15th Triennial World Congress, Barcelona, Spain

A REAL-TIME ARCHITECTURE FOR LOW-COST VISION BASED ROBOTS NAVIGATION

C. Bellini * S. Panzieri * F. Pascucci *

* *Dipartimento di Informatica e Automazione*
Università di Roma Tre
Via della Vasca Navale 79, 00146 Roma, Italy
{bellini, panzieri, pascucci}@dia.uniroma3.it

Abstract: Artificial vision can be used in many environments such as indoor, outdoor, space, and even in underwater contexts. Most of times, vision based localization requires complex algorithms and hardware resources when related to general environment features. However, the use of simple landmarks can reduce dramatically the cost and the complexity of the recognition system. In typical indoor environments, in particular in offices, ceiling lamps are all of the same type and are placed in a quite regular way. Moreover, they can be easily seen, as generally no obstacles can exist between them and the robot vision system. These peculiarities motivated a study on the possibility of implementing a very low cost localization procedure using a standard onboard webcam. A Kalman filtering approach has been used to fuse vision and odometric data for position estimation, and a navigation architecture, based on a real-time Linux kernel, has been set up to show its reliability and flexibility.

Keywords: Localization, Vision, Mobile Robots.

1. INTRODUCTION

Vision can be added to a localization system for mobile robots in order to reduce odometric errors due to the incremental nature of those sensors. The use of natural or artificial landmarks allows both simple resetting of odometric errors, so that sensors can restart each time with the correct absolute position information (Adam et al., 1999), and more complex sensor fusion algorithms that integrates all available information in a single estimation process (Panzieri et al., 2000).

Among the others, natural landmarks have the interesting features that avoid any change in the work environment. In indoor environments one can find many different reference points and marks that can be interpreted as natural landmarks, e.g. doors, corners, geometry of the floor. In particular in (Martinez et al., 1992) and (Adam et al., 1999) some of the ceiling characterizing elements are suggested as natural landmarks in order to avoid any occlusion problems.

Unfortunately, visual based control schema requires, generally, very complex algorithm and dedicated hardware (Amat et al., 2000). In this paper we have explored the possibility to realize a localization system for a mobile robot using very low cost standard hardware, i.e. a PC with a 60$ webcam.

To this end we have mounted the webcam on the mobile robot focused to the ceiling and used the lamps as reference points. Moreover, we have used a suitable topological representation of the environments, i.e., we represent the environment by means of a graph where each node is a location of interest and the arcs capture the connectivity of the space.

The proposed localization system has been experimentally tested on a mobile robot going through the passageway of the faculty and using ceiling lamps as reference landmarks.

From the software implementation point of view, our interest has been concentrated on GNU/Linux operating system that, compared with Microsoft systems, has the great advantage of being completely OpenSource and based on the Unix system. As all Unix systems, Linux scheduler is preemptive only at user-level, that means an high priority user process can suspend, when ready to be executed, a lower priority user process but not a Kernel process. Most recent versions of Linux kernel, called Real-Time Linux, introduce the possibility to define high priority processes in kernel mode that cannot be interrupted from kernel routines.

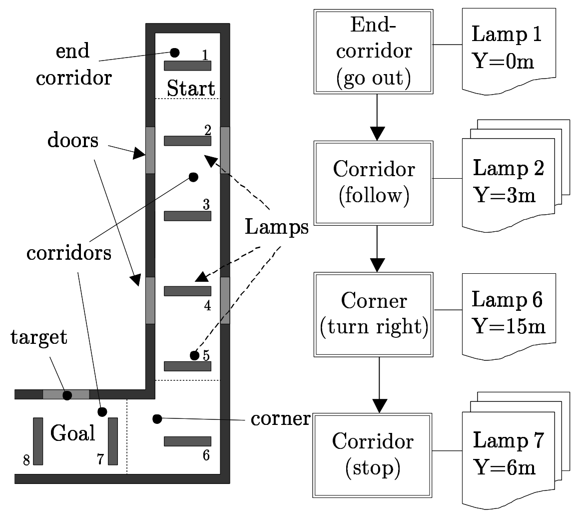

Fig. 1. Topological map

2. AN AUGMENTED TOPOLOGICAL MAP APPROACH

It is well known that an effective environment representation for a mobile robot must describe all the essential features necessary for self-localization, motion planning and navigation. Moreover, the robot should be able to extract the features directly from sensory data.

