MULTI-ROBOT SYSTEMS, TRENDS AND DEVELOPMENT

Edited by **Toshiyuki Yasuda and Kazuhiro Ohkura**

INTECHWEB.ORG

Multi-Robot Systems, Trends and Development
Edited by Toshiyuki Yasuda and Kazuhiro Ohkura

Published by InTech
Janeza Trdine 9, 51000 Rijeka, Croatia

Copyright © 2011 InTech
All chapters are Open Access articles distributed under the Creative Commons Non Commercial Share Alike Attribution 3.0 license, which permits to copy, distribute, transmit, and adapt the work in any medium, so long as the original work is properly cited. After this work has been published by InTech, authors have the right to republish it, in whole or part, in any publication of which they are the author, and to make other personal use of the work. Any republication, referencing or personal use of the work must explicitly identify the original source.

Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published articles. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

Publishing Process Manager Ana Nikolic
Technical Editor Teodora Smiljanic
Cover Designer Martina Sirotic
Image Copyright Vladimir Nikitin, 2010. Used under license from Shutterstock.com

First published January, 2011
Printed in India

A free online edition of this book is available at www.intechopen.com
Additional hard copies can be obtained from orders@intechweb.org

Multi-Robot Systems, Trends and Development, Edited by Toshiyuki Yasuda and Kazuhiro Ohkura
p. cm.
ISBN 978-953-307-425-2

INTECH OPEN ACCESS PUBLISHER

INTECH open

free online editions of InTech
Books and Journals can be found at
www.intechopen.com

Contents

11

Probabilistic Map Building, Localization and Navigation of a Team of Mobile Robots. Application to Route Following.

L. Payá, O. Reinoso, F. Amorós, L. Fernández and A. Gil
Miguel Hernandez University
Spain

1. Introduction

The applications that require the navigation of a robot or a team of robots through an environment need an internal representation (or map) of this environment. Thanks to it, the robots can estimate their position and orientation using the information captured with the different sensors the robots are equipped with. Despite the fact that there are several kinds of sensors to carry out that task, omnidirectional visual systems can be stood out due to the richness of the information they provide and the relatively low cost they have.

A typical problem in collaborative robotics implies a path following, e.g. to perform a surveillance task in an office environment or an assembly or delivery task in an industrial environment. Also, the problem of formations, where a team of robots must navigate keeping a relative position in a structure of robots can be seen as a problem of path following, where one or several robots must follow the path the leader is recording with an offset either in space or in time.

Classical researches into mobile robots provided with vision systems have focused on the extraction of natural or artificial landmarks from the image to build the map and carry out the localization of the robot (Booij et al., 2007; Burschka & Hager, 2001; Thrun, 2003). Nevertheless, it is not necessary to extract such kind of landmarks to recognize where the robot is. Instead of this, we can process the image as a whole. These approaches (known as *appearance-based*) are especially useful for complicated scenes in unstructured environments where appropriate models for recognition are difficult to create. As an example, (Matsumoto et al., 1999) address a method for route following where several low-resolution images along the route are stored, and (Payá et al., 2007) use an incremental compression method of the scenes that permits online multi-robot route following.

In the appearance-based approaches, images are saved without extracting any local feature, and the comparison is made directly, working with the whole information of the scenes. So the problem of finding the position of the robot in the environment consists in getting the best match for the current image among the reference images. Since no relevant information is extracted, an important problem of such approaches is the high computational cost they suppose. That is the reason why often it is necessary to apply some compression techniques.

Different researchers have shown how a manifold representation of the environment using some compression techniques can be used. For example, PCA (Principal Components Analysis) is a widely used method that has demonstrated being robust applied to image

processing, as (Kröse et al., 2004) do to create a database using a set of views with a probabilistic approach for the localization inside this database. Conventional PCA methods do not take profit of the amount of information that omnidirectional cameras offer, because they cannot deal with rotations in the plane where the robot moves. (Uenohara & Kanade, 1998) studied this problem with a set of rotated images, and (Jogan and Leonardis, 2000) applied these concepts to an appearance-based map of an environment. Despite its complexity and computational cost, it has the advantage of being a rotationally invariant method due to the fact that it takes into account the images with other orientations apart from the stored one. The approach consists in creating an eigenspace that takes into account the possible rotations of each training image, trying to keep a good relationship between amount of memory, time and precision of the map

Other authors use the Discrete Fourier Transform (DFT) as a generic method to extract the most relevant information from an image. In this field, (Menegatti et al., 2004) define the Fourier Signature for panoramic images, which is based on the 1D Discrete Fourier Transform of the image rows and gets more robustness dealing with different orientations and (Rossi et al., 2008) use spherical Fourier transform of the omnidirectional images.

On the other hand, (Dalal and Triggs, 2005) used a method based on the Histogram of Oriented Gradients (HOG) to pedestrians detection, proving that it could be a useful descriptor for computer vision and image processing using the objects' appearance.

In some applications, the use of a team of robots may help to make the achievement of the task quicker and more reliable. In such situations, each robot works with incomplete and changing information that has, also, a high degree of uncertainty. This way, only a suitable choice of the representation and an effective communication between the members of the team can provide the robots with a complete knowledge of the environment where they move. As an example, (Thrun, 2001) presents a probabilistic EKF algorithm where a team of robots builds a map online, while simultaneously they localize themselves. In (Ho & Newman, 2005) a map is build using visual appearance. From sequences of images, acquired by a team of robots, subsequences of visually similar images are detected and finally, the local maps are joined into a single map.

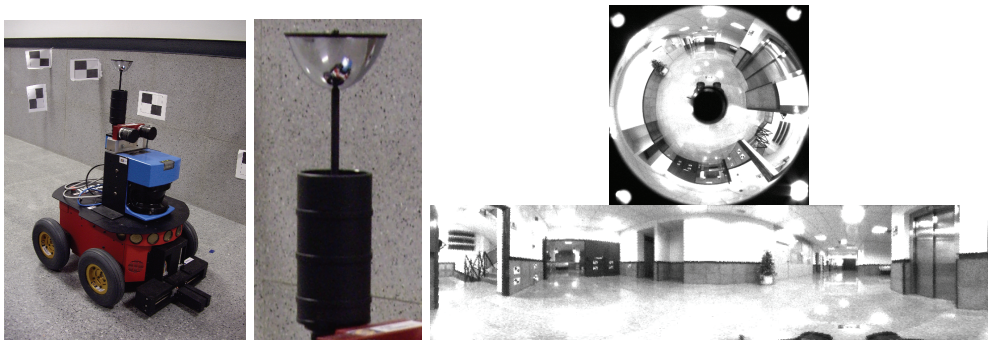
In this work, we present a framework that permits carrying out multi-robot route following, where an appearance-based approach is used to represent the environment and a probabilistic approach is used to compute the localization of the robot. Each robot carries out an omnidirectional camera in a fixed position on its top, and the localization and navigation tasks are carried out using a descriptor of the global appearance of the omnidirectional images captured by this camera. Along the chapter, we study and compare some alternatives to build an efficient appearance descriptor. We compare the descriptors in terms of computational cost, size of memory and its validity in localization tasks. We use three different approaches to build these descriptors: the Discrete Fourier Transform, Principal Components Analysis and Histogram of Oriented Gradients.

