

# Real-Time Robot Localization In Indoor Environments Using Structural Information

Pablo Espinace, Alvaro Soto, and Miguel Torres-Torriti  
Pontificia Universidad Católica de Chile  
Email: {pespinac, asoto, mtorrest}@ing.puc.cl

**Abstract**— This paper presents a novel approach for real-time mobile robot localization in structured indoor environments. The proposed method takes advantage of the available structural information by implementing a Monte Carlo Localization strategy over a map of line segments rather than a grid-based map, thus allowing for speed improvements. Another novel aspect is in the likelihood function, which is based on a Modified Hausdorff Distance between the expected line segments the robot should sense and the line segments extracted from actual measurements using a range finder. Additionally, the number of particles of the Monte Carlo Localization method is automatically adjusted, using a large number of particles in the global localization phase, where the position of the robot is unknown, and a reduced number of particles during the state tracking phase, where uncertainty about the robot position is restricted to a small area. The proposed approach has been implemented and tested in a real office environment, achieving true real-time performance. Results show a fast convergence from global localization to state tracking, as well as, robustness in position tracking. Experimental tests and comparisons with state-of-the-art methods validate the efficiency and robustness of our approach.

## I. INTRODUCTION

Self-localization is an essential task for mobile robot navigation, especially in environments where globally accurate positioning systems, such as GPS, are not available. Existing localization algorithms can be classified into two main groups: i) Position tracking algorithms needing an initial position estimate, and ii) Global localization algorithms that calculate the robot location without knowledge of its initial pose.

Most methods in the first group rely on the well known Kalman Filter and its variants; see for example [1], [2], [3]. Some of the limitations of the Kalman Filter arise from the linearity assumption (partially solved in the Extended Kalman Filter) and the Gaussian noise assumption. These assumptions limit the filters performance when the system is nonlinear and disturbances are non-gaussian. Despite these limitations, the Kalman Filter performs reasonably well and is one of the most employed filtering methods due to its simplicity and low computational cost.

The main problem with position tracking methods is that they fail to consider the global localization problem arising when the robot does not know its position at all, for example, when the robot starts from an unknown initial position or when it loses track of its position and needs to relocalize itself with no knowledge that a position change has occurred. The latter problem is known as the *kidnapped robot problem*

and occurs when the robot is lifted and relocated without updating the robot about its new position, or when the robot has a temporary sensors loss. To cope with this problem, some researchers have proposed to maintain multiple hypothesis about the current robot location [4], [5].

The most popular global localization method is the Monte Carlo Localization approach [6] based on Particle Filters. This technique is a sample-based implementation of the general Bayes Filter, which is able to handle multimodal noises and highly nonlinear functions. The main problem of Monte Carlo Localization is the trade-off between efficiency and accuracy, this is, the algorithm needs as much particles as possible in order to achieve an accurate localization, but as few particles as possible in order to perform efficiently in real time.

Several modifications to the original Monte Carlo Localization approach have been proposed to improve the algorithm's robustness and efficiency, such as: i) new particle resampling schemes [7], ii) inclusion of topological information to reduce the size of the particles set [8], iii) relocation of particles to prevent the robot from getting lost due to low particle likelihood and particles set degradation [9], iv) adaptive particle filtering using KLD-sampling to dynamically adjust the number of particles [10], [11], [12], v) likelihood function smoothing and uncertainty regions adjustment according to whether the robot is in global localization or tracking phase [13].

Many successful localization approaches use features extracted from the environment. Most feature-based approaches employ line segments. A line-based SLAM method that uses separate Particles Filters to estimate map object positions and robot position is introduced in [14]. Robot localization is performed in a limited area around the previously estimated location. In [15], a localization algorithm is proposed that tracks the position of a mobile robot by matching line segments extracted from laser range finder readings with line segments in a predefined map. Line segments extraction from sensor data is performed through a sequential segmentation method. A similar approach is presented in [16] using the Hough Transform for line segment extraction from sensor data. A lightweight SLAM algorithm that uses the orthogonality property of most indoor environments to create a lines map for the environment is proposed in [17]. The approach shows that an effective use of a particular environments information can lead to major improvements in efficiency with respect to other state of the art methods. An

experimental comparison of several widely used procedures for line extraction from laser range finder measurements is also provided in [18].

