Bird's–Eye View Image for the Localization of a Mobile Robot by Means of Trilateration

Lorenzo Sabattini [*] **Cristian Secchi** [**] **Cesare Fantuzzi** [**]
Alessandro Stefani [**]

[*] *Department of Electronics, Computer Sciences and Systems (DEIS), University of Bologna, Italy.*
e–mail: lorenzo.sabattini2@unibo.it
[**] *Department of Sciences and Methods of Engineering (DISMI), University of Modena and Reggio Emilia, Italy.*
e–mail: {cristian.secchi, cesare.fantuzzi}@unimore.it

Abstract: In this paper we introduce a method for the localization of a low–cost mobile robot, based on the use of a monocular camera. We consider a robot moving in a bi–dimensional environment, where some landmarks are placed in known positions. The acquired image of the environment is converted into a bird's–eye view image, used to measure the distance of the robot from the landmarks, to compute the robot's position by means of trilateration. The proposed strategy is able to compute both the position and the orientation of the robot.

Keywords: Position estimation, mobile robots, robot vision, autonomous vehicles, image distortion

1. INTRODUCTION

In this paper we introduce a method for the self localization of low–cost autonomous mobile robots moving in a bi–dimensional environment. This method is based on the use of a monocular camera to measure the distances from some known landmarks. We consider a bi–dimensional environment because our aim is to develop a visual localization system for low–cost wheeled mobile robots, moving on the ground floor. Our objective is to make the robot able to estimate its current position and orientation, starting from completely arbitrary and unknown initial conditions.

Since localization is a fundamental issue in the mobile robotics field, several strategies have been developed to solve it. Our method is based on the principle of trilateration, which is widely adopted in mobile robots localization. Using trilateration, the position of the mobile robot is computed based on the measurement of the distances from three (or more) landmarks, whose positions are known in advance. One of the most used techniques based on trilateration is the Global Positioning System (GPS), that is trilateration using satellites as landmarks. As is well known, this technique is very effective for outdoor localization, but is not suitable for indoor scenarios (Jin et al., 2006).

The use of vision sensors to solve the localization problem is motivated by the the fact that cameras are cheap sensors (if compared for example with laser scanners), but the acquired information is very rich. In fact, characteristics like colors and shapes can be easily extracted by images. By the way, the acquisition of images can be affected by changing in the light sources, or by the unexpected appearance of shadows. These problems must be taken into account during the image processing.

Most of the visual localization techniques are based on the use of multiple views, obtained by means of multiple cameras, or by means of multiple acquisitions of the same scene from different points of view (see for example (Davison et al., 2007) and references therein). The main drawback of this kind of techniques is in the fact that the elaboration of multiple images is computationally heavy, making it not suitable for obtaining good real–time performances on low–cost robots.

Our main idea is to use landmarks placed on the ground floor. This choice allows us to exploit some existing landmarks that are almost always present on the floor in industrial environment, like colored stripes. Furthermore, if there are not enough features that can be used as landmarks, it's easy to add some colored landmarks on the floor, without creating obstacles for the movement of the vehicles.

As described for example in (Calabrese and Indiveri, 2005), omnidirectional cameras can be used to acquire the positions of known landmarks. Omnidirectional cameras are obtained exploiting a camera facing a curved mirror, that gives the camera a 360–degree field of view. In order to obtain a simpler and cheaper system, our interest is in localization techniques based on the use of single (monocular) cameras. In (Betke and Gurvits, 1997) a powerful strategy for landmark–based localization of mobile robots has been introduced. This strategy can compute the position of the robot measuring the bearing of some known landmarks, given that they are not in some singularity configurations. In (Leung et al., 2008) the authors describe a localization method that exploits the prior knowledge of aerial images of an area, to compute the position of the robot. More specifically, the camera acquires the images, and landmarks (in this case, landmarks are the buildings' boundaries) are recognized. The current position of the robot is computed by means of particle filters, exploited to associate the acquired landmarks to the landmarks of the aerial image.

To avoid dealing with full aerial images of the environment, our localization strategy is based only on the knowledge of the position of a certain number (at least three) of small colored landmarks, placed on the ground floor. Their positions can be completely arbitrary, but must be known in advance, and stored in a look–up table. By means of a simple calibration procedure, it is possible to obtain the matrix that defines the planar homography that allows the system to convert the acquired image into a *bird's–eye view* image of the environment, that is a top–down image, equivalent to an aerial one. Once obtained the bird's–eye view image of the environment containing the landmarks, the position of the robot can be computed by means of the trilateration technique. By means of simple geometrical considerations, the orientation of the robot can be computed as well.

