

Omnidirectional Vision for Appearance-based Robot Localization

B.J.A. Kröse, N. Vlassis, and R. Bunschoten

Real World Computing Partnership, Novel Functions Laboratory SNN, Department of Computer Science, University of Amsterdam,
Kruislaan 403, NL-1098 SJ Amsterdam, The Netherlands
{krose,vlassis,bunschot}@science.uva.nl

Abstract. Mobile robots need an internal representation of their environment to do useful things. Usually such a representation is some sort of geometric model. For our robot, which is equipped with a panoramic vision system, we choose an appearance model in which the sensoric data (in our case the panoramic images) have to be modeled as a function of the robot position. Because images are very high-dimensional vectors, a feature extraction is needed before the modeling step. Very often a linear dimension reduction is used where the projection matrix is obtained from a Principal Component Analysis (PCA). PCA is optimal for the reconstruction of the data, but not necessarily the best linear projection for the localization task. We derived a method which extracts linear features optimal with respect to a risk measure reflecting the localization performance. We tested the method on a real navigation problem and compared it with an approach where PCA features were used.

1 Introduction

An internal model of the environment is needed to navigate a mobile robot optimally from a current state toward a desired state. Such models can be topological maps, based on labeled representations for objects and their spatial relations, or geometric models such as polygons or occupancy grids in the task space of the robot.

A wide variety of probabilistic methods have been developed to obtain a robust estimate of the location of the robot given its sensory inputs and the environment model. These methods generally incorporate some observation model which gives the probability of the sensor measurement given the location of the robot and the parameterized environment model. Sometimes this parameter vector describes *explicit* properties of the environment (such as positions of landmarks [8] or occupancy values [4]) but can also describe an *implicit* relation between a sensor pattern and a location (such as neural networks [6], radial basis functions [10] or look-up tables [2]).

Our robot is equipped with a panoramic vision system. We adopt the implicit model approach: we are not going to estimate the parameters of some sort of CAD model but we model the relation between the images and the robot location directly (*appearance modeling*).

In section 2 we describe how this model is used in a Markov localization procedure. Then we discuss the problem of modeling in a high dimensional image space and describe the standard approach for linear feature extraction by Principal Component Analysis (PCA). In order to evaluate the method we need a criterion, which is discussed in section 5. The criterion can also be used to find an alternative linear projection: the supervised projection. Experiments on real robot data are presented in sections 6 and 7 where we compare the two linear projection methods.

2 Probabilistic appearance-based robot localization

Let \mathbf{x} be a stochastic vector (e.g., 2-D or 3-D) denoting the robot position in the workspace. Similar to [1] we employ a form of *Markov localization* for our mobile robot. This means that at each point in time we have a belief where the robot is indicated by a probability density $p(\mathbf{x})$.

Markov localization requires two probabilistic models to maintain a good position estimate: a *motion* model and an *observation* model.

The motion model describes the effect a motion command has on the location of the robot and can be represented by a conditional probability density

$$p(\mathbf{x}_t|u, \mathbf{x}_{t-1}) \quad (1)$$

which determines the distribution of \mathbf{x}_t (the position of the robot after the motion command u) if the initial robot position is \mathbf{x}_{t-1} .

The observation model describes the relation between the observation, the location of the robot, and the parameters of the environment. In our situation the robot takes an omnidirectional image \mathbf{z} at position \mathbf{x} . We consider this as a realization of a stochastic variable \mathbf{z} . The *observation model* is now given by the conditional distribution

$$p(\mathbf{z}|\mathbf{x}; \boldsymbol{\theta}), \quad (2)$$

in which the parameter vector $\boldsymbol{\theta}$ describes the distribution and reflects the underlying environment model.

