Image database retrieval using adaptive non-linear image descriptions

Dick de Ridder[*†], Olaf Lemmers[*], Robert P.W. Duin[*] and Josef Kittler[†]

[*]Pattern Recognition Group, Dept. of Applied Physics,
Faculty of Applied Sciences, Delft University of Technology
Lorentzweg 1, 2628 CJ Delft, The Netherlands

[†]Centre for Vision, Speech and Signal Processing, Dept. of Electronic & Electrical Engineering,
School of Electronic Engineering, Information Technology and Mathematics,
University of Surrey, Guildford, GU2 5XH Surrey, United Kingdom

dick@ph.tn.tudelft.nl

Keywords: adaptive methods, image analysis, feature extraction, image database retrieval

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. If the adaptive subspace map is trained on data extracted from images, a description of the image content is obtained, which can then be used for various classification and clustering problems. Here, the method is applied to an image database retrieval problem, and is shown to give promising results.

1 Introduction

A method often used for representing image data is to use the image function on a grid of pixel positions, i.e. $I(x, y)$. A problem with this representation is that it is rather unnatural in a number of ways. In principle, although the space spanned by all pixel positions (x, y) is high-dimensional, it would seem to lend itself well for use in standard pattern recognition approaches. However, if the image is just slightly translated, rotated or scaled, the distribution of the data in the high-dimensional space changes completely, whereas to a human observer the data still looks very similar. In fact, transformed versions of an image all lie on an m-dimensional manifold in the high-dimensional space spanned by all pixel grey values, where m is the number of degrees of freedom present in the set of transformations [8]. Although this manifold may be low-dimensional, it is likely to fill a large portion of the high-dimensional space.

One way of tackling this problem is by using pixel coordinates in the data description as well as the grey values themselves [3, 9]. However, although this might help to express pixel vicinity in tasks such as segmentation, it is not straightforward to apply it to large-scale problems such as image classification: even though the pixel coordinate values remain approximately the same for small transformations, the grey values still change significantly.

The approach proposed here is to describe an image (or image patches) using subspaces in the high-dimensional space of all pixel values, rather than just points. A mixture of subspaces can then be used to model the manifold discussed earlier. It draws on earlier work by Kohonen, the adaptive subspace self-organising map (ASSOM) [7] and work by Kambhatla and Leen [5], Hinton et. al. [4], and Tipping and Bishop [16]. Their approaches will be discussed in section 2.

In section 3 the basic version of the proposed method, the adaptive subspace map, or ASM, will be introduced. Next, in section 4, it is applied to an image database retrieval problem. In this problem, an ASM is trained on each image. Then, a distance measure between the ASMs is used to retrieve images. Experiments on a small image database are presented in section 5. Finally, section 6 will discuss possible extensions, and section 7 will draw some conclusions.

2 Adaptive image description

Kohonen [7] proposed an extension of his self-organising map, which uses subspaces $\vec{S}_j$ in each node rather than just single weights. This adaptive subspace self-organising map, or ASSOM, is based on training not just using single samples but sets E_k of slightly translated, rotated and/or scaled signal or image samples, called *episodes*. These episodes are treated as a single entity, that is, samples are assigned as a group to a subspace based on a distance measure between an episode and a subspace, which is the minimum projection error of any sample $\vec{x} \in E_k$:

$$D(E_k, \vec{S}_j) = \min_{\vec{x} \in E_k} ||\vec{x} - \vec{\hat{x}}^j|| \qquad (1)$$

where $\vec{\hat{x}}^j$ is the projection of $\vec{x}$ onto subspace $\vec{S}_j$.

To train the ASSOM, samples drawn from a signal or an image are converted into episodes by creating slightly transformed versions of the original sample. The distance between each node and the episode is then calculated, and the winning node is defined as that node to which the episode has minimum distance. In the adaptation phase, the winning node's subspace, and that of its neighbours, is rotated to better fit the just presented episode.