The remainder of the paper is organized as follows. In section 2, the main features of the multi-robot route-following system are addressed. In section 3, some methods to describe the appearance of the scenes are detailed. The performance of these descriptors is tested in section 4 in a map building task, and in section 5 in a localization task. In both sections, the results of the experiments are commented. At last, the conclusions are presented.

2. Multi-robot route following

The main goal of the work is to present an architecture to carry out multi-robot route following using just visual information and with an appearance-based approach.

In this application, we work with a team of Pioneer P3-AT robots (fig. 1(a)) that are equipped with a catadioptric system (fig. 1(b)) consisting on a forward-looking camera and a parabolic mirror that provides omnidirectional images of the environment. To work with this information in an efficient way, the omnidirectional images are transformed to panoramic images, as shown on fig. 1(c). The size of these panoramic images is 41x256 pixels.



(a) (b) (c)

Fig. 1. (a) Pionner P3-AT robot and (b) catadioptric system that provides omnidirectional images. (c) Omnidirectional and (d) panoramic view of the environment captured by the catadioptric system.

In a multi-robot route following task, a leader robot goes through the desired route while a team of robots follow it with an offset in space or in time. To accomplish this goal, the leader robot performs a *learning* task while the follower robots perform an *autonomous navigation* task. The *learning* task consists in tele-operating the leader robot through the route to follow while it stores some general visual information along this route. In a general way, new omnidirectional images are continuously acquired, but a new image is stored only when its correlation with the previously stored image goes down a threshold. This way, images are stored more frequently when the information changes quicker (i.e. turnings and non structured environments) and fewer images are stored when the new information is quite similar. In this step, it is important to define correctly the representation of the environment to permit that any robot can follow the route of the leader one with an offset either in space or/and in time in an efficient way. In this work, we propose the use of appearance-based methods to describe globally the information in the scenes. In section 3 we show how we build an efficient descriptor to represent this visual information.

Fig. 2 shows a sample route in an indoor environment. The points where a new omnidirectional image has been stored are drawn as red dots. Each omnidirectional image has been transformed to the panoramic format, as show in the figure. In a posterior phase, a single descriptor per image will be computed. All these descriptors will constitute the database or map.

On the other hand, once the leader robot has created the database or while it is being built, the follower robots perform the *autonomous navigation* task. Before starting this process, the follower robot is situated in a point near the learned route. Then, it has to recognize which of the stored positions is the nearest to the current one and drive to tend to the route, following it till the end. This task must be carried out just comparing its current visual information with the information stored in the database. Two processes must run

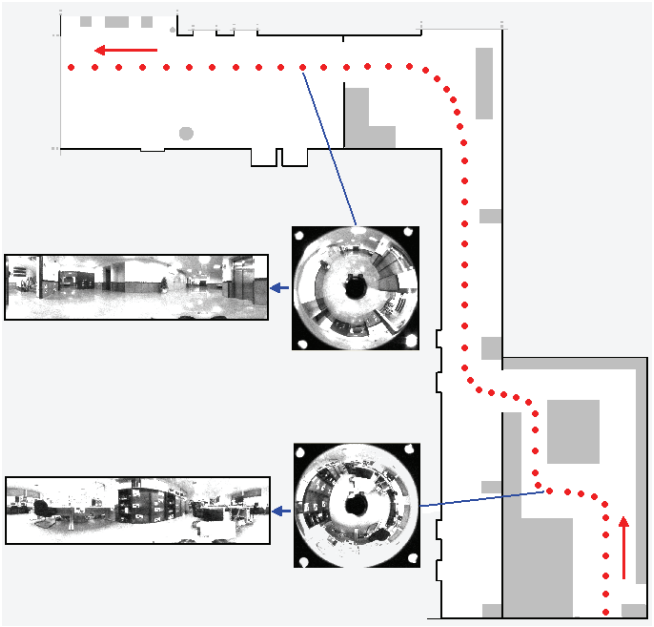


Fig. 2. Sample route including a hall, a corridor and a laboratory. The red dots show the places where the robot has acquired omnidirectional images to form the database.

successively: *auto-location* and *control*. During the *auto-location*, the current visual information is compared with the information in the database, and the most similar point is taken as the current position of the robot. This decision must be made taking into account the aperture problem in office environments. This way, the new position of the robot must be in a neighbourhood of the previous one, depending on the speed of the robot. Once we have a new matching, in the *control phase*, the current visual information must be compared with the matched information of the database, and from this comparison, a control law must be deducted so that the robot tends to the route and follows it till the end.

Once the principles of our route-following architecture have been exposed, in the next subsections each process in described in deep.

2.1 Task 1: learning the map

The map is built gradually while the leader robot is going through the route to record. This robot captures a new omnidirectional image when the correlation respect to the previous one goes down a threshold. This way, our database is composed of a set of omnidirectional views.

As our proposal consists in working with global appearance methods, that imply using the information in the images as a whole, we do not perform any feature extraction. This fact becomes a problem due to the growing size of the memory for long routes and the high computational cost in the localization phase to find the best match between the current image and all the images stored in the database. That is the reason why a compression method must be used to extract the most relevant information from the set of images.

When a new omnidirectional image is captured, first it is transformed into the panoramic image and then, the information is compressed to build a global descriptor of each

panoramic image. This way, each 41x256 image is compressed to a sequence of M numbers that constitute the descriptor of the image.

Each image $X_j \in \Re^{RxC}; j = 1 \ldots N$, being *R* the number of rows and *C* the number of columns, is transformed into a vector $\vec{z}_j \in \Re^M; j = 1 \ldots N$, being *M* the size of the global descriptor, *M* << *RxC*.

While building the descriptor $\vec{z}_j$, some principles must be taken into account. It must extract the most relevant information from the image, and its length must be as small as possible. Also, each image must be described as the leader robot is capturing them; the descriptor must be computed in an incremental procedure (i.e. the descriptor of each image must be computed independently of the rest of images). At last, the descriptor must contain enough information so that the follower robot can estimate the control law that makes it tend to the route. In section 3 we present some approaches to build the descriptor and we make a complete set of experiments to decide the optimal procedure to build it in our route-following application.

2.2 Task 2: robot navigation

After the *learning* task, the leader robot has created a database consisting of some feature vectors that are labelled as positions *1,..., N*. For each position, a descriptor represents the visual information taken at that position. The descriptors are $\vec{z}_j \in \Re^M; j = 1 \ldots N$.