Using the Bayes' rule we can get an estimate of the position of the robot after observing \mathbf{z} :

$$p(\mathbf{x}|\mathbf{z}; \boldsymbol{\theta}) = \frac{p(\mathbf{z}|\mathbf{x}; \boldsymbol{\theta})p(\mathbf{x})}{\int p(\mathbf{z}|\mathbf{x}; \boldsymbol{\theta})p(\mathbf{x})d\mathbf{x}}. \quad (3)$$

Here $p(\mathbf{x})$ gives the probability that the robot is at \mathbf{x} before observing \mathbf{z} . Note that $p(\mathbf{x})$ can be derived using the old information and the motion model $p(\mathbf{x}_t|u, \mathbf{x}_{t-1})$ repeatedly. If both models are known we can combine them and decrease the motion uncertainty by observing the environment again.

In this paper we will focus on the observation model (2). In order to estimate this model we need a dataset consisting of positions \mathbf{x} and corresponding observations \mathbf{z} ¹. We are now faced with the problem of modeling data in a high-dimensional space, particularly since the dimensionality of \mathbf{z} (in our case the omnidirectional images) is high. Therefore the dimensionality of the sensor data has to be reduced. Here we restrict ourselves to linear projections, in which the image can be described as a set of linear features. We will start with a linear projection obtained from a Principal Component Analysis (PCA), as is usually done in appearance modeling [5]

3 Principal Component Analysis

Let us assume that we have a set of N images $\{\mathbf{z}_n\}$, $n = 1, \dots, N$. The images are collected at respective 2-dimensional robot positions $\{\mathbf{x}_n\}$. Each image consists of d pixels and is considered as a d -dimensional data vector. In a Principal Component Analysis (PCA) the eigenvectors of the covariance matrix of an image set are computed and used as an orthogonal basis for representing individual images. Although, in general, for perfect reconstruction all eigenvectors are required, only a few are sufficient for visual recognition. These eigenvectors constitute the q , ($q < d$) dimensions of the *eigenspace*. PCA projects the data onto this space in such a way that the projections of the original data have uncorrelated components, while most of the variation of the original data set is preserved.

First we subtract from each image the average image over the entire image set, $\bar{\mathbf{z}}$. This ensures that the eigenvector with the largest eigenvalue represents the direction in which the variation in the set of images is maximal. We now stack the N image vectors to form the rows of an $N \times d$

¹ In this paper we assume we have a set of positions and corresponding observations: our method is *supervised*. It is also possible to do a simultaneous localization and map building (SLAM). In this case the only available data is a stream of data $\{\mathbf{z}^{(1)}, u^{(1)}, \mathbf{z}^{(2)}, u^{(2)}, \dots, \mathbf{z}^{(T)}, u^{(T)}\}$ in which u is the motion command to the robot. Using a model about the uncertainty of the motion of the robot it is possible to estimate the parameters from these data [8].

image matrix \mathbf{Z} . The numerically most accurate way to compute the eigenvectors from the image set is by taking the singular value decomposition [7] $\mathbf{Z} = \mathbf{U}\mathbf{L}\mathbf{V}^T$ of the image matrix \mathbf{Z} , where \mathbf{V} is a $d \times q$ orthonormal matrix with columns corresponding to the q eigenvectors \mathbf{v}_i with largest eigenvalues λ_i of the covariance matrix of \mathbf{Z} [3].

These eigenvectors \mathbf{v}_i are now the linear features. Note that the eigenvectors are vectors in the d -dimensional space, and can be depicted as images: the *eigenimages*. The elements of the $N \times q$ matrix $\mathbf{Y} = \mathbf{Z}\mathbf{V}$ are the projections of the original d -dimensional points to the new q -dimensional eigenspace and are the q -dimensional feature values.

4 Observation model

The linear projection gives us a feature vector \mathbf{y} , which we will use for localization. The Markov localization procedure, as presented in Section 2, is used on the feature vector \mathbf{y} :

$$p(\mathbf{x}|\mathbf{y}; \boldsymbol{\theta}) = \frac{p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})p(\mathbf{x})}{\int p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})p(\mathbf{x})d\mathbf{x}}, \quad (4)$$

where the denominator is the marginal density over all possible \mathbf{x} . We now have to find a method to estimate the observation model $p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})$ from a dataset $\{\mathbf{x}_n, \mathbf{y}_n\}, n = 1, \dots, N$.