The ASSOM gives good results, but is extremely slow in training. This is caused by the learning of the subspaces by rotating them, which demands careful and prudent setting of learning parameters, but also by the updating of neighbourhoods to obtain a topologically correct map. If one drops these demands, i.e. just finds the subspaces in a batch-mode operation (e.g. using principal component analysis (PCA)), and perform non-topologically correct clustering, the resulting system would be greatly simplified. Such a system would come close to the systems described by Kambhatla and Leen [5], who basically used a batch vector quantisation algorithm with the projection error onto a subspace as a distance measure; Hinton et al. [4], a mixture of local linear models learned using the EM algorithm; and Tipping and Bishop [16], who extended Kambhatla's work by defining a probabilistic PCA model (PPCA). However, these methods do not use the idea of episodes, and were mainly used on artificial data or small images containing entire objects (handwritten digits).

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 subspaces is used. An overview of our proposed system will be given in section 3.

3 The adaptive subspace map

The basic idea of the ASM is to find a number of subspaces which describe the episodes well. This calls for both a clustering mechanism, which assigns episodes to subspaces, and a subspace-finding method, which calculates subspace parameters based on the assigned episodes. Although many algorithms could in principle be used, here we only consider the PCA algorithm for finding subspaces and a k-means like clustering method we will refer to as the k-subspaces method. 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 E_k by translating, rotating and/or scaling the samples
2. Assign these episodes randomly to one of the subspaces $\left\langle \vec{S}_j, \vec{O}_j \right\rangle$, $j = 1, \ldots, n$
3. Re-calculate each subspace, using a PCA on the episodes assigned to that subspace to find $\vec{S}_j$ and setting the origin $\vec{O}_j$ of the subspace to be the mean of these episodes
4. Assign each episode E_k to the closest subspace, which is that subspace $\left\langle \vec{S}_j, \vec{O}_j \right\rangle$ to which the average sample projection distance is smallest:

   $$D(E_k, \left\langle \vec{S}_j, \vec{O}_j \right\rangle) = \frac{1}{|E_k|} \sum_{\vec{x} \in E_k} ||\vec{x} - \vec{\hat{x}}^j|| \qquad (2)$$

   where $\vec{\hat{x}}$ is the projection of $(\vec{x} - \vec{O}_j)$ onto $\vec{S}_j$
5. While not converged, go to 3.

Note that we use the average distance of all samples as opposed to the minimum distance used by Kohonen. This stabilised convergence and did not give very different results. Also, in the general ASM each subspace has its own origin $\vec{O}_j$.

3.1 Data set creation

Episodes are created by taking randomly shifted samples in a certain range (translation), by rotating the image over a certain range of angles and taking samples (rotation) or by scaling the image over a certain range of scales and taking samples (scaling). For the latter two, the rotation or scale range is divided into a number of equally large steps, with a small random offset. All resulting samples are packed into vectors.

After creation of the episodes, the data is preprocessed by subtracting the average grey value from each vector and normalising its standard deviation to 1. The advantage of this normalisation is that texture samples will be recognised independently of illuminisation, if illuminisation differences between images I^A and I^B are modelled as $I^A = \alpha I^B + \beta$. A major disadvantage, however, is that natural images often contain large non-textured regions, such as sky. Due to the normalisation to unit length, the noise in samples taken from this region will be blown up. We are currently investigating ways of normalising the data without becoming too sensitive to noise.

Note that since all samples are centered around the origin, use of a different origin for each subspace is not necessary. In the experiments described in this paper the subspace origins were therefore not used, i.e. each subspace was centered around the origin of the original high-dimensional space.

4 Image database retrieval

An ASM trained on episodes collected from images, or even a collection of images, can be seen as a descriptor of that image. This opens up a variety of possible applications, such as segmentation, object recognition etc. Here we focus on image database applications, in which it is often a problem to define measures such that image content can be described in a compact way. A large body of literature exists dealing with indexing images based on their texture content – see e.g. Antani et al. [2] for a review. As they note, it is impossible to define a good set of features beforehand for a wide variety of images; therefore, the best approach is to be adaptive. However, some mechanism should be present to shield the end user from the actual implementation.