Now, a different robot (the follower robot) can follow the same route carrying out successively two processes: *auto-location* and *control*.

2.2.1 Auto-location

When the follower robot captures a new image X_t, first, the global descriptor of this image $\vec{z}_t$ must be computed. Then, using this information, the robot must be able to compute which of the stored points is the closest one. With this aim, the distance between the descriptor of the current images and all the descriptors in the database is computed. We propose using the Euclidean distance as a measure. The distance between the descriptor of the current image $\vec{z}_t$ and the descriptors $\vec{z}_j$ in the database is:

$$d_{tj} = \sqrt{\sum_{l=0}^{M} \left(\vec{z}_t(l) - \vec{z}_j(l) \right)^2}; \quad j = 1 \ldots N \tag{1}$$

To normalize the values of the distances, a degree of similarity between descriptors is computed with eq. (2). The image of the database that offers the maximal value of γ_{tj} is the matching image, and then, the current position of the robot (the nearest one) is *j*.

$$\gamma_{tj} = \frac{\left(d_t^{\max} / d_{tj} \right) - 1}{\left(d_t^{\max} / d_t^{\min} \right) - 1}; \quad \gamma_{tj} \in [0,1]; \quad j = 1 \ldots N \tag{2}$$

where $d_t^{\max} = \max_{j=1}^{N} \left(d_{tj} \right)$ and $d_t^{\min} = \min_{j=1}^{N} \left(d_{tj} \right)$.

However, in office environments that present a repetitive visual appearance, this simple localization method tends to fail often as a result of the aperture problem (aliasing). This

means that the visual information captured at two different locations that are far away can be very similar. To avoid these problems, we propose the use of a probabilistic approach, based on a Markov process (Thrun et al., 2005) to solve the problem.

In this localization task, we are interested in the estimation of the nearest point of the database, i.e. the state x_t at time *t* using a set of measurements $\vec{z}_{1:t} = \{\vec{z}_1, \vec{z}_2, \ldots, \vec{z}_t\}$ from the environment and the movements $u_{1:t} = \{u_1, u_2, \ldots, u_t\}$ of the robot. In this notation, we consider that the robot makes a movement u_t from time *t-1* to time *t* and next obtains an observation $\vec{z}_t$. In this document the localization problem has been stated in a probabilistic way: we aim at estimating a probability function $p(x_t \mid \vec{z}_{1:t}, u_{1:t})$ over the space of all possible localizations, conditioned on all the data available until time *t*, the observations $\vec{z}_{1:t}$, movements performed $u_{1:t}$ and the map. In a probabilistic mobile robot localization, the estimation process is usually carried out in two phases:

Prediction phase:

In the Prediction Phase the motion model is used to compute the probability function $p(x_t \mid \vec{z}_{1:t-1}, u_{1:t})$, taking only motion into account. The motion model is a probabilistic generalization of robot kinematics. In this work the value of *u* is an odometry reading. Generally, we assume that the current state x_t depends only on the previous state x_{t-1} and the movement u_t. The motion model is specified in the form of the conditional density $p(x_t \mid x_{t-1}, u_t)$. As we have a set of discrete localizations, the probability function at the next step is obtained by the summation:

$$p(x_t \mid \vec{z}_{1:t-1}, u_t) = \sum p(x_t \mid x_{t-1}, u_t) \cdot p(x_{t-1} \mid \vec{z}_{1:t-1}, u_{1:t-1}) \tag{3}$$

where the function $p(x_t \mid x_{t-1}, u_t)$ represents the probabilistic movement model.

When the robot moves, the uncertainty over its pose generally increases. This is due to the fact that the odometry sensors are not precise. In consequence, the function $p(x_t \mid x_{t-1}, u_t)$ describes the probability over the variable x_t given that the robot was at the pose x_{t-1} and performed the movement u_t.

Update phase:

In the second phase, a measurement model is used to incorporate information from the sensors and obtain the posterior distribution $p(x_t \mid \vec{z}_{1t}, u_{1t})$. In this step, the measurement model $p(\vec{z}_t \mid x_t)$ is employed, which provides the likelihood of obtaining the observation $\vec{z}_t$ assuming that the robot is at pose x_t. The posterior $p(x_t \mid \vec{z}_{1t}, u_{1t})$, can be calculated using Bayes' Theorem:

$$p(x_t \mid \vec{z}_{1:t}, u_{1:t}) = \eta \cdot p(\vec{z}_t \mid x_t) \cdot p(x_t \mid \vec{z}_{1:t-1}, u_t) \tag{4}$$

where $p(x_t \mid \vec{z}_{1:t-1}, u_t)$ denotes the probability that the robot is on the position x_t before observing the image whose descriptor is $\vec{z}_t$. This value is estimated using the previous information and the motion model (eq. 3). $p(\vec{z}_t \mid x_t)$ is the probability of observing $\vec{z}_t$ if the position of the robot is x_t. This way, a method to estimate the observation model must be deducted. In this work, the distribution $p(\vec{z}_t \mid x_t)$ is modelled through a sum of Gaussian kernels, centred on the *n* most similar points of the route:

$$p(\vec{z}_t \mid x_t) = \frac{1}{n} \cdot \sum_{i=1}^{n} \left(\gamma_{ti} \cdot e^{-\left(\frac{x_t - x_i}{\sigma}\right)^2} \right) \tag{5}$$

Each kernel is weighted by the value of confidence γ_{ti} (eq. 2). Once the prediction and the update phase have been computed, the new position is considered at the point with highest probability contribution. After the update phase, this process is repeated recursively.
The knowledge about the initial state at time t_0 is generally represented by $p(x_0)$. In this case two different cases are generally considered. When the initial pose of the mobile robot is totally unknown, the initial state at time t_0, $p(x_0)$, is represented by a uniform distribution on the map. Then we say we are in the case of global localization. But if the initial pose of the mobile robot is partially known, the case of local localization or tracking, the function $p(x_0)$ is commonly represented by a Gaussian distribution centred at the known starting pose of the robot. In our route-following application, as the initial position is usually unknown, we use a uniform distribution to model $p(x_0)$.