The paper is organized as follows. In Section 2 the method to obtain the bird's–eye view image is described. In Section 3 the bird's–eye view image of the environment is exploited to compute the position and orientation of the robot. In Section 4 we describe the implementation of the localization strategy on real low–cost robots, and we present some experimental tests. Some concluding remarks are given in Section 5.

2. BIRD'S–EYE VIEW

To acquire an image of the environment, the robot is equipped with a monocular camera. The camera is in the front of the robot, opportunely inclined to be able to frame a significant portion of the floor. Since the camera is not on the ceiling, the floor is not acquired from a top–down perspective: thus, the ground floor appears deformed in the acquired image.

To measure the horizontal distance of the robot from some landmarks placed on the floor, we need a top–down image, which we define a *bird's–eye view* image of the environment. In fact, in a bird's–eye view image, the horizontal distance between two points can be directly measured, as in a map.

Thus, in this section we describe how to convert the robot's view of a scene into a bird's–eye view. What we need is the matrix that defines the planar homography that relates the ground plane with the camera plane. To do this, we refer to the procedure described in (Bradski and Kaehler, 2008).

Let Q be a point in the real reference frame, and q be the corresponding point in the camera reference frame. The real reference frame is three–dimensional, while the camera reference frame is bi–dimensional. More specifically, we can express Q and q as

$$Q = [X \ Y \ Z]^T \tag{1}$$

$$q = [x \ y]^T \tag{2}$$

Expressing these point into homogeneous coordinates, i.e.:

$$\begin{aligned} \tilde{Q} &= [Q^T \ \ 1]^T \\ \tilde{q} &= [q^T \ \ 1]^T \end{aligned} \tag{3}$$

the homography that relates $\tilde{Q}$ to $\tilde{q}$ can be represented by the matrix H:

$$\tilde{q} = H\tilde{Q} \tag{4}$$

This matrix H defines a rotation and a translation that, in our case, relate the ground plane to the camera plane. Thus, in order to define H, we need six parameters: three to define the rotation, and three to define the translation. Since our aim is to localize a robot moving on a plane, without loss of generality we can define this plane as the one with $Z = 0$. On this plane, each point is defined by a couple (X, Y). Acquiring three point, whose positions is known a priori, gives us six equations, that enable us to find the parameters that define the matrix H. Thus, the procedure to obtain matrix H is the following:

- placing on the ground plane (at least) three points in known positions;
- acquiring the image of the scene;
- solving the system of equations, to compute the parameters that define H.

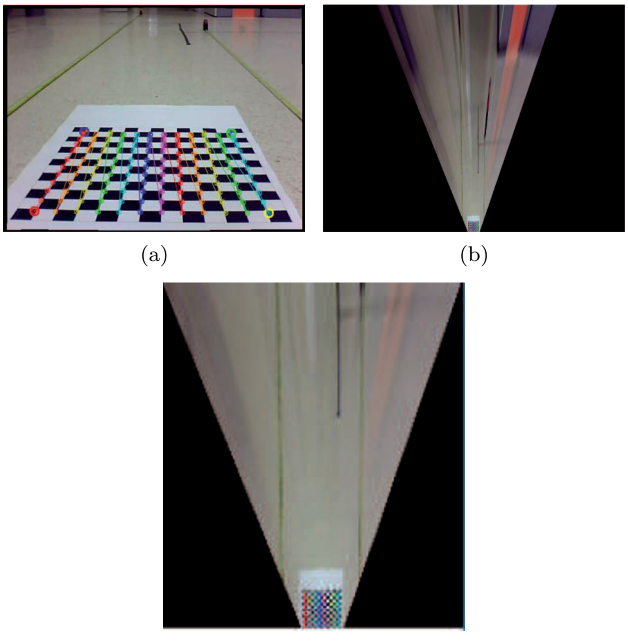
In principle, three points are enough. In practice, due to the errors in the acquisition system, it's better to acquire several points, and find out statistically the matrix H that best fits the observations. Chessboards are often used to compute these matrices: in fact, they provide several points that are easy to acquire (the corners), and they are described by a simple regular pattern, which is quite easy to deal with.