The mapping approaches, i.e., the way the world is represented, that have been proposed in literature can be grouped in two main classes (Borenstein et al., 1996): *metric maps* and *topological maps*. In the metric maps the environment is represented in terms of geometric relations between the objects and a fixed reference frame. On the other hand, in a topological map, only adjacency relations between objects are represented (Dudeck et al., 1991), avoiding metric information as far as possible.

Metric maps and topological maps are two different representations of the same environment: as a consequence, they exhibit complementary rather than opposite properties. To exploit the best of both approaches the authors in (Fabrizi et al., 2000) have suggested to put additional metric information in nodes as tags of particular interest related to the natural landmarks included in the node itself. This approach has been successfully applied for the representation of office-like environments: this kind of indoor environment is usually structured with standard elements like corridors, T-junctions, corners, and end-corridor, and very often a navigation task can be expressed as a sequence of places defining a path inside the environment such as “follow the corridor and turn right at the first corner”. So one has a graph representation as high-level view of the environment (useful for the integration of the system in an artificial intelligence framework), and, at the same time, metric information about length of the corridor, number of left and right doors, number of lamps in the ceiling and possibly their relative distance.

In our case, the extraction of some of the features mentioned before from a webcam image may be a very time-consuming task. The computational burden can be dramatically reduced if the extraction procedure is focused on particular landmarks as the lamps in the ceiling which can be easily extracted (Martinez et al., 1992; Adam et al., 1999). Indeed, they are well visible, easy to identify inside the image, generally all of the same type and placed in a quite regular pattern.

Limiting vision system to identify only the lamps in the ceiling imposes to the graph the inclusion in the *topological nodes* of a labeled sequence of landmark tags associated to the lamps. Nodes are used to decide the correct navigational behavior (follow corridor, turn right at corner) and eventually to perform an approximate localization; tags are employed to refine the localization process.

An example of the adopted representation is shown in Fig. 1 where the planned path from *Start* to *Goal* assumes that the robot *goes out* of the end-corridor, *follows* a corridor, *turns right* at the corner and find the *Goal* in the next corridor. Once finished this high-level motion, the target location (a door) can be reached with a fine motion.

On the way, the robot can always refine its localization using landmarks (lamps) that are associated to each topological node but only a low precision is required. Note that its odometry should be able to correctly label each lamp and to handle abnormal situation (e.g., when a lamp is out of order). Near the target the estimated position must be refined to correctly approach the target.

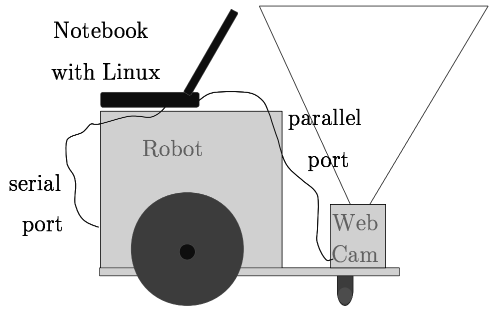

Fig. 2. Robotic system

It is important to stress that during the motion, the localization system use odometric information for a rough prediction of the landmark positions.

3. VISION BASED LOCALIZATION

3.1 *Image acquisition*

Accurate calibration of cameras is a crucial step for applications that involve quantitative measurements, such as the geometrical and the dimensional ones (Weng et al., 1992). Webcam lens aberrations must be evaluated and corrected. Using a low cost, wide angle camera this procedure become even more important. In this work we only consider camera distortion, which is related to the position of image points in the image plane, but not directly to the image quality, because we only use geometrical information in our localization system (see below). Moreover, we consider as accurately known the position of the camera in the robot frame. Calibration matrices are calculated and used as in (Panzieri *et.al*, 2001) where more details can be found.