In this paper we present a fast and robust implementation of the Monte Carlo Localization algorithm that takes full advantage of the implicit structural information available in indoor office environments. Our system employs a likelihood function based on a Modified Hausdorff Distance (MHD) suggested in [3] for scan-to-map matching. However, instead of employing a point to point distance as in [3], our approach computes the distance between the parameters of expected line segments observations as should be seen by the robot given the map and the parameters of line segments extracted from the robot's laser range finder measurements. Furthermore, to reduce the trade-off between efficiency and accuracy, the number of particles of the Monte Carlo Localization method is adapted using the revised KLD-Sampling method presented in [12]. Fast line extraction and comparison methods implemented provide the basis for an effective adjustment of the number of particles, thus enabling the algorithm to rapidly converge from the global localization phase to the state tracking phase, when the robot starts from an unknown positions, and guarantying robust and efficient position tracking, when the robot position is limited to a small area.

The remainder of this paper is organized as follows. Section 2 presents our localization method, explaining the line extraction procedures, a Modified Hausdorff Distance proposed in [19], the Monte Carlo Localization implementation based on a likelihood function that employs the Modified Hausdorff Distance as similarity measure, and the method for adapting the number of particles. Section 3 presents the experimental results and a comparison with a point to point method that prove the robustness and efficiency of our approach. Finally, Section 4 presents some concluding remarks and future work.

## II. LOCALIZATION SYSTEM

The proposed Monte Carlo Localization approach exploits the fact that most indoor environments like office floors are highly structured, and hence, facilitate the extraction of lines from range finder measurements as explained next. By comparing the lines expected given the map with those from the measurements it is possible to assess the likelihood that the measurements are correct and thus correct the belief about the position as also explained in the subsequent sections.

### A. Line Extraction

Given a robot pose and a predefined map containing the line segments corresponding to the different walls of a structured environment, a group of line segments that should be observed by the robot can be constructed. We call this group of line segments the Map Segments Group (*MSG*). Considering that the robot is equipped with a laser range finder that provides measurements of range and bearings to obstacles around it, the *MSG* is constructed as follows:

- 1) Initialize *MSG* as an empty set

- 2) For each laser ray do
  - a) Simulate trajectory and compute the map line segment hit by the ray
  - b) Solve the ray casting problem to find the hit point, i.e. intersect the ray trajectory and the map line segment to compute the corresponding hit point
  - c) If the current ray is the first ray of the laser reading, the corresponding hit point is settled as the initial point of the first segment in *MSG*
  - d) If the current ray hits a different line segment from that found on the previous ray casting, the previous ray hit point is defined as the end point of the last segment included in *MSG*, and the hit point of the current ray is defined as the initial point of a new segment in *MSG*
  - e) If the current ray is the last ray of the laser reading, the corresponding hit point is set as the end point of the last segment included in *MSG*

### 3) End

The computational complexity of computing *MSG* is  $O(N_l * N_r)$ , where  $N_l$  is the number of line segments in the map and  $N_r$  is the number of rays in a laser scan.

Using the current laser scan, a group of line segments observed by the robot can be constructed. We call this group of segments the Observed Segments Group (*OSG*). The *OSG* is built by applying the Split and Merge method for line extraction to the laser reading. The Split and Merge method has been shown to be one of the best algorithms for line extraction [18]. Its procedure can be summarized as follows:

- 1) Initialize *OSG* as an empty set and let  $P_1$  be the group of points scanned by the laser range finder. Put  $P_1$  in a list  $L$
- 2) Repeat until  $L$  is empty
  - a) Take the first group of consecutive points  $P_i$  remaining in  $L$  and find the line segment  $S_i$  that best fits those points
  - b) Find point  $p$  in  $P_i$  with maximum distance  $d$  to  $S_i$
  - c) If  $d$  is larger than a threshold,
    - i) Split  $P_i$  at  $p$  in two groups of points  $P_j$  and  $P_k$
    - ii) Put  $P_j$  and  $P_k$  in  $L$
 Otherwise, insert  $S_i$  in *OSG*.
  - d) If  $S_i$  was inserted in *OSG*, but  $S_i$  is collinear with the segment  $S_j$  inserted in *OSG* just before  $S_i$ , and the distance between  $S_i$  and  $S_j$  is lower than a threshold, merge  $S_i$  and  $S_j$  into a single segment  $S_k$ , remove  $S_i$  and  $S_j$  from *OSG* and insert  $S_k$  in *OSG*
  - e) Remove  $P_i$  from  $L$