The procedure described so far to define the matrix H is quite elaborate, and requires heavy computation. Nevertheless, since the matrix H is constant once the camera is fixed on the robot, it doesn't need to be computed at runtime. More specifically, the computation of the matrix H is performed only once, during the initial calibration procedure.

In Fig. 1 an example of the computation of a bird's–eye view image is shown. Fig. 1(a) shows the image of the environment that has been acquired by the camera. Fig. 1(b) shows the corresponding bird's–eye view, computed by means of the application of the homography defined by the matrix H.

We want to remark that in the bird's–eye view image only the objects that are on the ground floor are correctly represented. An example can be seen in Fig. 1: the red object in the top right–hand corner of Fig. 1(a) is on the wall, and it is heavily deformed in Fig. 1(b). Thus, to have meaningful measurements, all the reference points must be placed on the ground floor.

Fig. 1(c) shows the bird's–eye view image of the ground floor. It can be noted that, as desired, straight and parallel lines on the ground floor are represented as straight and parallel lines on the image. This means that the real distance between two objects is proportional to the distance between the same two objects in the image.



(a) (b) (c)

Fig. 1. Camera–view image and bird's–eye view image

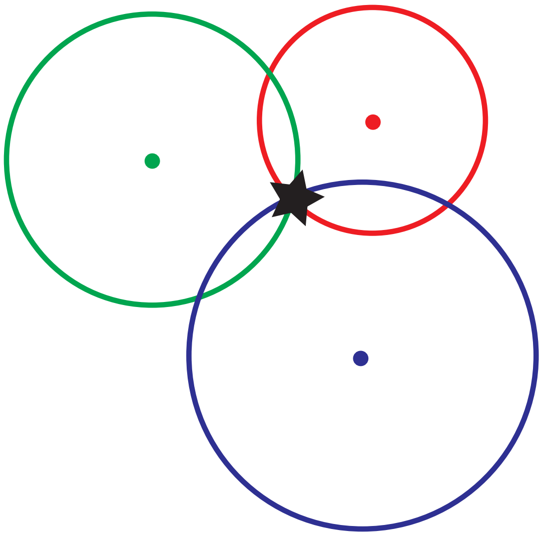


Fig. 2. Estimation of the position of the robot by means of trilateration

3. POSE ESTIMATION

In this section we will describe how to estimate the pose of the robot, by means of the measurement of its distances from some landmarks whose positions are known in advance. With the term *pose* we mean both the position and the orientation of the robot.

The position of the robot is estimated by means of the trilateration principle (Thomas and Ros, 2005), described in Fig. 2. The robot (the black star in the picture) measures its distance from three landmarks (the colored dots in the picture) whose positions are known in advance, with respect to some absolute reference frame. Let R_i be the distance of the robot from the i–th landmark, whose position is $x_i \in \mathbb{R}^2$: the robot is on a circumference with radius R_i and center x_i. Thus, acquiring the distance from three landmarks, the position of the robot is computed as the geometrical intersection of three circumferences with radius R_i and center x_i, with $i = 1, 2, 3$.

Since we assume that the camera is fixed on the robot, finding the orientation of the robot is equivalent to finding the orientation of the camera, as it is shown for example in Fig. 3, where θ is the orientation of the robot.

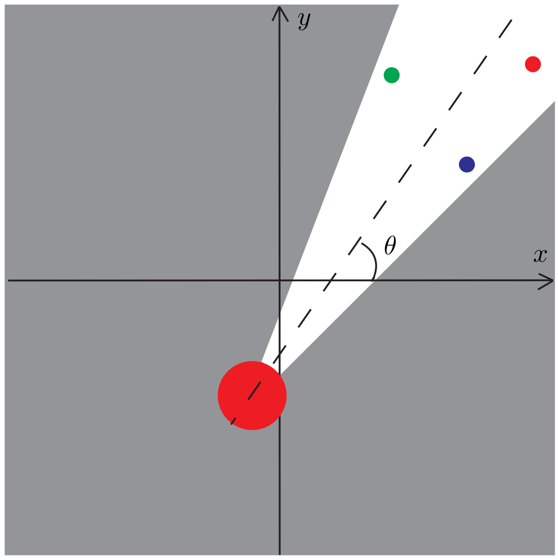


Fig. 3. Finding the orientation of the robot is equivalent to finding the orientation of the camera

