AISTA 2004 in Cooperation with the IEEE Computer Society Proceedings / ISBN : 2-9599776-8-8
15-18 November, Kirchberg - Luxembourg

Implementation of a vision-based navigation framework on a house mobile robot prototype

Guillaume Blanc*, Thierry Chateau*, Youcef Mezouar*, Omar Aït Aïder * and Philippe Martinet*
Laurent Eck†, Vincent Moreau†, and Arnaud Nadim†

*LASMEA
24 Avenue des Landais
63177 AUBIERE - FRANCE
Email: firstname.lastname@lasmea.univ-bpclermont.fr

†CEA/List
BP 6
92265 FONTENAY AUX ROSES - FRANCE
Email: firstname.lastname@cea.fr

***Abstract*— Autonomous robots conception has to face developing efficient localization and control algorithms which could be implemented in a small electronic system, embedded on a robot. The WACIF project, supported by the French government, takes up this challenge. It aims to design algorithms and electronics for an autonomous surveillance robot at home. Thanks to an original image-based framework, the developed prototype is able to navigate autonomously into an indoor environment. All the navigation algorithms are running on a programmable on-board hardware system controlling a two wheeled nonholonomic mobile robot, which is equipped with an embedded monocular vision system.**

The proposed framework includes the concept of visual memory of the environment. In an unknown environment, this memory results from a supervised learning step. A reference path, to be performed later in an autonomous way, is recorded as a set of ordered key images of the environment, called visual path. The robot performs the learnt path thanks to recently developed approaches of vision-based localization and visual servoing for nonholonomic mobile robots.

I. Introduction

Mobile robots are entering consumers houses. Industrial products are yet available for household, like "companion" toys, or vacuum cleaners, but whose navigation capabilities are limited because of industrial constraints on electronics performances since the production prices are hoped as low as possible. We have dealt with this issue since we have been developing a small mobile robot which aim is to navigate autonomously in an indoor environment, to achieve surveillance tasks. Although our developments have been constrained by industrial requirements, our navigation framework is exclusively vision-based, even if associated algorithms are renowned to be time consuming.

Implementing navigation algorithms for an autonomous mobile robot onto a small dedicated device is a great challenge. Algorithms efficiency and robustness must be guaranteed while the implementation support performances are limited. A good adequation between navigation approaches and algorithms development on one hand, and flexible but powerfull computation electronics devices on the other hand appears vital.

Fig. 1. The developed prototype

Classically, the three main subtasks of a navigation framework are environment learning, robot localization onto a model of its environment, and robot control along a planned path in order to reach a given goal. Looked at from that point of view, approaches based on simultaneous localization and mapping of the environment (SLAM) have been extensively studying [1]. During an autonomous run, a mobile robot can follow a planned path in the current map, which is updated thanks to the sensor observations. Although such methods have already been implemented on real autonomous plateforms [2], the resulting robotic systems are far from an everyday life mobile robot, either in terms of costs or device performances.

In indoor environments, telemeters and odometry are the most classical used sensors. By merging their data using for instance an extended Kalman filter, the robot location into its map can be enough accurately estimated in order to achieve a complete navigation task [3]. However, considering a low cost mobile robot, whose either hardware structure and sensors data are poorly accurate, other exteroceptive sensors have to be used for better results. Because of their decreasing costs and rich informations they can provide, vision systems are more and more used to develop navigation frameworks. In such approaches, landmarks recognition and tracking often holds a central rule for precisely navigating within a map [4], [5], [6], [7]. On one hand, this can provide an accurate positioning of an embedded camera with respect to the landmarks which are in the camera field of view. On an other hand, sensor-based approaches for trajectory planning and following can be applied using this kind of visual data [8], in order to increase the robustness of navigation processes [3].

The approach we propose is based on an original image-based framework, whose theoretical bases are described in [9] and [10]. Using low cost imaging CMOS sensor which is turned to the ceiling, this framework includes unknown environment learning, robot localization and control. During a learning stage, the robot is teleoperated by the user, and describes paths it should be able to perform during an autonomous run. That human-guided learning is exploited to build a "mosaic" of the ceiling. It is a plane where some interesting patterns of the ceiling, acquired during the training phase, are projected into [11]. The robot is located and controlled into the mosaic space, where the robot's mission is specified as a path to be performed automatically, called visual route. This path is described by a sequence of states of the robot in the mosaic, which results from a selection of key images along human-guided paths [9]. A key image corresponds to a robot configuration in the mosaic space. A visual route is built according especially to patterns visibility between two successive key images and main characteristics of the control law. Key images are used as successive references for the robot visual servoing. The developed vision-based control law conciders that camera motions suffer from the same nonholonomic constraint as the mobile base, on which it is mounted.