One approach is to find an optimal subset of standard features, as Alexandrov et al. [1] did. They used a large number (120) of Gabor filters, from which indices were created using feature selection. Of course, one can use any of a number of texture representation methods. Pentland et al. [12] used Wold components for textures. Kelly et al. [6] used histograms of texture masks evaluated at each pixel in an image and developed a method of approximating and comparing these histograms. Ramesh and Sethi [13] looked at edges, or more precisely, wavelet features extracted at points of high curvature in edge descriptions.

In our approach, the feature extraction and feature selection stages are rolled into one and performed automatically. All that remains is to define a way of using the ASMs to find distances between (classes of) images, say I^A and I^B. There are two possible strategies:

1. Train ASMs A and B on images (or classes of images) I^A and I^B and use these as a descriptor
2. Train an ASM A on image I^A (or class of images) and use the histogram of an image I^B mapped onto ASM A as a descriptor (in mapping an image, each pixel is assigned to a subspace using the window around it)

The second option seems to be preferable, as it defines distances between images I^A and I^B in terms of the content of both images and the size of the regions of similar content. However, it requires mapping each new image (or query image) onto all maps found so far, which can be a computationally intensive task. The first option is computationally much lighter, but potentially discards information on the size of the regions responsible for each subspace in the map. For the image database application, we used the first option, and incorporated a way of at taking the image region sizes at least partially into consideration.

4.1 Comparing ASMs

For the first approach, a distance measure between ASMs A, with n^A subspaces $\left\langle \vec{S}_i^A, \vec{O}_i^A \right\rangle$, and B, with n^B subspaces $\left\langle \vec{S}_j^B, \vec{O}_i^A \right\rangle$, can be defined:

$$\begin{aligned} D(A,B) &= \max(D'(A,B), D'(B,A)) \\ D'(A,B) &= \frac{1}{n^A} \sum_{i=1}^{n^A} \min_{j=1,\ldots,n^B} D''(\vec{S}_i^A, \vec{S}_j^B) \\ D''(\vec{S}_i^A, \vec{S}_j^B) &= \frac{1}{d} \sum_{k=1}^{d} ||\vec{s}_k^{A_i} - \hat{\vec{s}}_k^{A_i}||^2 \end{aligned} \quad (3)$$

where each d-dimensional subspace $\vec{S}$ is spanned by basis vectors $\vec{s}_1, \ldots, \vec{s}_d$ and $\hat{\vec{s}}_k^{A_i}$ is the projection of basis vector k of subspace $\vec{S}_i^A$ onto subspace $\vec{S}_j^B$ (after subtraction of $\vec{O}_j^B$). The idea behind this measure is to find, for each subspace in ASM A, the closest subspace* in B and average

*Experiments were also performed in which a more principled distance measure between subspaces called the *gap* [14] was used: $D''(\vec{S}_i^A, \vec{S}_j^B) = ||\vec{P}_i^A - \vec{P}_j^B||_2$, where $\vec{P} = \vec{S}(\vec{S}^T\vec{S})^{-1}\vec{S}^T$ is the projection matrix onto subspace $\vec{S}$. However, the computational burden of this method was much higher due to the singular value decomposition needed for the calculation of the norm, and results were more or less the same.

Figure 1: Example images from the five sequences, from left to right: `news reader`, `queen`, `cathedral`, `hut` and `flag`. The regions were used only in the KIDS system.

this distance over all subspaces in A. The same is done for B and the maximum is taken, like in the Haussdorf distance.