2.2.2 Control

Once the robot knows its position, it has to compute its heading, comparing to the heading that the leader had when it captured the image at that position. This information must be computed from the comparison between the image descriptors. We suppose that the descriptor of the current image is $\vec{z}_t$. After the probabilistic localization process, the descriptor of the corresponding location in the database is $\vec{z}_i$. If we suppose we can retrieve the relative orientation θ_t between the heading of the leader robot when it captured the image X_t and the heading of the follower robot when it captured the image X_i, then, a control action can be computed to control the robot. We propose the use of a controller with the following expression:

$$\begin{aligned} \omega_t &= k_1 \cdot \theta_t + k_2 \cdot [\theta_t - \theta_{t-1}]. \\ v_t &= k_3 \cdot p(x_t). \end{aligned} \tag{6}$$

where ω_t is the steering speed and v_t the linear speed that must be applied to the robot. For the calculation of the steering speed, we propose to use a proportional and derivative controller. The linear velocity will be proportional to the probability $p(x_t)$, what means that when the robot is bad localized (due to the aperture problem or to the fact that it is far from the route), the linear velocity is low to allow correcting the trajectory, but when the robot is well localized (i.e. the robot is following the route quite well and no aliasing is produced), the robot goes quicker.
Since the most important parameter to control the robot is the relative orientation of the robot, θ_t, in section 5 we make a detailed experimental study to determine how efficient is each descriptor when computing the relative orientation of the robot.

3. Representation of the environment through a set of omnidirectional images with appearance-based methods

In this section, we outline some techniques to extract the most relevant information from a set of panoramic images, captured from several positions along the route to follow. We have

grouped these approaches in three families: DFT (Discrete Fourier Transform), PCA (Principal Components Analysis) and HOG (Histogram of Oriented Gradients) methods. The last technique has proved previous promising results, although not in localization functions. This section completes the study presented in (Paya et al., 2009).

3.1 DFT-based techniques

As shown in (Menegatti et al., 2004), when working with panoramic images, it is possible to use a Fourier-based compact representation for the scenes. It consists in expanding each row of the panoramic image $\{a_n\} = \{a_0, a_1, \ldots, a_{N_y-1}\}$ using the Discrete Fourier Transform into the sequence of complex numbers $\{A_n\} = \{A_0, A_1, \ldots, A_{N_y-1}\}$.

This Fourier signature presents the same properties as the 2D Fourier Transform. The most relevant information concentrates in the low frequency components of each row, and it presents rotational invariance. However, it exploits better this invariance to ground-plane rotations in panoramic images. These rotations lead to two panoramic images which are the same but shifted along the horizontal axis (fig. 3). Each row of the first image can be represented with the sequence $\{a_n\}$ and each row of the second image will be the sequence $\{a_{n-q}\}$, being q the amount of shift, that is proportional to the relative rotation between images. The rotational invariance is deducted from the shift theorem, which can be expressed as:

$$\Im\left[\{a_{n-q}\}\right] = A_k e^{-j\cdot\frac{2\pi qk}{N_y}}; \qquad k = 0, \ldots, N_y - 1 \tag{7}$$

where $\Im\left[\{a_{n-q}\}\right]$ is the Fourier Transform of the shifted sequence, and A_k are the components of the Fourier Transform of the non-shifted sequence. According to this expression, the amplitude of the Fourier Transform of the shifted image is the same as the original transform and there is only a phase change, proportional to the amount of shift q.

Fig. 3. Two panoramic images captured in the same point of the environment but with different heading for the robot.

Then, with this method, a descriptor $\vec{z}_j$ can be built with the modules and the angles of the Fourier signature of the panoramic image I_j, retaining only the f first columns where the most relevant information is stored.

3.2 PCA-based techniques

When we have a set of *N* images with *M* pixels each, $I_j \in \Re^{Mx1}; j = 1 \ldots N$, each image can be transformed in a feature vector (also named 'projection' of the image) $\vec{z}_j \in \Re^{Kx1}; j = 1 \ldots N$, being *K* the PCA features containing the most relevant information of the image, $K \leq N$ (Kirby, 2000). The PCA transformation can be computed from the singular value decomposition of the covariance matrix *C* of the data matrix, *X* that contains all the training images arranged in columns (with the mean subtracted). If *V* is the matrix containing the *K* principal eigenvectors and *P* is the reduced data of size *K x N*, the dimensionality reduction is done by $P = V^T \cdot X$, where the columns of *P* are the projections of the training images, $\vec{z}_j$.
However, if we apply PCA directly over the matrix that contains the images, we obtain a database with information just with the orientation of the robot when capturing those images but not for other possible orientations. Some authors have studied how to build an appearance-based map of an environment using a variation of PCA that includes information not only about the localizations where the images were taken but also about all the possible orientations at that points (Jogan et al., 2000). However, in previous works we have proved that the computational cost of such techniques does not permit to use them in an online task. Instead of using these rotational invariant techniques, what we propose in this chapter is to build the descriptor $\vec{z}_j$ by applying PCA over the Fourier Signature components instead of the original image, obtaining the compression of rotational invariant information, joining the advantages of PCA and Fourier techniques.

3.3 HOG-based techniques

The Histogram of Oriented Gradient (HOG) descriptors (Dalal and Triggs, 2005) are based on the orientation of the gradient in local areas of an image. The first step to apply HOG to an image is to compute the spatial derivatives of the image along the *x* and *y*-axes (U_x and U_y). We have obtained these derivatives by calculating the convolution of the images with Gaussian filters with different variance. Once the convolution of the image is made, we can get the magnitude and direction values of the gradient at each pixel:

$$|G| = \sqrt{U_x^2 + U_y^2} \qquad \theta = \arctan\left(U_x / U_y\right) \tag{8}$$

After that, we compute the orientation binning by dividing the image in cells, and creating the histogram of each cell. The histogram is computed based on the gradient orientation of the pixels within the cell, weighted with the corresponding module value. The number of histogram divisions is 8 in our experiments, and the angle varies between -90º and 90º. Each image is represented through the histogram of every cell ordered into a vector.
An omnidirectional image contains the same pixels in a row although the image is rotated, but in a different order. We can take profit of this characteristic to carry out the location of the robot by means of calculating the histogram with cells having the same width as the image (fig. 4). This way, we obtain an array of rotational invariant characteristics.
However, to know the relative orientation between two rotated images vertical windows (cells) are used, with the same height of the image, and variable width and distance (fig. 4). Ordering the histograms of these windows in a different way, we obtain the same results as calculating the histogram of an image rotated a proportional angle to the *D* distance. The angle resolution that can be got between two shifted images is proportional to that distance:

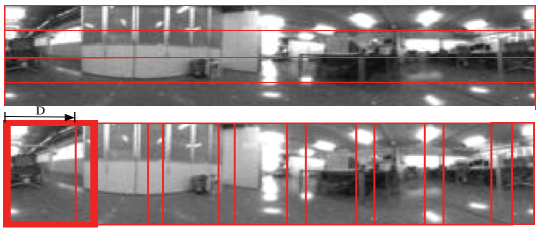


Fig. 4. Cells used to compute the current position and the orientation of the robot.

$$Angle(^{o}) = \frac{D * 360}{Width \quad of \quad the \quad image} \tag{9}$$

4. Performance of the descriptors in a map building task

In this section, the different methods to build the images' descriptors are compared using a database made up of a set images that were collected in several living spaces under realistic illumination conditions. It has been got from Technique Faculty of Bielefeld University (Möller et al., 2007). The images were captured with an omnidirectional camera, and later converted into panoramic ones with 41x256 pixels size. All the images belong to inner living spaces. Specifically, there are examples from a living room (L.R.), a kitchen and a combined area (C.A.), all of them structured in a 10x10 cm rectangular grid. Fig. 5 shows an example of image corresponding to each area. The objective is to test the memory requirements and computational cost of each descriptor when representing an environment.