We used a kernel density estimation or Parzen estimator. In a Parzen approach the density function is approximated by a sum of kernel functions over the N data points from the training set. Note that in a strict sense this is not a ‘parametric’ technique in which the parameters of some pre-selected model are estimated from the training data. Instead, the training points themselves as well as the chosen kernel width may be considered as the parameter vector $\boldsymbol{\theta}$. We write $p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})$ as

$$p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta}) = \frac{p(\mathbf{y}, \mathbf{x}; \boldsymbol{\theta})}{p(\mathbf{x}; \boldsymbol{\theta})} \quad (5)$$

and represent each of these distribution as a sum of kernel functions:

$$p(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^N g_y(\mathbf{y} - \mathbf{y}_n)g_x(\mathbf{x} - \mathbf{x}_n) \quad (6)$$

$$p(\mathbf{x}; \boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^N g_x(\mathbf{x} - \mathbf{x}_n). \quad (7)$$

where

$$g_y(\mathbf{y}) = \frac{1}{(2\pi)^{q/2}h^q} \exp\left(-\frac{\|\mathbf{y}\|^2}{2h^2}\right) \quad \text{and} \quad g_x(\mathbf{x}) = \frac{1}{2\pi h^2} \exp\left(-\frac{\|\mathbf{x}\|^2}{2h^2}\right) \quad (8)$$

are the q - and two-dimensional Gaussian kernel, respectively. For simplicity in our experiments we used the same width h for the g_x and g_y kernels.

5 Feature representation

As is made clear in the previous sections, the performance of the localization method depends on the linear projection, the number of kernels in the Parzen model, and the kernel widths. First we discuss two methods with which the model can be evaluated. Then we will describe how a linear projection can be found using the evaluation.

5.1 Expected localization error

A model evaluation criterion can be defined by the average error between the true and the estimated position. Such a risk function for robot localization has been proposed in [9]. Suppose the difference between the true position \mathbf{x}^* of the robot and the the estimated position by \mathbf{x} is denoted by the loss function $L(\mathbf{x}, \mathbf{x}^*)$. If the robot observes \mathbf{y}^* , the expected localization error $\varepsilon(\mathbf{x}^*, \mathbf{y}^*)$ is (using Bayes' rule) computed as

$$\begin{aligned}\varepsilon(\mathbf{x}^*, \mathbf{y}^*) &= \int_{\mathbf{x}} L(\mathbf{x}, \mathbf{x}^*) p(\mathbf{x}|\mathbf{y}^*) d\mathbf{x} \\ &= \int_{\mathbf{x}} L(\mathbf{x}, \mathbf{x}^*) \frac{p(\mathbf{y}^*|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y}^*)} d\mathbf{x}.\end{aligned}\quad (9)$$

To obtain the total risk for the particular model, the above quantity must be averaged over all possible observations \mathbf{y}^* obtained from \mathbf{x}^* and all possible \mathbf{x}^* to give

$$R_L = \int_{\mathbf{x}^*} \int_{\mathbf{y}^*} \varepsilon(\mathbf{x}^*, \mathbf{y}^*) p(\mathbf{y}^*, \mathbf{x}^*) d\mathbf{y}^* d\mathbf{x}^* \quad (10)$$

The *empirical* risk is computed when estimating this function from the data:

$$\begin{aligned}\hat{R}_L &= \frac{1}{N} \sum_{n=1}^N \varepsilon(\mathbf{x}_n, \mathbf{y}_n) \\ &= \frac{1}{N} \sum_{n=1}^N \frac{\sum_{l=1}^N L(\mathbf{x}_l, \mathbf{x}_n) p(\mathbf{y}_n|\mathbf{x}_l)}{\sum_{l=1}^N p(\mathbf{y}_n|\mathbf{x}_l)}.\end{aligned}\quad (11)$$

This risk penalizes position estimates that appear far from the true position of the robot. A problem with this approach is that if at a few positions there are very large errors (for example two distant locations have similar visual features and may be confused), the average error will be very high.

