

# Global Hough Localization for Mobile Robots in Polygonal Environments

Giorgio Grisetti, Luca Iocchi, Daniele Nardi  
Dipartimento di Informatica e Sistemistica  
Università di Roma “La Sapienza”  
Via Salaria 113, 00198, Roma, Italy  
{grisetti,iocchi,nardi}@dis.uniroma1.it

*Abstract*— Knowing the position of a mobile robot in the environment in which it operates is an important element for effectively accomplishing complex tasks requiring autonomous navigation. Among several existing techniques for robot self-localization, a new approach called Hough Localization was proposed for map matching in the Hough domain, that turned out to be reliable and efficient for position tracking in polygonal environments. In this paper we present an extension of Hough Localization able to deal with the global localization problem, in which the robot does not know its initial position in the environment.

*Keywords*— Mobile Robots, Global Self-Localization, Hough Transform.

## I. INTRODUCTION

The ability for a mobile robot to find its pose (position and orientation) in an environment is a crucial feature for autonomous navigation and for performing complex tasks over long periods of time. Among all the techniques that have been proposed for robot self-localization (see [1] for a survey), we can distinguish those methods that make the assumption that the robot knows at every time an estimate of its pose (also known as *position tracking problem*) from those methods that do not rely on any assumption on the current pose of the robot (*global localization problem*).

The assumption of knowing an estimate of the robot’s current pose at every time is often a strong one for a mobile robot performing complex tasks in dynamic environments, since several situations can occur that make this assumption not valid (e.g. crashes with other objects in the environment, temporary failure of sensor devices, etc.). On the other hand, global localization is usually computationally more expensive than position tracking, because it is not possible to exploit the additional information about the estimated pose of the robot.

Global localization has been addressed in many recent works on localization depending on the sensor used by the robots. When the environment can be appropriately structured, then beacon-based systems allows for a precise and reliable computation of the

robot’s pose in the environment covered by these beacons. However, when the environment cannot be modified the most common class of methods for absolute positioning is model matching, that is the process of determining the pose of the robot by a matching between a given model of the environment (a map) and the information acquired by the robot’s sensors.

Among model matching based methods for global localization, another distinction can be done between those methods that do not rely on the extraction of specific features from the sensor data, and those that are based on extracting and matching specific features detected in the environment. A notable example in the first group are: Markov Localization [2], in which an explicit representation of the probabilistic distribution of the robot’s pose is periodically updated according to the sensor readings, and Monte Carlo Localization [3] that is a more efficient version of Markov Localization, since it represents only a subsets of all the possible poses of the robot. Other methods are based on a local search in the space of the robot poses in order to find the best overlap between the current scan and the reference map. For example, the method described in [4] performs matching between raw sensor data and an unstructured representation of the environment.

Feature-based map matching are based on extracting features from sensor data and matching them with a representation of the environment in terms of these features. Lines are often used for localization in polygonal environment. For instance, in [5] an algorithm for computing robot poses by line matching is presented, while the localization method in [6] is based on certainty grids and on the use of the Hough Transform only for extracting lines from these points.

In this paper we present a method called Global Hough Localization, that extends the Hough Localization method described in [7], [8] in order to provide a solution for global localization. Global Hough Localization is based on matching a geometric reference map with a representation of range information acquired by the robot’s sensors. We exploit the proper-

ties of the Hough Transform for recognizing lines from a sets of points, as well as for calculating the displacement between the estimated and the actual pose of the robot. The main differences between our approach and others are: 1) by exploiting some properties of the Hough Transform, map matching is directly performed in the Hough space; 2) the probability distributions of the robot’s pose is represented as a sum of Gaussian. The combination of these two features makes our approach more efficient. In particular the probability distribution that is not represented in an explicit way allows for simplifying computation by still having enough power to maintain probabilistic multiple hypotheses on the robot’s pose.

We have fully implemented and tested this method in office-like environments as well as in the RoboCup environment by making use of vision based line extraction procedures performing as a range data sensor.

## II. HOUGH LOCALIZATION

In this section we recall the Hough Localization method for position tracking problem (see [7], [8] for a more detailed description).

Hough Localization is based on a matching between a known map of the environment and a local map built by the robot’s sensors. The matching is performed between the Hough representation of both the reference map and the local map. Sensor data are thus transformed in a parameter space by applying the Hough Transform given by the equation  $\rho = x \cos\theta + y \sin\theta$ .

