PATTERN TRACKING AND VISUAL SERVOING FOR INDOOR MOBILE ROBOT ENVIRONMENT MAPPING AND AUTONOMOUS NAVIGATION

O. Ait Aider, G. Blanc, Y. Mezouar and P. Martinet
LASMEA
UBP Clermont II, CNRS - UMR6602
24 Avenue des Landais, 63177 Aubiere, France
firstname.lastname@lasmea.univ-bpclermont.fr

Keywords: Mobile robot, autonomous navigation, pattern tracking, visual servoing.

Abstract: The paper describes a complete framework for autonomous environment mapping, localization and navigation using exclusively monocular vision. The environment map is a mosaic of 2D patterns detected on the ceiling plane and used as natural landmarks. The robot is able to localize itself and to reproduce learned trajectories defined by a set of key images representing the visual memory. A specific multiple 2D pattern tracker was developed for the application. It is based on particle filtering and uses both image contours and gray scale level variations to track efficiently 2D patterns even on cluttered ceiling appearance. When running autonomously, the robot is controlled by a visual servoing law adapted to its nonholonomic constraint. Based on the regulation of successive homographies, this control law guides the robot along the reference visual route without explicitly planning any trajectory. Real experiment results illustrate the validity of the presented framework.

1 INTRODUCTION

For an indoor mobile robot system, the ability to autonomously map its environment and localize itself relatively to this map is a highly desired property. Using natural rather than artificial landmarks is another important requirement. Several works using monocular vision for mapping and self localization exist. The most important difficulty is to achieve the generation of a sufficient number of landmarks which can be robustly recognized during navigation session with a near real time rate. Interest points (Se et al., 2001), straight lines (Talluri and Aggarwal, 1996) and rectangular patterns (Hayet, 2003) were used. The first approaches focused on producing efficient algorithms to match a set of observed patterns with a subset of the map primitives (Talluri and Aggarwal, 1996). Multiple sensor data fusion (odometry) was generally used to achieve real time computing by eliminating ambiguities. More recently, the success of real-time tracking algorithms simplified the matching process and allowed to use structure from motion technics to compute the 3D coordinates of the observed features (Se et al., 2001).

In this paper, we show how pattern tracking and visual servoing is used to provide a complete framework for autonomous indoor mobile robot application. The focus is done on three main points: i) environment mapping with autonomous localization thanks to pattern tracking, ii) path learning using visual memory, iii) path reproducing using visual servoing.

The environment map is a mosaic of 2D patterns detected on the ceiling plane and used as natural landmarks. This makes the environment representation minimalist and easy to update (only vector representations of the pattern are stored rather than entire images). During a learning session, landmarks are automatically detected, added to the map and tracked in the image. The consistency of the reconstruction is guaranteed thanks to the pattern tracker which enables robot pose updating.

A specific pattern tracker was developed for the application. This was motivated by the fact that for realistic robotics applications there is a need for algorithms enabling not only pattern tracking but also automatic generation and recognition. Among the large variety of existing tracking methods, model-based approaches provide robust results. These methods can use 3D models or 2D templates such as appearance models (Jurie and Dhome, 2001; Black and Jepson, 1996) or Geometric primitives as contour curves and CAD description (Marchand et al., 1999; Lowe, 1992; Pece and Worrall, 2002). Object recognition algorithms based on segmented contours (straight lines, ellipses, corners,...) have reached today a high level of robustness and efficiency. Thus, it seems judicious to investigates trackers which use contour based pattern models. Note that tracking in cluttered background and with partial occlusions is challenging because representing patterns with only contour information may produce ambiguous measurements. Extensive studies of either active or rigid contour tracking have been presented in literature (Isard and Blake, 1998; Zhong et al., 2000; Blake et al., 1993; Bascle et al., 1994; Kass et al., 1988). The used methods are usually defined in the Bayesian filtering framework assuming that the evolution of the contour curve state follows a Markov process (evolution model) and that a noisy observation (measurement model) is available. The contour state is tracked using a probabilistic recursive prediction and update strategy (Arulampalam et al., 2002). More recently, Particle filtering was introduced in contour tracking to handle non Gaussianity of noise and non linearity of evolution model (Isard and Blake, 1998). The pattern tracker used here, and whose first results were presented in [*], is a contour model-based one and takes into account the image gray scale level variations. It uses the condensation algorithm (Isard and Blake, 1998) to track efficiently 2D patterns on cluttered background. An original observation model is used to update the particle filter state.

The robot is also able to reproduce learned trajectories defined by a set of key images representing the visual memory. Once the environment map is built, a vision-based control scheme designed to control the robot motions along learned trajectories *(visual route)* is used. The nonholonomic constraints of most current wheeled mobile robots makes the classical visual servoing methods unexploitable since the camera is fixed on the robot (Tsakiris et al., 1998). However, motivated by the development of 2D 1/2 visual-servoing method proposed by Malis *et al* (see (Malis et al., 1999)), some authors have investigated the use of homography and epipolar geometry to stabilize mobile robots (Fang et al., 2002), (Chen et al., 2003). In this paper, because the notions of visual route and path are very close, we turn the nonholonomic visual-servoing issue into a path following one. The designed control law does not need any explicit off-line path planning step.

The paper is organized as follows: in section II, the autonomous environment mapping, robot localization and navigation using key images, and the pattern tracker are described. Section III deals with the design of the control scheme. An experimental evaluation is finally presented in section IV.

