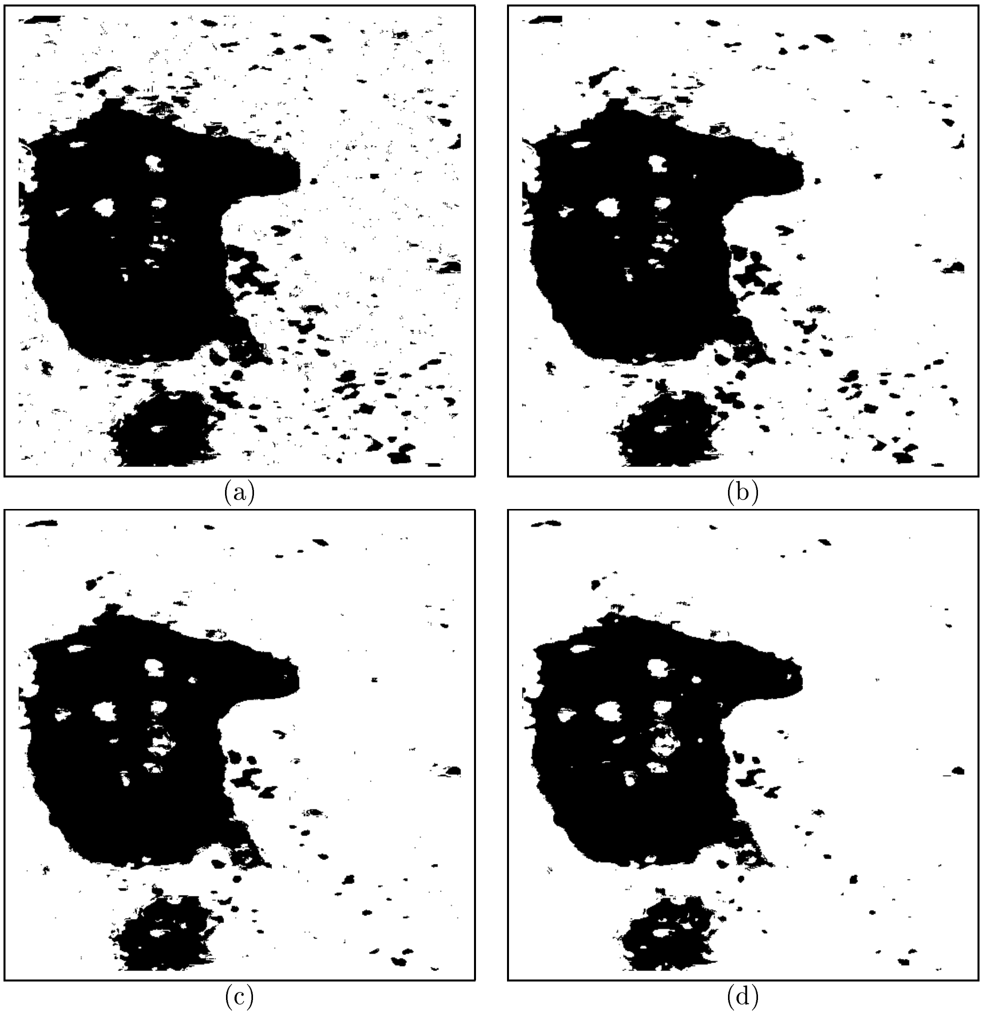


Figure 5. Surface image after segmentation with the Parzen estimated densities. (a) $d = 99\%$, $T = 0$, (b) $d = 99\%$, $T = 1$, (c) $d = 99.9\%$, $T = 0$, (d) $d = 99.9\%$, $T = 1$. Defects are marked with black color.

question has been how much an unknown sample can differ from the best-matching unit of the SOM before it is classified as a defect. One solution is to determine a threshold value for the distance between a feature vector of the unknown sample and the best-matching map unit. It is not, however, easy to choose such a threshold value, and in fact a common threshold value for all map units may not be enough. Instead, a distinct threshold value for each map unit would be necessary. The proposed method overcomes the threshold selection problem. It makes use of the Voronoi set of each map unit and defines a new rule for finding the best-matching map unit. For each component plane j of the Voronoi set F_i of the map unit i a confidence interval is defined. When assuming a uniform density function for a component plane j, which is actually the best possible assumption that can be made, the definition of the confidence interval is straightforward. However, these distributions are not necessary uniform. The Parzen estimation procedure can be used to estimate these densities if more accuracy is needed. Forming the Voronoi tessellation of the feature space with other estimator than the self-organizing map is also be possible. For a recent review of different estimators, see for example Ref. 37.