### 3) End

The Split and Merge algorithm was implemented using Total Least Squares for line fitting as in [18]. The results are illustrated in terms of the example shown in Figure 1 from which it is possible to observe that the algorithm finds



Fig. 1. Line extraction example assuming that the estimated pose is equal to the real one.

very similar groups of segments when the estimated pose is close to the real robot pose. Furthermore, the method has two key advantages over other methods for obtaining information from the environment: i) It is fast in obtaining the segment groups because it depends only on the number of laser rays per reading (usually in the order of hundreds) and on the number of lines in the map (also below a few hundred), ii) Even if an observed line is partially occluded by surrounding objects, the Split and Merge algorithm is able to detect it as a single line and not as several different lines partitioned by the surrounding objects, as long as these objects are below line separation thresholds or above the merge thresholds. As noted in [17] this allows the approach to cope well with the inherent dynamism of common environments.

### B. Segments Comparison Using a Modified Hausdorff Distance

Once we have two groups of segments to be compared,  $MSG$  and  $OSG$ , an appropriate distance measure between them must be defined. We have chosen to use a Modified Hausdorff Distance (MHD) proposed in [19]. This distance does not constitute a metric in a strict mathematical sense, but provides a similarity measure by calculating the average of the minimum distances between each segment in  $OSG$  to the segments in  $MSG$ . The formal Hausdorff Distance measures the largest deviation between two groups of points [3]. Given two groups of points,  $P$  and  $Q$ , the Hausdorff Distance is defined as

$$H(P, Q) = \max(h(P, Q), h(Q, P)) \quad (1)$$

where

$$h(P, Q) = \max_{p \in P} \min_{q \in Q} \text{dist}(p, q)$$

is known as a *directed* Hausdorff Distance between  $P$  and  $Q$ , and  $\text{dist}(p, q)$  is a distance measure between two points  $p$  and  $q$ .

Let  $|P|$  denote the number of elements in  $P$ . It has been shown in [19] that the MHD defined by (1), with  $h(P, Q)$  replaced by:

$$h(P, Q) = \frac{1}{|P|} \sum_{p \in P} \min_{q \in Q} \text{dist}(x, y),$$

has the best performance in terms of object matching accuracy amongst other variants of the Hausdorff Distance, including the formal Hausdorff Distance. For most purposes, the distance measure  $d(p, q)$  can be defined in terms of

the Euclidean distance between the  $p$  and  $q$ . It is worth pointing out that in our approach, the Euclidean distance is not computed between the parameters of the segments, such as slope and intercept, but rather the start and end points. This has the advantage of being fast, while keeping into account differences between segments not only in slope and position, but also in length.

### C. Monte Carlo Localization

Monte Carlo Localization (MCL) is one of the most popular methods to solve the robot localization problem. It relies on a sample-based implementation of the general Bayes Filter [20] and therefore is suitable for multimodal distributions and nonlinear stochastic processes. Each sample of the MCL procedure represents a possible robot pose. MCL involves three main steps: i) a propagation step, where each sample is moved according to the robot odometry readings and an appropriate motion model, ii) an observation step or weighting step, where each sample receives a weight according to a likelihood function based on the robot sensors readings, the robot pose represented by the corresponding particle and a predefined map, and iii) a resampling step, where a new set of samples is built by selecting particles with a probability that is proportional to their weights. A single particle may be selected one or more times, or it may not be selected at all.