Algorithms have been thought and developed with taking into account integration objectives. Furthermore, electronics have been designed to fit algorithms as well as possible. A prototype supported by a Pekee$^{\mathrm{TM}}$ robot platform results from this cooperation, as presented by Figure 1.

The next Section presents backgrounds on vision-based localization principles and computer vision tools. Section III details how we formalize the robot navigation mission and the involved controller. Then Section IV describes the robot prototype architecture. Experimental results, presented in Section V, show the behavior of our prototype in non cooperative environment.

II. VISION-BASED LOCALIZATION

The localization of the wheeled indoor mobile robot is achieved into the mosaic. This modeling of the robot environment is based on a geometrical arrangement of models of natural landmarks projection into the embedded camera image. These models result from an analysis of patterns projection into reference images, taken during the environment learning step. A tracking algorithm has been developed to track patterns lying on the same 3D plane, which is the ceiling hereafter, while the robot is navigating. This modelization does not require any accurate 3D reconstruction of the robot environment since it uses only 2D geometrical data which are extracted from images.

A. A mosaic of planar patterns

From a visual point of view, indoor environment abounds in planes. The geometrical structure of walls, the ceiling, doors or furniture suggests using planes for an image based navigation framework. Nevertheless the scene rigidity is strongly required to localize the robot onto it. In a room, the most static elements are generally in height. That is a reason why we choose to turn the embedded camera facing the ceiling.

If two patterns are lying on a static plane Π, their respective configurations onto Π can be simplily linked through a constant rigid transformation, which only consists of one rotation around the normal vector to Π and two translations along two orthogonal vectors of Π. Unfortunately, the perspective projection of these patterns onto the image affects the above mentioned transformation with an homography, relative to Π. In a general case, this homography can be computed thanks to existing algorithms, as in [12], based on multiviews geometry. However, an interesting particular case is obtained since the camera optical axis is colinear with the normal vector to Π. Indeed, the transformation in the image plane between the projection of two landmarks consists of the same rotation as the rigid transformation between these two landmarks in Π, and the same translations up to a constant scale factor $\alpha = \frac{f}{d}$, where f denotes the focal length of the camera. Moreover, when the Euclidean distance d between the camera optical center and Π remains constant while the camera is moving, the motion of every planar patterns from Π in the image plane is only characterized by a planar transformation with three degrees of freedom.

This particular case can be considered in pratice when Π is the ceiling plane, and the robot is constrained to move onto a ground plane parallel to the ceiling. Then, since a model of a pattern lying on Π can be detected and tracked in the current image, the transformation between current and initial model's attitude can be computed. The knowledge of this transformation allows the localization of a frame attached to the camera image in a plane where an observed pattern projection is lying on. Moreover, when a new interesting pattern is detected in the current image during the robot navigation, it can be easily anchored in this plane with respect to every other known pattern models, since one of them is visible. The resulting set of models of planar patterns, which are anchored in the same plane, constitutes the mosaic $\mathcal{M}$ of Π [11], [9]. Figure 5 illustrates this mosaic building.

As a consequence, the robot is not localized in a absolute Euclidean space. While the robot is navigating, the current camera image is in fact localized in a 2D space thanks to the observation of the attitude of known patterns in this space. A central clue of the proposed approach thus lies in the development of an efficient method to track models of projected patterns from the ceiling.

B. Tracking of planar patterns

This section deals with the method used to track each local pattern into a real time video sequence. The approach is based on the method presented into [13].

The goal of the tracking is to find the probability distribution function (pdf) $p(\mathbf{x}_k|\mathbf{Z}_k)$, from the pdf $p(\mathbf{Z}_k|\mathbf{x}_k)$; where $\mathbf{x}_k$ is the state vector composed by the model parameters and $\mathbf{Z}_k$ is the observation vector composed by the history of the measures $\mathbf{Z}_k = \{\mathbf{z}_1, ..., \mathbf{z}_k\}$. At time k, the state of each local pattern is defined by $\mathbf{x}_k = \{\mathbf{Pc}_k, sc_k, \theta_k\} \in \mathbb{R}^4$ with $\mathbf{Pc}_k = \{xc_k, yc_k\}$ the position of the pattern center, θ_k, a rotation parameter and sc_k, a scale factor. Considering the canonical camera configuration we are here interesting in, sc_k is not supposed to vary. $\mathbf{x}_k$ defines the transformation between the pattern model and its projection into the image at time k.

