Proceedings of ICAR 2003
The 11th International Conference on Advanced Robotics
Coimbra, Portugal, June 30 - July 3, 2003

Performance Evaluation of Incremental Eigenspace Models for Mobile Robot Localization

R.Freitas[†,‡], J.Santos-Victor[‡], M.Sarcinelli-Filho[†], T.Bastos-Filho[†]

[‡]Instituto de Sistemas e Robótica
Instituto Superior Técnico
Lisboa - Portugal
{roger,jasv}@isr.ist.utl.pt

[†]Departamento de Engenharia Elétrica
Universidade Federal do Espírito Santo
Vitória, ES - Brasil
{roger,mario.sarcinelli,tfbastos}@ele.ufes.br

Abstract

We address the problem of robot localization based on eigenspace representations of a collection of views of the workspace, often referred to as appearance based methods. We study several approaches to build such eigenspaces in an incremental fashion (subspace tracking), thereby allowing the simultaneous localization and mapping of a mobile vehicle. We see how different ways of determining whether or not new information (images) should be included in the eigenspace model lead to different error values and model size (complexity). The performance evaluation allows us to know which criterion and parameters to use in a specific application to meet error specifications subject to computational constraints.

1 Introduction

We address the problem of mobile robot localization based on video images. Rather than localizing from known geometric features we rely on appearance-based methods to solve this problem [3, 7]. The idea consists in representing the robot environment as a topological map, storing a (usually large) set of landmark images. To speedup the comparison of the robot views with these landmark images, it is advantageous to use low-dimensional approximations of the space spanned by the original image set. One example is to use principal component analysis (PCA) that uses the set of input images to extract an orthonormal basis (or model) of a lower dimensional subspace (eigenspace) that approximates the input images.

When building such a map, images taken from positions close to each other are likely to be strongly correlated. Eigenspace models are then used to build a compact representation of the set of images. In the traditional approach to calculate these eigenspace models, known as *batch method*, the robot must capture all the images needed to build the map and then, using either eigenvalue decomposition of the covariance matrix or singular value decomposition of the data matrix, calculate the model. This approach has some drawbacks. Since the entire set of images is necessary to build the model, it is impossible to make the robot build a map while visiting new positions. Update of the existing model is only possible from scratch, which means that original images must be kept in order to update the model, requiring a lot of storage capability.

To overcome these problems, some authors [8, 2, 4, 5, 6] proposed algorithms that build the eigenspace model incrementally (sometimes referred to as subspace tracking in the communications literature). The basic idea behind these algorithms is to start with an initial subspace (described by a set of eigenvectors and associated eigenvalues) and update the model in order to represent new acquired data. This approach allows the robot to perform simultaneous localization and map building. There is no need to build the model from scratch each time a new image is added to the map, and it is thus easier to deal with dynamic environments. Whenever the robot acquires a new image, the first step consists in determining whether or not this image is well represented by the existing subspace model. The component of the new image that is not well represented by the current model is added to the basis as a new vector. Then, all vectors in the basis are "rotated" in order to reflect the new distribution of the energy in the system.

Recently, Artač et al [1] improved Hall's algorithm [4] by suggesting a way of updating the low dimensional projections of the images, allowing to discard the image as soon as the model has been updated. New data represents an increase in the model dimensionality and, as a consequence, in the computational cost. Therefore, we look for criteria to guide the decision on whether or not to increase the dimensionality of the model. Deciding not to increase the dimensionality of the model, we are discarding information that will affect the accuracy of the representations.

In this work we describe the results achieved by implementing and testing the incremental PCA algorithm presented in [4] and improved by [1]. Although these methods have been proposed by other authors, very little or no performance evaluation has been done regarding criteria that affect the dimensionality of the model. In this paper, we compare the set of methods usually chosen to decide whether or not a new vector must be added to the existing basis.

Our implementation is slightly different from the Artač's algorithm, as we calculate the covariance matrix in the way proposed by [8]. The implementation was done in MATLAB®. We used a set of omnidirectional images taken from a corridor in our lab to test the algorithm. We also show preliminary results achieved when testing the algorithm for mobile robot localization. In Section 2 we describe the traditional approach for building eigenspace models and the incremental approach. In Section 3 we show results in terms of reconstruction error and dimensionality of the model when using different criteria to control the update of the model. We conclude outlining the work done and presenting some directions to the future.