5.2 Measure of multimodality

An alternative way of evaluating the linear projection from \mathbf{z} to \mathbf{y} is to consider the average degree of *modality* of $p(\mathbf{x}|\mathbf{y})$ [11]. The proposed measure is based on the simple observation that, for a given observation \mathbf{z}_n which is projected to \mathbf{y}_n , the density $p(\mathbf{x}|\mathbf{y} = \mathbf{y}_n)$ will always exhibit a mode on $\mathbf{x} = \mathbf{x}_n$. Thus, an approximate measure of multimodality is the *Kullback-Leibler* distance between $p(\mathbf{x}|\mathbf{y} = \mathbf{y}_n)$ and a unimodal density sharply peaked at $\mathbf{x} = \mathbf{x}_n$, giving the approximate estimate $-\log p(\mathbf{x}_n|\mathbf{y} = \mathbf{y}_n)$ plus a constant. Averaging over all points \mathbf{y}_n we have to minimize the risk

$$\hat{R}_K = -\frac{1}{N} \sum_{n=1}^N \log p(\mathbf{x}_n|\mathbf{y} = \mathbf{y}_n) \quad (12)$$

with $p(\mathbf{x}|\mathbf{y} = \mathbf{y}_n)$ computed with kernel smoothing as in (5). This risk can be regarded as the negative average log-likelihood of the data given a model defined by the kernel widths and the specific projection matrix. The computational costs of this is $O(N^2)$, in contrast with the $O(N^3)$ for \hat{R}_L .

5.3 Supervised projection

We use the risk measure to find a linear projection $\mathbf{y} = \mathbf{W}^T \mathbf{z}$ alternative to the PCA projection. The smooth form of the risk \hat{R}_K as a function of the projection matrix \mathbf{W} allows the minimization of the former with nonlinear optimization. For constrained optimization we must compute the gradient of \hat{R}_K and the gradient of the constraint function $\mathbf{W}^T \mathbf{W} - \mathbf{I}_q$ with respect to \mathbf{W} ,

and then plug these estimates in a constrained nonlinear optimization routine to optimize with respect to \hat{R}_K . We followed an alternative approach which avoids the use of constrained nonlinear optimization. The idea is to parameterize the projection matrix \mathbf{W} by a product of Givens (Jacobi) rotation matrices and then optimize with respect to the angle parameters involved in each matrix (see [12] for details). Such a *supervised projection* method must give better results than an unsupervised one like PCA (see experiments in this paper and [11]). In the next sections we will experimentally test both methods.

6 Experiments using PCA features

First we want to know how good the localization is when using PCA features. In particular we investigate how many features are needed.

6.1 Datasets and preprocessing

We tested our methods on real image data obtained from a robot moving in an office environment, of which an overview is shown in figure 1. We made use of the MEMORABLE robot database. This database is provided by Tsukuba Research Center, Japan, for the Real World Computing Partnership and contains a dataset of about 8000 robot positions and associated measurements from sonars, infrared sensors and omni-directional camera images. The measurements in the database were obtained by positioning the robot (a Nomad 200) on the grid-points of a virtual grid with distances between the grid-points of 10 cm. One of the properties of the Nomad 200 robot is that it moves around in its environment while the sensor head maintains a constant orientation. Because of this, the state of the robot is characterized by the position \mathbf{x} only.

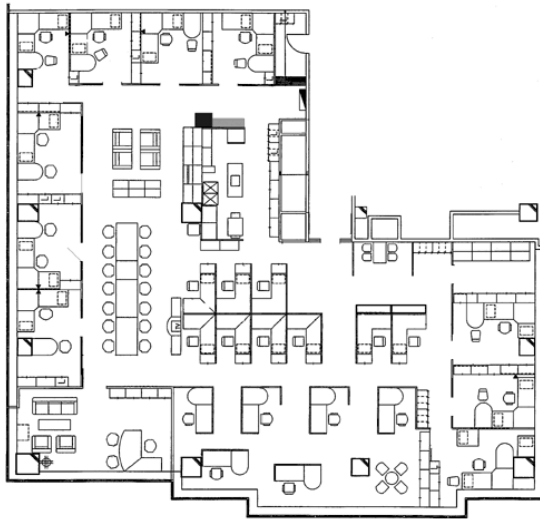


Fig. 1. The environment from which the images were taken.