Tracking is achieved by a particle filter based approach. Details about particle filters can be find into [14], where a general description of Monte-Carlo methods is done, and in [15], where particle filters are used for visual tracking through CONDENSATION algorithm. The main idea of the particle filter is to estimate the pdf $p(\mathbf{x}_k|\mathbf{Z}_k)$ with a finite set of components, called particles $S_k = \{(\mathbf{s}_k^{(n)}, \pi_k^{(n)})|n = 1...N\}$. Each component consists of an hypothesis on the value of the state vector $(\mathbf{s})$ and a value of probability (π) associated with this hypothesis. The evolution of the set S_k is updated with a propagation of each particle following an evolution model (constant velocity for example). A weight is then affected to each particle according to the observation (measures).

$$\pi_k^n = p(\mathbf{Z}_k|\mathbf{x}_k = \mathbf{s}_k^n) \qquad \sum_{n=1}^{N} \pi_k^n = 1$$

$p(\mathbf{Z}_k|\mathbf{x}_k = \mathbf{s}_k^n)$ is obtained from a criterion based on the observation of image measurements $\mathbf{Z}_k$, associated to $\mathbf{x}_k$. For each particle, this criterion quantifies the global adequation between the tracked model and the prediction of its state in the current image.

The set of particles S_k is then restored by a random draw of N particles where each particle can be drawn with a probability of $\pi_k^{(N)} = p(\mathbf{z}_k|\mathbf{x}_k = \mathbf{s}_k^{(N)})$. Each particle can be draw several times. An estimation of the state vector, for each iteration is given by:

$$\hat{\mathbf{x}}_\mathbf{k} = E(S) = \sum_{n=1}^{N} \pi_k^{(n)} \mathbf{s}_k^{(n)}$$

The particle filter insures in its own ways a form of robustification. However this robustness is linked to the number of used particles and to the diffusion noise. It induces a cost in terms of computations and precision. To deal with this issue, a dedicated hardware has been designed to efficiently process all the algorithm steps. The measurement function used to compute observation criteria has particularly appeared as a central clue (see Section IV).

Building the model of a local pattern consists in acquiring in a reference image a set of points of interest which are chosen by computing directional gradient extrema from a specified center. Thus, local image data are associated with each point of interest. These data are used to compute local criteria for each predicted state of the model. The global adequation above mentioned criterion results from these local criteria, that increases the tracker robustness to partial occlusions.

This tracker takes place in all steps of the proposed vision-based navigation framework. It allows the localization of the camera image into $\mathcal{M}$ since it tracks an anchored model to $\mathcal{M}$. Of course it is usefull during the mosaic building, but also when the robot is navigating autonomously. The proposed navigation principle consists in acquiring reference paths during a learning step in the view of performing them during a posterior autonomous run. Thanks to the presented vision-based localization module, these paths can be learnt while the mosaic is under construction, or later, during a tele-operated run of the robot. The next Section describes firstly how these paths are referred into the mosaic and then explains briefly the used control law to perform these paths according to the robot nonholonomic constraint.

III. Robot control along visual route

The strategy of the proposed global navigation framework consists in memorizing key images of the environment while the robot is teleoperated. This *visual path* describes with a set of key images a path the robot should autonomously be able to reproduce thanks to an adequate control law.

A. A visual route

While the robot is teleoperated by the user, its location into the mosaic is computed at each image acquisition, thanks to tracking of known patterns models (*cf* Section II). Thus, camera motion can be observed in the mosaic. Instead of storing for each learnt path the whole camera trajectory, only a set $\{\mathcal{F}_i \mid i = 1, \dots, p\}$ of p significative states of a frame $\mathcal{F}_I \triangleq (O_I, \mathbf{x}_I, \mathbf{y}_I)$, which is attached to the image frame in $\mathcal{M}$, are memorized. Each $\mathcal{F}_i$ is associated to a grabbed key image of the environment while the robot is learning a path. Two thresholds - one on the distance covered by O_I into $\mathcal{M}$ and the other one on the variation of $\mathbf{x}_I$ orientation - triggers the acquisition of a new key image. The whole set of all stored key images during learning phases constitutes the *visual memory* of the robot environment.
Therefore, a navigation mission is described by a *visual route* into this visual memory as a set of key images. This visual route results from the concatenation of pieces of visual paths, and contains a sequence of reference images for the robot vision-based control.