As said before, the problem with this distance measure is that all information about the size of the regions responsible for the subspaces is discarded. A possible solution for this is to weigh the distance between a subspace $\vec{S}_i^A$ and an ASM B with the relative importance of $\vec{S}_i^A$ in ASM A:

$$D'(A,B) \quad = \quad \frac{1}{n^A}\sum_{i=1}^{n^A} y_i^A \min_{j=1,\ldots,n^B} D''(\vec{S}_i^A, \vec{S}_j^B) \tag{4}$$

There are various ways of expressing the importance y_i^A of subspace $\vec{S}_i^A$. First of all, we can just weigh all subspaces equally: $y_{i,0}^A = 1, \ \forall \vec{S}_i^A$. Two other possible measures used in this work are:

$$y_{i,1}^A \quad = \quad \frac{\vec{h}_i^A}{\sum_{j=1}^{n^A} \vec{h}_j^A} \tag{5}$$

where $\vec{h}_i^A$ is the i^{th} bin of the histogram $\vec{h}^A$ of the mapping of the training image onto ASM A, and

$$y_{i,2}^A \quad = \quad \frac{\vec{h}_i^A(1-\bar{\vec{e}}_i^A)}{\sum_{j=1}^{n^A} \vec{h}_j^A} \tag{6}$$

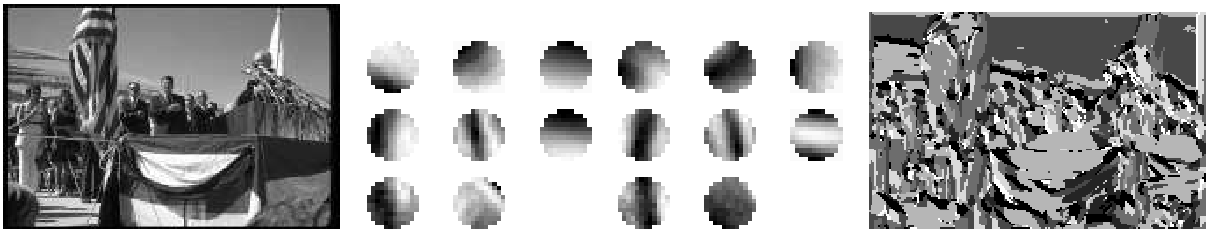
where $\bar{\vec{e}}_i^A$ is the average projection error of the pixels assigned to subspace $\vec{S}_i^A$.

5 Experiments

A small image database (200 images) was created containing images taken randomly from the MPEG7 database and stills from an hour-long video sequence of Sky news. Besides these images, five sequences of similar images of a news reader, Her Royal Highness the Queen, Guildford cathedral, a hut, and a US flag were added. The goal of the experiment was, given an image in a sequence, to find the other images in that sequence. Examples of the images in these sequences are given in figure 1. All 200 database images, originally 24-bit colour, were converted to grey values and histogram stretched before samples were taken.

5.1 ASM-based retrieval

On each individual image, ASMs were trained for various settings of the subspace dimensionality, the number of subspaces n and the sample window size w. To learn about the influence of the episodes, two experiments were run. In the first, for training, 200 episodes were created, each containing 5 samples translated randomly in a $[-5,5]$ pixel range. In the second, episodes contained 15 samples: 5 translated as before, 5 rotated over a $[-45,45]$ degree range and 5 scaled in a range of $[1.0,1.5]$ times the original size. In both cases a round sampling window was used, with a diameter of w pixels. The experiments were run using ASMs with 2D and 4D subspaces, respectively.



(a) (b) (c) (d)

Figure 2: One of the images in the `flag` sequence (a); the first (c) and second (d) basisvectors of a translation-only, $n = 8, w = 12$ ASM trained on that image and the resulting segmentation (d). Note the vertical and horizontal textures in the flag learned in subspaces 5, 4 and 6, respectively.

After training, for each sequence r one image I_r^{tst} was used as a test image. The distance $D(A_r^{tst}, A_i), i = 1, \ldots, 199$ between the ASM A_r^{tst} trained on this image and all other ASMs was calculated using equations 3-5 and the images were ordered by the distance of their ASM to that of the test image. Finally, the ranks of the other images in sequence r, $I_{r,i}^{trn}$ were noted. An example of a trained ASM and the corresponding segmented database image is given in figure 2.

The first thing to note is that for most queries, using subspace weight measure $y_{i,2}^{A}$ (cf. equation 6) gives by far the best results. As an example, figure 3 shows an overview of the average retrieval positions for the `news reader` query, using the three different subspace weighing methods. All results reported from now on have been found using this subspace weight measure.