Thus every point  $P(x, y)$  in the Cartesian plane (representing a sensor range information) determines a curve in the Hough domain  $(\theta, \rho)$  and we denote with  $HT^S(\theta, \rho)$  the Hough Transform of the sensor data  $S = \{(x_i, y_i) \mid i = 1, \dots, n\}$ . At the same time, a point in the Hough domain corresponds to a line in the Cartesian plane  $(x, y)$ . In particular, local maxima of the Hough Transform correspond to the best fitting lines of the sensor points and they will be matched against the map reference points for computing the displacement needed for correct positioning of the robot.

A localization process can be considered as the task of evaluating at every time  $t$  the most likely pose  $l = (x, y, \theta) \in \mathbb{R}^2 \times [0, 2\pi)$  of the robot, that is the probability distribution of the robot’s pose  $p(l \mid A^t, S^t)$ , given a reference map  $\mathcal{M}$  and the information on the robot’s status and on the environment taken at time  $t$  from a relative positioning system  $A^t$  and from a range sensor  $S^t$ .

This task is usually performed in two steps:

- *Prediction* of the new pose of the robot by dead reckoning from the previous position;
- *Update* of the robot’s pose with the results of a map matching process between  $S^t$  and  $\mathcal{M}$ ;

Summarizing, the Hough Localization method consists in: 1) computing the Hough Transform  $HT^S(\theta, \rho)$  of sensor points  $S$  in the  $(x, y)$  plane, 2) determining the local maxima of  $HT^S(\theta, \rho)$ , 3) finding correspondences between local maxima and reference points, 4) measuring displacements between local maxima and the corresponding reference points in the Hough domain (that corresponds to the displacements between the estimated and the current pose of the robot), 5) integrating map displacements with odometric information and computing a new probability distribution.

The critical step of this procedure is the third one, that is finding the correct correspondence between local maxima and reference points. In [7] we have assumed a *small error assumption* on the robot’s pose, that is the case in which we can consider the actual position of the robot very close to the estimated one (leading to the position tracking problem). In this case the probability distribution of the robot’s pose  $p(l)$  was represented as a single Gaussian, whose mean  $l_k$  is the most likely position of the robot and the covariance matrix  $P_k$  represent the variance of this information and the correspondence problem was easily solved by adopting a closest matching approach. In other words, the HT grid was partitioned in a number of regions (one for each reference point in  $\mathcal{M}$ ), such that a matching was considered only if a local maximum of  $HT^S$  was within the corresponding region.

In this article, we drop the small error assumption on the robot’s pose and consider the general case in which the robot does not rely on any information about its current pose.

## III. GLOBAL HOUGH LOCALIZATION

By removing the assumption that the robot is only affected by a *small positioning error*, the correspondence between local maxima of the Hough Transform and reference points is not trivial, because it is not possible to rely on an initial estimate of the robot’s position.

Moreover, in order to detect and to deal with ambiguous situations, we need a more powerful representation of the robot pose than a simple Gaussian used in our previous work. On the other hand, we do not want to explicitly represent a distributed probability over the whole configuration space of the robot as in [2], since it tends to be computationally inefficient. Therefore we choose to represent the probability distribution  $p(l)$  as a sum of a finite (limited) number  $N$  of Gaussians. The number of Gaussian  $N$  is varying over time and depends on the number of possible ambiguities detected during map matching. In this way ambiguous situations will be characterized by the presence of  $N > 1$  Gaussians. Moreover the variances of

the Gaussians can also be used to determine the most likely position at each time.

In this section we describe how to compute a set of Gaussians (denoting all the possible robot's poses) from a matching in the Hough domain between sensor data and a reference map, and without the knowledge of an initial pose of the robot.

Let  $\mathcal{M} = \{m_i = (\rho_i, \theta_i, P_i, \lambda_i) \mid i = 1, \dots, M\}$  be a representation of the environment as a set of segments  $m_i$ , denoted by the line containing the segment  $(\rho_i, \theta_i)$ , the center point  $P_i$  and its length  $\lambda_i$ ; and let  $S = \{(x_k, y_k) \mid k = 1, \dots, S\}$  be the current sensor data (in the robot's reference system).