B. Robot Control

A visual route $\Upsilon = \{\mathcal{F}_i \mid i = 1, \dots, p\}$ of p key images can be considered as a global visual servoing task to perform by splitting it into p successive subtasks. In [10], a visual servoing control law to perform a visual path in a general case is presented. This control allows a nonholonomic robot to perform a visual route under the hypothesis that an homography can continuously be estimated along an admissible path for robot between two consecutive key images. Here, as in [9], the p frames $\mathcal{F}_i$ refer the path into $\mathcal{M}$. Instead of interpolating all the configurations $\mathcal{F}_i$ to create a global path to be tracked by $\mathcal{F}_I$, the adopted control strategy consists in regulating successively $\mathcal{F}_I$ on straight lines $\Gamma_i = (O_i, \mathbf{y}_i)$. Assuming that the camera frame is equivalent to the robot controlled frame, a nonholomic kinematic model of a cart-like vehicle is associated to $\mathcal{F}_I$. This non linear kinematic model is turned into an exact linear one by using a chained form transformation. Therefore, thanks to classical linear automatics, a stable control law

$$\omega_c(y, \phi) = -V \cos^3 \phi (K_p y + K_d \tan \phi)$$

is obtained, where $(K_p, K_d) \in \mathbb{R}^{2+}$ are two non null gains. The rotational velocity ω_c of the camera around its optical axis is computed to regulate to zero both the angular deviation ϕ and a lateral deviation y with respect to the current Γ_i to track. Control performances are tuned by K_p and K_d. These two gains determines the theoretical settling distance covered by O_I when tracking Γ_I. Theoretically, the control performances does not depend on the longitudinal velocity V of the camera. V is let free for an eventual supervisor which could adapt it to navigation context. In practice, suitable control performances and efficiency of the pattern tracking relies on tuned values for K_p, K_v and V.

IV. Algorithm/Architecture Adequation and Partitioning

The developed algorithms have been integrated and validated on a small mobile robot (*cf* Figure 1). The hardware architecture should be able to interface easily with the already existing system architecture of the Pekee$^{\mathrm{TM}}$ robot from Wany Robotics. It should cope with the computation requirements of the navigation algorithms, while fitting in place in the robot and having a reasonable power consumption. High integration constraints are usually solved by developing dedicated System On Chip chipset such as those that can be found in smartphones, PDA, etc. In this aim, we used a programmable chip (FPGA) with an embedded hardware processor core. The programmable logic allows the design of various interfaces, of powerful data management and computation units, while the processor allows flexibility for other tasks needed by the navigation algorithms. Another determining factor was the availability of a prototyping board for the targeted device. Providing flash and dynamic memories, several expansion connectors, this board allowed us to design, integrate and validate at an early stage of the project all the navigation algorithms directly on the robot in real situations. It was then possible to design safely a smaller computational board fitting more efficiently our needs.

An EPXA10 chip from ALTERA was used as the main part of the navigation module. It associates a high density FPGA, an embedded 32 bits RISC ARM9$^{\mathrm{TM}}$ processor running at 200 MHz and several peripherals (interrupt controller, dynamic memory controller, timer, UART, ...). The FPGA device drives a CMOS image sensor (the ceiling camera), performs all the navigation algorithms, and sends speed and steering commands to the robot. The system is completed with a WiFi wireless link already available on the robot essentially used during the learning stage for teleoperation. The main functional blocs of the FPGA are shown in Figure 2.