2 ENVIRONMENT MAPPING AND TRAJECTORY LEARNING

2.1 Problem Formalization

The mobile robot system is composed of three frames as shown in Figure 1: the world frame $\mathcal{F}_W$, the robot frame $\mathcal{F}_R$ and the camera frame $\mathcal{F}_C$. To localize the robot in its environnement one have to estimate the transformation ${}^W\mathbf{T}_R$ between the world frame and the robot frame. Assuming that the transformation ${}^C\mathbf{T}_R$ between the camera frame and the robot frame is known thanks to a camera-robot calibrating method, the image is corrected so that it corresponds to what would be observed if the camera frame fits the robot frame i.e. ${}^C\mathbf{T}_R = \mathbf{I}$. As the visual landmarks are on the ceilling plane, the correction to apply on the observed data is a homography ${}^C\mathbf{H}_R$. Knowing ${}^C\mathbf{T}_R$, ${}^C\mathbf{H}_R$ can be expressed as follows:

$$ {}^C\mathbf{H}_R = \mathbf{K}\left(\mathbf{R} - \mathbf{t}\mathbf{n}^T/d\right)\mathbf{K}^{-1} \qquad (1) $$

Where $\mathbf{K}$ is the camera intrinsic parameters matrix, $\mathbf{R}$ and $\mathbf{t}$ are respectively the rotation matrix and the translation vector in ${}^C\mathbf{T}_R$, $\mathbf{n}$ is the vector normal to the ceilling plane and d is the distance from the robot frame to this plane. After applying the computed correction to image data, one can assume for the clarity of the presentation and without loss of generality that ${}^C\mathbf{T}_R$ is set to the identity matrix.

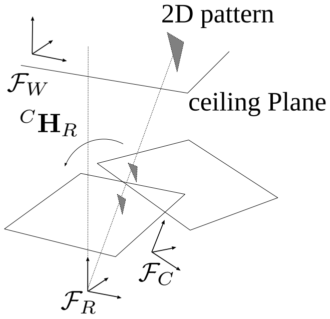

Figure 1: The camera-robot system.

2.2 On-line Robot Pose Computing

Let us denote $\mathcal{F}_M$ the 2D frame defined by the x and y-axis of $\mathcal{F}_W$ and related to the ceiling plane. We assume that a set of 2D landmarks $\mathbf{m}_i$ detected on this plane are modeled and grouped in a mosaic represented by a set $\mathcal{M} = \left\{\left(\mathbf{m}_i, {}^M\mathbf{T}_i\right), i = 1, ..., n\right\}$ where the planar transformation matrix ${}^M\mathbf{T}_i$ between $\mathcal{F}_M$ and a frame $\mathcal{F}_i$ related to $\mathbf{m}_i$ defines the pose of $\mathbf{m}_i$ in the mosaic (Figure 2). We have ${}^M\mathbf{T}_i =$

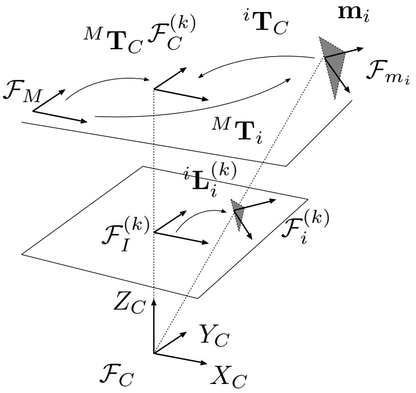

Figure 2: Robot pose computing using visual data of a 2D model.

$\begin{bmatrix} {}^M\mathbf{R}_i & {}^M\mathbf{P}_i \\ \mathbf{0} & 1 \end{bmatrix}$ where ${}^M\mathbf{R}_i = \begin{bmatrix} C\theta_i & -S\theta_i \\ S\theta_i & C\theta_i \end{bmatrix}$ expresses the rotation of the model and ${}^M\mathbf{P}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ its position. Localizing the robot at an instant k consists in computing ${}^M\mathbf{T}_C^{(k)}$ which defines the homogenous transformation between the projection $\mathcal{F}_C^{(k)}$ of the camera frame $\mathcal{F}_C$ on the mosaic plane as shown in Figure 2. At the instant k, the robot grabs an image $\mathcal{I}_k$ of the ceiling. Let $\mathcal{F}_I^{(k)}$ be a 2D frame lied to the image plane. The pose of the projection of an observed model $\mathbf{m}_i$ on the image plane is defined by the transformation ${}^I\mathbf{L}_i^{(k)} = \begin{bmatrix} \mathbf{{}^Ir_i^{(k)}} & \mathbf{{}^Ip_i^{(k)}} \\ \mathbf{0} & 1 \end{bmatrix}$ between $F_I^{(k)}$ and a frame $F_i^{(k)}$ lied to the projection of $\mathbf{m}_i$ where ${}^M\mathbf{r}_i = \begin{bmatrix} C\theta_i^{(k)} & -S\theta_i^{(k)} \\ S\theta_i^{(k)} & C\theta_i^{(k)} \end{bmatrix}$ express the rotation of the model and ${}^M\mathbf{p}_i = \begin{bmatrix} u_i^{(k)} \\ v_i^{(k)} \end{bmatrix}$ its position in the image. Considering the inverse of the perspective projection, we obtain the transformation between the projections $F_C^{(k)}$ and F_i of the camera frame and the model frame respectively on the mosaic plane (Figure 1):

$${}^C\mathbf{T}_i^{(k)} = \begin{bmatrix} {}^C\mathbf{R}_i & {}^C\mathbf{P}_i \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} {}^I\mathbf{r}_i^{(k)} & \frac{Z}{f}{}^I\mathbf{p}_i^{(k)} \\ \mathbf{0} & 1 \end{bmatrix}$$

where Z is the distance from the origin of the camera frame to the ceiling and f the focal length. We can thus express the robot pose relatively to $\mathcal{M}$ with respect to the parameters of a seen model $\mathbf{m}_i$ by

$${}^M\mathbf{T}_C^{(k)} = {}^M\mathbf{T}_i \left({}^C\mathbf{T}_i^{(k)}\right)^{-1} \qquad (2)$$

Note that the robot position is computed up to a scale factor $\frac{Z}{f}$. The absolute position can be retrieved if the camera is calibrated. Otherwise, the computed pose is sufficient to achieve navigation using visual servoing as we will see in section 3.