2 Principal Component Analysis

In the traditional (batch) approach, the images taken are resized to column vectors $\mathbf{x}_i \in \mathbb{R}^{m\times 1}$; i=1…n, where m is the number of pixels in the image and n is the number of images. Then, the eigenspace model is calculated by solving the EVD of the covariance matrix $C \in \mathbb{R}^{m\times m}$ composed as

$$C = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T \tag{1}$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the mean image vector.

The eigenvectors $\mathbf{u}_i, i = 1\ldots n$ associated to nonzero eigenvalues of the matrix C form an orthonormal basis spanning at most $min(m,n)$ dimensions. We can discard eigenvectors associated with small eigenvalues and keep just $k \ll min(m,n)$ eigenvectors which will form the basis. Each one of the n images is then projected in this subspace, generating n k-dimensional points.

2.1 Incremental PCA

The algorithm proposed by [4] assumes that we already have an eigenspace model built and we want to update this model with new data (a new image). The first step consists in determining whether or not this image is well represented by the existing subspace model. We then project the new image to the current basis. The component of the new image that is not well represented by the current model is added to the basis as a new vector. Then, all vectors in the basis are "rotated" in order to reflect the new distribution of the energy in the system. The rotation is represented by a matrix of eigenvectors obtained by the eigenvalue decomposition of a special matrix. This matrix is composed taking into account the current eigenvalue distribution and the projection of the new data in the current basis. These steps are detailed in the equations below.

The existing eigenspace model can be represented by $\Omega = (\bar{\mathbf{x}}, \mathbf{U}, \mathbf{\Lambda}, \mathbf{n})$, defining the mean vector, the set of eigenvectors kept in the basis, the corresponding eigenvalues and the number of images respectively.

We calculate the initial basis as proposed by [8]. This method allows us to calculate an implicit, or "low dimensional" covariance matrix:

$$\tilde{\mathbf{C}} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{X} - \bar{\mathbf{x}})^T(\mathbf{X} - \bar{\mathbf{x}}) \tag{2}$$

The eigenvalues and eigenvectors of C and $\tilde{C}$ are related by

$$\lambda_j = \tilde{\lambda_j} \tag{3}$$

and

$$\mathbf{u}_j = (n\tilde{\lambda_j})^{-1/2}(\mathbf{X} - \bar{\mathbf{x}})\tilde{\mathbf{u_j}},\ j\text{=}1\ldots n, \tag{4}$$

where λ_j and $\tilde{\lambda_j}$ are the j-th eigenvalues of $\mathbf{C}$ and $\tilde{\mathbf{C}}$, $\mathbf{u}_j$ and $\tilde{\mathbf{u}_j}$ are the j-th eigenvectors of $\mathbf{C}$ and $\tilde{\mathbf{C}}$, n is the number of images in the initial set, $\mathbf{X} = [\mathbf{x}_1\mathbf{x}_2\cdots\mathbf{x}_n]$ is the data matrix, and $\bar{\mathbf{x}}$ is the mean vector. In practice, additional care must be taken in order to avoid numerical instabilities: very small eigenvalues and eigenvectors associated must be discarded before calculating the basis for matrix $\mathbf{C}$. This approach allows a reduction in the problem complexity by solving an eigenvalue problem in $\mathbb{R}^{n\times n}$ instead of $\mathbb{R}^{m\times m}$, given that $n \ll m$ when working with images. Again, we can retain the k most representative eigenvectors, with $k \leq n$. With the initial eigenspace model calculated, we follow the method proposed by [4] and improved by [1], which we describe next.

To update the existing basis to take a new image $\mathbf{x}_{n+1}$ into account, we update the mean:

$$\bar{\mathbf{x}}' = \frac{1}{n+1}(n\bar{\mathbf{x}} + \mathbf{x}_{n+1}) \tag{5}$$

Then, we project the new image to the current basis $\mathbf{U}$. $\mathbf{a}_{n+1}$ is the vector that represents the new image in the current basis:

$$\mathbf{a}_{n+1} = \mathbf{U}^T(\mathbf{x}_{n+1} - \bar{\mathbf{x}}) \tag{6}$$

The residual vector, orthogonal to each basis vector $\mathbf{u}_i$, is given by:

$$\mathbf{h}_{n+1} = (\mathbf{U}\mathbf{a}_{n+1} + \bar{\mathbf{x}}) - \mathbf{x}_{n+1} \tag{7}$$

To update the basis we use the normalized residual vector:

$$\hat{\mathbf{h}}_{n+1} = \begin{cases} \frac{\mathbf{h}_{n+1}}{\|\mathbf{h}_{n+1}\|_2} \text{ if } \|\mathbf{h}_{n+1}\|_2 \neq \epsilon \\ \mathbf{0} \text{ otherwise} \end{cases} \tag{8}$$

where ϵ is a threshold in the range of machine precision.