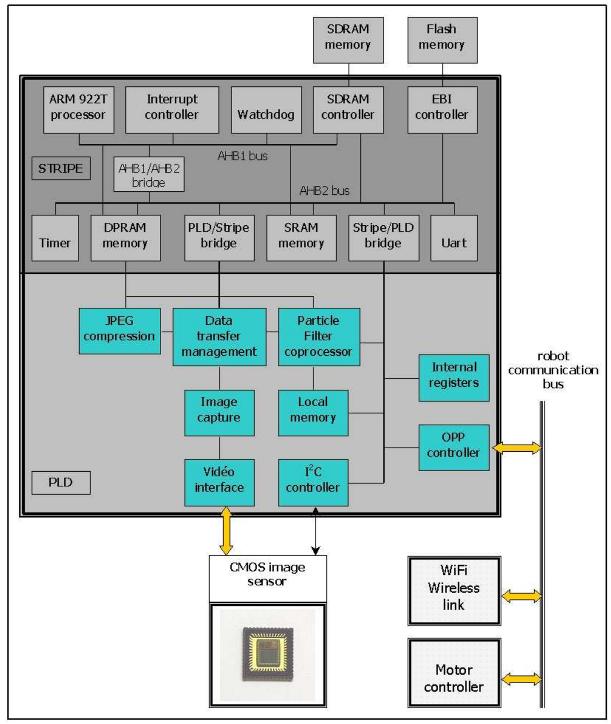


Fig. 2. FPGA main functional hardware blocs

A. The ceiling camera

The ceiling camera consists of a single chip 1/3" VGA CMOS pixel sensor capturing gray scale still or motion images and converting them to a digital raw data stream. While storing this data stream on memory, the FPGA also performs automatic gain and exposure control to adapt in real time for illumination changes. The image sensor is associated with a miniature glass lens offering a 87° diagonal field of view, necessary for covering a wide ceiling area. Due to the statistical nature of the particle filter used for the tracking, image distortion does not need to be corrected and the navigation algorithms have been found to work efficiently.

B. The particle filter coprocessor

Ceiling images are used by the particle filter for tracking planar patterns. Computing requirements for this task are quite heavy and would not fit on the processor alone with the real time constraints of the robot navigation. So most computational part of the particle filter has been implemented in hardware (*cf* Figure 3), relieving the embedded ARM9 processor and insuring good real time performances for the video tracker. Local memories are used to store the particle state and the planar model to be tracked. A model is defined during the supervised learning step with the polar coordinates of characteristic points corresponding to an extremum of image gradient. For each particle, the corresponding state vector (xc, yc, sc, θ) is applied to the model and the new

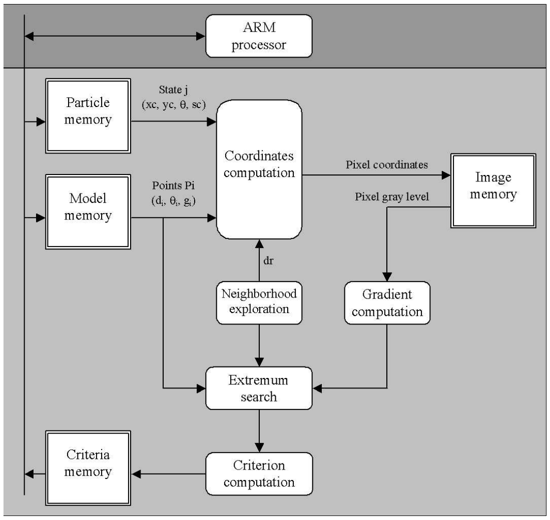


Fig. 3. Block diagram of the particle filter coprocessor

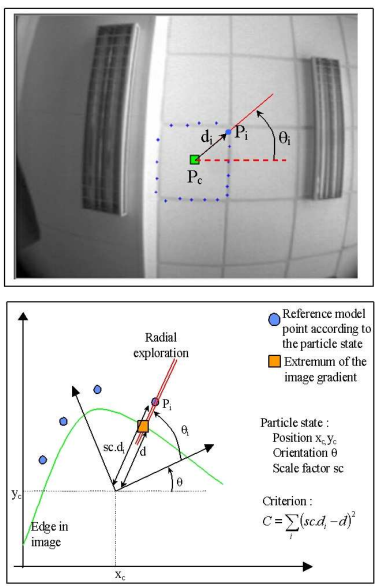


Fig. 4. Top: definition of a planar model. Blue points correspond to local extremum of image gradient. Bottom: particle criterion computation

coordinates of the points are computed (*cf* Figure 4). For each predicted point, a radial exploration is performed in the current image to extract the local extremum of the image gradient. The position of this extremum along the radial axis is compared with the considered particle point position. The resulting error is squared and cumulated over each point, providing a measurement criterion for the particle. This step is repeated for all particles and the corresponding criteria are then stored in local memory. All hardware computations are done using fixed point binary representation. Operators are pipelined, and NxPxV cycles are needed to fill the criteria memory, where N is the number of particles, P is the number of points defining a planar model and V is number of points for neighborhood exploration. In this application the number of particles is limited to 256 particles, the number of points for a model definition is limited to 64 points, and neighborhood exploration is set to 20 points. With the current logic gate array clocked at 40 MHz, and for the above maximum conditions, this leads to an execution time of 8.2ms which is compatible with the real time video data processing. The time required for the overall tracking task is 24ms. It includes image acquisition, particle filter processing and tracker state estimation.