The omni-directional imaging system consists of a vertically oriented standard color camera and a hyperbolic mirror mounted in front of the lens. This results in images as depicted in figure 2. Using the properties of the mirror we transformed the omni-directional images to 360 degrees panoramic images. To reduce the dimensionality we smoothed and subsampled the images to a resolution of 64×256 pixels (figure 3). A set of 2000 images was randomly selected from the total set to derive the eigenimages and associated eigenvalues. We found that for 80% reconstruction error we needed about 90 eigenvectors. However, we are not interested in the reconstruction of images, but in the use of the low-dimensional representation for robot localization.

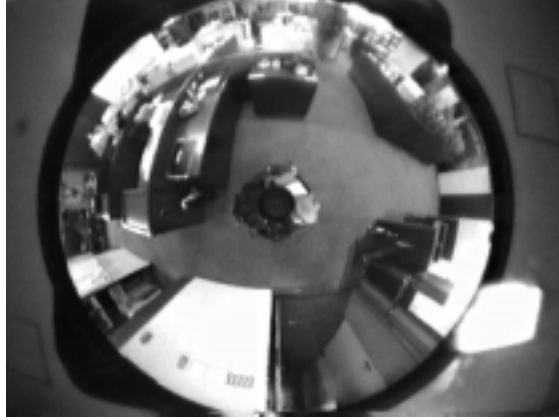


Fig. 2. Typical image from a camera with a hyperbolic mirror



Fig. 3. Panorama image derived with the omnidirectional vision system.

6.2 Observation model

In section 4 we described the kernel estimator as a way to represent the observation model $p(\mathbf{y}|\mathbf{x}; \boldsymbol{\theta})$. In such a method usually all training points are used for the modeling. In our database we have 8000 points we can use. If we use this whole dataset this means that in the operational stage we should calculate the distance to all 8000 points for the localization, which, even though the dimensions of \mathbf{x} and \mathbf{y} are low, is computationally too slow. We are therefore interested in taking only a part of these points in the kernel density estimation model. In the following sections a set of about 300 images was selected as a training set. These images were taken from robot positions on the grid-points of a virtual grid with distances between the grid-points of 50 cm.

Another issue in the kernel method is a sensible choice for the width of the Gaussian kernel. The optimal size of the kernel depends on the real distribution (which we do not know), the number of kernels and the dimensionality of the problem. When modeling $p(\mathbf{x}, \mathbf{y})$ for a one-dimensional feature vector \mathbf{y} with our training set we found that $h \approx 0.1$ maximized the likelihood of an independent test set. The test set consisted of 100 images, randomly selected from the images in the database not designated as training images. When using more features for localization (a higher dimensional feature vector \mathbf{y}) the optimal size of the kernel was found to be higher. We used these values in our observation model.

6.3 Localization

In a Markov localization procedure, an initial position estimate is updated by the features of a new observation using an observation model. The initial position estimate is computed using the motion model, and gives an informed prior in the Bayes rule. Since we are only interested in the performance of the observation model, we assume a flat prior distribution on the admissible positions \mathbf{x} .