The new eigenvectors are obtained by appending $\hat{\mathbf{h}}_{n+1}$ to the current basis and rotating them:

$$\mathbf{U}' = [\mathbf{U}, \hat{\mathbf{h}}_{n+1}]\mathbf{R}. \qquad (9)$$

To update the representations $\mathbf{a}_i, i = 1 \ldots n$, we reconstruct each image using the old basis:

$$\mathbf{x}_i = \mathbf{U}\mathbf{a}_i + \bar{\mathbf{x}}, i = 1 \ldots n, \qquad (10)$$

and project them to the new basis, thus generating new low dimensional representations $\mathbf{a}_i, i = 1 \ldots n+1$:

$$\mathbf{a}_i = (\mathbf{U}')^T(\mathbf{x}_i - \bar{\mathbf{x}}'), i = 1 \ldots n+1. \qquad (11)$$

Once the basis is updated, we can discard the original image and the old basis. The $n+1$ images are now represented in a $(k+1)$-dimensional basis.

2.2 The rotation matrix R

In order to better understand the role of matrix **R** presented in Section 2.1, we give a two-dimensional example (see Figure 1) . Given the set of points in the

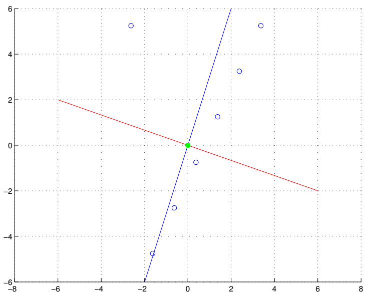

Figure 1: *Eigenvectors calculated by batch method.*

figure, we calculate principal components by using the batch method.

Suppose now that we want to incrementally construct the basis using the same data set. The initial basis can be calculated using the six aligned points. Then we must update the basis in order to represent the point that is not aligned. The mean is updated, the new point is projected to the current basis, the normalized residue vector is calculated and appended to the current basis. Figure 2 shows the eigenvectors obtained *before* applying the rotation **R** . Comparing Figure 2 and Figure 1 we can see that is necessary to apply a rotation in order to correctly align the eigenvectors.

2.2.1 Computing rotation R.

The columns of the rotation matrix **R** are the eigenvectors of a matrix **D**, which is composed by taking into account the

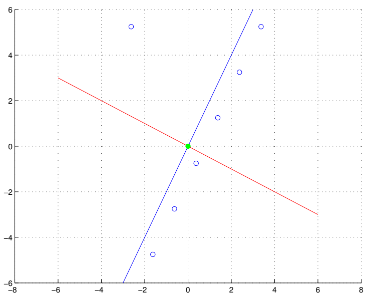

Figure 2: *Eigenvectors before applying the rotation.*

current eigenvalue distribution and the projection of the new data in the current basis. For a complete explanation on composing matrix **D**, refer to [4]. Matrix **D** is given by:

$$D = \frac{n}{n+1}\begin{bmatrix} \mathbf{\Lambda} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + \frac{n}{(n+1)^2}\begin{bmatrix} \mathbf{a}\mathbf{a}^T & \gamma\mathbf{a} \\ \gamma\mathbf{a}^T & \gamma^2 \end{bmatrix}, \quad (12)$$

where γ represents the portion of the new image that is not well represented by the current basis:

$$\gamma = \hat{\mathbf{h}}_{n+1}^T(\mathbf{x}_{n+1} - \bar{\mathbf{x}}). \qquad (13)$$

Solving the eigenproblem:

$$\mathbf{D}\mathbf{R} = \mathbf{R}\mathbf{\Lambda}', \qquad (14)$$

we obtain **R**.

