

Real Time Self Localization Using a Single Frontal Camera

Francesco Marando¹, Maurizio Piaggio² and Alessandro Scalzo³

D.I.S.T., University of Genoa,
Via Opera Pia 13, I-16145 Genova, Italy

Abstract. In this paper we present a novel feature based localization method which relies on an efficient feature extraction technique exploiting a single frontal camera, combined with Kalman filtering. The approach has been tested in the RoboCup scenario in which robot positioning is normally a requirement for successful coordination and overall team behaviour.

1 Introduction

The problem of knowing the position in which a robot is situated in the environment is a central issue in intelligent robotics and has been deeply investigated in literature. In general the existing systems assume that the robot has some sort of approximate information about its position in the world: the robot incrementally estimates its own position by means of *odometry* or *dead-reckoning techniques*. Self-localisation techniques are intended as a way to periodically reduce the errors that accumulate during the robot's motion by comparing the acquired sensorial data with a pre-stored model, a map of the environment

Map-based positioning has been long investigated since the first usage of proximity sensors in mobile robotics. Traditionally, the matching problem has been faced in two different ways, iconic and feature-based [1], which usually have a correspondence with the way the robot interprets and stores the sensorial information it acquires from the environment: occupancy grids[2][3] for the former, line segments models[4] and topological models [5][6][7] for the latter. Iconic self-localisation methods attempt to minimise the discrepancy between the raw sensed data and the pre-stored model by means of correlation [8] or Markov inspired [9] techniques. Feature-based self-localisation methods attempt to recognise significant features in the environment and then match such features against the corresponding ones in a pre-stored model of the environment [10][11].

In this paper we present a novel feature based method which relies on an efficient feature extraction technique exploiting a single frontal camera, combined with Kalman filtering. The approach has been tested in the RoboCup scenario in which robot positioning is normally a requirement for successful coordination and overall team behaviour. Moreover in this scenario the problem is particularly challenging because of the sudden and frequent crashes between the robots during the match which make dead reckoning completely unreliable.

2 System's Architecture

The software architecture is depicted in figure 1. It has been developed in ETHNOS [16], a programming environment for real time control of multi-robot applications providing effective support for inter and intra robot agent communication and multithread programming with preemptive, rate monotonic real time scheduling.

¹ kciccio@hotmail.com

² piaggio@dist.unige.it

³ scalzo@dist.unige.it

An ETHNOS application consists of a *kernel* and of several agents (called *experts*) each dedicated to a specific function. Communication between the experts is based on a *publish/subscribe* technique: an expert can subscribe to a set of message types; it will then receive messages of such types whenever another expert publishes them. Local subscription allows transparent intra-robot expert communication whereas global (club) subscription allows transparent inter-robot communication.