2.3 Environment Mapping

We will now explain the process of building the mosaic of 2D landmarks. At time $k = 0$, the robot grabs an image $\mathcal{I}_0$, generates a first 2D model $\mathbf{m}_0$ and associates to it a frame $\mathcal{F}_0$. In fact, this first model will serve as a reference to the mosaic. Thus, we have $\mathcal{F}_0 = \mathcal{F}_M$ and ${}^M\mathbf{T}_0^{(k)} = \mathbf{I}$ ($\mathbf{I}$ is the identity matrix). In the image $\mathcal{I}_0$, a tracker τ_0 is initialized with a state ${}^I\mathbf{L}_i^{(0)}$. As the robot moves, the state of τ_0 evoluates with respect to k. The system generates other models. At each generation of a new model $\mathbf{m}_i$ at the instant k, a new tracker τ_k is initialized with a state ${}^I\mathbf{L}_i^{(k)}$. Due to the mosaic rigidity, the transformation between the two model frames is time independent and equal to ${}^0\mathbf{L}_i = {}^0\mathbf{L}_I^{(k)}\,{}^I\mathbf{L}_i^{(k)}$ (Figure3). Noting ${}^0\mathbf{L}_i = \begin{bmatrix} {}^0\mathbf{r}_i & {}^0\mathbf{p}_i \\ \mathbf{0} & 1 \end{bmatrix}$ and projecting this transformation onto the mosaic plane we obtain the pose ${}^M\mathbf{T}_i = \begin{bmatrix} {}^0\mathbf{r}_i & \frac{Z}{f}\mathbf{{}^0p_i} \\ \mathbf{0} & 1 \end{bmatrix}$ of the new model m_i in $\mathcal{M}$:

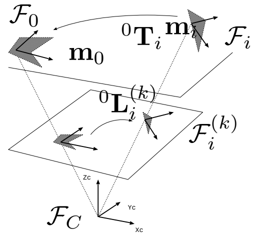

Figure 3: Computing the pose of a new model in the mosaic.

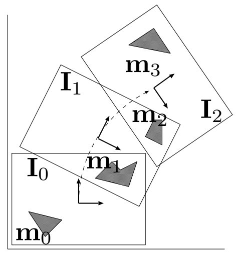

Figure 4: An example of key images forming a trajectory.

Of course, as the robot moves, $\mathbf{m}_0$ may eventually disappear at an instant k when creating a new model $\mathbf{m}_i$. In fact, it is sufficient that at least one model $\mathbf{m}_j$, already defined in the mosaic, is seen at the instant k. To compute the pose ${}^M\mathbf{T}_i$, we first calculate ${}^j\mathbf{L}_i$ and project it to obtain ${}^{\mathbf{j}}\mathbf{T_i}$. The model $\mathbf{m}_i$ is then added to $\mathcal{M}$ with the pose ${}^{M}\mathbf{T}_i ={}^0 \mathbf{T}_j^j\mathbf{T}_i$. Figure 5 shows an example of mosaic construction using the sequence presented on the left hand side. The set of images represents the detected 2D models. The right image is a representation of the constructed mosaic and the robot locations (gray triangles) during navigation.

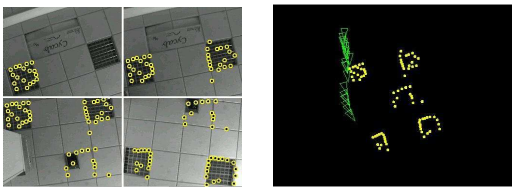

Figure 5: An example of mosaic: (a) a set of key-images with detected 2D patterns, (b) The mosaic and robot locations.

2.4 Path Learning and Visual Navigation

The idea is to enable the robot to learn and reproduce some paths joining important places in the environment. Let us consider a trajectory executed during the learning phase and joining a point A to a point B. A set of so called key images is chosen among the sequence of video images acquired during the learning stage. A key image $\mathcal{I}_k$ is defined by a set of mosaic models and their poses in the image (Figure 4): $\mathcal{I}_k^{(AB)} = \left\{\left(\mathbf{m}_i,^I L_i^{(k)}\right), i = 1, 2, ...\right\}$. A trajectory relating A and B is then noticed $\Phi_{AB} = \left\{\mathcal{I}_k^{(AB)}, k = 1, 2, ...\right\}$. Key images are chosen so that the combination of the elementary trajectories between each couple of successive key images $\mathcal{I}_k^{(AB)}$ and $\mathcal{I}_{k+1}^{(AB)}$ forms a global trajectory which approximatively fits the learned trajectory. Some conditions have to be satisfied when creating Φ_{AB}: i) two successive key images must contain at least one common model of the mosaic, ii) the variation of the orientation of a model between two successive key images is smaller than a defined threshold. The first condition is necessary to visual servoing. The second is motivated by the fact that several different paths joining two poses do exist. It is thus necessary to insert additional key images to reduce the variation of orientation between two successive images (Figure 4).Visual servoing is used to carry the robot from a key image to another.

2.5 Pattern Tracking

The tracking problem can be expressed as the estimation of the state of a dynamical system basing on noisy measurements done at discrete times. Let us consider that, at time k, the state of a system is defined by a vector $\mathbf{X}_k$ and the measurements by a vector $\mathbf{Z}_k$. Based on a Bayesian approach, the tracking consists in iteratively predicting and updating the posterior density function (pdf) of the system state using respectively the dynamical and the observation models. The pdf $p\left(\mathbf{X}_k|\mathbf{Z_{1:k}}\right)$ is thus estimated as the vector $\mathbf{Z}_{1:k} = \left(\mathbf{Z}_i, i = 1, ..., k\right)$ containing the latest measurements becomes available online.