In addition to these specific tasks inherent in tracking algorithm, the FPGA also manages interfaces communication with the others parts of the system. It includes I2C controller and video interface for the image sensor, specific robot communication protocol controller, direct memory data transfer management and jpeg compression. Indeed for monitoring and debug purpose, a wireless link is established between the robot and a remote PC running GUI software. It allows to download configuration for the navigation module and to upload ceiling image, tracker state in the image and robot state in the mosaic. As image transfer could required a large amount of data, the jpeg compression, implemented in hardware, is used to reduce the needed bandwidth.

Finally, management of the navigation task is implemented by the ARM9 processor. It consists of providing the particle filter with models from the mosaic that are possibly visible in the current image. Estimation of the model state in the image is then transposed back to the mosaic plane to give an estimation of the robot localization in the mosaic. From this localization, speed and steering trajectory tracking controls are computed and sent to the motor controller as to drive the robot to the next key image. The cycle time for the overall system is about 70ms, which is compatible with the robot dynamic. The software program size is 92KB while the hardware implementation uses 26% of the FPGA logic resources, among the 38400 logic blocs available in this device. For the FPGA and the associated memories, power consumption measured during experiments was 2.5W.

V. EXPERIMENTS

A. The teleoperated learning step

The learning stage is characterized by a human guided teleoperation of the robot. Starting from an initial point, an image capture is performed by the FPGA and a planar model is automatically generated. This first model will be the anchor point for the mosaic (*cf* Figure 5). Then the particle filter is initialized with this first pattern and the localization is started. When the robot is moved, new models are generated and added to the mosaic with reference to the current tracked model to cover the newly explored space. Trajectories are learnt at the same time than mosaic construction, or they could be learnt later. While being localized in the mosaic, a predefined threshold robot displacement triggers the recording of current robot localization, corresponding to a key image. Together with the mosaic, the collection of all these records obtained during teleoperation of the robot are used to build the navigation visual memory of the environment as a database. This database is small as it consists only of model's points coordinates and robot tracked path coordinates.

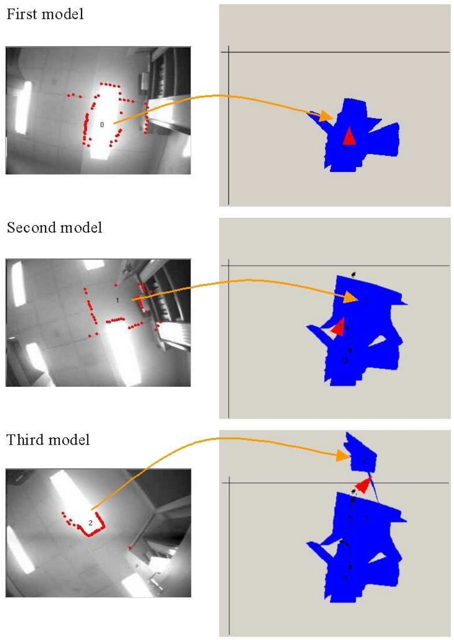


Fig. 5. Incremental building of the mosaic. Left: planar model to add to the mosaic. Right: resulting mosaic plane, the new model is placed relating to the current robot localization

Using a camera facing the ceiling for the navigation provides the benefit that the robot can navigate a learnt path in the reverse way easily. Theoretically if the camera is considered to be ideal and placed at the robot's rotation axis, key images have only to be reversed. Practically the camera and the robot are not ideal, experiments have shown that the main error source comes from misalignment between the lens and the image sensor, which is difficult to avoid on a low cost system. Instead of following hard mechanical constraints, a rough self calibration strategy, illustrated by Figure 6 has been developed to compensate for this error:

- the particle filter is initialized with a planar model
- the model state (x_1, y_1, θ_1) get from the particle filter is memorized
- the robot spins around, until the particle filter exhibits a $180°$ rotation
- the new model state $(x_2, y_2, \theta_1 + 180)$ is memorized

An estimation of the image center of rotation is obtained as the middle point between $(x1, y1)^\top$ and $(x2, y2)^\top$. Coordinates for the learnt path (represented with light blue triangles in Figure 7) and the robot localization (represented with red triangle in Figure 7) are then corrected according to this calibration. With this compensation, experiments show that the robot is able to follow a trajectory in both way without showing significant deviation.