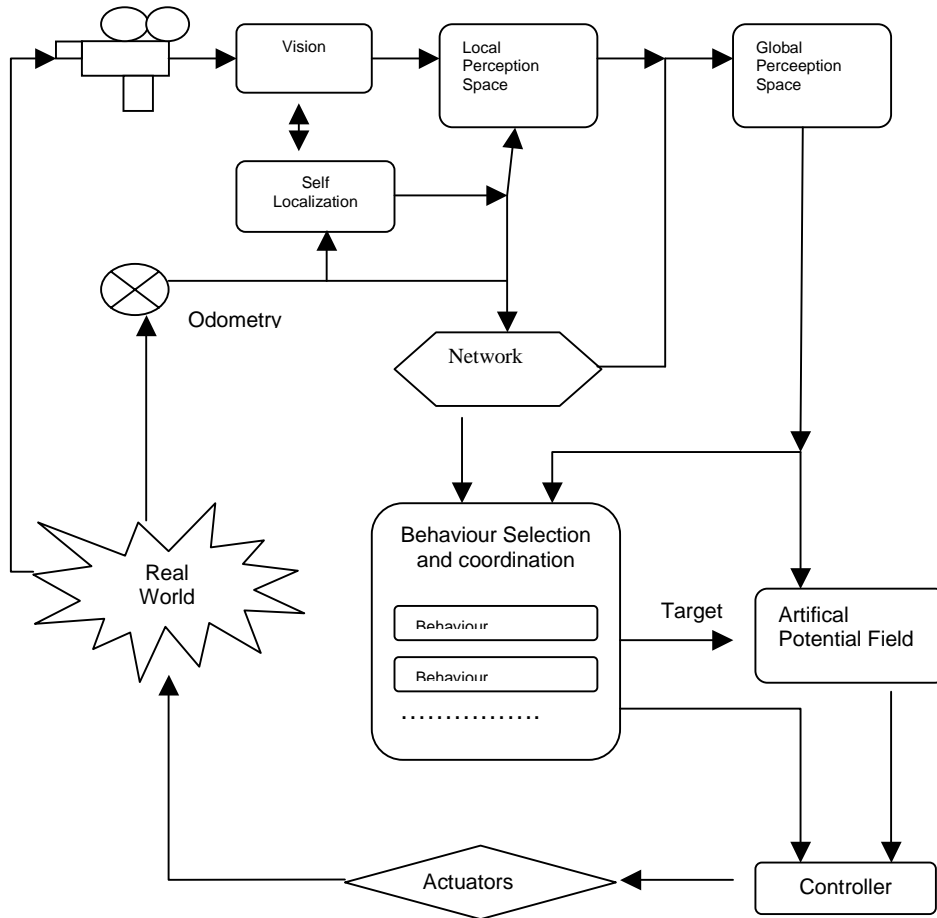


Figure 1 Software architecture

The general information flow within the architecture is the following: the Vision Expert processes the image acquired from the camera in order to extract the symbolic information about the environment (ball position, obstacle position, goals, field border lines, etc.). This information is used to update the LPS - Local Perception Space as well as for the self-localization algorithm. The information in the LPS is transmitted to the other robots and integrated with the information received in the GPS – Global Perception Space. Local and global information are used for coordination and behaviour selection as well as for artificial potential field based control.

The next section deals in more detail with field line extraction and the self-localization expert on which this paper focuses. For development and test purposes we also designed and exploited a dedicated monitoring application which can integrate and visualize the robot transmitted information. In this way it is possible to visualize in real time the color segmented view of the camera (compressed with an ad-hoc RLE technique), the extracted field lines and the robot position. A typical snapshot is depicted in figure 2.



Figure 2 Snapshot of the monitoring application

3 Self Localization

The self localization algorithm can be divided in two main steps:

1. In the first step the visible field border lines are extracted. The algorithm exploits the characteristics of the RoboCup field in which the different structures are assigned different colors. The field is composed by a green rectangular playing ground limited by white 50 cm walls and in which the two goals have different colour.
2. The extracted field lines are compared with the complete field model and the position is updated by exploiting a Kalman filter.

It is worth mentioning that, although the two steps have been optimized for the RoboCup scenario, the method is not limited to this application but can be exploited in all environments in which similar structure considerations can be made. In fact, in many buildings the corridors, doors and floors are very structured and easily comparable.

We would also like to remark that we specifically decided to rely only on the field border lines for positioning. In fact methods based on triangulation using the goalposts references were excluded because seldom applicable during a match and subject to significant positioning errors. Analogously methods relying on the identification of complex landmarks such as the middle-field circle or similar features were excluded because computationally too expensive and seldom reliable. Our approach has intentionally a low computational impact so that it can be executed at the same rate of the images acquisition (10 Hz), compensating, in this way, for the low informative contents of each image with the amount of information per second.

3.1 Landmark Detection

Landmark detection is carried out on a image consisting of a 320 x 240 pixels bitmap, analyzing only a line out of two in order to speed up computation. The camera has been turned of 90° in order to achieve the best vertical resolution and favor the precision in the computation of the distance of the relevant objects. Moreover scanning columns instead of rows further reduces the processing time as we will see shortly.

The column image analysis is intended to find the color transitions from white to green in order to detect field border lines. The algorithm is executed in synergy with image segmentation and clustering in order to carry out all necessary operations in one step and exploiting common intermediate results. Because of the acquisition noise and scene lighting differences color transition is detected on the basis of a statistical measure after a local lowpass filter. White and green pixels are counted in a vertical position mask that is translated over the columns. By applying suitable thresholds it is possible to establish the presence of a transition. The column scan is interrupted either when the transition is found or when the number of green pixels increases above a certain threshold identifying the field playground. This has a double advantage: it further reduces the computation cost because only a limited portion of the image is scanned and it limits the detected lines to field border lines (in fact the border lines are the more external lines, situated, because of the prospectical projection, in the highest part of the image compared with the more internal field zones). Moreover the filter mask solution also permits to effectively solve the problem of the black writings on the white walls that do not disturb processing.

The result of this first step is a series of points, ordered from right to left, belonging to the field border. The relative world coordinates of such point positions is determined throughout a previously calibrated look up table performing the perspective inversion after the correction of the distortion introduced by the camera lens. Notice that distorsion is a crucial issue because, if the were not compensated, it would introduce an error on the identification of the border lines that could be erroneously interpreted as curves.