3.2 *Lamps recognition*

After the calibration step the algorithm proceeds analyzing the image. Remember that typical image processing algorithms use complex strategies to reduce the computational burden and good public domain implementation are quite common on the world-wide web.

First, a graph search of all connected components is performed comparing, for each pixel, the luminance information with a threshold: if the pixel luminance exceeds the threshold then is considered, otherwise is neglected.

After this scanning, more than one connected component are usually retrieved. Sometimes reflections can produce small connected components that can be discriminated evaluating their area. Our lamps produce an image that is around 6000 pixels; we suppose to have lamps only if the area is greater then 3500 pixels. From the area we also know how much of the lamp is in the viewing

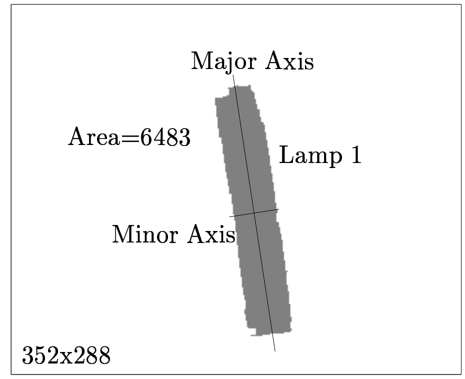

Fig. 3. Major and minor axes, center of mass and labeling

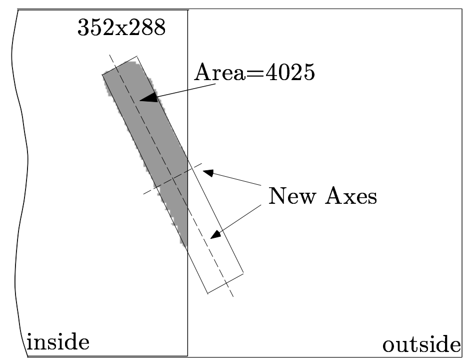

Fig. 4. Completing partial view

window and how much is left out. After that, to better the quality of the image the morphological operators of erosion and dilation are applied to reduce fringes caused by small reflection along borders of the lamp. This effect is more evident when lamps are far from the center of the image and the light reaches the camera after reflecting on the common metallic grids that are always set over the luminescent elements.

At this point we deduce the robot position and orientation using the lamps configuration in the image plane. To do this we compute the center of gravity (CoG) and the orientation of each lamp (see Fig. 3).

With this procedure, the system is able to recognize when a new lamp appears in the image and when the lamp disappear from it. Using the odometry (on a short range) a new incoming lamp is labeled and its CoG and orientation is calculated. To correctly use these measures the system needs a good estimate of those quantities and this is not possible when only a partial view is available.

Due to the limited view of the camera some problems may occur when only a partial view of a lamp is available. This forced us to develop a recovery algorithm able to complete the missing area and correctly estimate CoG and orientation. In Fig. 4 the result of this procedure, based on the knowledge of the reconstructing shape, is shown. The procedure is performed only when the visible area is greater then 4000 pixels. The algorithm uses edges found in the area and tries to interpolate them with a least square method. The reliability of such process is not extremely high but the information retrieved is a valued one in any case.

4. KALMAN BASED DATA FUSION

At the end of the image processing phase, estimated position and orientation values are available. This process can be repeated with a rate that in general depends on the frames acquisition hardware and on the complexity of algorithms performed. With a low cost hardware this is limited to few images per second and in the experiments that we present this rate is only 10 frames per second. On the contrary the algorithm that controls the motion of the robot is performed with a higher rate and needs an estimate of the full state (position, orientation, and their derivatives) at each sampling time; even if no lamps are in the view of the webcam. Then, the estimate must be obtained merging odometry, always available, with vision data. This can be done by means of an Extended Kalman Filter. The only drawback is that a delay exists between the frame acquisition instant and the time in which geometrical data becomes available to the filter. The solution adopted, like in (Panzieri et al., 2000), was to preserve in a buffer some past state values and odometric data and, each time a new frame (relative to time t_f) is processed, go back with the filtering algorithm to $t = t_f$, include the geometrical data in its evaluation, and then propagate to the actual instant the state estimate using the odometric data previously stored.