B. Autonomous navigation

A GUI running on a remote PC was built to configure the robot's navigation module. It allows to display the mosaic, the learnt visual paths and the current robot localization within the mosaic. In Figure 7 the small black dots, linking the light blue triangles, are samples of the robot localization during an autonomous navigation. It gives a view of the performed path within the mosaic while the robot is automatically driven to follow a desired visual route (*cf* Section III), represented in Figure 7 by the sequence of light blue triangles.

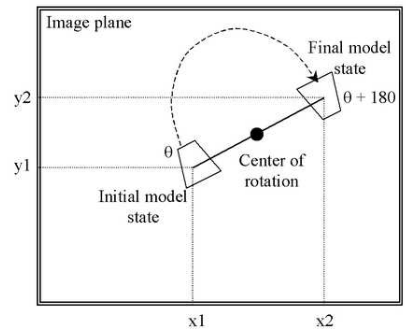


Fig. 6. Estimation for the image center of rotation

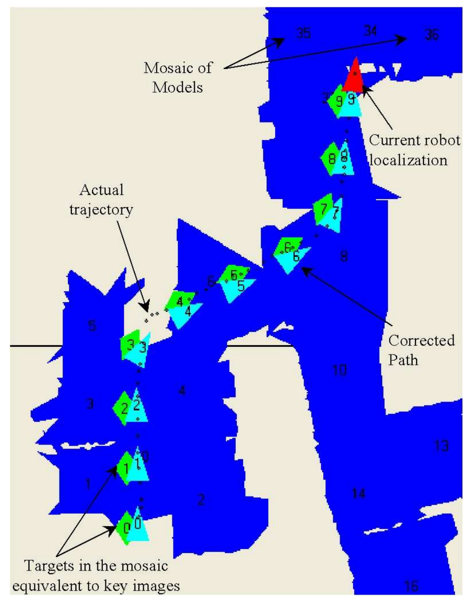
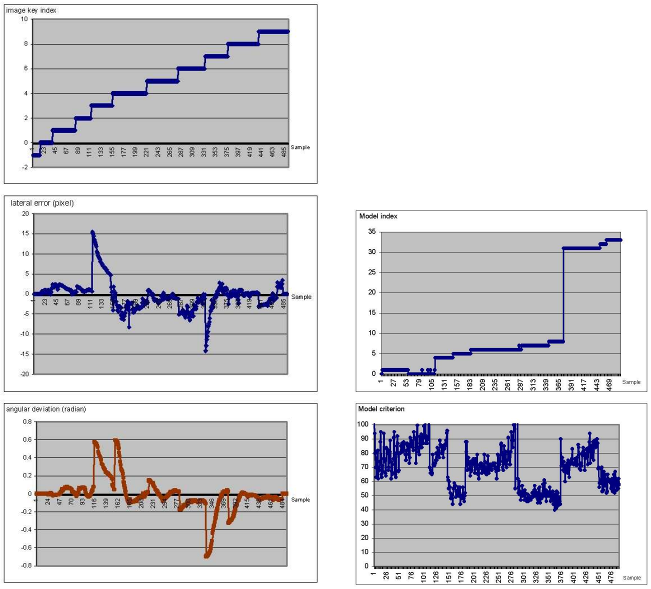


Fig. 7. GUI display for mosaic and trajectory

C. Control performances

There is a trade off to find between the accuracy of the command and the behavioral of the robot. If controller's gains K_p and K_d are set too high, the robot will exhibit sudden direction changes when switching from one key image to the other. To decide whether the key image is reached or not, the distance in the mosaic between the current robot localization and the key image is computed. When the distance is minimum, the robot is considered to have reach his target. Figure 8 (a) shows the lateral error and the angular deviation records between the robot localization and the path target along a trajectory when switching from one key image to the other. In spite of noise on the robot localization measurement, it shows that the robot converges to the successive key images, thus following the correct path.



(a) (b)

Fig. 8. (a): Lateral error and angular deviation for successive key images during an autonomous navigation; (b): Evolution of the selected model (top) and the criterion (bottom) during an autonomous navigation

D. Mosaic use during autonomous navigation

When the robot navigates in an autonomous way, localization is possible if there is at least one model visible in the ceiling image. To ensure this, the mosaic is parsed to select the appropriate model for the particle filter by computing the distance between the robot localization and the model in the mosaic. If this distance is less than a predefined threshold the model is said to be visible. This computation is very fast and a mosaic of 100 models is parsed in less than $300\mu s$. If multiple models are candidates, the most centered in the image is chosen to minimize optical distortion. This selection step takes about $100\mu s$ per visible model. Usually, there is less than 5 models visible together in the same image so this computation is also costless.