The results of experiments with surface images are encouraging. When compared with simple threshold segmentation based on gray level histograms, the proposed scheme has obvious advantages. The characteristics of a fault-free surface are learned automatically from samples. The use of confidence intervals makes it possible to fine-tune the accuracy of segmentation. The method is general in the sense that it can be applied to defect detection of different types of surfaces. However, it may be necessary to reselect features to take into account the specific properties of the surface type.

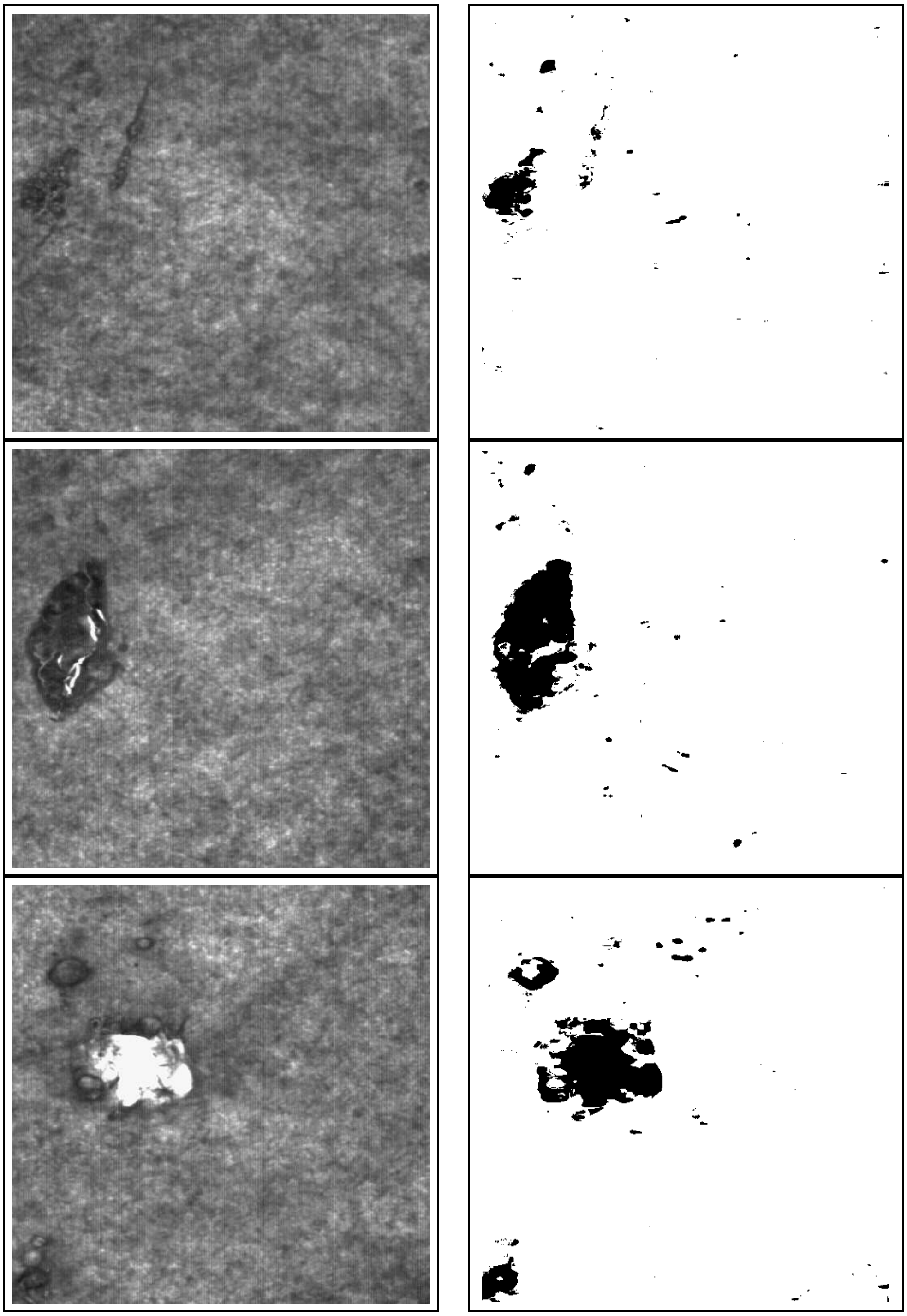


Figure 6. Surface images and their segmentations with the Parzen estimated densities with $d = 99.9\%$ and $T = 1$. Defects are marked with black color.

ACKNOWLEDGMENTS

The authors wish to thank the ABB Industry Oy and the Technology Development Centre of Finland for financial support (TEKES's grant 4172/95).

REFERENCES

1. S. K. Fu and J. K. Mu, “A survey on image segmentation,” *Pattern Recognition* **13**, pp. 3–16, 1981.
2. A. K. Jain and R. C. Dubes, *Algorithms for clustering data*, Prentice Hall, Englewood Cliffs, NJ, 1988.
3. B. D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, 1996.
4. T. Kohonen, *Self-Organizing Maps*, Springer-Verlag, Berlin, 1995.
5. J. Lampinen and E. Oja, “Self-organizing maps for spatial and temporal ar models,” in *Proceedings of the 6th Scandinavian Conference on Image Analysis*, pp. 120–127, (Oulu, Finland), June 18–21 1989.
6. A. Visa, “Identification of stochastic textures with multiresolution features and self-organizing maps,” in *Proceedings of the 10th International Conference on Pattern Recognition*, vol. I, pp. 518–522, (Atlantic City, New Jersey, USA), June 16–21 1990.
7. A. Visa, “Unsupervised image segmentation based on a self-organizing feature map and a texture measure,” in *Proceedings of the 11th IAPR International Conference on Pattern Recognition*, vol. III, pp. 101–104, (The Hague, The Netherlands), Aug. 30–Sept. 3 1992.
8. A. Visa, K. Valkealahti, and O. Simula, “Cloud detection based on texture segmentation by neural network methods,” in *International Joint Conference on Neural Networks*, vol. 2, pp. 1001–1006, (Singapore), Nov. 18–21 1991.
9. A. Visa, K. Valkealahti, J. Iivarinen, and O. Simula, “Experiences from operational cloud classifier based on self-organising map,” in *Applications of Artificial Neural Networks V*, S. K. Rogers and D. W. Ruck, eds., Proc. SPIE 2243, pp. 484–495, 1994.
10. A. Visa and J. Iivarinen, “Evolution and evaluation of a trainable cloud classifier,” *IEEE Transactions on Geoscience and Remote Sensing* **35**, pp. 1307–1315, Sept. 1997.