4.1 *State prediction*

In designing the Kalman Filter, the following points must be taken into account:

- the measures provided by the webcam are noisy, but do not grow with the traveled path;
- the vision system accuracy is correlated to the robot velocity;
- a delay of about 0.1 sec arises for computations and communications.

The prediction equation are simply obtained by the unicycle kinematic model. Define the vector state as the robot configuration, $X_k = (x_k, y_k, \phi_k)$ and the input $U_k = (\Delta S_k, \Delta R_k)$ as the vehicle displacement and rotation along the trajectory in the k-sampling interval. The inertial prediction is then:

$$\hat{X}_{k+1/k} = f(\hat{X}_k, U_k) =$$

$$= \begin{bmatrix} \hat{x}_k + \Delta S_k \cos\left(\phi_k + \dfrac{\Delta R_k}{2}\right) \\ \hat{y}_k + \Delta S_k \sin\left(\phi_k + \dfrac{\Delta R_k}{2}\right) \\ \hat{\phi}_k + \Delta R_k \end{bmatrix} \qquad (1)$$

The model inputs are computed using encoder data as follows. The robot displacement is measured from right end left encoder readings that give respctively the values ΔS_k^r and ΔS_k^l as

$$\Delta S_k = \frac{\Delta S_k^r + \Delta S_k^l}{2}. \qquad (2)$$

The platform rotation is estimated as

$$\Delta S_k = \frac{\Delta S_k^r - \Delta S_k^l}{2a}. \qquad (3)$$

where a is the semiaxis length.

The covariance matrix associated with the prediction error is written as

$$P_{k+1|k} = J_f^X\left(X_{k+1|k}, U_k\right) \cdot P_k \cdot J_f^X\left(X_{k+1|k}, U_k\right)^T + Q(U_k) \qquad (4)$$

where $J_{fk}^U(\cdot)$ is the Jacobian matrix of $f(\cdot)$ with the respect to $\hat{X}_k$, P_k is the covariance matrix at time istant t_k and $Q(U_k)$ is given by

$$Q(U_k) = J_{f_k}^U(U_k) \cdot C \cdot J_{f_k}^U(U_k)^T \qquad (5)$$

The (5) show that $Q(U_k)$ is the projection of the noise affecting the inputs on the state.

4.2 *Observation prediction*

The observation prediction is given by the vector

$$\hat{Z}_{k+1} = \hat{X}_{k+1/k} \qquad (6)$$

The innovation term and the associated covariance, being $\hat{Z}_{k+1}$ the measured output from the webcam, are computed as

$$V_{k+1} = Z_{k+1} - \hat{Z}_{k+1} \qquad (7)$$

$$S_{k+1} = J_h^X(\hat{X}_{k+1/k})\, P_{k+1/k}\, \left(J_h^X(\hat{X}_{k+1/k})\right)^T + R_{k+1} \qquad (8)$$

where $J_h^{k+1}(\cdot)$ is the Jacobian matrix of $h(\cdot)$ respect to $\hat{X}_{k+1/k}$. The first term used to compute S_{k+1} represents the uncertainty on the observation due to the uncertainty on the inertial prediction. The second term is the observation noise covariance matrix.

Actually the webcam neasures are available at a slower rate and some delay. This problem has been worked around as follows: when the webcam

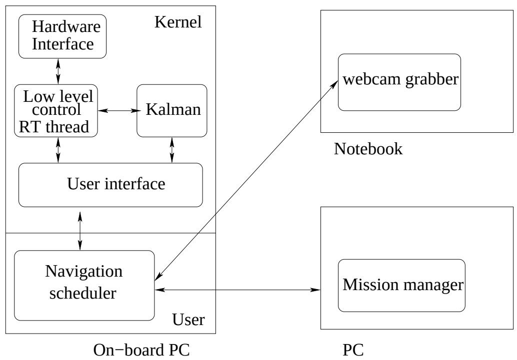

Fig. 5. Navigation architecture

information are not available, the Kalman filter is not executed and the position estimation is given by the odometric prediction. As the estimations proceed, all the input and state values are stored together with a time stamp. When the webcam measures became available the nearest time stamp is searched for and the above equations are applied. In this way the past state is corrected. From that instant on, the EKF is executed again using the stored measures and the new state estimations.