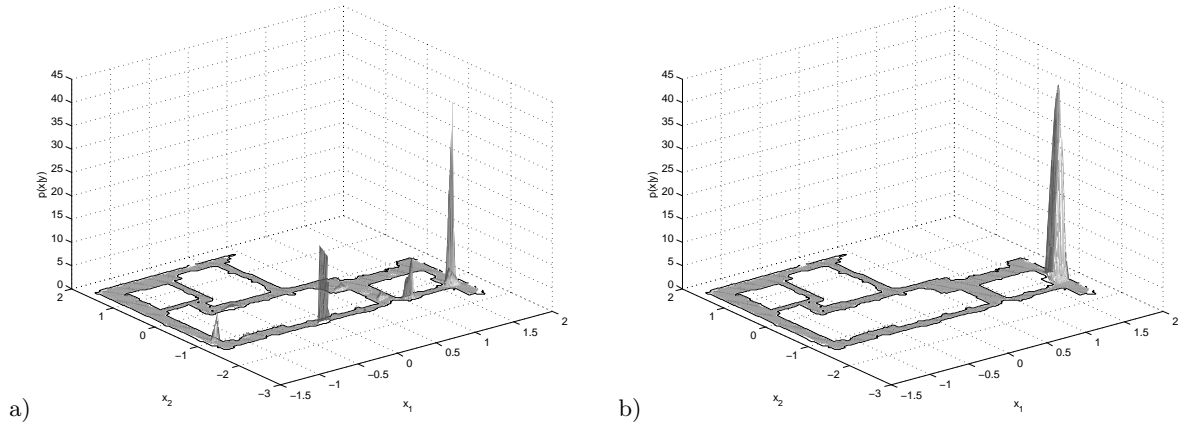


Fig. 4. An image at position (1.74, -0.96) is taken. The figure depicts the probability distribution over the learned locations. a) The first eigenvector (with the highest eigenvalue) is used as feature. b) The first 5 eigenvectors are used.

In the current experiments we studied how many of the principal components are needed for good localization. In figure 4 we see the distribution $p(\mathbf{x}|\mathbf{y})$ for an image which was taken at position (1.74, -0.96), for two different number of features: five eigenimages or one eigenimage. We observe that the distribution when using a single feature has multiple peaks, indicating multiple hypotheses for the position. This is solved if more features are taken into account. In both situations the maximum *a posteriori* value is close to the real robot position. This illustrates that the model gives a good prediction of the robot's real position. In some cases we observed a maximal value of the distribution at an erroneous location if too few features were used. So we need sufficient features for correct localization. The effect of the number of features is depicted in figure 5 where we plotted localization risk for different number of features. We see that for this dataset (300 positions) the performance levels out after about 10-15 features.

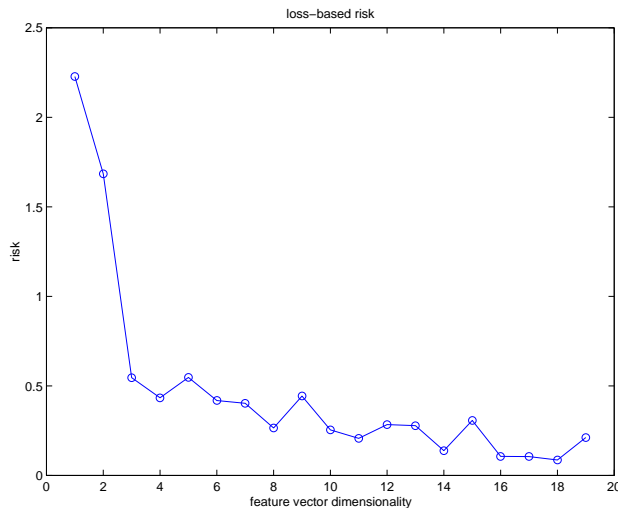


Fig. 5. Performance for different number features

7 Comparing PCA with a supervised projection

We also compared the localization of the robot when using PCA features and when using supervised projection features. Here we used data collected in our own laboratory. The Nomad robot follows a predefined trajectory in our mobile robot lab and the adjoining hall as shown in Fig. 6. The data set contains 104 omnidirectional images (320×240 pixels) captured every 25 centimeters along the robot path. Each image is transformed to a panoramic image (64×256) and these 104 panoramic images together with the robot positions along the trajectory constitute the training set of our algorithm. A typical panoramic image shot at the position A of the trajectory is shown in Fig. 7.