Extracted points are clustered and interpolated through a linear regression algorithm in order to minimize the average square error in the different clusters. The algorithm is again designed for computational efficiency. The border straight lines are expressed in the image plane by the form: $x = a \cdot y + b$, where x represents the coordinate with respect to the vertical axis of the image, that is the one associated to the prospectical depth, while y represents the horizontal axis. Coefficients a e b are calculated as follows:

$$\begin{aligned} \text{sum_x} &= \sum X_i \\ \text{sum_y} &= \sum Y_i \\ \text{sum_xy} &= \sum X_i \cdot Y_i \\ \text{sum_y2} &= \sum Y_i \cdot Y_i \\ \Delta &= N \cdot \text{sum_y2} - \text{sum_y} \cdot \text{sum_y}; \\ a &= (N \cdot \text{sum_xy} - \text{sum_x} \cdot \text{sum_y}) / \Delta; \\ b &= (-\text{sum_xy} \cdot \text{sum_y} + \text{sum_x} \cdot \text{sum_y2}) / \Delta; \end{aligned}$$

where X_i e Y_i are the coordinates in the image plane of the i^{th} border point P_i found, N the number of points considered.

Segments of straight lines are computed in an incremental way. As a first step a straight line is calculated over the beginning K points. From this straight line a statistical evaluation is carried out counting, as new points $P_i, i > K$, are gradually considered, how many out of them included

in the $P_i..P_{i-8}$ set, have distance d from the straight line greater than a threshold H . This point set will be shifted at every step. Every time the amount of points that have a distance, from the straight line, greater than a threshold H exceeds a second threshold, the previous segment is considered to be completed and a new segment is extracted.

Considering that the algorithm is performed at a high frequency, we preferred to favour the information quality rather than its quantity, thus discarding all the extracted segments considered ambiguous, incorrect or imprecise. In particular we did not consider segments with the following characteristics:

- Obtained from too few pixels
- Too much inclined in the image plane
- Such as to correspond, when projected, to a real length lower than 40 cm
- With an extreme too high in the view field, i.e too far and thus not enough precisely localized
- Belonging to straight lines intersecting with an angle different to 90° (in contrast with the field model)
- Three or more segments presents together in the scene (the case of three walls framed is actually very rare)

The remaining information turns out to be very reliable (see the experiments section) and it is used for position estimation based on Kalman filtering. It is described in detail in the following subsection.

3.2 Position computing

In order to identify the position of an orientated point in a plane it is necessary to have one reference to correct the orientation and to know the position of two more references to determine the cartesian coordinates. When one or more references are missing or their identification is ambiguous, this is the case of the field edges, it is necessary to exploit the previously known position information, introducing probabilistic criterions.

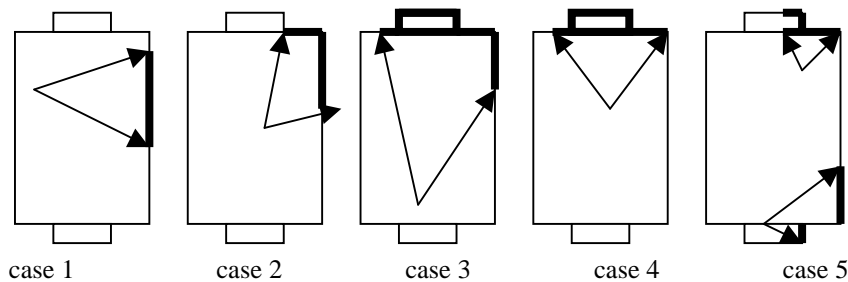


Figure 3 Possible situations in which the positioning information can be exploited

In particular let \mathbf{x} the robot position, we need finding a \mathbf{x} that maximizes $P(\mathbf{x}/r_1..r_n, x)$ where $r_1..r_n$ are the visual references and x the odometric position. A survey of the available informations (also depicted in figure 3) is:

- 1) One visible straight line
- 2) Two visible straight lines
- 3) Two visible straight lines and the goal
- 4) One visible straight line and the goal within it
- 5) One visible straight line and the goal outside it

Out of these five cases we decided to rule out the goal information corresponding to the fifth case because it could be not reliable or imprecise. In the first and second case, the identification of the straight lines is obtained by odometry and especially by odometric orientation. This choice comes out from a simple consideration: an orientation error ranging from $\pm 45^\circ$ allows the correct matching of the observed straight lines with the field edges, on the other hand, even in case of absence of localization-activity for long time, hardly a so big angular error can be accumulated. On the contrary, if we consider the cartesian coordinates, it is easy to verify that, even with an error of only 15° , covering once the field length (9 meters) a nearly two and an half meters error is accumulated, this equals half the field width, enough to make vain every attempt to carry out matching based on the wall distance.