4.3 *Extended Kalman Filter*

The Extended Kalman Filter is used to correct the inertial configuration estimate on the basis of the validated observations. Particularly, the final configuration estimate is obtained as

$$\hat{X}_{k+1} = \hat{X}_{k+1/k} + K_{k+1}[Z_{k+1} - \hat{Z}_{k+1}] \quad (9)$$

where K_{k+1} is the Kalman gain matrix

$$K_{k+1} = P_{k+1/k}(J_h^X(\hat{X}_{k+1/k}))^T S_{k+1}^{-1}. \quad (10)$$

The covariance associated with the final configuration estimate $\hat{X}_{k+1}$ is given by

$$P_{k+1} = P_{k+1/k} - K_{k+1}\, S_{k+1}\, K_{k+1}^T. \quad (11)$$

5. NAVIGATION ARCHITECTURE

The architecture that realize the navigation system (see Fig.5) can be divided into several modules:

Mission manager uses the topological map to plan the high level motion, as we explained in section 2;

Navigation scheduler manages the comunications between the robot, the vision system, and the mission manager. It maps the mission into low level tasks, that the robot can execute;

Webcam grabber implements the lamps recognition algorithm;

User interface handles the messages between the user and the kernel space, through a character device;

Kalman module realizes the EKF: when the grabber start the image acquisition, it receives a command and starting to store odmetric data. At the end of the grabbing procedure it receives the image data and returns the new position estimation;

Low level controller is the low level motion control system. It is a real time thread, executed periodically, every 5 ms. It uses processor for about 60 μs, so it leave the CPU free for non real time application for a significant slot of time;

Hadware interface is driver of the motion control board.

As the figure 5 shown, some modules run on the onboard PC of the robot, while other are executed on different computers. The connection between navigation scheduler, webcam grabber and mission manager are realized using TCP/IP socket. In this way the three modules can be distributed on a local network.

6. EXPERIMENTAL RESULTS

The proposed algorithm has been tested using the mobile robot super M.A.R.I.O., a Philips Vesta Pro Scan and a Notebook (AMD Athlon 1GHz). The mobile platform is a unicycle robot built in our Department having a front castor and two fixed wheels on the same axle actuated by two independent motors. The robot sensory system is composed by two incremental encoders mounted on the two motors and connected to the on board robot mini-computer where runs the low-level control algorithm and the navigation scheduler. The webcam grabber and the mission planner run on the Notebook, connected to the on-board PC through a ethernet link. The webcam is mounted on the robot focused to the ceiling (see Fig. 2). The distance between the vision system and the landmarks (rectangular lamps) is about 2.50m, so each pixel is about 5mm at the 356×288 resolution and about 10mm at the 176×144 resolution.

In Fig. 6 a path traveled by the mobile platform is shown. Four lamps are met before turning left into a door. In this experiment the feedback has been closed around the odometry and the initial position has been set. A left-turn has been programmed in front of the door and the path labeled as *odometry* is the precise odometric result of the control algorithm executed. The path labeled as *Kalman* is the trajectory computed by the Kalman Filter and the one that has been actually followed by the robot. About 300 frames have been captured (one for each second), and 50% of them have given no information because black.