We compute the orientation θ by means of the computation of the positions of two auxiliary points, namely p_1 and p_2. As shown in Fig. 4, p_1 and p_2 are the intersections

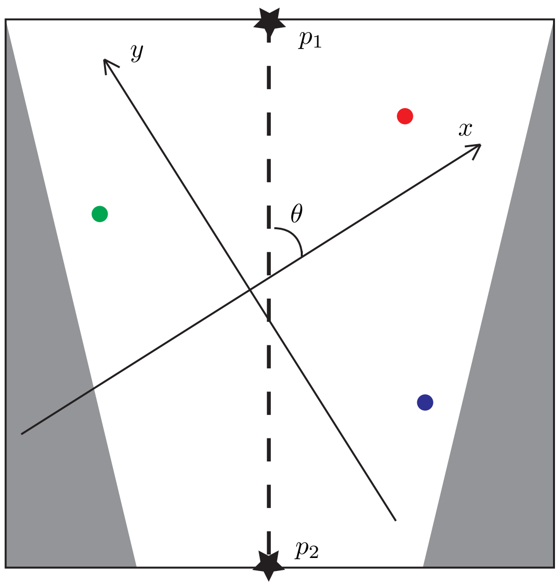


Fig. 4. Computation of the orientation

of the camera axis with the bounding of the bird's–eye view image. Once estimated the positions of p_1 and p_2 by means of the trilateration principle, the orientation θ can be computed as follows:

$$\theta = atan2\left(\left(x_{p_1} - x_{p_2}\right), \left(y_{p_1} - y_{p_2}\right)\right) \qquad (5)$$

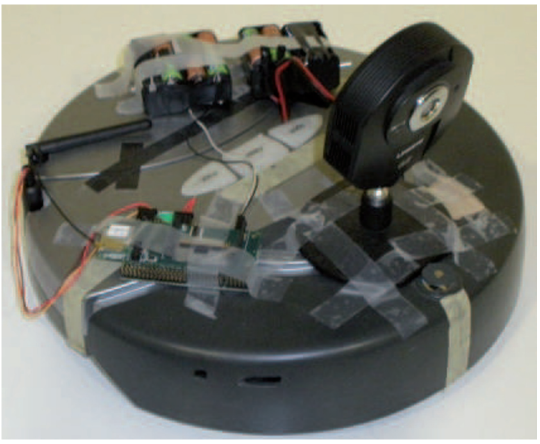


Fig. 5. iRobot Roomba mobile robot equipped with Gumstix computer and Linksys IPCamera

where $atan2$ is the two–arguments four quadrants arctangent, $p_1 = (x_{p_1}, y_{p_1})$, and $p_2 = (x_{p_2}, y_{p_2})$.

4. IMPLEMENTATION AND EXPERIMENTS

4.1 Experimental setup

To validate our localization strategy, we developed some experimental tests on a low–cost mobile robot. The total cost of our experimental setup is approximately 600 euros: thus, this is quite appropriate for educational purposes. Inspired by (Matarić et al., 2007), our mobile robot is based on an iRobot Roomba vacuum cleaner [1], controlled by means of a Gumstix Connex board [2]. Gumstix is a miniaturized Linux computer, that can be connected to the Roomba via serial port, controlling the wheels' motors and reading data from the sensors. Furthermore, it is equipped with an expansion board that provides WiFi connectivity. As shown in Fig. 5, we equipped our mobile robot with a camera. More specifically, we used a Linksys WiFi IP–Camera, generally used for video surveillance purposes.

The robot is controlled by means of the Player Robot Device Interface [3], while the acquisition and elaboration of the images is developed by means of the OpenCV libraries (Bradski and Kaehler, 2008). The Gumstix computer runs the Player server, while the control strategy and the elaboration of the images are implemented on a remote personal computer, which controls the robot exploiting a WiFi network. We want to remark that the use of a remote personal computer is motivated only by the limited computational resources of the Gumstix board.

In our implementation, the landmarks are identified by means of small–size colored dots. A look–up table is stored into the memory of the robot that allows the software to associate the color of each landmark with the position of its barycenter. To find the position of the barycenter on the bird's–eye view image, a filter is applied for each color that defines a landmark, to find the pixels in the image that match each landmark's color. Then, the barycenter is computed averaging the coordinates of these pixels.