The tracker developed for this application uses the condensation algorithm (Isard and Blake, 1998) which is based on particle filtering theory (Arulampalam et al., 2002). First, a pattern model is defined in the image inside an interest window. It is composed of contours approximated by segments and arcs. The model contains also a list of vectors whose elements represent the evolution of image gray scales in the gradient direction around points sampled on the contours (Figure 6). Polygonal representation of the contours is used for automatic pattern generation and recognition. This enables tracking initialization. During the tracking, only gray scale vectors are used to estimate the state of the pattern. To formalize this model, let us consider a window of interest in the image with a center (x_c, y_c) and dimensions Δ_x and Δ_y. Each segmented contour is sampled in a set of image points $\left\{\mathbf{m}^{(j)} = \left(u^{(j)}, v^{(j)}\right), j = 1, ..., N_m\right\}$ where N_m is the number of points. At each point, we built a vector $\mathbf{V}^{(j)} = \left(g_1^{(j)}, g_2^{(j)}, ..., g_l^{(j)}, ...\right)^T$ composed of N_s gray scale value samples from the image following the gradient direction at the pixel $\mathbf{m}^{(j)}$ and with a fixed step size δ. $g_l^{(j)}$ is a bilinear approximation of the gray scale values of the nearest four pixels. A pattern model can thus be expressed as follows:

$$\mathbf{M} = \left\{\left(\mathbf{U}^{(j)}, \mathbf{V}^{(j)}, \mathbf{W}^{(j)}\right), j = 1, ..., N_m\right\} \quad (3)$$

with $\mathbf{U}^{(j)} = \left[x^{(j)}, y^{(j)}, \phi^{(j)}\right]^T$, where $x^{(j)} = \frac{u^{(j)}-x_c}{\Delta_x}$ and $y^{(j)} = \frac{v^{(j)}-y_c}{\Delta_y}$ are the normalized coordinates of $\mathbf{m}^{(j)}$ inside the interest window and $\phi^{(j)}$ the gradient orientation at $\mathbf{m}^{(j)}$. The vector $\mathbf{W}^{(j)} = \left(a^{(j)}, b^{(j)}, ...\right)^T$ is composed of a set of parameters defining a function $\tilde{C}_{GG}^{(j)}$ which is an approximation of the one-dimensional discrete normalized auto-correlation function $C_{GG}^{(j)}$ of $G^{(j)}$ where $G^{(j)}\left(l\right) = g_l^{(j)}$ for $l = 1, ..., N_s$, and $G^{(j)}\left(l\right) = 0$ elsewhere. We have

$$C_{GG}^{(j)}(\lambda) = \left(1/ \parallel \mathbf{V}^{(j)} \parallel^2\right) \sum_{l=-N_S}^{N_S} G^{(j)}(l)\, G^{(j)}(l-\lambda)$$

The simplest expression of $\tilde{C}_{GG}^{(j)}$ is a straight line equation (Figure 7). $\mathbf{W}^{(j)}$ is then one-dimensional and equal to the slope.

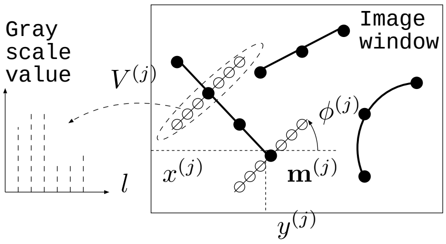

Figure 6: Pattern model: sampled points on segmented contours and corresponding gray scale vectors in the gradient direction.

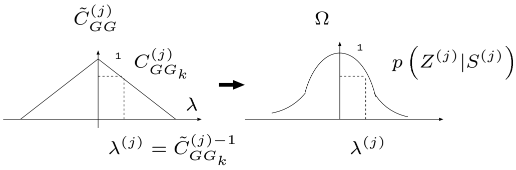
$\tilde{C}_{GG}^{(j)}$
1
$C_{GG_k}^{(j)}$
λ
$\lambda^{(j)} = \tilde{C}_{GG_k}^{(j)-1}$
Ω
1
$p\left(Z^{(j)}|S^{(j)}\right)$
$\lambda^{(j)}$

Figure 7: Deriving the probability measure from the inter-correlation measure.

The state vector $\mathbf{X}_k$ defines the pattern pose in image $\mathcal{I}_k$ at time k :

$$\mathbf{X}_k = (x_k, y_k, \theta_k, s_k)^T \tag{4}$$

x_k, y_k, θ_k are respectively the position of the pattern center and its orientation in the image frame and s_k is the scale factor. The key idea is to use a Monte Carlo method to represent the pdf of $\mathbf{X}_k$ by a set of samples (particles) $\mathbf{S}_k^{(i)}$. A weight $w_k^{(i)}$ is associated to each particle. It corresponds to the probability of realization of the particle. Starting from a set $\left\{\mathbf{S}_k^{(i)}, i = 1, ..., N_p\right\}$ of N_p particles with equal weights $w_k^{(i)}$, the algorithm consists in iteratively: 1) applying the evolution model on each particle to make it evolve toward a new state $\mathbf{S}_{k+1}^{(i)}$, 2) computing the new weights $w_{k+1}^{(i)}$ using the observation model, 3) re-sampling the particle set (particles with small weights are discarded while particles which obtained high scores are duplicated so that N_p remains constant).