Given the sensor data  $S$ , we want to compute a set of possible poses of the robot in the map  $\mathcal{M}$  and represent them as a set of Gaussians  $\mathcal{L} = \{l_j^R = (x_j^R, y_j^R, \theta_j^R), \Sigma_j^R\}$ , with means  $l_j^R$  and covariance matrices  $\Sigma_j^R$ . The probability distribution of the robot's pose will be thus defined as<sup>1</sup>

$$p(l \mid S) = \sum_{j=1}^N G_j(l; l_j^R, \Sigma_j^R) \quad (1)$$

An important property of the Hough Transform is the *Decomposition Property* (for details see Property 1 in [7]), which allows for decomposing the map matching process in two phases: first we only consider a rotation of the robot and thus we determine a set of possible rotation angles  $\theta_\lambda^*$  that will correct the orientation of the robot, and then for every rotation  $\theta_\lambda^*$  we compute possible translations  $\rho_{\lambda\mu}^*$ . As we will see later, this property provides for an efficient and easy way to compute these displacements separately.

The generation of the set  $\mathcal{L}$  will be performed in four steps:

**Step 1. Computing  $HT^S(\theta, \rho)$**

In the first step the Hough Transform is computed for all the sensor points in  $S$ .

**Step 2. Computing  $\theta_\lambda^*$**

In the second step we compute  $\theta_\lambda^*$ ,  $\lambda = 0, \dots, \Lambda$  representing a set of possible orientation corrections for the robot. If  $\Lambda = 0$  then the map matching process terminates and the set of Gaussians  $\mathcal{L}$  will be empty.

**Step 3. Computing  $\rho_{\lambda\mu}^*$**

In the third step we compute for each  $\theta_\lambda^*$  the values  $\rho_{\lambda\mu}^*$ , for  $\mu = 0, \dots, \Gamma_\lambda$ , that will represent possible translation corrections for the robot related to  $\theta_\lambda^*$ .

**Step 4. Computing  $(x_j^R, y_j^R, \theta_j^R)$  and  $\Sigma_j^R$**

From the set  $\{(\theta_\lambda^*, \rho_{\lambda\mu}^*)\}$  of possible orientation and translation correction, we compute a set of possible pose  $(x_j^R, y_j^R, \theta_j^R)$  of the robot and their covariance

<sup>1</sup>We denote with  $G(l; l^R, \Sigma^R)$  a 3D Gaussian with mean  $l^R$  and covariance matrix  $\Sigma^R$

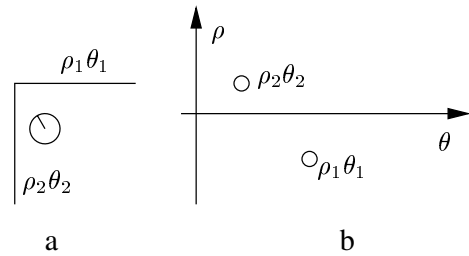


Fig. 1. Example

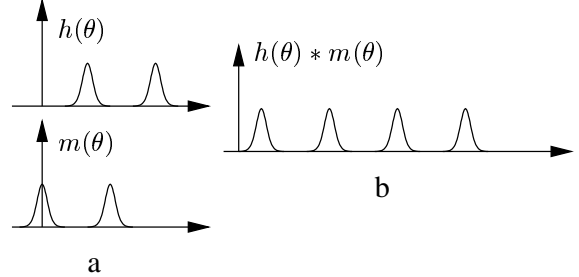


Fig. 2. Angular correlation

matrices  $\Sigma_j^R$ . In this step some of the poses could be discarded (for instance if the variance is too large).

Let us examine these steps in detail, by considering the simple situation in Fig. 1, in which the robot faces a corner without any information about its current position.

The first step consists in transforming all the points detected by the range sensor (Fig. 1b) in the Hough domain as already described in the previous sections.

As for the second step, the *decomposition property* of HT states that during a rotation of the robot the  $\rho$  component of  $HT^S$  does not change. Therefore when looking for an angular correction we can aggregate the values of  $\rho$  in  $HT^S$  and in the Hough Transform of the reference segments in  $\mathcal{M}$  by defining respectively

$$h(\theta) = \sum_{\rho} HT^S(\theta, \rho) \quad m(\theta) = \sum_{\rho} HT^{\mathcal{M}}(\theta, \rho)$$

Observe that the function  $h(\theta)$  is very similar to the *angle histogram* [9] of the sensor data.