In order to take advantage of the structural information available in the environment, the proposed MCL approach employs the line extraction procedures of section 2.A and the MHD presented in section 2.B, as the basis for the computation of the likelihood function. For each particle  $p$ , the  $MSG$  and  $OSG$  groups of segments are computed. A likelihood function  $L(P)$  that decreases as the MHD between  $MSG$  and  $OSG$  increases can then be defined as follows:

$$L(P) = \frac{1 - \tanh\left(\frac{2 * ((\text{Dist}(MSG, OSG)) - IP)}{IP}\right)}{2}$$

where  $\text{Dist}(x, y)$  is the MHD of Section 2.B, and  $IP$  is the point where the likelihood function has a value of 0.5. By adjusting  $IP$  it is possible to handle how fast particle weights decrease as the difference between  $MSG$  and  $OSG$  grows. Figure 2 shows that low values of  $IP$  result in narrow likelihood functions, which provide high weights only to particles that are very close to the real robot pose. On the other hand, high  $IP$  values generate smooth likelihood functions that assign a higher plausibility to particles that may be far from the robot's real pose.

The main weakness of the proposed likelihood function is that it is very sensitive to small changes in the robot's orientation, even when  $IP$  is high. To deal with this problem, a very simple and highly effective method was developed to adjust the robot's sampled orientation value. The method once again takes advantage of the structure of the environment and is applied to each particle after the line extraction procedure, but before the weight is assigned. It works as follows:

- 1) Select  $S_1$ , the longer line segment in  $OSG$
- 2) Find  $S_2$ , the line segment in  $MSG$  that has minimum distance to  $S_1$

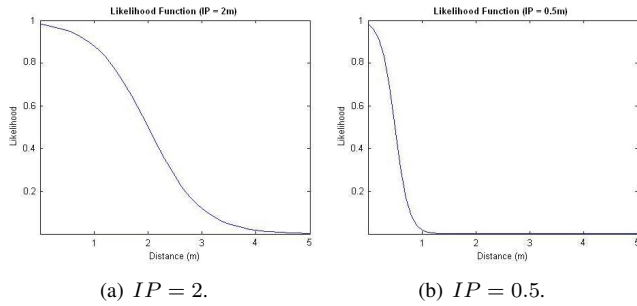


Fig. 2. Likelihood function for two different values of  $IP$ .

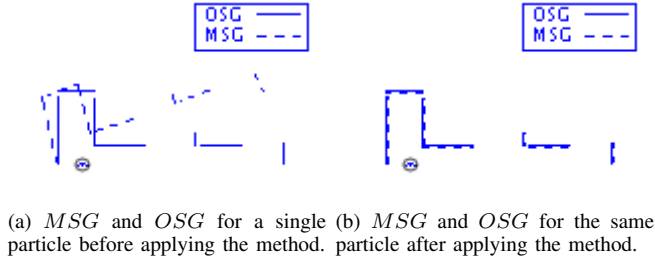


Fig. 3. Example of the method for adjusting particles orientation.

- 3) Compute  $\theta$ , the angle difference between  $S_1$  and  $S_2$
- 4) Change the orientation value of the particle in  $\theta$
- 5) Rotate the line segments in  $MSG$  by  $\theta$  in order to update them consequently with the new orientation value of the particle
- 6) End

By using the previously described method, the likelihood values of particles close to the real robot's pose, but differing in orientation, grow considerably, hence, the overall performance of the localization algorithm becomes more efficient, as the particles set converges faster to the real robot pose. In Figure 3, the  $MSG$  and  $OSG$  groups of segments having initially a large distance between them are aligned very accurately using the proposed approach, as long as they only differ in angle.

#### D. Adaptive Sample Set

As mentioned earlier, the main problem of MCL localization is the trade-off between efficiency and accuracy. This means that the algorithm needs as much particles as possible to achieve an accurate localization, but as few particles as possible to perform efficiently in real time. A natural way to handle this trade-off is to adapt the number of particles in order to use many particles in the global localization phase, when the uncertainty about the robot location is high, and fewer particles in the state tracking phase, during which the uncertainty about the robot location decreases.

In our approach, we include an implementation of the Revised KLD-Sampling method presented in [12]. Revised KLD-Sampling allows to reduce the number of particles as the robot pose converges, by maintaining this number proportional to the variance of the particles set. By doing so, the state tracking phase uses as few particles as possible.



Fig. 4. Initial situation for the localization procedure with 5000 uniformly distributed particles.