Table 1 shows the results for the ASMs, using optimal overall settings for each type of ASM. Of course, it is possible to use a large number of settings for the number n of subspaces to use and the sampling window size w. Unfortunately, space does not permit giving these results. For most other settings the overall results were worse, although on individual sequences they would sometimes be slightly better.

5.2 Comparison to KIDS

As a comparison, the same queries were performed using a state-of-the-art system called KIDS [9, 10]. This system can also handle colour features, but for the comparison only texture features (local variance in 9 DCT filtered versions, 8 Gabor filtered versions and 4 wavelet filtered versions of the original image) were used. Basically, KIDS segments each database image using colour and texture information, and stores segment locations, sizes and median feature values. Retrieval is based on training a two-output unit feed-forward neural network to distinguish between a region the user specified in the query image (output 1) and a randomly drawn set of regions in the database (output 2). Any region in the database for which network output 1 is larger than a pre-set threshold T is then labelled as a match. Finally, a similarity measure for entire images is based on a scoring function incorporating both the network output and the size of any matched regions.

The regions we specified for KIDS are shown in figure 1. A network output threshold of $T = 0.5$ was used which gave optimal overall results. The KIDS results are also given in table 1.

5.3 Discussion

The table shows that the ASM method gives promising results, even compared to KIDS. For most of the query images the other images in the sequence are ranked high. Also, training on episodes with translated, rotated and scaled samples pays off for most sequences. A notable exception is the `cathedral` sequence. This is the only sequence consisting of high-resolution images, with areas of fine texture; it was shot using a digital video camera[†]. There is a suspicion the camera might have introduced artefacts into the texture due to aliasing. The ASMs code this texture quite precisely, so that different views of the cathedral give quite different ASMs. Interestingly, this is where KIDS performs best, indicating that the two techniques might be complementary.

To verify that it was indeed the high frequencies in the `cathedral` images that played a role, we re-sized the `cathedral` images to one half their original size, using bi-cubic interpolation, and reran the query (excluding the original `cathedral` images). Performances, also shown in table 1, did indeed improve to a more reasonable level.

This fact, and inspection of the maps, lead us to

[†]The `hut` sequence was shot using the same camera, but does not contain texture as fine as the `cathedral` sequence.

Sequence	Translation invariance ASMs		All-invariance ASMs		KIDS
	2D subspaces, $n = 20, w = 12$	4D subspaces, $n = 8, w = 16$	2D subspaces, $n = 20, w = 8$	4D subspaces, $n = 12, w = 8$	
`news reader`	1,6,7,18	1,2,3,7	1,2,4,12	1,2,3,4	17,19,20,23,25
`queen`	4,10,16,20	4,6,16,22	1,5,20,21	4,6,20,36	14,15,17,21
`cathedral`	55,124,184,198	110,122,179,193	126,192,194,198	110,186,197,198	2,4,5,6
`hut`	2,5,16,50	1,2,3,40	7,9,12,57	1,2,3,83	1,2,3,5
`flag`	1,2,4,5,10	11,28,29,44,73	1,2,3,4,13	1,2,3,8,34	24,25,30,35,50
`cathedral (2)`	12,50,59,92	16,17,25,108	1,22,50,111	4,5,23,97	-

Table 1: Results on the image database retrieval problem, using subspace weight measure $y_{i,2}^{A}$ in the ASM distance calculations. Numbers shown are ranks of the images sought in the retrieval order. For each ASM column, the optimal settings (over all queries) for the number of subspaces n and sampling window size w were picked. The last line gives performances on the `cathedral` query using subsampled versions of the sequence.

believe that the ASMs are more succesful in coding large-scale image structure (i.e. edges) than they are in coding texture very precisely. If texture is modelled precisely, as it was for the original `cathedral` query, the same texture under different viewing angles will create two subspaces with a relatively large distance. The idea that ASMs code shape more than texture is also supported by some of the query returns, such as the ones shown in figure 4. Note how for the `hut` query, which contains a large number of horizontal and vertical edges, images containing quite different textures but identical edges are found.