If we define:

- x, y, θ : robot coordinates
- ψ_i : wall orientation in a relative reference system
- $\varphi_i = \psi_i - \theta$: wall orientation in an absolute reference system
- d_i : distance edge-robot
- kl_i : Kalman coefficient for the distance of the edge i
- kh_i : Kalman coefficient for the angle of the edge i

Four cases can be distinguished:

- 1) $|\varphi| > \frac{3}{4}\pi$
 $X_{err} = kl_i (-XMAX + d_i - x)$
 $\theta_{err} = kh_i (\varphi - \pi)$
- 2) $\frac{1}{4}\pi < \varphi < \frac{3}{4}\pi$
 $Y_{err} = kl_i (-YMAX + d_i - y)$
 $\theta_{err} = kh_i (\varphi - \frac{1}{2}\pi)$
- 3) $-\frac{3}{4}\pi < \varphi < -\frac{1}{4}\pi$
 $Y_{err} = kl_i (YMAX - d_i - y)$
 $\theta_{err} = kh_i (\varphi + \frac{1}{2}\pi)$
- 4) $|\varphi| < \frac{1}{4}\pi$
 $X_{err} = kl_i (XMAX - d_i - x)$
 $\theta_{err} = kh_i \varphi$

When the goal information is available the algorithm can correct the position in a safe way, because it does not exploit odometric data. The procedure distinguishes two cases depending on which goal is seen. In the case in which two straight lines have been found, the Y coordinate of the intersection point $intY$, in the image reference system, is calculated.

Let a_0 , b_0 and a_1 , b_1 the coefficients of the two lines, $\text{intY} = (b_0 - b_1)/(a_1 - a_0)$
 For instance, in the case in which the opposing goal has been seen, let goalY be the Y coordinate (in the image) of the goal. It follows that:

```

if (goalY > intY)
{
 $\theta_{\text{err}} = kh_1 (\psi_1 - \theta) + kh_0 (\psi_0 - \frac{1}{2}\pi - \theta)$ ;
 $X_{\text{err}} = kl_1 (X_{\text{MAX}} - d_1 - x)$ ;
 $Y_{\text{err}} = kl_0 (-Y_{\text{MAX}} + d_0 - y)$ ;
}
else
{
 $\theta_{\text{err}} = kh_1 (\psi_1 + \frac{1}{2}\pi - \theta) + kh_0 (\psi_0 - \theta)$ ;
 $X_{\text{err}} = kl_0 (X_{\text{MAX}} - d_0 - x)$ ;
 $Y_{\text{err}} = kl_1 (Y_{\text{MAX}} - d_1 - y)$ ;
}

```

Calculated errors are not directly used to update the position, but they are filtered by a simplified Kalman filter. Notice that it is difficult to model the odometric error since the noise changes in a discontinuous way depending on the most different factors like crashes or skids. However, since the algorithm works at high rate, we supposed an odometric error constant and equal to the initial one. The filter parameters kl_i and kh_i vary depending on the measured distance and have been experimentally tuned to obtain a good compromise between stability and quickness during error correction.

4 Experimental results

The experiments have been carried out on our robot Relè depicted in figure 4. It is based on an *Activmedia™ Pioneer1* equipped with an on board processing system composed of a common PC motherboard with an AMD K6 266 MHz processor, 64 MB SRAM and 4 GB HD. A camera has been mounted on top and it has been oriented in order to give a view ranging from few centimeters close to the robot front, up to the horizon (infinite). In this way it is possible to have both feedback for the ball control while pushing it toward the goal and, at the same time, a sufficiently extended field view.



Figure 4 Relè – the robot soccer player

Both experiments illustrate the efficiency and accuracy of the approach which compare favourably with all vision based techniques used in the same competition. Such works are discussed in the following section.

5 Related Work

This problem has been widely investigated in many different application domains leading to systems based on natural or artificial landmarks, stereoscopic vision or sonar sensors. In particular the RoboCup community has faced the problem from a perspective in which the computational efficiency and the real time applicability were fundamental issues, often more important than accuracy. The approaches in this domain can be categorized in relation to the type of sensors used: methods based on Laser Range Finders, on an omnidirectional vision sensor and on a single frontal camera. Moreover, since the sensors are in some cases used not only for positioning but for perception in general (ball detection, opponent tracking, etc.) the methods are not directly comparable.

Method based on range finders in general allow for a high accuracy. For example the CS-Freiburg and Stuttgart-Cops can determine their position on both axis with an accuracy of 1 and 5 cm, respectively. The major drawback is the significantly high and often prohibiting cost of the sensor system as well as the need of field walls in order to work correctly.