A key point in particle filtering is the definition of the observation model $\mathbf{Z}_k$ in order to estimate $w_k^{(i)} = p\left(\mathbf{Z}_k|\mathbf{S}_k^{(i)}\right)$. For each predicted particle $\mathbf{S}_k^{(i)}$, the model is fitted to the image $\mathcal{I}_k$. Around each image point coinciding with a model point $\mathbf{m}_j$, an observed vector $\mathbf{V}_k^{(j)} = \left(g_1^{(j)}, g_2^{(j)}, ..., g_l^{(j)}, ...\right)^T$ of gray scales is built following a direction which is the transformation of the gradient orientation of the model. For each observed vector we compute the normalized inter-correlation between the vector stored in the model and the observed vector components:

$$C_{GG_k}^{(j)} = \frac{\mathbf{V}^{(j)}.\mathbf{V}_k^{(j)}}{\parallel \mathbf{V}^{(j)} \parallel \parallel \mathbf{V}_k^{(j)} \parallel}$$

The question is now how to use the inter-correlation measure to estimate $p\left(\mathbf{Z}_k|\mathbf{S}_k^{(i)}\right)$? We first compute the probability $p\left(\mathbf{Z}_k^{(j)}|\mathbf{S}_k^{(i)}\right)$ that each model point $\mathbf{m}^{(j)}$ is placed according to the state vector particle $\mathbf{S}_k^{(i)}$ on the corresponding point in the observed pattern. The maximum of probability is expected at $\mathbf{m}^{(j)}$. The inter-correlation measure $C_{GG^{(k)}}^{(j)}$ can yeld an estimate of the deviation $\lambda^{(j)} = \tilde{C}_{GG}^{(j)-1}\left(C_{GG_k}^{(j)}\right)$ between the observed and the predicted gray scale vectors, $\tilde{C}_{GG}^{(j)-1}$ being the inverse of the auto-correlation function. Assuming that the probability that the observed gray scale vector fits the predicted one follows a Gaussian distribution with respect to $\lambda^{(j)}$ (Figure 7) we can reasonably approximate $p\left(\mathbf{Z}_k^{(j)}|\mathbf{S}_k^{(i)}\right)$ by $\Omega_\sigma(\lambda)$, the one-dimensional Gaussian function with standard deviation σ. Assuming that the probabilities $p\left(\mathbf{Z}_k^{(j)}|\mathbf{S}_k^{(i)}\right)$ are mutually independent, it results that

$$p\left(\mathbf{Z}_k|\mathbf{S}_k^{(i)}\right) = \prod_{j=1}^{N_p} p\left(\mathbf{Z}_k^{(j)}|\mathbf{S}_k^{(i)}\right) \tag{5}$$

more details about handling partial occlusions and scale changing effects can be found in [*].

3 VISUAL ROUTE FOLLOWING

Visual-servoing is often considered as a way to achieve positioning tasks. Classical methods, based on the task function formalism, are based on the existence of a diffeomorphism between the sensor space and the robot's configuration space. Due to the nonholomic constraints of most of wheeled mobile robots, under the condition of rolling without slipping, such a diffeomorphism does not exist if the camera is rigidly fixed to the robot. In (Tsakiris et al., 1998),

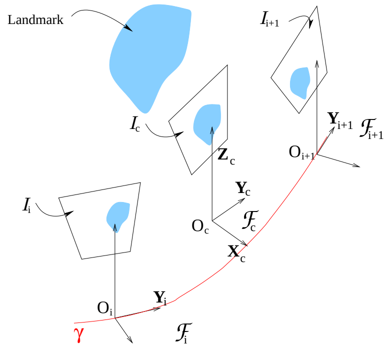

Figure 8: Frames and images: $\mathcal{I}_i$ and $\mathcal{I}_{i+1}$ are two consecutive key images, acquired along a teleoperated path γ.

the authors add extra degrees of freedom to the camera. The camera pose can then be regulated in a closed loop.

In the case of an embedded and fixed camera, the control of the camera is generally based on wheeled mobile robots control theory (Samson, 1995). In (Ma et al., 1999), a car-like robot is controlled with respect to the projection of a ground curve in the image plane. The control law is formalized as a path following problem. More recently, in (Chen et al., 2003), a partial estimation of the camera displacement between the current and desired views has been exploited to design vision-based control laws. A trajectory following task is achieved. The trajectory to follow is defined by a prerecorded video. In our case, unlike a whole video sequence, we deal with a set of relay images which have been acquired from geometrically spaced out points of view.

A visual route following can be considered as a sequence of visual-servoing tasks. A stabilization approach could thus be used to control the camera motions from a key image to the next one. However, a visual route is fundamentally a path. To design the controller, described in the sequel, the key images of the reference visual route are considered as consecutive checkpoints to reach in the sensor space. The control problem is formulated as a path following to guide the nonholonomic mobile robot along the visual route.

3.1 Assumptions and Models

Let $\mathcal{I}_i$, $\mathcal{I}_{i+1}$ be two consecutive key images of a given visual route to follow and $\mathcal{I}_c$ be the current image. Let us note $\mathcal{F}_i = (O_i, \mathbf{X_i}, \mathbf{Y_i}, \mathbf{Z_i})$ and $\mathcal{F}_{i+1} = (O_{i+1}, \mathbf{X_{i+1}}, \mathbf{Y_{i+1}}, \mathbf{Z_{i+1}})$ the frames attached to the robot when $\mathcal{I}_i$ and $\mathcal{I}_{i+1}$ were stored and $\mathcal{F}_c = (O_c, \mathbf{X_c}, \mathbf{Y_c}, \mathbf{Z_c})$ a frame attached to the robot in its current location. Figure 8 illustrates this setup. The origin O_c of $\mathcal{F}_c$ is on the axle midpoint of a cart-like robot, which evolutes on a perfect ground plane.