The number of images that compose the database varies depending on the experiment, since, in order to assess the robustness of the algorithms, the distance between the images of the grid we take will be expanded, getting less information. I.e., instead of using all the images in the database to make up the map, we use just every two, three or four available images. In table 1 the size of the grid and the number of elements appear depending on the selected grid.

	Size	10x10	20x20	30x30	40x40
L.R.	22x11	242	66	32	18
Kitchen	12x9	108	30	12	9
C.A.	36x11	396	118	48	27
TOTAL		**746**	**204**	**92**	**54**

Table 1. Size of the database and number of images selected depending on the grid.

Fig. 5. Living room, kitchen and combined area sample images.

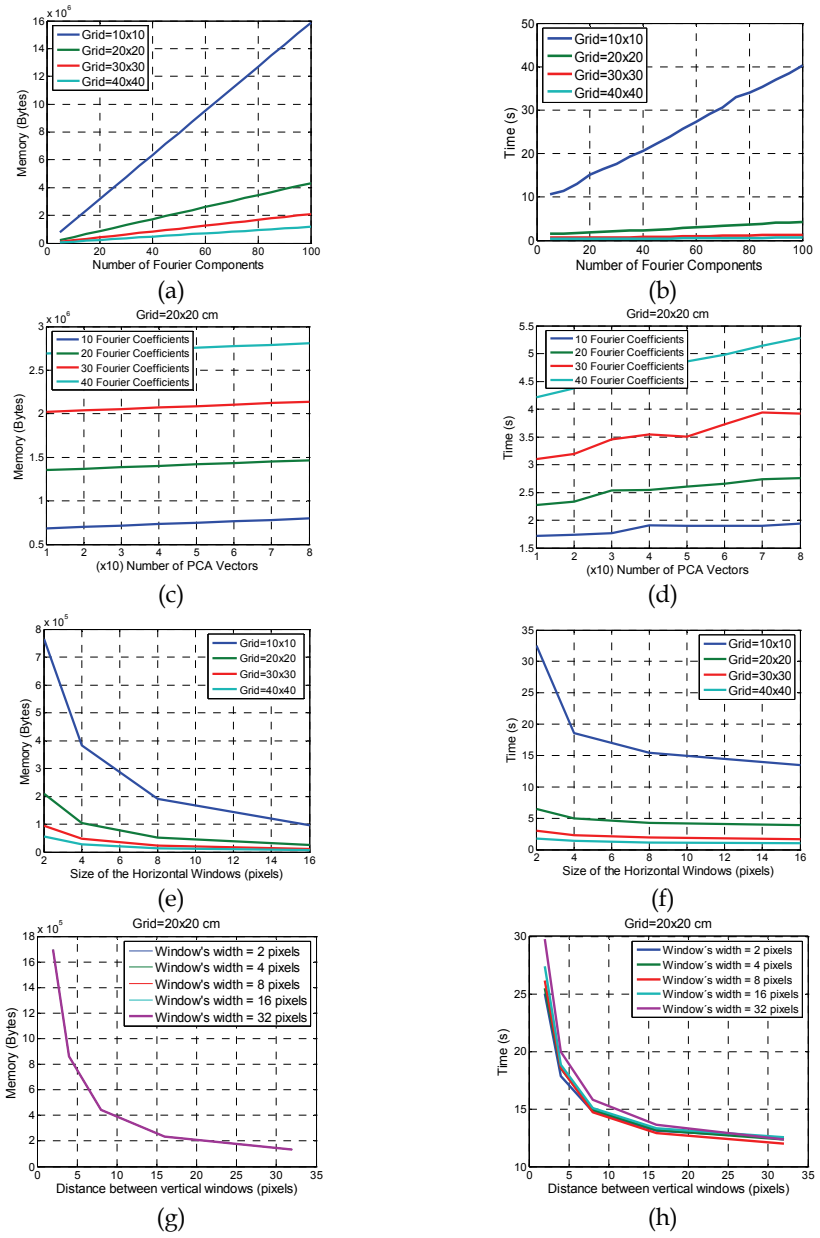


Fig. 6. (a) Amount of memory and (b) processing time to build the map with the Fourier-based algorithm. (c) Amount of memory and (d) processing time to build the map using PCA over Fourier Signature method. (e) Amount of memory and (f) processing time to build the map with HOG, varying horizontal windows parameters. (g) Amount of memory and (h) processing time to build the map with HOG, varying vertical windows parameters.

Once the images of the map are selected, we compute the image descriptors using each method to obtain a dataset, which represents the map of the environment.
Fig. 6 shows the amount of memory and time necessary to build the map, depending on the number of images and/or the different parameters that affect each algorithm as described in the previous section. These parameters are, in the case of Fourier signature, the number of Fourier coefficients per row (*f*). In the case of PCA over the Fourier signature, they are the number of vectors selected from the PCA decomposition (*K*), apart from the number of Fourier coefficients (*f*). At last, in the case of HOG, both the size and separation between windows must be decided in the case of the horizontal (position estimation) and the vertical (orientation estimation) windows.
Fig. 6(a) and 6(b) show the memory requirements and processing time to build the map with the Fourier method. Naturally, there is a proportional increase of the memory and time requirements as we get more coefficients of each row (since more information is computed and stored), and a decrease of both parameters when the grid step increases (since it supposes fewer images in the map).
Fig. 6(c) and 6(d) show the results when applying PCA to the Fourier Signature for a 20x20 cm grid. The results are more dependent on the number of Fourier coefficients than on the number of PCA components. In all the experiments, the processing time is larger than in the first method if the same number of Fourier components is taken, due to the fact that apart from calculating the Fourier signature, it is necessary to compute the PCA algorithm too.
Fig. 6(e) and 6(f) show the results when using HOG and varying the size of the horizontal windows and grid step. In these experiments, the vertical windows have a fixed width and gap of 8 pixels (i.e., degree step of 11,25º). These figures show a strong correlation between the size of the horizontal window and the memory and time requirements, since the bigger is the window, the lesser windows exist (no overlapping is used in horizontal windows).
Fig. 6(g) and 6(h) show the results when using HOG and varying the size of and the distance between the vertical windows, for a 20x20 grid step. In this case, overlapping between windows is allowed. In these figures, we appreciate that the number of windows is much more influential in the results than its size. That is because calculating the histogram of a bigger window is computationally less expensive than computing the histogram of a new cell. In fig. 6(g) the graphs are overlapped because the information stored (the histogram of each cell) has the same size although the windows get bigger.