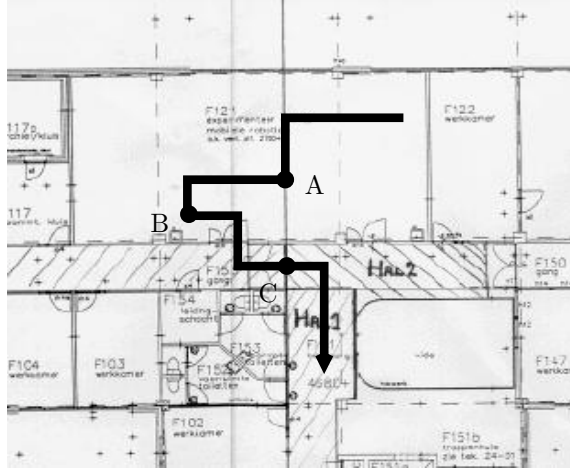


Fig. 6. The robot trajectory in our building.

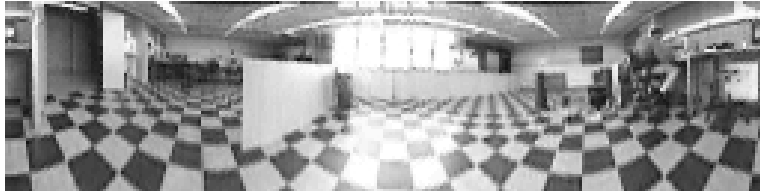


Fig. 7. A panoramic snapshot from position A in the robot trajectory.

In order to apply our supervised projection method, we first sphered the panoramic image data. Sphering is a normalization to zero-mean and identity covariance matrix of the data. This is always possible through PCA. We kept the first 10 dimensions explaining about 60% of the total variance. The robot positions were normalized to zero mean and unit variance. Then we applied the supervised projection method (see [12] for details of optimization) projecting the sphered data points from 10-D to 2-D.

In Fig. 8 we plot the resulting two-dimensional projections using (a) PCA, and (b) our supervised projection method. We clearly see the advantage of the proposed method over PCA. The risk is smaller, while from the shape of the projected manifold we see that taking into account the robot position during projection can significantly improve the resulting features: there are fewer self-intersections of the projected manifold in our method than in PCA which, in turn, means better robot position estimation on the average (smaller risk). In [11] we also show that localization is more accurate when using the supervised projection method.

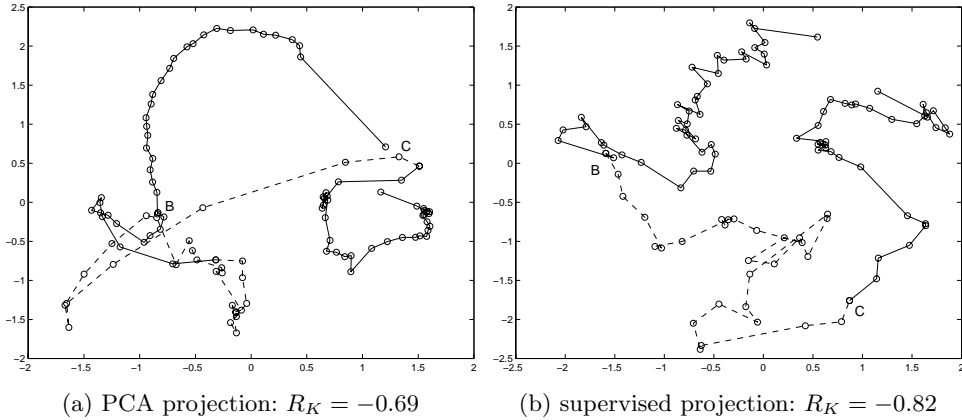


Fig. 8. Projection of the panoramic image data from 10-D. (a) Projection on the first two principal components. (b) Supervised projection optimizing the risk R_K . The part with the dashed lines corresponds to projections of the panoramic images captured by the robot between positions B and C of its trajectory.

Finally, the two feature vectors (directions in the image space on which the original images are projected) that correspond to the above two solutions are shown in Fig. 9. In the PCA case these are the familiar first two eigenimages of the panoramic data which, as is normally observed in typical data sets, exhibit low spatial frequencies. We see that the proposed supervised projection method yields very different feature vectors than PCA, namely, images with higher spatial frequencies and distinct local characteristics.