The criterion of the selected model is also monitored. When the criterion is high too many times, meaning that the tracking is less reliable, the decision is taken to switch to another model. Figure 8 (b) shows the evolution of criteria during an autonomous run. Looking at the model index vs sample records in Figure 8, we can see that around samples 84, 96 and 101, the system hesitates between model number 0 and 1 because both have bad criteria. As the robot go on, model number 4 becomes the most centered and is selected because of its best criterion.

VI. Conclusion

This paper presents developments and implementation of an exclusively vision-based navigation framework for an indoor wheeled mobile robot. An original mosaic of the ceiling is used as visual memory of the environment. The robot is located into this mosaic and is able to follow a predefined path, described by a set of key images of the mosaic. The development of a robust tracker of planar pattern into the embedded camera image is a central clue of this approach, as the automatic control of the camera motion with taking into account the robot nonholonomic constraint. The challenge of implementing the navigation algorithms on a dedicated small device has been successfully taking up, in spite of hard requirements due to industrial perspectives.

Acknowledgment

This work is supported by the WACIF project (2002-2004), a French government RNTL project.

References

[1] G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-White, and M. Csorba, "A solution to the simultaneous localisation and map buildding (slam) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–242, June 2001.

[2] W. Burgard, A. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot," *Artificial Intelligence*, vol. 114, no. 1-2, pp. 3–55, 1999.

[3] A. C. Victorino and P. R. ans J-J. Borrelly, "Mobile robot navigation using a sensor-based control strategy," in *IEEE International Conference on Robotics and Automation (ICRA 2001)*, Seoul, Korea, May 2001, pp. 2753–2758.

[4] A. J. Briggs, D. Scharstein, D. Braziunas, C. Dima, and P. Wall, "Mobile robot navigation using self-similar landmarks," in *IEEE International Conference on Robotics and Automation (ICRA 2000)*, San Francisco, April 2000, pp. 1428–1434.

[5] J. Hayet, F. Lerasle, and M. Devy, "A Visual Landmark Framework for Indoor Mobile Robot Navigation," in *Proc. Int. Conf. on Robotics and Automation (ICRA'02)*, Washington DC, USA, 2002, pp. 3942–3947.

[6] F. Launay, A. Ohya, and S. Yuta, "A corridors lights based navigation system including path definition using a topologically corrected map for indoor mobile robots," in *IEEE International Conference on Robotics and Automation (ICRA 2002)*, Washington DC, USA, May 2002, pp. 3918–3923.

[7] M. Mata, J. Armingol, A. de la Escalera, and M. Salichs, "Using learned visual landmarks for intelligent topological navigation of mobile robots," in *Proc. of the 2003 IEEE Int. Conf. on Robotics and Automation*, Taipei,Taiwan, September 2003, pp. 1324–1329.

[8] J. Kosecka, "Visually guided navigation," in *Proc. 4th Int. Symp. on Intelligent Robotic Systems (SIRS'96)*, Lisbon, Portugal, July 1996.

[9] G. Blanc, O. A. Aider, Y. Mezouar, T. Chateau, and P. Martinet, "Autonomous image-based navigation in indoor environment," in *IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, Lisboa, Portugal, July 2004.

[10] G. Blanc, Y. Mezouar, and P. Martinet, "An image driven vision-based control framework for wheeled mobile robots," in *International Symposium on Robotics and Automation*, vol. 1, Querétaro, México, August 2004, pp. 507–512.

[11] O. A. Aider, T. Chateau, and J. T. Lapresté, "Indoor autonomous navigation using visual memory and pattern tracking," in *British Machine Vision Conference*, vol. 2, Kingston, England, September 2004, pp. 657–665.

[12] E. Malis and F. Chaumette, "2 1/2 d visual servoing with respect to unknown objects through a new estimation scheme of camera displacement," *International Journal of Computer Vision*, vol. 37(1), pp. 79–97, June 2000.

[13] T. Chateau and J. T. Lapresté, "Real time tracking with occlusion and illumination variations," in *ICPR, International Conference on Pattern Recognition*, vol. IV, aug 2004, pp. 763–766.

[14] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[15] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *IJCV : International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.