To take full advantage of the potential given by Revised KLD-Sampling, the  $IP$  value is adjusted in order to adapt the likelihood function according to the variance of the particles set. During global localization, where many particles are available, a narrow likelihood function is used to select samples that are close to the real robot position. During state tracking, where fewer particles are available, a wider likelihood function is used to lower the probability of the robot getting lost because of low weights for all particles in the set. Even if it is possible to change the likelihood function according to the current phase, we have found that the localization algorithm is able to perform robustly and efficiently by maintaining the likelihood function unchanged and keeping the number of particles to a minimum in the state tracking phase. The minimum number of particles that ensure a good performance was empirically determined to be at least 80.

### III. EXPERIMENTAL RESULTS

In this section we describe the results of our experimental tests in a real world office environment, the Computer Science Department (DCC) of our university. For this purpose, we built a lines map of the DCC, where each line corresponds to a wall. We start the section providing general results for the proposed algorithm. Next, we compare our approach with a classical implementation of the Monte Carlo Localization algorithm, that uses a point to point based likelihood function.

#### A. General Results

We tested our system in two different situations: i) In a simulated DCC environment, to test the system under ideal conditions, ii) In the real DCC, to test the system in a real world situation. All our tests start with a uniformly distributed particles set, as we have initially no knowledge about the robot location (Figure 4).

Simulation results were fully successful. In every tested situation, the particles set converged to a small area around the real robot position, however, the convergence speed depended on the similarity of the starting robot location with other places in the environment. As can be seen in Figure 5, when the robot starts in a place that is different from any other place in the environment, the system provides a fast convergence. After 7 iterations of the algorithm, most places in the environment are already discarded, and the number of particles is reduced from 5000 to 900. After 50 iterations, the robot has changed its position within the environment, however, it has successfully tracked its position using the minimum number of particles (80).



(a) Situation after 7 iterations of the algorithm. The robot pose is limited to a small area within the environment. The number of particles is reduced to 900.



(b) Situation after 50 iterations of the algorithm. The robot is localized. The number of particles is reduced to 80.

Fig. 5. Evolution of the localization algorithm in the first example.

When the robot starts in a place that looks very similar to other places in the environment, it must maintain several hypothesis about its current location. In the example in Figure 6, the robot starts in one of the small upper corridors of the environment. After 7 iterations, the robot is able to discard every place in the environment, except for the five corridors that look equal to each other. As there are still several possible locations for the robot pose, the number of particles is not yet considerably reduced. After the robot moves out of the corridor, it has enough information to decide that its real location is in the corridor to the left of the map. As the position uncertainty is now very small, the number of particles is quickly reduced to its minimum of 80.

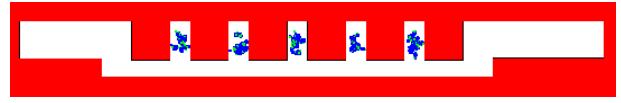
In the real world environment tests, we found two considerable differences with respect to the simulated environment case: i) The real world environment is only partially represented by the given map, ii) There are dynamic objects, specially people walking near the robot.

Problem i) produces a reduction in the calculated likelihood values, however, this is not a major problem for our approach, as likelihood values decrement is mainly proportional between the different particles.

Problem ii) produces a reduction in the calculated likelihood values that, if the laser rays occlusions are high, is quite bigger than in the previous case. As said previously, the method discards short lines to handle dynamics up to a certain point, however, if laser rays are strongly occluded, the performance of the algorithm is seriously affected.

Some of the consequences of the above mentioned problems are: i) A slower convergence from a global localization phase to a state tracking phase, producing a decrement in the efficiency of the system, and ii) A collapse of the particles set, because of a low weight in every particle, forcing the system to restart from a new uniformly distributed particles set. It is important to remark that this last situation only happens under very hard occlusion of the sensor readings for a considerable period of time. In our tests, it only happened in situations where many people came near the robot, however, the system could always restart and relocalize properly.

The only case where our system failed in the global localization phase was in a situation where the robot started



(a) Situation after 7 iterations of the algorithm. The robot pose is limited to the five corridors that look equal to each other. The number of particles is not reduced yet.



(b) Situation after the robot moves out of the corridor. The robot is localized. The number of particles is reduced to 80.

Fig. 6. Evolution of the localization algorithm in the second example.