5. Performance of the descriptors in a localization task

Apart from studying the performance of the descriptors when representing the scenes, it is also necessary to test the accuracy they offer when computing the position and the orientation of the robot in the environment, since both results are crucial in the localization and control phases of the route-following aplication. In this section, we measure this accuracy. When a new image arrives, the position and orientation the robot had when capturing it must be computed comparing the descriptor of the new image with the descriptors in the map. To carry out the experiments, we use as test images all the available images in the database, independently of the grid selected to make the map, and 15 artificial rotations of each one (every 22.5º). So, the total number of test images is 11,936.
We study separately the computation of the position and the orientation of the robot. Position is studied as binary results, considering if we obtain the best possible match or not,

and the information is showed with recall and precision measurements (Gil et al., 2009). Each chart shows the information about if a correct location is in the Nearest Neighbour (N.N.), i. e., if it is the first result selected, or between Second or Third Nearest Neighbours (S.N.N or T.N.N). Regarding the rotation, we represent the results accuracy in bar graphs depending on how much they differ from the correct ones. If the experiment error is bigger than ±10 degrees, it is considered as a failure. The processing time to calculate the position and orientation of the robot is shown in different tables, in seconds.

5.1 Fourier signature technique

The map obtained with the Fourier signature descriptor is represented with two matrices per image: the module and the phase of the Fourier coefficients. When a new image arrives, the first step is to compute its Fourier signature. Then, the location is estimated calculating the Euclidean distance between its module matrix and the module matrices stored in the map. The best match is obtained as the image in the map with minimun distance. Once the position of the robot is known, the phases' matrix associated to the most similar image is used to compute the relative orientation of the robot. Table 2 shows the processing time depending on the images grid step and the number of Fourier components per row. There is a bigger dependency on the number of components than on the grid step (i.e. the number of images in the map). This is due to the orientation estimation, since it is the most computational heavy part of the algorithm and it depends only on the number of components per row.

GRID	2 Comp	5 Comp	10 Comp	20 Comp
10x10	0.0099	0.0130	0.0219	0.0487
20x20	0.0072	0.0107	0.0192	0.0446
30x30	0.0070	0.0104	0.0187	0.0440
40x40	0.0069	0.0103	0.0185	0,0438

Table 2. Processing time (in seconds) of the position and orientation estimation using the Fourier signature, varying the number of components per row and the grid step.

Fig. 7 shows recall and precision measures. When more components per row are taken, the location is better, but there is a limit over which it is not interesting to raise the number of components because the results do not improve. In this case, with 10 components the location is successful. The phase accuracy also improves when more coefficients are used to compute the angle, although it is quite constant when we take 8 or more components.

5.2 PCA over fourier signature

After applying PCA over the Fourier signature modules matrix, we obtain another matrix containing the main eigenvectors, and the projection of the training images onto the space made up with that vectors. These projections are used to calculate the position of the robot. On the other hand, we keep the phases matrix of the Fourier signature directly to estimate the orientation.

To know where the robot is, first the Fourier signature of the current image must be computed. After retaining the desired number of components per row, the vector of modules is projected onto the eigenspace. The most similar image in the map is obtained by

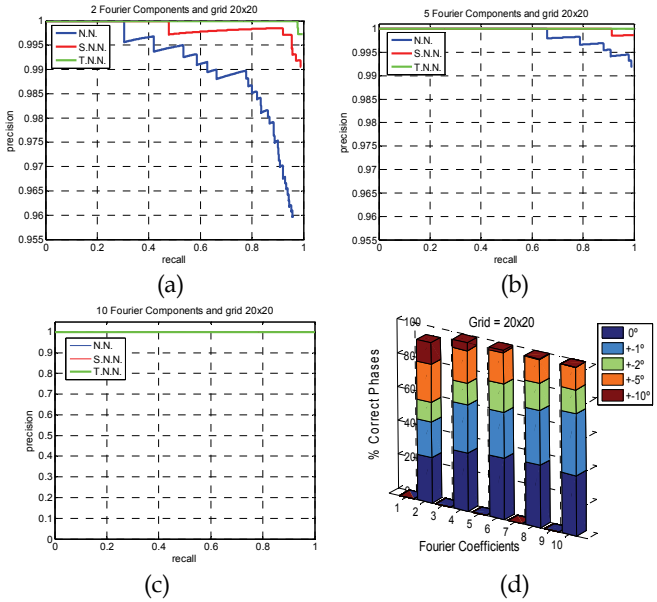


Fig. 7. (a), (b), (c) Recall-Precision charts with Nearest Neighbour (N.N.), Second Nearest Neighbour (S.N.N.) and Third Nearest Neighbour (T.N.N) and (d) accuracy in the calculation of the orientation using the Fourier Signature method.

GRID	2 Comp	5 Comp	10 Comp	20 Comp
10x10	0.1113	0.2420	0.2844	0.3362
20x20	0.0355	0.0613	0.0985	0.1462
30x30	0.0233	0.0483	0.0838	0.1321
40x40	0.0205	0.0459	0.0815	0.1294

Table 3. Processing time (in seconds) of the pose estimation using PCA over Fourier signature when selecting 10 PCA components.

GRID	2 Vectors	5 Vectors	10 Vectors	20 vectors
10x10	0.1505	0.1492	0.1497	0.1497
20x20	0.1294	0.1293	0.1300	0.1295
30x30	0.1278	0.1280	0.1271	0.1273
40x40	0.1272	0.1273	0.1272	0.1270

Table 4. Processing time (in seconds) of the pose estimation using PCA over Fourier signature when selecting 40 Fourier coefficients per row.

calculating the minimum Euclidean distance of the new image's projection and the map ones. When the position is known, the phase is calculated in the same way than when we do not apply PCA since the phase matrix is not projected.

Table 3 presents the processing time to get the position and the orientation of the robot from the moment the current image arrives. To measure this time, the number of eigenvectors we keep is constant. The results show that the elapsed time rises as the number of images in the map or the components in the Fourier signature increase. However, except in the first case, the number of coefficients we take has more influence since the computation of the phase is computationally more expensive than the localization, and it depends on this parameter.

Table 4 shows the processing time when applying PCA over Fourier signature, varying the number of eigenvectors. These results show again that the number of vectors is not very influential since the time spent in the algorithm does not change significantly. Moreover, as far as PCA effects are concerned, there are no important variations in the time when the grid step varies, i.e. projecting the characteristic vector onto the eigenspace is a fast process and there are no important differences when having more images or using more vectors to build the projection space.

As we can see in fig. 8, if a high accuracy in the localization task is needed, it is required a high number of PCA eigenvectors, what means loosing the advantages of applying this method. Moreover, in the majority of experiments, the number of Fourier coefficients we need is bigger than when we do not use PCA, incrementing the memory requirements.

