The adaptive subspace map for texture segmentation

Dick de Ridder and Josef Kittler
Centre for Vision, Speech and Signal Processing
Dept. of Electronic & Electrical Engineering
School of EEITM, University of Surrey
Guildford, GU2 5XH Surrey, United Kingdom

Olaf Lemmers and Robert P.W. Duin
Pattern Recognition Group
Dept. of Applied Physics
Delft University of Technology
Lorentzweg 1, 2628 CJ Delft, The Netherlands

E-mail: dick@ph.tn.tudelft.nl

Abstract

In this paper, a non-linear mixture-of-subspaces model is proposed to describe images. Images or image patches, when translated, rotated or scaled, lie in low-dimensional subspaces of the high-dimensional space spanned by the grey values. These manifolds can locally be approximated by a linear subspace. The adaptive subspace map is a method to learn such a mixture-of-subspaces from the data. Due to its general nature, various clustering and subspace-finding algorithms can be used. In this paper, two clustering algorithms are compared in an application to some texture segmentation problems. It is shown to compare well to a standard Gabor filter bank approach.

1. Introduction

Over the past years, adaptive image description techniques have received some attention. One development was the Adaptive Subspace Self Organising Map (ASSOM) by Kohonen [6, 8]. This is an extension of the standard SOM which uses subspaces in each node instead of just weights, enabling each node to represent a certain texture in a translation-, rotation- or scale-invariant manner. The idea is based on the observation that translated, rotated or scaled image patches lie on a low-dimensional manifold in the high-dimensional pixel space. This manifold can then, at least locally, be approximated by linear subspaces, giving a sparse representation of image content. The results are very interesting, but the ASSOM so far has only been applied to one or two real images [6] and images containing coloured noise [8]. Heidemann and Ritter [3] used a similar technique, the local linear map (LLM), for object recognition. Their system, however, did not work directly on pixel data, but on features extracted using optimised Gabor filters.

At the same time, there has been some interest in adaptive techniques that might be called "mixtures of local linear estimators". Hinton et al. [4] described a method in which image manifolds were modelled as subspace-specific principal components (PCAs). Later work by Tipping and Bishop [9] generalised this approach by developing a probabilistic model for PCA. The specific advantages and disadvantages of these methods will be discussed in section 2.

In this paper a system is described which combines the best aspects of these two approaches. On the one hand, knowledge of the invariances one desires the models to display can be used when creating the training set, as was done by Kohonen. On the other hand, to avoid the extremely long training times of the ASSOM, a mixture of local PCA models is used. An overview of the proposed system, using two different clustering methods, will be given in section 3. The system is then applied to an artificial segmentation problem and to a real texture segmentation problem in section 4. Its performance will be compared to one of the standard approaches to image segmentation, which is based on Gabor filters. Some conclusions and discussion are given in section 5.

2. Adaptive texture description

In [6] the adaptive subspace self-organising map (ASSOM) was proposed, based on ideas developed earlier for the learning subspace method (LSM, [7]). Basically, an ASSOM is an ordinary SOM of which the nodes do not just represent points in a feature space but subspaces S_j. In training, the ASSOM uses not just single samples, but *episodes* E_k: sets of slightly translated, rotated or scaled samples, which are treated as a single entity. The distance measure used is also different from that used in a normal SOM: it is the minimum projection error of any sample $\mathbf{x}_i \in E_k$ onto subspace S_j.

0-7695-0750-6/00 $10.00 © 2000 IEEE

Training the ASSOM consists of presenting one episode to the network in each iteration (note that this means the learning is done on-line). The winning node is defined as that node onto which the sample gives the lowest projection error. This node's subspace, and that of its neighbours, is then rotated towards the sample; i.e. the subspace is learned. The ASSOM gives good results, but is extremely slow in training. This is mainly due to the rotation of subspaces, to learn them, and the fact that neighbourhoods of nodes are updated instead of just single nodes, to obtain topologically correct maps.

Neither of these features are necessary when the goal is simply to construct a model which describes an image well. First, the subspace need not be learned, but can easily be found by applying a principal component analysis (PCA) in a batch-learning version. Second, there is no need for a topologically correct map in most applications. Dropping these features leads to a model which resembles the mixture of local linear models proposed by Hinton et al. [4]. However, in their work no use was made of episodes; instead, the invariances were incorporated into the covariance matrix used when calculating a PCA, with weights indicating the relative importance of each invariance. Also, a pre-specified number of clusters was sought, with a pre-specified dimension.