Methods based on vision are in general less expensive and, depending on the algorithms used, they do not necessarily require the presence of field walls which will presumably be removed in future editions of the competition.

Pedro Lima and Carlos Marques [12] propose an omni-directional catadioptric (vision + mirror) system to determine the robot position from the observation of natural environment landmarks such as straight lines resulting from the intersection between the walls and the ground, as well as from an *a priori* knowledge of the environment geometry. The Hough Transform is applied in order to extract the lines. Typical position errors range from 0 to 5 cm in each coordinate, but for some images the error may reach 10 cm. Similarly the orientation error is usually in the 0-3 degrees range, but in one case it reached 21 degrees. The processing time required is high and, for this reason, the on-line applicability is not clear because only off-line experiments have been presented.

R. Hanek and T. Schmitt of "Technische Universitaet Moenchen" [13] for the "Agilo Team" use a single frontal camera. Localization is cooperative with external centralized processing unit. A feature extraction based on the image processing library HALCON provides the necessary data for the onboard scene interpretation. These features as well as the odometric data are checked on the master PC with regard to consistency and plausibility. The results are distributed to all robots. Tests with motionless robot for long time (30 seconds) produced generally quite good results (errors from 1 up to 26 cm) depending on the robot location within the field. Less significant are the test performed with moving robots.

Iocchi and Nardi [14] also use a single frontal camera and match the lines with a field model using the Hough Transform. Experimental results have not been presented but minimal theoretical errors due to the adopted discretization are 10 cm and 3 degrees, the computational weight does not allow the algorithm execution at the picture acquisition rate.

References

- [1] G. Shaffer, J. Gonzalez, A. Stentz, Comparison of two range-based pose estimators for a mobile robot, *Proc. SPIE Conf. Mobile Rob.*, Boston, MA, 1992.
- [2] A.E. Elfes, A Sonar-Based Real-World mapping and Navigation, *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 3, June 1987
- [3] H.P. Moravec, Sensor fusion in Certainty Grids for Mobile Robots, *AI Magazine*, 1988.
- [4] M. Drumheller, Mobile Robot Localization Using Sonar, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 9, No. 2, March 1987.
- [5] D. Kortenkamp and T. Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing, *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, July, 1994.
- [6] G. Dudek, P. Freedman, S. Hadjres, *Mapping in Unknown Graph-Like Worlds*, *Journal of Robotic Systems* 13(8), 1996.
- [7] B. Kuipers and Y. Byun. A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations, *Journal of Robotics and Autonomous Systems*, No. 8, 1991
- [8] S. Thrun, Learning Metric-Topological Maps for Indoor Mobile Robot Navigation, *AI Journal* 99(1), 1998.
- [9] W. Burgard, A. Derr, D. Fox, and A.B. Cremers. Integrating Global Position Estimation and Position Tracking for Mobile Robots: The Dynamic Markov Localization Approach. *International Conference on Intelligent Robots and Systems*, October 1998.
- [10] P. MacKenzie and G. Dudek, Precise Positioning Using Model-Based Maps, *Proceedings IEEE International Conference on Robotics and Automation*, San Diego, CA, May 1994.
- [11] M. J. Mataric, "A Distributed Model for Mobile Robot Environment-Learning and Navigation" , *MIT EECS Master's Thesis*, Jan 1990, *MIT AI Lab Tech Report AITR-1228*, May 1990
- [12] C. Marques, P. Lima. A localization Method for a Soccer Robot Using a Vision-Based Omni-Directional Sensor. *The Fourth International Workshop on RoboCup*, 2000.
- [13] R. Hanek and T. Schmitt. Vision-Based Localization and Data Fusion in a System of Cooperating Mobile Robots. In *Proc. of the IEEE Intl. Conf. on Intelligent Robots and Systems*, pages 1199-1204. IEEE/RSJ, 2000.
- [14] L. Iocchi, D. Nardi. Self-Localization in the RoboCup Environment. In *Proc. of 3rd International Workshop on RoboCup*. Stockholm, Sweden, 1999.
- [15] C. Balkenius, L. Kopp. Elastic Template Matching as a Basis for Visual Landmark Recognition and Spatial Navigation. *Proc. AISB Workshop on "Spatial Reasoning in Mobile Robot and Animals, Manchester 1997*
- [16] Maurizio Piaggio, Antonio Sgorbissa, Renato Zaccaria, "A Programming Environment for Real Time Control of Distributed Multiple Robotic Systems ", *Advanced Robotics*, The International Journal of the Robotics Society of Japan, Vol. 14, N.1, VSP Publisher, 2000.