11. J. Iivarinen, J. Rauhamaa, and A. Visa, “Unsupervised segmentation of surface defects,” in *Proceedings of the 13th International Conference on Pattern Recognition*, vol. IV, pp. 356–360, (Wien, Austria), Aug. 25–30 1996.
12. S. Lu, J. E. Hernandez, and G. A. Clark, “Texture segmentation by clustering of gabor feature vectors,” in *International Joint Conference on Neural Networks*, vol. I, pp. 683–688, (Seattle, WA), July 8–12 1991.
13. H. Greenspan, R. Goodman, and R. Chellappa, “Texture analysis via unsupervised and supervised learning,” in *International Joint Conference on Neural Networks*, vol. I, pp. 639–644, (Seattle, WA), July 8–12 1991.
14. A. P. Dhawan and L. Arata, “Segmentation of medical images through curvature scale space,” in *Proceedings of the IEEE International Conference on Neural Networks*, vol. III, pp. 1277–1282, (San Francisco, California), Mar. 28–Apr. 1 1993.
15. H. Yin and N. M. Allinson, “Self-organised segmentation for textured images,” in *Proceedings of the International Conference on Artificial Neural Networks*, vol. 2, pp. 1149–1152, (Sorrento, Italy), May 26–29 1994.
16. A. Finch and J. Austin, “A neural network for dimension reduction and its application to image segmentation,” in *Proceedings of the International Conference on Artificial Neural Networks*, vol. 2, pp. 1141–1144, (Sorrento, Italy), May 26–29 1994.
17. S. Oe, M. Hashida, M. Enokihara, and Y. Shinohara, “A texture segmentation method using unsupervised and supervised neural networks,” in *Proceedings of the IEEE International Conference on Neural Networks*, vol. IV, pp. 2415–2418, (Orlando, Florida), June 27–29 1994.
18. S. Haring, M. A. Viergever, and J. N. Kok, “Kohonen networks for multiscale image segmentation,” *Image and Vision Computing* **12**, pp. 339–344, July/Aug. 1994.
19. H. Chao, S. R. Ray, and N. Zheng, “Texture segmentation using joint time frequency representation and unsupervised classifier,” in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, pp. 304–309, (Vancouver, Canada), Oct. 22–25 1995.
20. S. M. Bhandarkar, J. Koh, and M. Suk, “Multiscale image segmentation using a hierarchical self-organizing map,” *Neural Computation* **14**(3), pp. 241–272, 1997.
21. J. Ruiz-del-Solar and M. Köppen, “A texture segmentation architecture based on automatically generated oriented filters,” *Journal of Microelectronic Systems Integration* **5**(1), pp. 43–52, 1997.
22. K. Y. Song, M. Petrou, and J. Kittler, “Texture defect detection: A review,” in *Applications of Artificial Intelligence X: Machine Vision and Robotics*, Proc. SPIE 1708, pp. 99–106, 1992.
23. M. Unser and F. Ade, “Feature extraction and decision procedure for automated inspection of textured materials,” *Pattern Recognition Letters* **2**, pp. 185–191, Mar. 1984.