Although the mixture of local linear models comes close to the system advocated here, our method is more flexible as several cluster algorithms and subspace-finding methods can be plugged in. In the next section, two variants using different clustering algorithms will be discussed. Also, there is no need to specify relative importances of invariances, as these will be learned automatically.

3. The adaptive subspace method

3.1. The *k*-subspace ASM

Training the adaptive subspace map (henceforth referred to as *ASM*) is basically a combination of clustering high-dimensional data, grouped in episodes, and then calculating subspaces based on the cluster members. In this framework, various clustering algorithms can be used. The simplest is probably the k-subspaces algorithm, derived from the k-means algorithm. Instead of finding just a mean of several data points, it searches for subspaces. In each iteration, each subspace is re-calculated by applying PCA to all its member episodes. An episode E_k is a member of that subspace $\langle \mathbf{S}_j, \mathbf{O}_j \rangle$ to which it has the minimum distance. This distance is defined as:

$$D(E_k, \langle \mathbf{S}_j, \mathbf{O}_j \rangle) = \frac{1}{|E_k|} \sum_{i=1}^{|E_k|} ||\mathbf{x}_i - \hat{\mathbf{x}}_i^j|| \qquad (1)$$

where $\hat{\mathbf{x}}_i^j$ is the projection of image sample $\mathbf{x}_i \in E_k$ onto subspace $\mathbf{S}_j$ with origin $\mathbf{O}_j$. Note that this measure uses the average error over all episode samples instead of the minimum, as Kohonen did. This was done to stabilise the algorithm. The origin $\mathbf{O}_j$ of each subspace is found by taking the mean of all the member episodes. Before projection of a sample, this origin is subtracted from the sample.

Although this algorithm is quick and converges reasonably well, there is the need to specify the desired number of subspaces beforehand. This can be a problem in cases where the exact number of segments an image should be split into is not known beforehand.

3.2. The subspace shift ASM

Another clustering algorithm used is the subspace shift algorithm, an iterative scheme, based on the probabilistic *mean shift* clustering algorithm proposed in [1]. In short, the mean shift algorithm for normal feature space clustering follows a density gradient estimate by repeatedly calculating the mean of a number of points falling inside a hypersphere of radius h. The centre of the hypersphere is then placed at this mean, and the algorithm is iterated until convergence. The probabilistic version of this algorithm does not apply k hyperspheres at once, but starts with one and removes the points inside it when it has converged. It then reapplies the method on the remaining points until there are no more points left.

Here, we replace the hypersphere by a subspace and the radius by a maximum distance (as defined in equation (1)) to that subspace. In each iteration, a PCA is performed on the episodes within a distance h to that subspace.

The basic algorithm for the adaptive subspace method thus is:

1. Create a data set by extracting samples from one or more images and make episodes by translating, rotating and/or scaling the samples
2. Using the subspace shift algorithm, find one subspace $\langle \mathbf{S}_i, \mathbf{O}_i \rangle$ for which all contributing episodes lie within a distance h_i
3. Remove the contributing episodes from the data set
4. Repeat 2. until there are fewer than r episodes left

The subspace shift algorithm works as follows:

1. Pick r episodes $E_k, k = 1, ..., r$ at random from the data set and find a subspace $\langle \mathbf{S}_k, \mathbf{O}_k \rangle$ for each
2. Set the seed to be that episode E_j for which the projection error onto its subspace, $D(E_j, \langle \mathbf{S}_j, \mathbf{O}_j \rangle)$, is smallest
3. Set h for this subspace to $h_i = m \cdot D(E_j, \langle \mathbf{S}_j, \mathbf{O}_j \rangle)$
4. Find all episodes with a projection error smaller than h_i

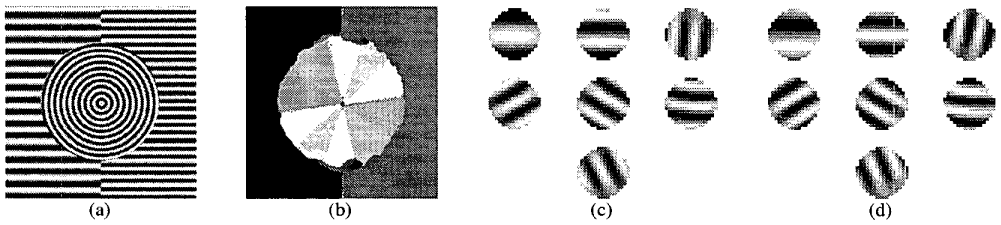

Figure 1. (a) Artificial texture. (b) Subspace shift ASM segmentation. (c) 1st and (d) 2nd basis vectors.