The set of possible angle differences between the sensor data and the map are found by a correlation between  $h(\theta)$  and  $m(\theta)$ . Therefore  $\theta_\lambda^*$  will be defined as the local maxima of the correlation function  $h(\theta) * m(\theta)$  as shown in Fig. 2.

If no local maxima are found the map matching process fails, since current sensor data are not useful for correcting the pose of the robot.

In the example, the function  $h(\theta) * m(\theta)$  has 4 local maxima corresponding to 4 possible orientation for the robot. The reason for this result is that at this point

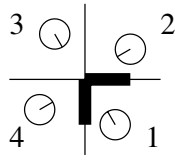


Fig. 3. Possible poses for the robot

we are considering lines and not segments, and thus four different orientations are possible.

The step 3 is performed for every  $\theta_\lambda^*$  as follows:

**Step 3.a** The robot is virtually rotated by  $\theta_\lambda^*$  and  $HT^S$  is updated according to the following equations for roto-translation in the Hough domain (see [7] for details):

$$\begin{aligned}\theta' &= \theta + \theta_R \\ \rho' &= \rho + T_x \cos(\theta + \theta_R) + T_y \sin(\theta + \theta_R)\end{aligned}$$

**Step 3.b** For every  $\theta_\mu^M$  such that  $m(\theta_\mu^M) > 0$ , we define the following functions:

$$\begin{aligned}h'_{\lambda\mu}(\rho) &= HT^S(\theta_\mu^M + \theta_\lambda^*, \rho) \\ m'_\mu(\rho) &= HT^M(\theta_\mu^M, \rho)\end{aligned}$$

We then compute the correlation function  $h'_{\lambda\mu}(\rho) * m'_\mu(\rho)$  and define  $\rho_{\lambda\mu\nu}^*$  as the local maxima of this function. In other words, we are computing a correlation of the columns in  $HT^S$  corresponding to  $\theta_\lambda^*$  and the associated column in  $HT^M$ . In this way it is possible to compute the translation of the robot with respect to the map.

In the example, the map includes two lines with different orientation  $\mu = 1, 2$ . Therefore, for each of the four possible orientations  $\lambda = 1, \dots, 4$ , the two functions  $h'_{\lambda\mu}(\rho) * m'_\mu(\rho)$  computed present a single local maximum  $\rho_{\lambda\mu 1}^*$ .

Finally, step 4 is performed for computing new possible poses of the robot (i.e. the centers of the Gaussians)  $(x_j^R, y_j^R, \theta_j^R)$  and their associated covariance matrices  $\Sigma_j^R$ , from the set  $\{(\theta_\lambda^*, \rho_{\lambda\mu\nu}^*)\}$  of possible displacement between the map and the sensor data.

More specifically, for every  $\lambda = 1, \dots, \Lambda$ , this set of data have been computed in the previous steps:  $\theta_\lambda^*, \rho_{\lambda\mu\nu}^*$ , with  $\mu = 1, \dots, M_\lambda$  and  $\nu = 1, \dots, N_{\lambda\mu}$ . We will distinguish three cases depending on the value of  $M_\lambda$ .

**Case  $M_\lambda = 0$ .** In this case we can only compute the orientation displacement of the robot. Therefore we define  $dl = (0, 0, \theta_\lambda^*)$  and a covariance matrix  $\Sigma_l = \text{diag}\{\infty, \infty, \sigma_\theta^2\}$  with large variances on the translation components.

**Case  $M_\lambda = 1$ .** In this case we have detected a matching with all parallel lines, therefore it is not possible to find a complete translation of the robot. In-

stead we can express just a relation between the translation  $T_x, T_y$  of the robot and  $(\theta_\lambda^*, \rho_{\lambda 1\nu}^*)$ , that is is given by

$$\rho_{\lambda 1\nu}^* = T_x \cos \theta_1^M + T_y \sin \theta_1^M$$

since  $\theta_1^M = \theta_\lambda^S + \theta_\lambda^*$ .

The set of possible position corrections  $dl_{\lambda\nu}$  can be given by  $(T_x, T_y, \theta_\lambda^*)$  subject to the equation above. In some cases it is possible to determine a translation along one direction: for instance if  $\theta_1^M = 0$  we have  $T_x = \rho_{\lambda 1\nu}^*$  and thus  $dl_{\lambda\nu} = (T_x, 0, \theta_\lambda^*)$  and  $\Sigma_{l_{\lambda\nu}} = \text{diag}\{\sigma_x^2, \infty, \sigma_\theta^2\}$ .