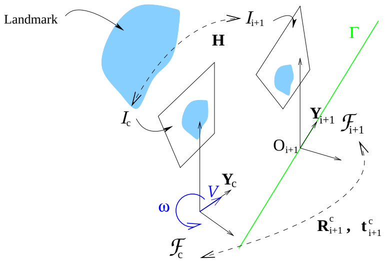

Figure 9: Control strategy.

The control vector of the considered cart-like robot is $\mathbf{u} = [V, \omega]^T$ where V is the longitudinal velocity along the axle $\mathbf{Y_c}$ of $\mathcal{F}_c$, and ω is the rotational velocity around $\mathbf{Z_c}$. The hand-eye parameters (*i. e.* the rigid transformation between $\mathcal{F}_c$ and the frame attached to the camera) are supposed to be known.

The state of a set of visual features $\mathcal{P}_i$ is known in the images $\mathcal{I}_i$ and $\mathcal{I}_{i+1}$. Moreover $\mathcal{P}_i$ has been tracked during the learning step along the path ψ between $\mathcal{F}_i$ and $\mathcal{F}_{i+1}$. The state of $\mathcal{P}_i$ is also assumed available in $\mathcal{I}_c$ (*i.e* $\mathcal{P}_i$ is in the camera field of view). The task to achieve is to drive the state of $\mathcal{P}_i$ from its current value to its value in $\mathcal{I}_{i+1}$.

3.2 Principle

Consider the straight line $\Gamma = (O_{i+1}, \mathbf{Y}_{i+1})$ (see Figure 9). The control strategy consists in guiding $\mathcal{I}_c$ to $\mathcal{I}_{i+1}$ by regulating asymptotically the axle $\mathbf{Y}_c$ on Γ. The control objective is achieved if $\mathbf{Y}_c$ is regulated to Γ before the origin of $\mathcal{F}_c$ reaches the origin of $\mathcal{F}_{i+1}$. This can be done using chained systems. Indeed in this case chained system properties are very interesting. A chained system results from a conversion of a mobile robot non linear model into an almost linear one (Samson, 1995). As long as the robot longitudinal velocity V is non zero, the performances of path following can be determined in terms of settling distance (Thuilot et al., 2002). The settling distance has to be chosen with respect to robot and perception algorithm performances.

The lateral and angular deviations of $\mathcal{F}_c$ with respect to Γ to regulate can be obtained through partial Euclidean reconstructions as described in the next section.

3.3 Evaluating Euclidean State

The tracker presented in Section 2.5 can provide a set of image points $\mathcal{P}_i = \{\mathbf{p}_{ik}, k = 1 \cdots n\}$ belonging to $\mathcal{I}_{i+1}$. These points are matched with the set of image points $\mathcal{P}_c = \{\mathbf{p}_{ck}, k = 1 \cdots n\}$ of $\mathcal{I}_c$. Let Π be a 3D reference plane defined by three 3D points whose projections onto the image plane belong to $\mathcal{P}_i$ (and $\mathcal{P}_c$). The plane Π is given by the vector $\pi^T = [\mathbf{n}^* \; d^*]$ in the frame $\mathcal{F}_{i+1}$, where $\mathbf{n}^*$ is the unitary normal of Π in $\mathcal{F}_{i+1}$ and d^* is the distance from Π to the origin of $\mathcal{F}_{i+1}$. It is well known that there is a projective homography matrix $\mathbf{G}$, relating the image points of $\mathcal{P}_i$ and $\mathcal{P}_c$ (Hartley and Zisserman, 2000): $\alpha_k \mathbf{p}_{ik} = \mathbf{G}\mathbf{p}_{ck}$ where α_k is a positive scaling factor. Given at least four matched points belonging to Π, $\mathbf{G}$ can be estimated by solving a linear system. If the plane Π is defined by 3 points, at least five supplementary points are necessary to estimate the homography matrix (Hartley and Zisserman, 2000). Assuming that the camera calibration $\mathbf{K}$ is known, the Euclidean homography of plane Π is estimated as $\mathbf{H} = \mathbf{K}^{-1}\mathbf{G}\mathbf{K}$ and it can be decomposed into a rotation matrix and a rank 1 matrix: $\mathbf{H} = {}^{i+1}\mathbf{R}_c + {}^{i+1}\mathbf{t}_c \frac{\mathbf{n}^{*\top}}{d^*}$. As exposed in (Faugeras and Lustman, 1988), it is possible from $\mathbf{H}$ to determine the camera motion parameters, that is ${}^{i+1}\mathbf{R}_c$ and $\frac{{}^{i+1}\mathbf{t}_c}{d^*}$. The normal vector $\mathbf{n}^*$ can also be determined, but the results are better if $\mathbf{n}^*$ has been previously well estimated (note that it is the case in indoor navigation with a camera looking at the ceiling for instance). In our case, the mobile robot is supposed to move on a perfect ground plane. Then an estimation of the angular deviation γ between $\mathcal{F}_c$ and $\mathcal{F}_{i+1}$ can be directly extracted from ${}^{i+1}\mathbf{R}_c$. Furthermore, we can get out from $\frac{{}^{i+1}\mathbf{t}_c}{d^*}$ the lateral deviation z up to a scale factor between the origin of $\mathcal{F}_c$ and a straight line Γ (refer to Figure 9).