2.3 Model's Dimension

The specific application or the system's storage capability can make us to keep the dimensionality low instead of adding a new vector to the basis. On the other hand, keeping low the dimensionality will reduce the accuracy of the representations. We will need a criterion to decide whether or not to keep the dimensionality of the basis. We tested four criteria to deal with the model's dimensionality:

a) Adding a new vector whenever the size of the residue vector (Equation 7) exceeds an absolute threshold;

b) Adding a new vector when the percentage of energy carried by the last eigenvalue in the total energy of the system exceeds an absolute threshold, or equivalently, defining a percentage of the total energy of the system that will be kept in each update;

c) Discarding eigenvectors whose eigenvalues are smaller then a percentage of the first eigenvalue;

d) Keeping the dimensionality constant.

In the next section we will present results of the performance evaluation of these criteria.

3 Experimental Results

We tested the algorithm using a sequence of 53 gray level images, taken every 20cm apart with a mobile robot equipped with an omnidirectional camera. The robot used in our experiments was a TRC Labmate (see Figure 3). It is a differential-driven robot equipped with an on board computer (Pentium IV - 1.5 GHz - 384 Mbytes of RAM). The vision system consists of an omnidirectional catadioptric system, composed by the camera pointing upwards to a spherical mirror. This system is mounted on top of the mobile robot with its axis coincident to the platform's rotation axis. The set of omnidirectional images were taken in a corridor of our lab. The implementation was done following the procedure described in Section 2.1. Actually, we implemented four versions of the algorithm, changing the criterion used to decide whether or not to increase the dimensionality of the model. We started by calculating the initial model

Figure 3: *The mobile robot Labmate.*

from a subset of the images of the sequence, then we presented the next images, one by one, updating the model in each step. The same procedure was used in the four versions of the algorithm.

In Figures 4 to 7 we show the results in terms of reconstruction error (root mean squared error - RMS) and the resulting dimension of the model, as a function of the norm of the residue vector (Figure 4); as a function of the fraction of the total energy contained in the eigenvalues (Figure 5); as a function of the fraction of the energy contained in the most important eigenvalue (Figure 6); and as a function of a threshold in the number of vectors kept in the basis (Figure 7). The error is calculated as follows:

$$Error = \sqrt{\frac{\sum_{i=1}^{m}(x_i - \hat{x}_i)^2}{m}}, \qquad (15)$$

where x_i is the brightness of the i-th pixel of an original image, $\hat{x}_i$ is the brightness of the i-th pixel of the

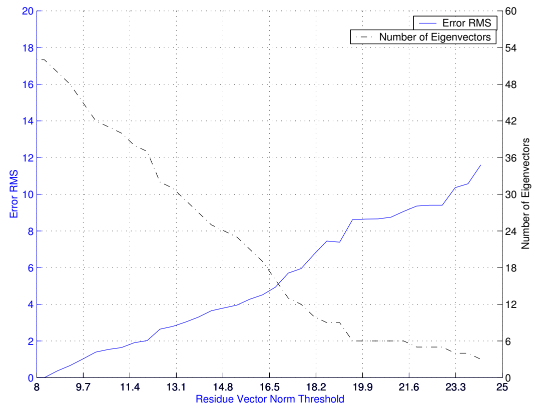

Figure 4: *Error RMS and Number of Eigenvectors obtained when using the norm of the residue vector as a threshold.*

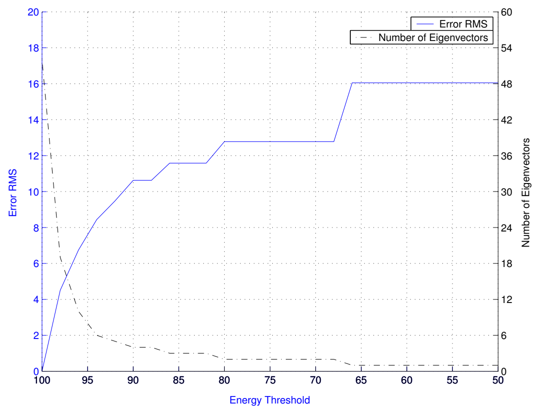

Figure 5: *Error RMS and Number of Eigenvectors obtained when using a fraction of the total energy as a threshold.*

reconstructed image and m is the number of pixels in the image.

In Figure 5, a threshold of 95% means that in each update, 95% of the *actual* energy of the system will be retained. The eigenvectors associated with eigenvalues outside this range will be discarded. In Figure 6, a threshold of 5% means that in each update, only the eigenvectors associated with eigenvalues whose energy represents 5% or more of the energy contained in the most important eigenvalue will be kept. The eigenvectors associated with eigenvalues outside this range will be discarded.