[1] http://www.irobot.com
[2] http://www.gumstix.com
[3] http://playerstage.sourceforge.net

The position of the barycenter of the robot is defined as the position of the point p_2 in Fig. 4 plus an offset. This

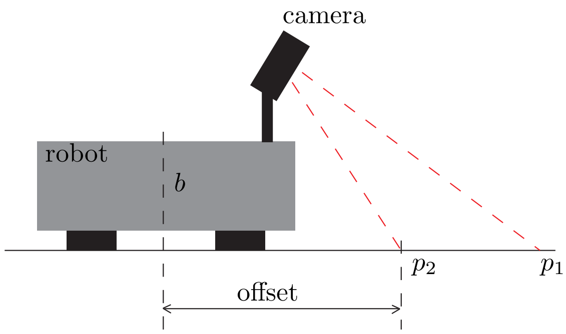


Fig. 6. The position of the barycenter b of the robot is defined as the position of point p_2 plus a constant offset

offset (Fig. 6) is constant, once the camera is fixed on the robot, and is measurable during the calibration procedure. Thus, once computed the position of the each landmark, the distances of the robot from each landmark can be easily computed. The distances are computed in pixels, but can be converted into meters by means of an appropriate scale factor. Even this scale factor is constant once the camera is fixed on the robot, and is measurable during the calibration procedure.

4.2 Measurement errors

The measurement of the distances is affected by an error. This error is caused by many factors, like imprecisions during the computation of the homography matrix H, shadows due to imperfect light conditions, or imperfection of the camera lenses.

As described before, the trilateration principle describes the position of the robot as the geometrical intersection of three circles (Fig. 2). With real measurements, the situation will be similar to the one described in Fig. 7: due to the errors in the measurements of the distances, the circumferences do not intersect in one point. As described for example in (Menegatti et al., 2006), one way to deal

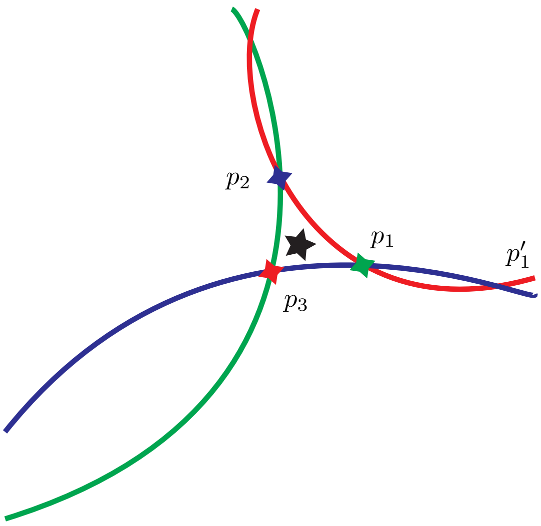


Fig. 7. Errors in the measurements of the distances

with uncertain measurement is to use optimization techniques like Monte Carlo algorithm. Another strategy to deal with these uncertainties is to compute the intersection of probabilistic (Gaussian) distribution, as described for example in (Liu and Zhou, 2007). Since these methods are computationally heavy, we describe a simplified method to find the position with uncertain measurements:

- We compute the geometrical intersection of two circumferences, namely the first one and the second one (e.g. the red one and the blue one in Fig. 7), thus finding two points (p_1 and p_1' in Fig. 7).
- We search which one of the previously defined two points best fits the third circumference (p_1 in Fig. 7).
- We iterate the procedure, exchanging the role of the circumferences. This allow us to find two other points (p_2 and p_3 in Fig. 7).
- The position of the robot is computed as the average of the positions of these three points (p_1, p_2 and p_3 in Fig. 7).

This technique is quite easy to implement, and is quite fast even on low–performance devices.

4.3 Experimental tests

The experimental test we developed aims at highlighting the improvement in the localization accuracy of a mobile robot using this visual localization technique based on trilateration on a bird's–eye view image of the environment.

More specifically, we want to compare the accuracy in the localization using pure odometry, and using the method described in this paper. Regarding the localization using the odometry, we compensated the systematic errors by means of the strategy described in (Borenstein and Feng, 1996). Nevertheless, as well known, odometry is affected by accidental errors (e.g. due to wheels slippage) that accumulate during the movement of the robot, thus making the localization error diverge.