6 Discussion

Although we were able to find reasonable overall settings for the dimensionality of the subspaces, the number of subspaces n and the sample window size w, the actual optimal settings were different for each query. This is only to be expected, as discriminative features will be found on different scales and in different quantities in each image. For easy applicability, though, the settings would have to automated automatically, for example by:

- automatically finding a good subspace dimensionality, per subspace, by demanding that a certain percentage of the variance in the original data be explained by the subspace (e.g. 80%).
- optimising the number of subspaces to use by applying another clustering mechanism, e.g. ISODATA or mean shift clustering [3].

The only parameter that cannot be optimised well is the window size, as it needs to be fixed throughout the database to allow ASM comparison. However, one could imagine using ASMs with different window sizes and combining the resulting distance measures.

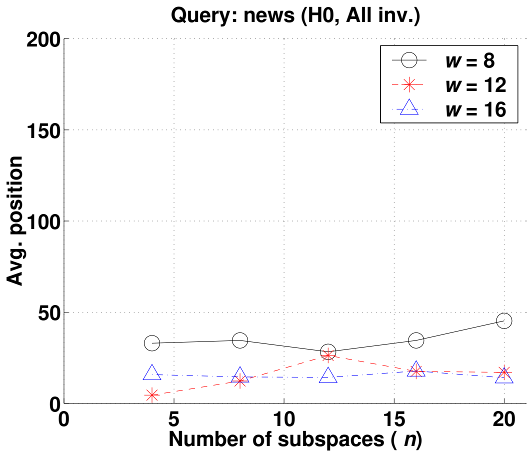
Finally, to be able to describe images better and circumvent the noise enhancement problem due to normalisation discussed earlier (section 3.1), normalisation should be dropped. This means that learning will become more cumbersome, as the origins of each subspace would have to be found as well. A few experiments in this direction indicate that using the origin usually works, but that training does not converge as well as on normalised samples. Perhaps a different distance measure between a point and a subspace should be used, such as the combination between distance-in-feature-space and distance-from-feature-space proposed by Moghaddam and Pentland [11] and Sung and Poggio [15].

Note that dropping normalisation is a prerequisite for extending the system to use color information. The latter is quite desirable, as the literature on image databases clearly shows color information to be the most important to use in image database retrieval [2].

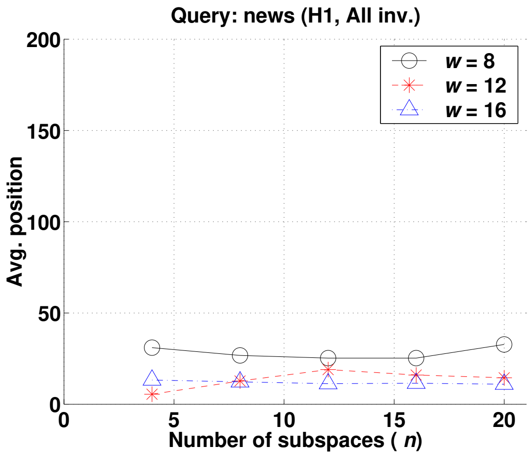
7 Conclusions

An adaptive image description method was presented, which uses subspaces to describe translated, rotated and/or scaled versions of patches extracted from images. The resulting description (ASM) can be used to segment images, but in this paper the focus was on using it for image description purposes.

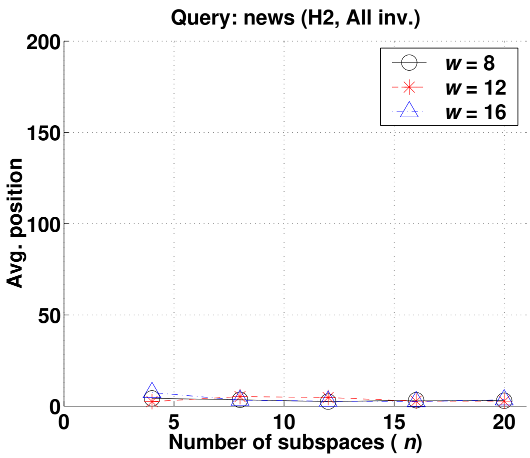
Given the distance measure between ASMs introduced in section 4, ASMs can be used to compare images. This method was shown to be applicable to image database retrieval problems. Although the database used was small, the results were quite good compared to a state-of-the-art system, given the fact that only texture and edge