**Case  $M_\lambda \geq 2$ .** In this case we have found matches with at least two non-parallel lines. In the case  $M_\lambda = 2$  the possible translation for the robot is computed by solving the following linear system for each  $\rho_{\lambda\mu_1\nu_1}^*, \rho_{\lambda\mu_2\nu_2}^*$ , with  $\mu_1 \neq \mu_2$ .

$$\begin{pmatrix} \cos \theta_{\mu_1}^M & \sin \theta_{\mu_1}^M \\ \cos \theta_{\mu_2}^M & \sin \theta_{\mu_2}^M \end{pmatrix} \begin{pmatrix} T_x \\ T_y \end{pmatrix} = \begin{pmatrix} \rho_{\lambda\mu_1\nu_1}^* \\ \rho_{\lambda\mu_2\nu_2}^* \end{pmatrix}$$

For each solution of the system, the displacement will be given by  $dl_\lambda = (T_{x\lambda}^*, T_{y\lambda}^*, \theta_\lambda^*)$ . The variance matrix  $\Sigma_l = \text{diag}\{\sigma_x^2, \sigma_y^2, \sigma_\theta^2\}$  is also computed by evaluating the precision of the map matching process over the coordinates  $(x, y, \theta)$ . When  $M_\lambda > 2$  the linear system is redundant and a solution can always be found by using a specific method.

Once  $dl$  and  $\Sigma_l$  are computed in one of the cases above, the mean and the covariance matrix of the relative Gaussian will be respectively  $l_j^R = l_{curr} + dl$  and  $\Sigma_j^R = \Sigma_l$ , with  $l_{curr}$  being the current pose of the robot (when sensor data  $S$  were taken).

At this point some of the poses computed could be wrong, since lines have been taken into account instead of segments. Therefore it is necessary to use the information in the map regarding the center and the length of the segments in the map to discard some of the poses.

Consider again the example of the corner: the third step computes the set of data  $\{(\theta_\lambda^*, \rho_{\lambda\mu\nu}^*)\}$  with  $\lambda = 1, \dots, 4$ ,  $\mu = 1, 2$ , and  $\nu = 1$ . Therefore step 4 with  $M_\lambda = 2$  computes 4 possible poses  $(T_{x\lambda}^*, T_{y\lambda}^*, \theta_\lambda^*)$ , with  $\lambda = 1, \dots, 4$ , corresponding to the 4 positions shown in Fig. 3.

By considering the information in the map  $\mathcal{M}$  about the center of the segments  $P_i$  and their length  $\lambda_i$ , it is easy to detect the positions that are not compatible with the actual sensor data. Therefore positions 2,3 and 4 can be discarded since they do not correspond to a possible robot's position and the final result of the global localization will be a single pose represented by a Gaussian centered in  $l = (\rho_{111}^*, \rho_{121}^*, \theta_1^*)$ . In this case no ambiguities have been detected.

We have described so far the process that is needed in order to compute a set of Gaussians  $\mathcal{L} = \{l_j^R, \Sigma_j^R\}$

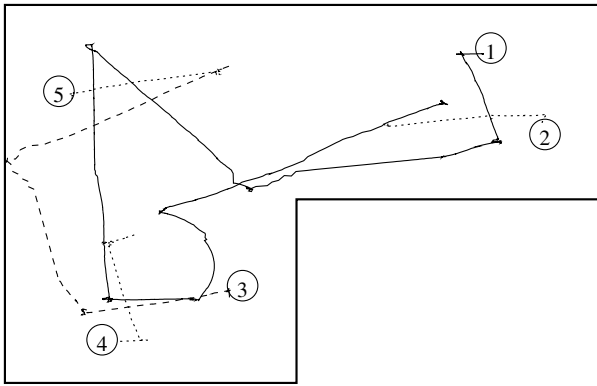


Fig. 4. Experiments for global localization

representing possible poses of the robot in the map  $\mathcal{M}$  given the current sensor readings  $S^t$ . The integration of these data with odometric information and the update of the probability distribution  $p(l)$ , can be again performed by making use of the Extended Kalman Filter operating on every possible pose computed.