Another problem of the localization using pure odometry is in the setting of the initial pose: in fact, the initial pose must be exactly known, and small errors in the initial pose (in particular in the initial orientation) generate big errors in the localization. Conversely, using the trilateration, the initial pose can be completely arbitrary and unknown, since the robot can compute its pose using an image of the environment.

The experimental test we have developed is quite simple (Fig. 8). The task of the robot is to reach a goal position. The robot's initial pose is completely arbitrary and unknown in advance. Thus, the robot first localizes itself looking for the landmarks (turning around itself), to compute it's initial pose. Then, using odometry, it moves to the goal position. When the position computed by the odometry is equal to the goal position, the robot localizes itself with the trilateration, looking again for the landmarks (turning around itself).

We repeated our experimental test several times, always using different arbitrary initial positions (and orientations), chosen randomly inside a circle with radius equal to 2 meters, centered in the goal position. From the analysis of the results, it turned out that the localization error

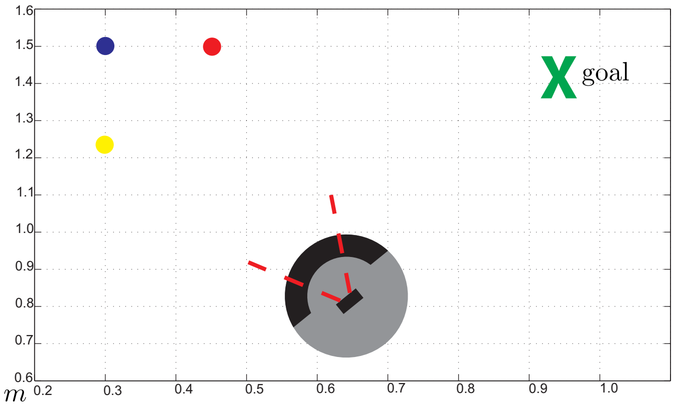


Fig. 8. Experimental tests: starting from unknown initial position, the robot has to localize itself and reach a given goal position

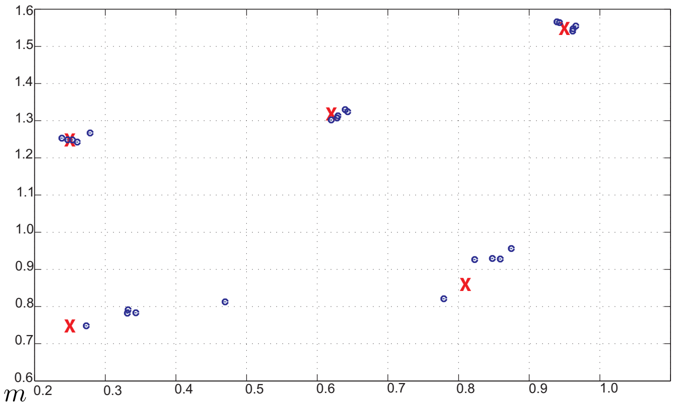


Fig. 9. The localization algorithm has been tested for different positions of the robot: the red crosses correspond to the real positions of the robot, the blue dots correspond to the positions estimated by the localization algorithm

decreased by approximately 20% with respect to the pure odometry localization.

At a first glance, the improvement appears not to be much relevant. But we want to remark that the distance traveled using only the odometry for localization is really short: as is well known, on short distances the odometry performs quite good.

We chose not to make the robot travel a longer path because otherwise the odometry measurement would have been almost completely meaningless, due to the error accumulation. Conversely, the trilateration on the bird's–eye view image of the environment is not affected by the traveled distance. From our experimental tests, it turns out that the mean error in the measurement of the position is approximately $7cm$, with a standard deviation of $3cm$. Some of the data collected during our experimental tests are shown in Fig. 9, while some statistics about the same data are shown in Fig. 10

5. CONCLUSIONS

In this paper we have described a strategy for the self localization of an autonomous mobile robot, based on the use of a monocular camera. The key point of the strategy