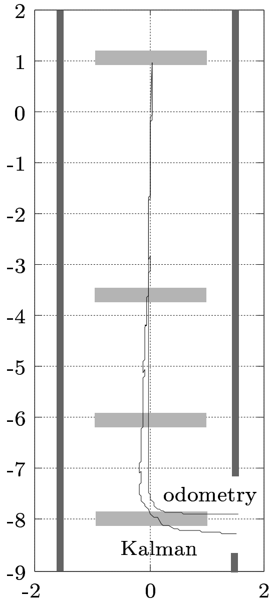

Fig. 6. Odometric and Filtered position

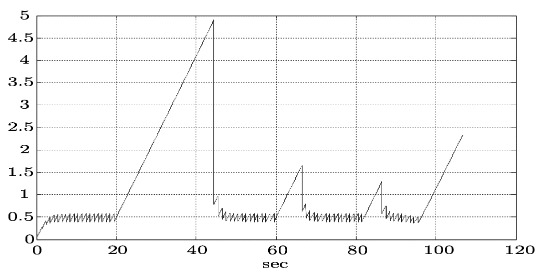

Fig. 7. Norm of covariance along path

In Fig. 7 the norm of the state covariance has been reported showing how, during the odometric navigation (no lamps in view), it slowly increases meaning that uncertainty on robot position and orientation becomes greater. Once a lamp is again in the field of the webcam, uncertainty is brought back to a value that depends on the visual sensor precision.

7. CONCLUSIONS

In this paper we have presented works on the use of artificial vision in a navigation architecure for a mobile robot using real-time architecture for low level control. The proposed approach, well suited for office-like environments, uses an inexpensive web-cam with standard hardware. This imposes that the extraction feature algorithm must be very light. The localization of the robot inside the environments is performed on the base of an augmented topological map.

Clearly, in order to correctly execute the motion plan, the robot must self-localize on the map, that is, the robot has to recognize in which node it is. For that which concerns the work presented in this paper we have assumed that the robot starts from an home position. Future works will be devoted to use vision system to localize autonomously the robot on the base of a set of images.

8. REFERENCES

A. Adam, E. Rivilin, and H. Rotstein (1999). Fusion of fixation and odometry for vehicle navigation. *IEEE International Conference on Robotic and Automation*, Los Alamitos, USA, pp. 1638-1643.

S. Panzieri , S. Renzi, G. Ulivi (2000). A stereo vision based docking procedure for a mobile robot. *Proc. 6th Int. IFAC Symp. On Robot Control SYROCO 2000*, Vienna, Austria, pp. 403-408.

A.B. Martinez, J. Climent, J.M. Asensio, and J. Batle (1992). Absolute positioning for indoor mobile robots guidance. *International Symposium on Industrial Robots*, Barcellona, pp. 529-532.

J. Amat, J. Fernandez, A. Casals (2000). Vision Based Navigation System for Autonomos Vehicles. *Intelligent Automation Systems*, ISO Press.

J. Weng, P. Cohen, M. Herniou (1992). Camera Calibration with Distorsion Models and Accuracy Evaluation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.14, n.10.

J. Borenstein, H.R. Everett, and L. Feng (1996). *Navigating mobile robot: sensors and techniques,* Peters A K, Ltd., Wellesley, MA.

G. Dudeck, M. Jenkin, E. Milios, and D. Wilkes (1991). Robotic exploration as graph construction. *IEEE Trans. on Robotics and Automation*, vol.7, n.6, pp. 859–865.

R. Schaphorst (1999). *VideoConferencing and Videotelephony: Technology and Standards*, Artech House.

B. Kuipers, Y.T. Byun (1991). A robot exploration and mapping strategy based on semantic hierarcchy of spatial representation. *Journal of robotics and autonomous sistems*, vol. 8, pp. 47–63.

S. Thrun (1999). Learning Metric-Topological Maps for Indoor Mobile Robotn Navigation. *Artificial Intelligence* vol. 1, pp. 21–71.

E. Fabrizi, S. Panzieri, and G. Ulivi (2000). Extracting topological features of indoor environment from sonar-based fuzzy maps. *Intelligent Autonomous Systems 6*, E. Pagello, F. Groen, T. Arai, R. Dillmann, A. Stentz (eds), IOS Press, Amsterdam, pp. 596–603.

S. Panzieri, F. Pascucci, R. Setola, G. Ulivi (2001). A Low Cost Vision Based Localization System for Mobile Robots. *9th Mediterranean Conf. on Control and Automation*, Dubrovnik, Croatia.