By using this kind of integration [10], the system can reduce over time the number of ambiguous situations as the amount of data on the environment increases, since those hypotheses that are not confirmed by sensor data will be discarded.

#### A. Implementation and experiments

The method described here has been implemented on our mobile robots and some experiments have been done for evaluating its performance. Here we report the result of a typical experiment performed in order to evaluate the capability of the robot to maintain a set of hypotheses of its pose and to update the Gaussians associated to these poses over time according to sensor readings. In this experiment we have placed our robot facing a corner of the environment represented in Fig. 4 and after a short time the set of five possible poses (indicated in the figure by numbered circles) have been computed. Then the robot has been moved on a predefined path and the initial poses have been updated. For the hypotheses 2, 3, 4 and 5, at a certain time the matching of sensor data with reference map is not correct and this allows for lowering the probability associated to these Gaussians. Therefore, in the figure, after some time, the trajectories associated to hypotheses 2 to 5 finish denoting that the probabilities of these poses are lower than a certain threshold. Hypothesis 1 instead is maintained since map matches confirm this pose over time.

#### IV. COMPUTATIONAL COMPLEXITY

For computing the complexity of our global localization method we distinguish those variables that in-

crease with the resolution of the range sensor from those ones that increase with the complexity of the environment. If the sensor gives  $|S|$  proximity readings within the maximum range of  $W$  and the  $HT$  is discretized with  $\Delta\rho$  and  $\Delta\theta$ , then computing  $HT^S(\rho, \theta)$  and  $h(\theta)$  costs respectively  $O(|S|\frac{180}{\Delta\theta})$  and  $O(\frac{180}{\Delta\theta}\frac{W}{\Delta\rho})$ .

Let the reference map  $\mathcal{M}$  made up by  $M_\theta$  sets of  $M_\rho$  parallel lines, and the local map  $\mathcal{Z}$  by  $Z_\theta$  sets of  $Z_\rho$  parallel lines. The computational effort for computing the set of possible poses is proportional to the size of this set, which is  $M_\rho^3 M_\theta^2 (M_\theta - 1) (Z_\theta Z_\rho)^2$  [10].

The complexity of the algorithm, without considering the terms depending on the discretization of the  $HT$  that alters only the resolution of the position estimation and does not depend on the size of the environment and on the sensor data, is  $O(|\mathcal{M}|^3 |\mathcal{Z}|^2 + |S|)$ , where  $|\mathcal{M}|$  is the size of the reference map,  $|\mathcal{Z}|$  is the size of the local map (acquired through the robot's sensors), and  $|S|$  is the number of the sensor readings.

This value can be greatly reduced using some pruning criteria to be applied within step 3 and 4, that take into account additional information, such as height of local maxima or the metric data in  $S$ . The use of those criteria does not modify the complexity of the method, but improves the average case.

In practical applications the number of simultaneously visible segments  $\mathcal{Z}$  is often small and does not increase with the complexity of the environment, so the localization algorithm runs with a cubic upper bound with respect to the environment and a linear upper bound with the number of proximity readings  $|S|$ .

Moreover, we may build a set of partial reference maps  $\{\mathcal{M}_i\}$  from the original one  $\mathcal{M}$ , each of them containing only simultaneously visible lines, and such that  $\bigcup_{i=1}^n \mathcal{M}_i = \mathcal{M}$ . Running the algorithm on each  $\mathcal{M}_i$  reduces the size of the possible poses set, by pruning only unfeasible positions. In fact, as in [5] under the hypothesis that the lines in the map are uniformly distributed (this is a typical case in office-like environments) the number of these subsets  $\mathcal{M}_i$  is linear with  $\mathcal{M}$  and the computation for each  $\mathcal{M}_i$  can be considered constant with respect to the size of the environment. Therefore in practical cases grouping the reference map into these sets of simultaneously visible segments leads to a linear complexity in the size of  $\mathcal{M}$ .

The computational complexity of our work may be compared with a similar approach of global localization by using line matching described in [5], that has the same worst-case complexity for line matching  $O(|\mathcal{M}|^3 |\mathcal{Z}|^2)$ . As for the line extraction, the use of Hough Transform allows for an additional complexity of only  $O(|S|)$ , while other approaches may have a higher computational time in case of multiple lines,

when the problem of point clustering (i.e. assigning every point to the line to which it belongs) must be addressed. This is mainly due to the fact the Hough Transform allows not only for detecting lines from a set of points, but also for an automatic clustering of points belonging to each line. Therefore working in the Hough space gives a substantial computation time advantage with respect to other approaches working in the map or robot's coordinate space.