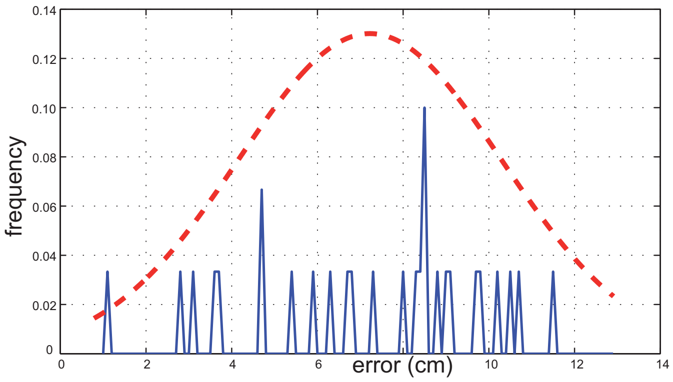


Fig. 10. Frequency of the estimation errors (in blue, solid line) and approximated normal distribution (in red, dashed line)

is to convert the image of the environment acquired by the camera into a top–down *bird's–eye view*, to be used to measure the distance of the robot from some landmarks. The landmarks are placed on the ground floor, and their positions are known in advance, and recorded into a look–up table. By means of a simple geometrical analysis of the bird's–eye view image, it is also possible to compute the orientation of the robot.

To validate our strategy, we developed several experimental tests, to highlight the improvement in the localization accuracy with respect to the use of pure odometry. The main advantage in using a localization technique based on the vision is the fact that the initial pose can be completely arbitrary and unknown in advance. Furthermore, we found out that, as expected, our strategy does not accumulate error while the mobile robot travels, which is one of the main problems with the odometry.

The aim of this paper is to describe a low–cost vision based localization strategy for mobile robots. This objective is obtained simplifying the online computation as more as possible. In fact, the matrix that defines the projection is to be computed only once, during the initial calibration phase. After that, once obtained the bird's–eye view of the environment, the measurement of the distances is trivial, and computationally simple. Despite this computational simplicity, with this localization strategy the robot is able to autonomously find its position and orientation with respect to a given reference frame.

Regarding the localization accuracy, we obtained that the mean error in the measurement of the position is approximately $7cm$, with a standard deviation of $3cm$. To improve this result, it is possible to use a camera with better performances, which introduces less deformations, and which is less sensitive to changes in the light conditions of the environment. Another point to improve the performances is to refine the strategy to deal with the measurement errors, described in Section 4.2. In fact, to avoid heavy computation we have introduced a simplified averaging strategy, but probably considering the composition of probability distributions could improve the accuracy.

Current work aims at implementing this localization strategy in cluttered environment, where several landmarks are placed on the ground floor, but only a reduced number of them are visible at each time.

Furthermore, we are studying how to reduce the influence of the light conditions, to be able to use this localization strategy also in outdoor environment.

REFERENCES

Betke, M. and Gurvits, L. (1997). Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, 13, 251–263.

Borenstein, J. and Feng, L. (1996). Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation*, 869–880.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV – Computer Vision with the OpenCV library.* O'Reilly.

Calabrese, F. and Indiveri, G. (2005). An omni–vision triangulation–like approach to mobile robot localization. In *Proceedings of the IEEE International Symposium on Intelligent Control*, 604–609.

Davison, A.K., Reid, I.D., Molton, N.D., and Stasse, O. (2007). MonoSLAM: Real–time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 1052–1067.

Jin, G., Lu, X., and Park, M. (2006). An indoor localization mechanism using active RFID tag. In *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing.*

Leung, K.Y.K., Clark, C.M., and Huissoon, J.P. (2008). Localization in urban environments by matching ground level video images with an aerial image. *Proceedings of the IEEE International Conference on Robotics and Automation*, 551–556.

Liu, W. and Zhou, Y. (2007). Recovering the position and orientation of a mobile robot from a single image of identified landmarks. *Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems*, 1065–1070.

Matarić, M.J., Koenig, N., and Feil–Seifer, D. (2007). Materials for enabling hands–on robotics and STEM education. In *AAAI Spring Symposium on Robots and Robot Venues: Resources for AI Education.*

Menegatti, E., Pretto, A., Scarpa, A., and Pagello, E. (2006). Omnidirectional vision scan matching for robot localization in dynamic environments. *IEEE Transactions on Robotics*, 22, 523–535.

Thomas, F. and Ros, L. (2005). Revisiting trilateration for robot localization. *IEEE Transactions on Robotics*, 21, 93–101.