The results obtained confirmed the general idea that more accurate results, in terms of image reconstruction, are obtained by increasing the number of vectors in the basis, at the cost of additional computational complexity.

There is a difference between the two energy methods. The method based on a fraction of the energy

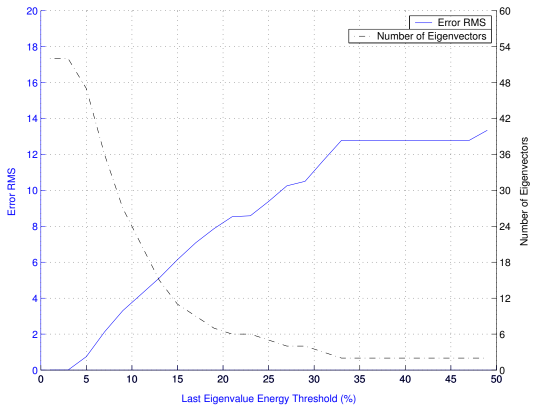

Figure 6: *Error RMS and Number of Eigenvectors obtained when using a fraction of the energy of the most important eigenvalue as a threshold.*

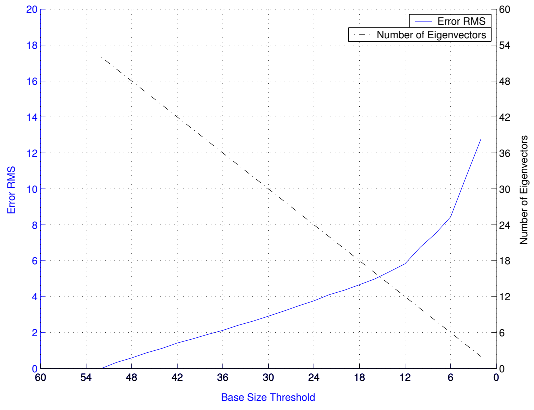

Figure 7: *Error RMS and Number of Eigenvectors obtained when making constant the number of vectors in the basis.*

of the most important eigenvalue seems to be more "conservative" on discarding eigenvectors. This is due to the fact that the first eigenvalue remains approximately constant over the entire experiment (and it may even increase in some cases). The criterion of keeping a fixed number of vectors in the basis seems to be more suitable when the application requires a fixed time response or limited computational resources are available.

A comparison can also be made regarding the reconstruction error obtained by the incremental method and the batch method. Figure 8 shows the reconstruction error - RMS for the set of 53 images obtained by the batch method and the incremental method. The threshold was based in a fixed number of vectors in the basis equal to 25. The batch method is represented with the same number of vectors to enable the comparison. We can see that the incremental method approximates the batch method in a reasonable manner.

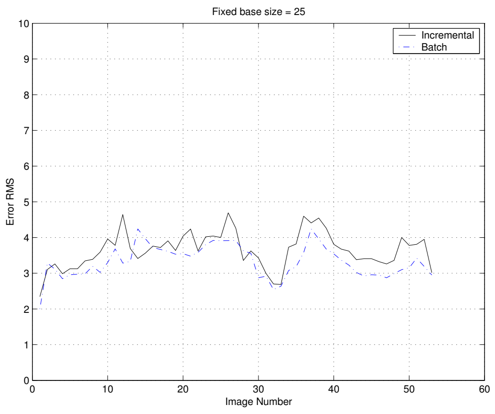

Figure 8: *Reconstruction error RMS for each image in the set when using 25 eigenvectors as a fixed threshold.*

Figure 9 shows the first image of the set and the results achieved when the reconstruction was made with a basis of 10 and 25 eigenvectors by the batch and the incremental methods.

In terms of localization, the image reconstruction error may not be the most representative error metric. For that purpose we conducted a set of preliminary experiments with the mobile robot to assess the quality of the localization as a function of the chosen criterion and parameters.

We used a set of 103 omnidirectional images of the same corridor. The basis was built with images 1, 4, 7, 10,... and so on. We then tried to localize the other images in the set, projecting them to the basis and measuring the distance to the projections of the images used to build the subspace. We considered a correct match only if the image could be correctly localized with respect to the two nearest reference images. The error rate is defined as the number of incorrect matches divided by the total number of tested images. Figure 10 shows the error rate in localization and the number of eigenvectors obtained in the basis as a function of the norm of the residue vector.

Invariance to rotation could be obtained, for example, by transforming the omnidirectional images to panoramic images and working in frequency domain.