As a consequence, the control problem can be formulated as following Γ in regulating to zero z and γ before the origin of $\mathcal{F}_c$ reaches the origin of $\mathcal{F}_{i+1}$

3.4 Control Law

Exact linearization of nonlinear models of wheeled mobile robot under the assumption of rolling without slipping is a well known theory, which has already been applied in many vehicle guidance applications, as in (Thuilot et al., 2002) for a car-like vehicle, and in our previous works (see [**]). The used state vector of the robot is $\mathbf{Z} = \left[\begin{array}{ccc} c & z & \gamma \end{array}\right]^\top$, where c is the curvilinear coordinate of a point $\mathbf{M}$, which is the orthogonal projection of the origin of $\mathcal{F}_c$ on Γ. The derivative of this state give the following state space model:

$$\dot{c} = V\cos\gamma; \quad \dot{z} = V\sin\gamma; \quad \dot{\gamma} = \omega_c \qquad (6)$$

The state space model (6) is converted into a chained system of dimension 3 $\left[\begin{array}{ccc} a_1 & a_2 & a_3 \end{array}\right]^\top$. Deriving this system with respect to a_1 gives an almost linear system. By choosing $a_1 = c$ and $a_2 = z$, and thanks to classical linear automatics, it is then possible to design an asymptotically stable guidance control law, which performances are theoretically independent to the longitudinal velocity V:

$$\omega(z,\gamma) = -V\cos^3\gamma K_p z - |V\cos^3\gamma| K_d \tan\gamma \quad (7)$$

K_p and K_d are gains which set the performances of the control law. They must be positive for the control law stability. Their choice determine a settling distance for the control, *i. e.* the impulse response of z with respect to the covered distance by the point $\mathbf{M}$ on Γ.

4 EXPERIMENTAL EVALUATION

4.1 Pattern Tracker Evaluation

The evaluation of the tracker was done by superimposing an image patch (pattern) with known state vector trajectory on a cluttered background image sequence. The goal was to analyse the accuracy of the tracker with respect to the pattern displacement magnitude. The number of particles was set to 200 and the standard deviations of the evolution model parameters were $\sigma_x = 5$ pixels, $\sigma_y = 5$ pixels, $\sigma_\phi = 3$ degrees, $\sigma_s = 0.1$. Figure 10 shows displacements and scale variations estimated by the tracker with respect to the ground truth values. It can be noticed that the tracker remains accurate inside $[-3\sigma, +3\sigma]$.

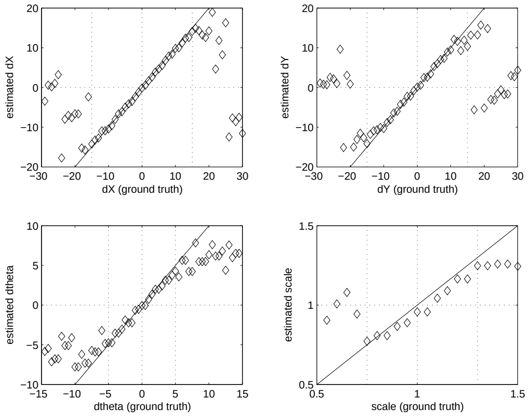

Figure 10: State vector variation estimates (dX and dY in pixels and dtheta in degrees) with respect to real variation values (The straight line represents the ideal curve).

4.2 A Navigation Task

The proposed framework is implemented on a Pekee$^{\text{TM}}$ robot which is controlled from an external PC. A small 1/3" CMOS camera is embedded on the robot and looks at the ceiling which is generally a plane parallel to the ground. Then, it is quite easy to give a good approximation of the normal vector $\mathbf{n}^*$ to the reference plane Π in order to evaluate an homography. If the camera frame is confounded with the robot frame $\mathcal{F}_c$, we can assume that $\mathbf{n}^* = [\begin{array}{ccc} 0 & 0 & 1 \end{array}]$. The displacement between $\mathcal{F}_{i+1}$ and $\mathcal{F}_c$ only consists of one rotation $\gamma \mathbf{Z}_c$ and two translations $t_{Xc}\mathbf{X}_c$ and $t_{Yc}\mathbf{Y}_c$. Therefore, $\mathbf{H}$ has only three degrees of freedom. Only two points lying on Π and matched in $\mathcal{I}_{i+1}$ and $\mathcal{I}_c$ are theoretically necessary to estimate $\mathbf{H}$. The angular deviation γ and the lateral deviation z with respect to Γ can be estimated directly from the computation of $\mathbf{H}$. The Figure 12 illustrates the evolution of planar patterns tracked during the robot motion along a given visual route. These tracked planar patterns have been extracted while the user was creating a visual path which is included into the visual route to be followed.

At the first step of an autonomous run, the current camera image has to be located into the visual memory. The tracking of learnt planar pattern in this image is then automatic. As a consequence, the user must have chosen at least one reference attitude of the robot which has to be associated with one key image. If the robot has ever achieved a mission since it has been started up, the current image is already supposed to be closed to a key image. At each frame, the tracker provides the coordinates of a current tracked planar pattern. $\mathbf{H}$ is then computed thanks to the knowledge of this pattern in the key image to reach $\mathcal{I}_{i+1}$. A key image is assumed to be reached when a distance between the current points coordinates and the desired one goes under a fixed threshold. The reference path, which is represented on the Figure 11 by the white squares which are lying on the ground, has been acquired as a visual route of fifteen key images. The corresponding path length is about $10m$. The longitudinal velocity V was fixed to $0.2m.s^{-1}$. When the robot stops at the end of the visual route, the final errors in the image corresponds to a positioning error around $5cm$ and an angular error about $3°$. Nevertheless, note that the robot has been stopped roughly, by setting V to zero since the last key image of the visual route has been detected. Moreover, both camera intrinsic and hand-eye parameters has been roughly determined. The positioning accuracy depends above all on the threshold which determines if a key image is reached. Our future works will improve that point.