Phase results are not included because the results are exactly the same as showed in fig. 4 since its calculation method does not vary.

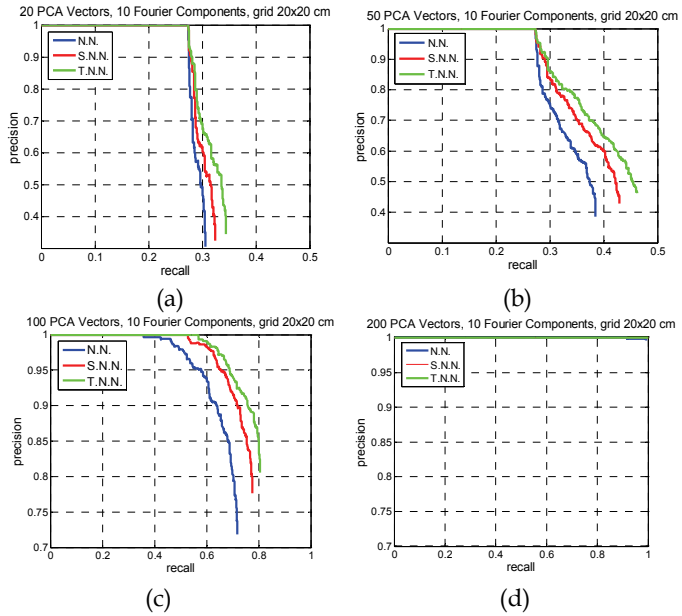


Fig. 8. Recall-Precision charts with Nearest Neighbour (N.N.), Second Nearest Neighbour (S.N.N.) and Third Nearest Neighbour (T.N.N) using PCA over Fourier Signature varying the number of PCA vectors.

5.3. Histogram of oriented gradient

When a new image arrives, first, its histogram of oriented gradient is computed using cells (windows) with the same size as when the map was built. So, the time needed to find the pose of the robot varies depending on both vertical and horizontal cells. To find the position of the robot, the horizontal cells information is used, whereas to compute the phase it is necessary to use the information in the vertical cells. In both cases, the information is found by calculating the Euclidean distance between the histogram of the new image and the stored ones in the map.

GRID	2 pixels	4 pixels	8 pixels	16 pixels
10x10	0.1113	0.2420	0.2844	0.3362
20x20	0.0355	0.0613	0.0985	0.1462
30x30	0.0233	0.0483	0.0838	0.1321
40x40	0.0205	0.0459	0.0815	0.1294

Table 5. Processing time (in seconds) in the pose estimation using HOG, varying horizontal cells height.

GRID	2 pixels	4 pixels	8 pixels	16 pixels
10x10	0.6599	0.5259	0.4979	0.4857
20x20	0.2777	0.1598	0.1240	0.1136
30x30	0.2546	0.1301	0.0982	0.0866
40x40	0.2497	0.1247	0.0943	0.0822

Table 6. Processing time (in seconds) in the pose estimation using HOG, varying vertical cells distance.

When a smaller cells' size is used, more windows appear, so more histograms have to be computed. So, when we reduce the height of the cells, the processing time is bigger (table 5). Moreover, when the map is made up of more images, we have to make more comparisons to find the nearest image, what means to spend more time. But with fewer cells, it is more difficult to recognise the correct position, as shown in fig. 9.
In table 6 the effect of varying the distance between vertical cells is assessed. This parameter conditions the phase computation. The more distance we have, the less time we need, because the number of histograms to compute is lower. However, to improve the angle accuracy, that distance must be reduced.

6. Conclusion

In this paper, we have presented an appearance-based multi-robot route following scheme. The proposed solution uses low resolution panoramic images obtained through a catadioptric system that is mounted on each robot in the system, and appearance based methods to build the map and compute the localization.
In our approach, a leader robot goes through the desired rote while it creates a database with some visual information captured along the route. This database is shared with the follower robots, which have to follow this route from a distance (as in space or in time). To

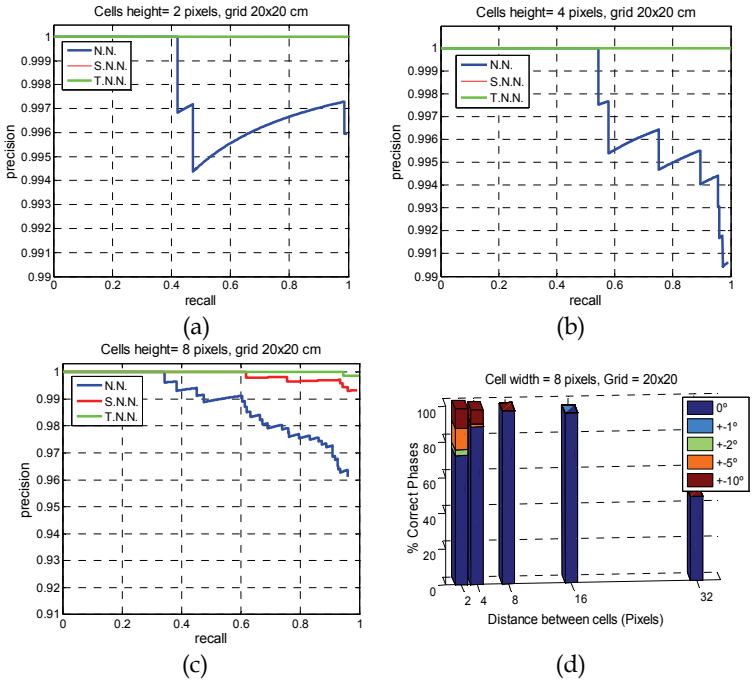


Fig. 9. (a), (b), (c) Recall-Precision charts with Nearest Neighbour (N.N.), Second Nearest Neighbour (S.N.N.) and Third Nearest Neighbour (T.N.N). (d) Phase accuracy using HOG.

do it, a probabilistic algorithm, based on a Markov process has been implemented to calculate their current position among those that the leader has stored, and a controller has been implemented, also based on the appearance of the scenes, to calculate the linear and turning speeds of the robot. To allow a new robot can follow the route that another robot is recording at the same time, an incremental algorithm is presented.

The work is mainly focused in the study of the feasibility of the techniques based on the global appearance of the panoramic scenes to solve the problem. With this aim, we use dimensionality reduction to extract the most relevant information from the environment and build a robust and efficient descriptor of each scene.

We have compared the performance of three different approaches when working with panoramic images: Fourier signature, Principal Components Analysis applied to the Fourier signature and Histogram of Oriented Gradients. Our contribution in the comparison of these three methods is twofold. First, we have studied their performance when representing the environment. We have carried out a set of experiments to test the computational cost to compute each kind of descriptor and the memory requirements to store them. Secondly, we have also studied their validity in localization tasks. In this case, the experiments have allowed us to know the efficiency in calculating the position and the orientation of the robot by comparing its current visual information with all the descriptors previously stored in the database.