5. Calculate a new subspace $\langle \mathbf{S}_i, \mathbf{O}_i \rangle$ on these episodes
6. While not converged, go to 4

Setting h_i dynamically is necessary, as it cannot be expected that all subspaces have the same properties, and beneficial, as the user does not have to specify h beforehand. The multiplier m does have to be specified, but seems a more natural parameter than h. The only other parameter that has to be set is r, but we found that the method is not sensitive to its setting, provided it is large enough, say $r \geq 10$.

3.3. Subspace merging

Although the subspace shift algorithm gives good results, it has a tendency to over-segment images. This happens when certain image content, such as highly unstructured textures, cannot be modelled well using a subspace. The method then finds a large number of subspaces which represent the data equally poorly. Therefore, an optional post-processing step can be performed in which subspaces are merged. This is done by projecting the original training image(s) onto each subspace and merging subspaces for which the average difference in projection error is smaller than a threshold t. A small difference indicates that the subspaces perform more or less the same function. No new subspaces are calculated; the subspaces are just given the same label.

4. Experiments

4.1. Artificial texture segmentation

An artificial texture image, shown in figure 1, was created containing sinewaves of different frequency and orientation. 100 master samples were extracted using a round, 16-pixel diameter window, and each was turned into an episode by creating 5 samples containing vertical translations of the master sample, in the range $[-5,5]$ pixels, and 5 samples containing rotated versions of the master sample in the $[-45,45]$ degree range. A subspace shift ASM was trained on these episodes ($m = 1.5$). The resulting basis vectors and the segmentation of the image are shown in figure 1. The algorithm has no trouble finding a good segmentation for this image. The basis vectors are pairs of wavelet-like filters, tuned to the specific image frequencies and orientation ranges.

4.2. Brodatz texture segmentation

In the same way as the artificial texture, 7 images consisting of 2 Brodatz textures each were segmented. In this case, episodes consisted of 5 samples horizontally and vertically translated over the range $[-5,5]$ pixels; 5 samples rotated over the range $[-45,45]$ degrees; and 5 samples scaled in the range $[1.0,1.5]$ times the original size. Samples were taken using a round window with a diameter of 16 pixels. ASMs with 4 dimensions per subspace were used, as 2 dimensions are too few to adequately represent these textures well. The images and the resulting segmentations by various ASMs, found by assigning each central pixel of a window to its nearest subspace, are shown in figure 2.

The k-subspace ASM was trained with $k = 2$. The subspace shift ASMs were trained with $m = 2$, giving some oversegmentation, and post-processed with $t = 0.05$. Although the result of this post-processing was sensitive to the setting of t, it was not hard to find a t that worked well for most texture images.

As a comparison, the same images were segmented by using k-means clustering in a 12-dimensional feature space generated by applying a Gabor filter bank (see [5]). It is very hard to define a merging algorithm similar to that used after the application of the ASM, as the average distances between the points in Gabor feature space and the cluster centers vary too much. That means that for each image an optimal threshold would have to be set.

The images show that the k-subspace ASMs give quite good results. Image 5 cannot be segmented well, since it is very irregular. Image 7 also gives problems. In some cases (images 2 and 4) the segmented form is correct, but not its size, due to the relatively large sample window used.

The subspace shift algorithm performs a little better. It automatically finds a good number of clusters to use, and with a relatively easy setting of the threshold parameter t most images can be segmented reasonably well. Image 7 shows again that the method has problems with unstructured textures. On the most highly unstructured texture, image 5,