24. P. Dewaele, L. Van Gool, P. Wambacq, and A. Oosterlinck, "Texture inspection with self-adaptive convolution filters," in *Proceedings of the 9th International Conference on Pattern Recognition*, vol. 1, pp. 56–60, (Rome, Italy), Nov. 14–17 1988.
25. F. S. Cohen, Z. Fan, and S. Attali, "Automated inspection of textile fabrics using textural models," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**, pp. 803–808, Aug. 1991.
26. J. Iivarinen, J. Rauhamaa, and A. Visa, "An adaptive two-stage approach to classification of surface defects," in *Proceedings of The 10th Scandinavian Conference on Image Analysis*, vol. I, (Lappeenranta, Finland), June 9–11 1997.
27. C. W. Therrien, *Decision, Estimation and Classification*, Wiley, New York, 1989.
28. K. Heikkinen and P. Vuorimaa, "Hardware architecture of the som for real-time applications," in *Proceedings of the 5th European Congress on Intelligent Techniques and Soft Computing*, vol. III, pp. 2495–2499, (Aachen, Germany), Sept. 8–11 1997.
29. R. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-3**, pp. 610–621, Nov. 1973.
30. J. S. Weszka, C. Dyer, and A. Rosenfeld, "A comparative study of texture measures for terrain classification," *IEEE Transactions on Systems, Man, and Cybernetics* **SMC-6**, pp. 269–285, Apr. 1976.
31. R. Conners and C. Harlow, "A theoretical comparison of texture algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2**, pp. 204–222, May 1980.
32. R. Conners and C. Harlow, "Toward a structural textural analyser based on statistical methods," *Computer Graphics and Image Processing* **12**, pp. 224–256, 1980.
33. M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis and Machine Vision*, Chapman & Hall Computing, London, 1993.
34. R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, vol. 1, Addison-Wesley, 1992.
35. J. Iivarinen and A. Visa, "An adaptive texture and shape based defect classification," in *Proceedings of the 14th International Conference on Pattern Recognition*, (Brisbane, Australia), Aug. 16–20 1998.
36. J. T. Alander, M. Frisk, and L. Holmström, "Process error detection using self-organizing feature maps," in *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, eds., vol. 2, pp. 1229–1232, North-Holland, 1991.
37. A. Hämäläinen, *Self-Organizing Map and Reduced Kernel Density Estimation*. PhD thesis, University of Helsinki, Rolf Nevanlinna Institute, Helsinki, 1995.