(a) $y^A_{i,0}$



(b) $y^A_{i,1}$



(c) $y^A_{i,2}$

Figure 3: Plots of average retrieval positions (the lower, the better) for the `news reader` query, using all-invariance, 4D ASMs, for various settings of the number of subspaces n and the sample window size w, and three different subspace weight measures.

(a)

(b)

(c)

Figure 4: Query returns, ordered from top-left to bottom-right, for the (a) `news reader`, (b) `queen` and (c) `hut` queries, using an all-invariance ASM with 4D subspaces, $n = 12$ and $w = 8$ and subspace weight $y^A_{i,2}$. The original query images are always shown in the top-left.

information is used (since the average grey value is removed from the samples).

In the experiments performed some important settings, such as the number of subspaces to use within each ASM and the sampling window size, were optimised by hand. For easier applicability, it would be interesting to investigate optimising these automatically, or perhaps to use a multi-scale approach. Another possible extension is the use of colour information. Finally, we intend to investigate the use of other subspace-finding algorithms, such as independent component analysis (ICA) or even non-linear methods such as multi-dimensional scaling (MDS) or diabolo networks, which might be more applicable to some problems.

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. Finally we would like to thank Kieron Messer for helping us in comparing the ASM to KIDS.

References

[1] A.D. Alexandrov, W.Y. Ma, A. El Abbadi, and B.S. Manjunath. Adaptive filtering and indexing for image databases. In W. Niblack and R.C. Jain, editors, *Storage and retrieval for image and video databases III*, volume 2420 of *Proceedings of SPIE*, Bellingham, WA, 1995. SPIE, SPIE.

[2] S. Antani, R. Kasturi, and R. Jain. Pattern recognition methods in image and video databases: past, present and future. In A. Amin, D. Dori, P. Pudil, and H. Freeman, editors, *Advances in Pattern Recognition, Proceedings of SPR'98 and SSPR'98*, pages 31–53, Berlin, 1998. IAPR, Springer-Verlag.

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

[4] G.E. 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] N. Kambhatla and T.K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

[6] P.M. Kelly and M. Cannon. Query by image example: the CANDID approach. In W. Niblack and R.C. Jain, editors, *Storage and retrieval for image and video databases III*, volume 2420 of *Proceedings of SPIE*, pages 239–248, Bellingham, WA, 1995. SPIE, SPIE.

[7] 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.

[8] H. Lu, Y. Fainman, and R. Hecht-Nielsen. Image manifolds. In N.M. Nasrabadi and A.K. Katsaggelos, editors, *Applications of Artificial Neural Networks in Image Processing III, Proceedings of SPIE*, volume 3307 of *SPIE Proceedings Series*, pages 52–63, Bellingham, WA, 1998. SPIE, SPIE.

[9] K. Messer. *Automatic image database retrieval system using adaptive colour and texture descriptors.* PhD thesis, University of Surrey, Guildford, Surrey, 1999.

[10] K. Messer and J. Kittler. A region-based image database system using colour and texture. *Pattern Recognition Letters*, 20(11-13):1323–1330, 1999.

[11] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):696–710, 1997.

[12] A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: tools for content-based manipulation of image databases. Technical report, Perceptual Computing Section, The Media Laboratory, MIT, Cambridge, MA, 1994.

[13] N. Ramesh and I. Sethi. Feature identification as an aid to content-based image retrieval. In W. Niblack and R.C. Jain, editors, *Storage and retrieval for image and video databases III*, volume 2420 of *Proceedings of SPIE*, Bellingham, WA, 1995. SPIE, SPIE.

[14] G.W. Stewart. Error and perturbation bounds for subspaces associated with certain eigenvalue problems. *SIAM review*, 15:727–764, 1973.

[15] K.-K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, 1998.

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