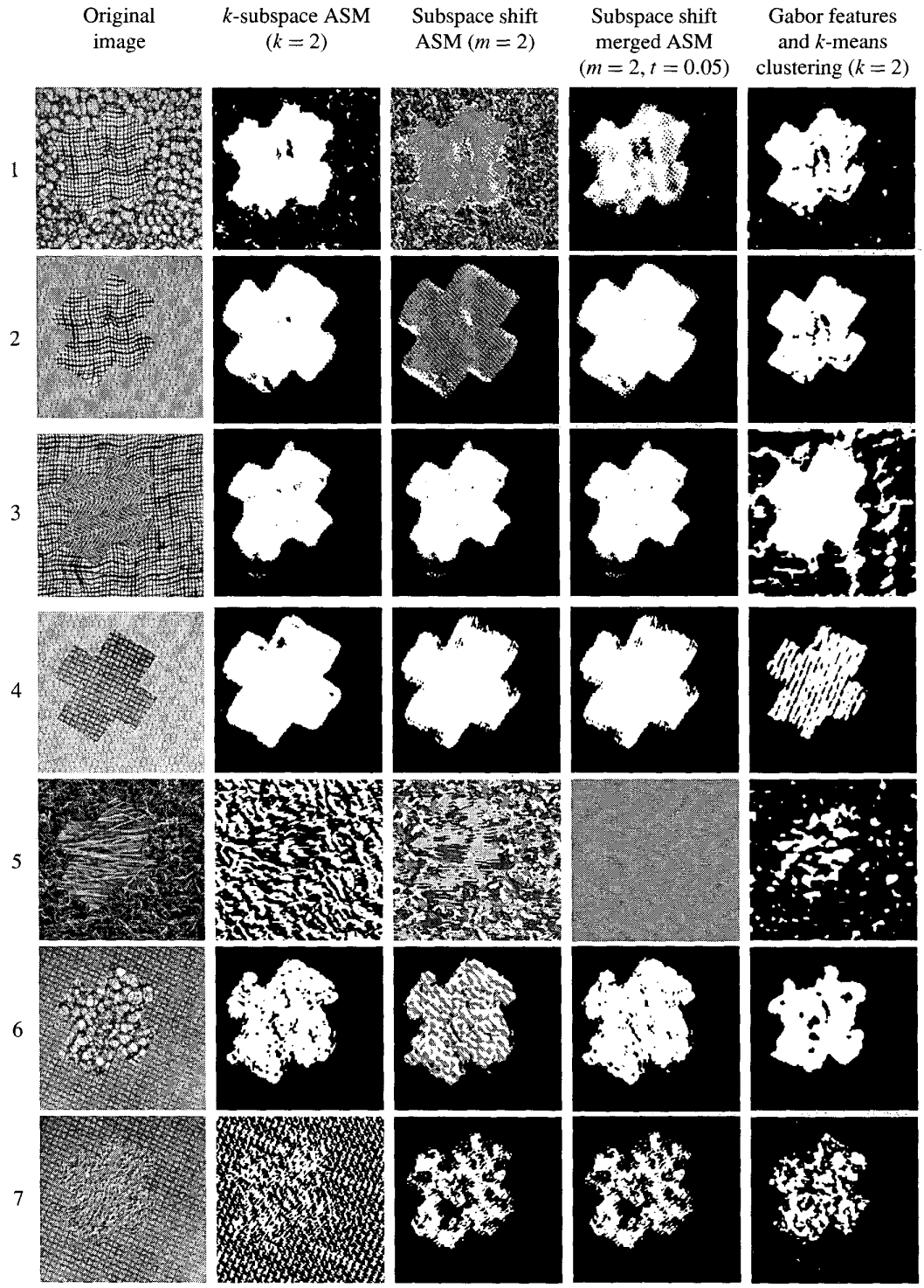

Figure 2. Seven composite Brodatz texture images and segmentations using various algorithms. The subspace shift ASMs segmented the images into 13, 5, 2, 2, 5, 3 and 2 segments, respectively. All segmentations by the subspace shift, merged ASM (except that of image 5) contain 2 segments.

it fails completely: all regions are merged into one. If t was set lower for this image, it would be segmented into two segments.

The Gabor filter bank gives similar results, sometimes slightly better (e.g. images 2 and 4) and sometimes slightly worse (e.g. images 3 and 7). In some cases the effect of the Gabor filter's limited rotation and scale invariance as compared to the ASM can be seen: in images 2 (foreground), 3 (background) and 6 (foreground), the Gabor filter bank has some relatively large blobs which are segmented incorrectly. Figure 3 illustrates this; it shows the behaviour of both the k-subspaces ASM and Gabor filter bank, trained on image 3, when applied to slightly rotated versions of image 3. Overall, the ASM seems to perform somewhat better than the Gabor filter bank.