## V. CONCLUSION

Localization for mobile robots has been extensively studied in the past years and several different approaches have been proposed, both for the position tracking problem and for the global localization problem. Among all, the most common methods are based on map matching for determining the metric absolute pose of the robot in the environment and on the integration of this matching with relative positioning information. Global Hough Localization presented here follows from our previous work in [7], [8]. We have implemented the method and the first experiments in office-like environments have shown the effectiveness and efficiency of the approach for computing the absolute pose of the robot in polygonal environments.

Similar previous works include the method in [5], that presents an algorithm for matching lines extracted by the sensors with reference lines, while the work in [6] is based on two certainty grids (one containing data from the reference map, the other containing data acquired by a range sensor) and uses the Hough Transform for detecting segments from the two grids, while the association problem is addressed by comparing in the Cartesian space each pair of segments in order to detect similarities. The difference with our approach is that in these works matching is performed in the Cartesian space, while our work is based on matching in the Hough space, and this allows for improving the efficiency of map matching.

Other global localization methods do not rely on the extraction of geometric features for map matching, but they use raw range readings. These approaches can also be used in unstructured environments. Markov Localization [2] maintains an explicit representation of the probability distribution of the robot's pose in the environment, and updates it according to the sensor readings during robot operation. The method is effective in any environment and it is adequate for dynamic environments. However, because of the large representation space, it tends to be computationally expensive and thus some heuristics must be applied in order to reduce computational time. A similar approach is Monte Carlo Localization [3] that does not represent the whole environment but just a portion of

it in which the robot may be. The main difference between our approach to global localization and Markov-based localization methods is that we represent the probability distribution of the robot poses as a set of Gaussians (representing a limited number of possible poses for the robot) instead of using an explicit (partial) representation of the whole environment.

Notice that we may still have an infinite number of possible poses represented by a finite number of Gaussians (by using large variances on one dimension of a Gaussian). Therefore, in polygonal environments, the representation using a set of Gaussians does not limit the possibility of catching all the possible hypotheses for the robot's poses, while it allows for defining a computationally efficient algorithm.

The approach to global localization presented here has some advantages for its use in polygonal environments: 1) it is computationally efficient and adequate for real-time execution, 2) it is robust in dynamic environments since Hough Transform can detect lines without any bias in presence of partially occluded lines and noise given by isolated points representing (mobile) objects that are not modelled in the map. The method has been implemented and successfully applied in different environments, including the dynamic environment provided by RoboCup soccer matches.

## REFERENCES

- [1] J. Borenstein, H. R. Everett, and L. Feng, *Navigating Mobile Robots: Systems and Techniques*, A. K. Peters, Ltd., 1996.
- [2] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of Artificial Intelligence Research*, vol. 11, pp. 391-427, 1999.
- [3] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," in *Proc. of the 16th National Conference on Artificial Intelligence (AAAI99)*, 1999.
- [4] F. Lu and E. Milius, "Robot pose estimation in unknown environments by matching 2D range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, pp. 249-275, 1997.
- [5] J.-S. Gutmann, T. Weigel, and B. Nebel, "Fast, accurate, and robust self-localization in the robocup environment," in *RoboCup-99: Robot Soccer World Cup III*, 1999, pp. 304-317.
- [6] B. Schiele and J. Crowley, "A comparison of position estimation techniques using occupancy grids," *Robotics and Autonomous Systems*, vol. 12, pp. 163-171, 1994.
- [7] L. Iocchi, D. Mastrantuono, and D. Nardi, "A probabilistic approach to Hough Localization," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA2001)*, 2001, pp. 4250-4255.
- [8] L. Iocchi and D. Nardi, "Self-localization in the RoboCup environment," in *RoboCup-99: Robot Soccer World Cup III*, 1999, pp. 318-330.
- [9] R. Hinkel and T. Knieriemer, "Environment perception with a laser radar in a fast moving robot," in *Proc. of Symposium on Robot Control (SYROCO'88)*, 1988, pp. 68.1-68.7.
- [10] G. Grisetti, "Localizzazione robusta e affidabile per robot mobili in un ambiente dinamico," M.S. thesis, University "La Sapienza", Rome, 2001, (In Italian).