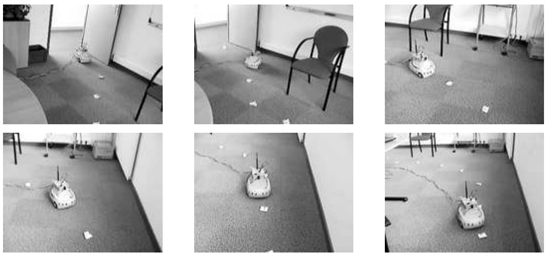

Figure 11: Following a visual route: the previously learnt visual path, about $10m$ long, is materialized on the ground. The pictures were taken during an autonomous run.

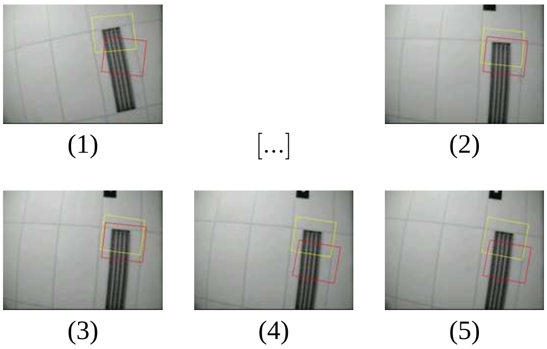

Figure 12: Evolution of the image when the robot is regulated between two consecutive key image : in each image, the yellow square is the current state of the tracker, the red one is the state to reach. The image (3) is considered close enough to the key image: the control has succeeded. At image (4), a new reference state to reach is then provided to the control law.

5 CONCLUSION

An efficient particle filter tracker with specific observation probability densities was designed to track planar modelled patterns on cluttered background. Scale changing and partial occlusions were taken into account. The obtained algorithm runs in near real time rate and is accurate and robust in realistic conditions according to experimental validation. The tracker was used for autonomous indoor environment mapping and image-based navigation. Visual route is performed thanks to a visual-servoing control law, which is adapted to the robot nonholonomy. Experimental results confirm the validity of the approach. Future works will deal with the adaptation of the tracker to 3D movement tracking using homographies in order to enable autonomous mapping of 3D indoor environment using planar patterns detected on the walls.

REFERENCES

Arulampalam, S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50:174–188.

Bascle, B., Bouthemy, P., Deriche, R., and Meyer, F. (1994). Tracking complex primitives in an image sequence. In

In Proc. of the 12th int. Conf. on Pattern Recognition, pages 426–431.

Black, M. and Jepson, A. (1996). Eigentracking: robust matching and tracking of articulated objects using a view-based representation. In *In Proc. European Conf. on Computer Vision*, pages 329–42.

Blake, A., Curwen, R., and Zisserman, A. A. (1993). A framework for spatiotemporal control in the tracking of visual contours. *Int. J. Computer Vision*, 11:127–145.

Chen, J., Dixon, W. E., Dawson, D. M., and McIntire, M. (2003). Homography-based visual servo tracking control of a wheeled mobile robot. In *Proceeding of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, pages 1814–1819, Las Vegas, Nevada.

Fang, Y., Dawson, D., Dixon, W., and de Queiroz, M. (2002). Homography-based visual servoing of wheeled mobile robots. In *Conference on Decision and Control*, pages 2866–2871, Las Vegas, NV.

Faugeras, O. and Lustman, F. (1988). Motion and structure from motion in a piecewise planar environment. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508.

Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

Hayet, J. (2003). *Contribution à la navigation d'un robot mobile sur amers visuels texturés dans un environnement structuré*. PhD thesis, Universit Paul Sabatier, Toulouse.

Isard, M. and Blake, A. (1998). Condensation-conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29:5–28.

Jurie, F. and Dhome, M. (2001). Real-time template matching: an efficient approach. In *In the 12th Int. Conf. on Computer Vision*, Vancouver.

Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: active contours. *Int. J. Computer Vision*, 1:321–331.

Lowe, D. (1992). Robust model-based motion tracking through the integration of search and estimation. *Int. J. Computer Vision*, 8:113–122.

Ma, Y., Kosecka, J., and Sastry, S. S. (1999). Vision guided navigation for a nonholonomic mobile robot. *IEEE Transactions on Robotics and Automation*, pages 521–37.

Malis, E., Chaumette, F., and Boudet, S. (1999). 2 1/2 d visual servoing. *IEEE Transactions on Robotics and Automation*, 15(2):238–250.

Marchand, E., Bouthemy, P., Chaumette, F., and Moreau, V. (1999). Robust real-time visual tracking using a 2d-3d model-based approach. In *In Proc. IEEE Int. Conf. on Computer Vision, ICCV'99*, pages 262–268, Kerkira, Greece.

Pece, A. E. C. and Worrall, A. D. (2002). Tracking with the em contour algorithm. In *In Proc. of the European Conf. on Computer Vision*, pages 3–17, Copenhagen, Danemark.

Samson, C. (1995). Control of chained systems. application to path following and time-varying stabilization of mobile robots. *IEEE Transactions on Automatic Control*, 40(1):64–77.

Se, S., Lowe, D., and Little, J. (2001). Vision-based mobile robot localization and mapping using scale invariant features. In *Int. Conference on Robotics and Automation (ICRA'01)*.

Talluri, R. and Aggarwal, J. (1996). Mobile robot self-location using model-image feature correspondence. *IEEE Trans. on Robotics and Automation*, 12(1):63–77.

Thuilot, B., Cariou, C., Martinet, P., and Berducat, M. (2002). Automatic guidance of a farm tractor relying on a single cp-dgps. *Autonomous Robots*, 13:53–71.

Tsakiris, D., Rives, P., and Samson, C. (1998). Extending visual servoing techniques to nonholonomic mobile robots. In D. Kriegman, G. H. and Morse, A., editors, *The Confluence of Vision and Control*, volume 237 of *LNCIS*, pages 106–117. Springer Verlag.

Zhong, Y., Jain, A. K., and Dubuisson, M. P. (2000). Object tracking using deformable templates. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:544–549.