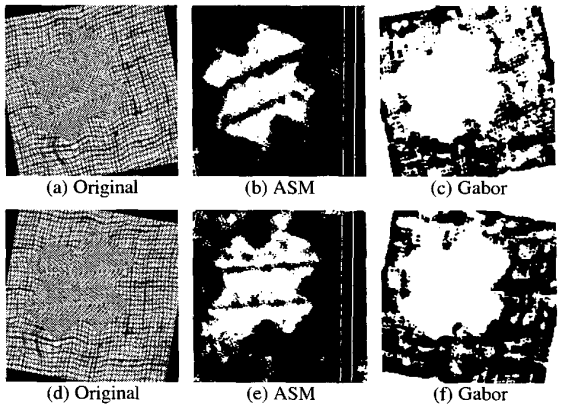

Figure 3. (a,d) Brodatz texture image 3 rotated over $-10°$ and $+10°$; segmentations using an ASM (b,e) and the Gabor filter bank (c,f).

5. Conclusions and discussion

Two methods were presented for adaptive description of texture in terms of a number of linear subspaces. The best performing one, the subspace shift ASM, depends on only a small number of parameters, the main one of which is a parameter influencing the extent of the subspace: the multiplier m in the subspace shift algorithm. The method automatically finds the optimal number of subspaces. The ASMs were applied to an artificial segmentation problem and several Brodatz texture segmentation problems, and were shown to perform quite well, especially on structured textures. Although the method is very general in nature, in this application it works just as well as a Gabor filter bank, and has built-in invariance to small rotations of the texture.

Possible extensions of the method include the automatic determination of the subspace dimensionality, e.g. for PCA by specifying the amount of variance that has to be preserved in the data when it is projected. For the k-subspace ASM this is quite straightforward, but it is not a trivial problem for the subspace shift ASM, as it interacts severely with the setting of the parameter m. Also, the method could be extended to use other subspace finding algorithms, such as ICA, or non-linear mappings such as multi-dimensional scaling. As long as one can define both a way of finding a subspace given a number of samples and a distance measure between samples and subspaces, there should be no problem fitting it into the framework.

In related work [2] the applicability of ASMs to image classification and image database retrieval is investigated. ASMs are ideally fitted for such a task as they present a tuned, sparse representation of image content.

Acknowledgements

This work was partly supported by the Foundation for Computer Science Research in the Netherlands (SION), the Dutch Organisation for Scientific Research (NWO) and the Engineering and Physical Sciences Research Council (EPSRC) in the UK (grant numbers GR/M90665 and GR/L61095). The first author would like to thank the Centre for Vision, Speech and Signal Processing and the EPSRC for allowing him to visit the centre as a visiting fellow for six months.

References

[1] D. Comaniciu and P. Meer. Distribution free decomposition of multivariate data. *Pattern Analysis and Applications*, 2(1):22–30, 1999.

[2] D. de Ridder, O. Lemmers, R. Duin, and J. Kittler. The adaptive subspace map for image description and image database retrieval. In *Proceedings S+SSPR*, 2000. To appear.

[3] G. Heidemann and H. Ritter. A neural 3-D object recognition architecture using optimized Gabor filters. In *Proceedings of the 13th IAPR International Conference on Pattern Recognition, Vol. IV*, pages 70–74, Los Alamitos, CA, 1996. IAPR, IEEE Computer Society Press.

[4] G. Hinton, P. Dayan, and M. Revow. Modelling the manifolds of images of handwritten digits. *IEEE Transactions on Neural Networks*, 8(1):65–74, 1997.

[5] A. Jain and F. Farrokhnia. Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

[6] T. Kohonen. *Self-organizing maps*. Springer Series in Information Sciences. Springer Verlag, Berlin, 1995.

[7] T. Kohonen, K.-J. Bry, M. Jalanko, H. Riittinen, and G. Németh. Spectral classification of phonemes by learning subspaces. In R. Olson, editor, *Proceedings of the ICASSP*, pages 97–100, Piscataway, NJ, 1979. IEEE Service Center.

[8] T. Kohonen, S. Kaski, and H. Lappalainen. Self-organized formation of various invariant-feature filters in the Adaptive-Subspace SOM. *Neural Computation*, 9(6):1321–1344, 1997.

[9] M. Tipping and C. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999.