4 Conclusions

In this paper we addressed the problem of mobile robot localization using appearance-based methods. These methods are used to build eigenspaces models that hold low dimensional representations of the images used for constructing a topological map. We explained the difference between the batch and the incremental way of constructing such a model. We implemented an incremental version of this method

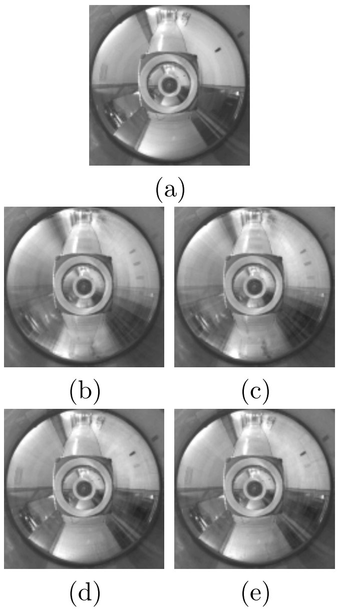

Figure 9: *The original first image (a), reconstructed image using 10 eigenvectors by batch method (b), reconstructed image using 10 eigenvectors by incremental method (c), same using 25 eigenvectors by batch method (d), same using 25 eigenvectors by incremental method (e).*

that allows simultaneous localization and map building and discussed some criteria regarding the dimensionality of the model.

The performance curves presented in this paper can be used in various ways. Firstly, they provide a mean for comparing different criteria to build incremental eigenspaces for localization. Secondly, given concrete application specifications or requirements, in terms of accuracy and computational load, the performance curves allow us to decide which method to use and which parameters to select.

In the future we plan to exploit other strategies as to control the update process of the subspace model, possibly taking into account the tradeoff between the accuracy gain arising from incorporating new data and the associated increase in computational cost.

Acknowledgments

This work is supported by *CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (Brazil) and project POSI/SRI/41506/261 Omnisys through *Fundação para a Ciência e Tecnologia* (Portugal).

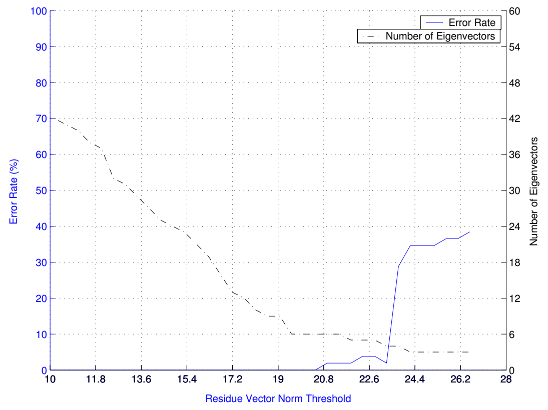

Figure 10: *Localization error rate and Number of Eigenvectors obtained when using the norm of the residue vector as a threshold.*

References

[1] M. Artač, M. Jogan and A. Leonardis, "Mobile Robot Localization Using an Incremental Eigenspace Model," *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*,Washington DC,May 2002.

[2] S. Chandrasekaran, B. S. Manjunath, Y. F. Wang, J. Winkeler and H. Zhang, "An eigenspace update algorithm for image analysis," *Graphical Models and Image Processing*, Vol. 59, pp. 321-332, Sept. 1997.

[3] J. Gaspar, N. Winters and J. Santos-Victor, "Vision-based Navigation and Environmental Representations with an Omni-directional Camera," *In IEEE Transactions on Robotics and Automation,* Vol. 16, Number 6, pp. 890-898, December 2000.

[4] P. Hall, D. Marshall and R. Martin, "Incremental eigenanalysis for classification," *British Machine Vision Conference*, Vol. 14, pp. 286-295, Sept. 1998.

[5] P. Hall, D. Marshall and R. Martin, "Merging and splitting eigenspace models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, Number 6, pp. 1042-1048, 2000.

[6] P. Hall, D. Marshall and R. Martin, "Adding and subtracting eigenspaces with eigenvalue decomposition and singular value decomposition," *Image and Vision Computing*, 20 (13-14), pp. 1009-1016, 2002.

[7] A. Leonardis and H. Bischof, "Robust recognition using eigenimages," *Computer Vision and Image Understanding*, Vol. 78, Number 1, pp. 99-118, 2000.

[8] H. Murakami and B.V.K.V. Kumar, "Efficient calculation of primary images from a set of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(5), pp. 511-515, 1982.