All the description methods have demonstrated to be valid to represent the environment and to carry out the estimation of the pose of a robot inside the map. However, the Fourier

signature has proved to be the most efficient method since with few components of the Fourier Transform of each row we obtain good results. So, the map does not need a huge amount of memory to be stored, and the comparison is not a computationally heavy process. No advantage has been found in applying PCA to the Fourier signature, since creating the eigenspace is a computationally expensive task, and in order to have good results it is needed to keep the great majority of the eigenvectors obtained. Moreover, because we need more Fourier coefficients, the size of the map is not reduced. Regarding HOG, although the results demonstrate it is a robust method, it is not very flexible due to its time and memory requirements, since the histogram computation is a time-consuming task, and if we want to improve the orientation accuracy, the map's memory increases notably.
This paper shows again the wide range of possibilities of appearance-based methods applied to mobile robotics, and its promising results encourage us to continue studying them in deep, looking for new available techniques. Our future work includes studying how to build more sophisticated topological maps of the environment and how to cope with some typical problems in indoor environments such as occlusions, changes in the lighting conditions and changes in the position of some objects of the scene. Also, these new maps will require the use of new probabilistic localization methods, such as Monte Carlo approaches, whose applicability to appearance-based data must be explored.

7. Acknowledgements

This work has been supported by the Spanish government through the project DPI2010-15308. 'Exploración integrada de entornos mediante robots cooperativos para la creación de mapas 3D visuales y topológicos que puedan ser usados en navegación con 6 grados de libertad'.

8. References

Booij, O.; Terwijn, B.; Zivkovic, Z. & Kröse, B. (2007). Navigation using an Appearance Based Topological Map. *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 3297-3932, ISBN 1-4244-0602-1, Roma, Italy, IEEE

Burschka, D. & Hager, G. (2001). Vision-Based Control of Mobile Robots, *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 1707-1713, ISBN 0-7803-6578-X, Seoul, Korea, May 2001, IEEE

Dalal, N. & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 886-893, ISBN: 0-7695-2372-2, San Diego, USA, Jun. 2005. IEEE Press.

Gil, A.; Martinez, O.; Ballesta, M. & Reinoso, O. (2009). A comparative evaluation of interest point detectors and local descriptors for visual SLAM. *Machine Vision and Applications*, (Apr. 2009) ISSN: 0932-8092.

Ho, K. & Newman, P. (2005). Multiple Map Intersection Detection using Visual Appearance. *Proceedings of 3rd International Conference on Computational Intelligence, Robotics and Autonomous Systems*, Singapore, Dic. 2005.

Jogan, M. & Leonardis, A. (2000). Robust Localization Using Eigenspace of Spinning-Images. *Proceedings of IEEE Workshop on Omnidirectional Vision*, pp. 37-44, ISBN 0-7695-0704-2, Hilton Head Island, USA, Jun 2000, IEEE.

Kirby, M. (2000). *Geometric data analysis. An empirical approach to dimensionality reduction and the study of patterns*, Wiley Interscience, ISBN 0-4712-3929-1

Kröse, B.; Bunschoten, R.; Hagen, S.; Terwijn, B. & Vlassis, N. (2004). Household robots: Look and learn. *IEEE Robotics & Automation magazine*, Vol. 11, No. 4, (Dec 2004) 45-52, ISSN 1070-9932

Matsumoto, Y.; Ikeda, K.; Inaba, M. & Inoue, H. (1999). Visual Navigation Using Omnidirectional View Sequence. *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, pp. 317-322, ISBN: 0-7803-5184-3, New York, USA, Oct 1999, IEEE Press

Menegatti, E.; Maeda, T. & Ishiguro, H. (2004). Image-based memory for robot navigation using properties of omnidirectional images. *Robotics and Autonomous Systems*. Vol. 47, No. 4, (Jul 2004), 251-276, ISSN 0921-8890

Möller, R.; Vardy, A.; Kreft, S. & Ruwisch, S. (2007). Visual homing in environments with anisotropic landmark distribution. *Autonomous Robots*, Vol. 23, No. 3, (Oct. 2007) pp. 231-245, ISSN: 0929-5593.

Payá, L.; Fernández, L.; Reinoso, O.; Gil, A. & Ubeda, D. (2009). Appearance-based Dense Maps Creation: Comparison of compression techniques with panoramic images, *Proceedings of 6th International Conference on Informatics in Control, Automation and Robotics*, pp. 250-255, ISBN: 978-989-674-000-9, Milan, Italy, Jun. 2009, INSTICC Press.

Payá, L.; Reinoso, O.; Gil, A.; Pedrero, J. & Ballesta, M. (2007). Appearance-Based Multi-Robot Following Routes Using Incremental PCA. *Lecture Notes in Artificial Intelligence. Knowledge-Based Intelligent Information and Engineering Systems*. Vol. 4693, (Sep. 2007) pp. 1170-1178, ISSN: 0302 -9743.

Payá, L.; Reinoso, O.; Gil, A. & Sogorb. J. (2008). Multi-robot route following using omnidirectional vision and appearance-based representation of the environment. *Lecture Notes in Artificial Intelligence. Hybrid Artificial Intelligence Systems*, Vol. 5271, (Sep 2008) pp. 680-687, ISSN 0302-9743.

Rossi, F.; Ranganathan, A.; Dellaert, F. & Menegatti, E. (2008). Toward topological localization with spherical Fourier transform and uncalibrated camera. *Proceedings of International Conference on Simulation, Modeling and Programming for Autonomous Robots*, pp. 319-330, ISBN 978-88-95872-01-8, Venice, Italy, Nov 2008.

Thrun, S. (2001). A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, Vol. 20, No. 5, (May. 2001), pp. 335-363, ISSN: 0278-3649.

Thrun, S. (2003). Robotic Mapping: A Survey, In: *Exploring Artificial Intelligence in the New Milenium*, 1-35, Morgan Kaufmann Publishers, ISBN 1-55860-811-7, San Francisco, USA

Thrun, S.; Burgard, W. & Fox D. (2005). *Probabilistic Robotics*. MIT Press, ISBN: 0-262-20162-3, Cambridge, USA.

Ueonara, M. & Kanade, T. (1998). Optimal approximation of uniformly rotated images: relationship between Karhunen-Loeve expansion and Discrete Cosine Transform. *IEEE Transactions on Image Processing*. Vol. 7, No. 1, (Jan. 1998) pp. 116-119, ISSN: 1057-7149.

Vasudevan, S.; Gächter, S.; Nguyen, V. & Siegwart, R. (2007). Cognitive maps for mobile robots – an object based approach. *Robotics and Autonomous Systems*, Vol. 55, No. 5, (May. 2007) pp. 359-371, ISSN 0921-8890.