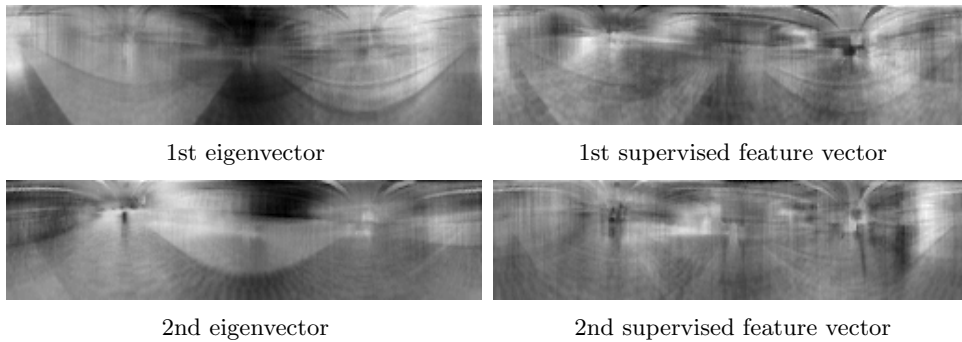


Fig. 9. The first two feature vectors using PCA (left) and our supervised projection method (right).

8 Discussion and conclusions

We showed that appearance-based methods give good results on localizing a mobile robot. In the experiments with the PCA features, the average expected localization error from our test set is about 40cm if around 15 features are used and the environment is represented with 300 training samples. Note that we studied the worst-case scenario: the robot has no prior information about its position (the ‘kidnapped robot’ problem), and combined with a motion model the localization accuracy should be better. A second observation is that the environment can be represented by only a small number of parameters. For the 300 15-dimensional feature vectors the storage capacity is almost negligible and the look-up can be done very fast.

The experiments with the supervised projection showed that this method resulted in a lower risk, and therefore a better expected localization. In [11] we describe an experiment where we used the full Markov procedure to localize the robot. The supervised projection method gave significantly better results than the PCA features.

Both experiments were carried out with extensive data sets, with which we were able to get good estimates on the accuracy of the method. However, the data were obtained in a static environment, with constant lighting conditions. Our current research in this line focuses on investigating which features are most important if changes in the illumination will take place.

9 Acknowledgment

We would like to thank the people in the Real World Computing Partnership consortium and the Tsukuba Research Center in Japan for providing us with the MEMORABLE robot database.

References

1. W. Burgard, A. Cremers, D. Fox, G. Lakemeyer, D. Hähnel, D. Schulz, W. Steiner, Walter, and S. Thrun. The interactive museum tour-guide robot. In A. P. Press, editor, *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1998.
2. J. L. Crowley, F. Wallner, and B. Schiele. Position estimation using principal components of range data. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Leuven, Belgium, May 1998.
3. I. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, 1986.
4. K. Konolige and K. Chou. Markov localization using correlation. In *Proc. International Joint Conference on Artificial Intelligence*, pages 1154–1159. Morgan Kaufmann, 1999.
5. H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *Int. Jnl of Computer Vision*, 14:5–24, 1995.
6. S. Oore, G. E. Hinton, and G. Dudek. A mobile robot that learns its place. *Neural Computation*, 9:683–699, 1997.
7. W. H. Press, S. A. Teukolsky, B. P. Flannery, and W. T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.
8. S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998.
9. S. Thrun. Bayesian landmark learning for mobile robot localization. *Machine Learning*, 33(1), 1998.
10. N. Vlassis and B. Kröse. Robot environment modeling via principal component regression. In *IROS'99, Proceedings of 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp 677–682, 1999.
11. N. Vlassis, R. Bunschoten, and B. Kröse. Learning task-relevant features from robot data. In *IEEE International Conference on Robotics and Automation*, Seoul, Korea, May 2001. pp 499–504, 2001.
12. N. Vlassis, Y. Motomura and B. Kröse. Supervised Dimension Reduction of Intrinsically Low-dimensional Data. *Neural Computation*, to appear, 2001.