Fig. 7. Failed run of the global localization phase.

at the large corridor in the lower part of the map, and there was a box in front of it that permanently occluded the laser rays (Figure 7). As a consequence, the system detected a line that was not a wall, and decided that it corresponded to the wall at the end of the corridor, making the localization procedure to converge to an area that was not around the real robot pose.

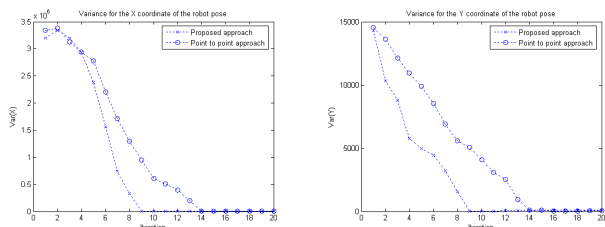
### B. Comparison with Point to Point Implementation

Now we provide an experimental comparison for our system with a classical implementation of the Monte Carlo Localization algorithm, based on a point to point likelihood function. This likelihood function consists on the Mean Square Error among the distances obtained from a laser reading, and the value each of these distances should have, according to the pose represented by the current particle and a grid based map of the DCC, built by the SLAM algorithm in [21]. Mean Square error has the advantage of punishing the error estimate when some of the rays are too far from their expected value.

We compared both algorithms under three different metrics. First, we tested the number of particles that the point to point method needs in order to provide a similar efficiency to the proposed approach using 5000 particles, in terms of number of iterations per minute. Over an average of ten runs, results show that efficiency for the proposed approach is 85% bigger. Second, we tested the convergence speed for both algorithms in terms of iterations, starting from a situation of uniform distribution of 5000 particles, until more than 95% of the particles lay within 1 square meter from the actual robot pose. Over an average of 10 runs, the proposed algorithms converges 55% faster. Finally, we tested the likelihood values of the best particles for both systems after convergence. Over an average of ten runs, the likelihood values are 3% higher in the case of the proposed algorithm.

TABLE I  
COMPARISON OF THE PROPOSED METHOD WITH A POINT TO POINT  
BASED LOCALIZATION METHOD.

	Number of particles. (same speed)	Iterations to converge.	Average likelihood (best sample).
Comp. method	2700	14	0.91
Prop. method	5000	9	0.94



(a) Evolution of the variance for the X coordinate of the robot pose. (b) Evolution of the variance for the Y coordinate of the robot pose.

Fig. 8. Variance Evolution.

A summary of the comparison results is shown in Table I.

Additionally, to provide a deeper test about the convergence speed of the compared algorithms, we tested the variance evolution for both, the X and Y coordinate of the robot pose. As can be seen in Figure 8, in both cases the variance in the point to point approach decreases slower than in the proposed approach. It can also be seen that, at the beginning, the variance in the Y coordinate is lower than the variance in the X coordinate, because of the rectangular nature of the environment.

As a final issue, we observed that the likelihood values in the case of the proposed approach are more stable when tested under similar levels of dynamism, as the particles set collapsed considerably less.

#### IV. CONCLUSIONS AND FUTURE WORK

In this paper we have shown that structural information, such as hallways and walls in office buildings, can be efficiently used to improve current localization methods for this particular type of environments. By building a compact line-based map representation and implementing fast line extraction and comparison methods, we have developed a system that outperforms a classical point to point approach in all aspects.

In order to improve the proposed localization strategy, ongoing research considers improving the line based map representation to account for the dynamics of the environment, such as furniture being moved. An automatic construction and update of the map, using an algorithm like the one proposed in [17], is a possible way to perform this improvement. Additionally, the following actions can be implemented to increase the robustness of the approach: i) making a better use of the information available in the environment by using 3D map representations. In this way, problems such as the box example shown in Section

III-A may be avoided, ii) detecting and selecting natural landmarks that may help to differentiate places that look alike if only structural information is employed, iii) detecting people nearby in order to determine which part of the laser measurements correspond to occlusions caused by people in the vicinity of the robot.

#### REFERENCES

- [1] S. Kwon, K. Yang, S. Park, and Y. Ryuh, "Robust mobile robot localization with combined kalman filter-perturbation estimation," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2005.
- [2] S. Kwon, K. Yang, and S. Park, "An effective kalman filter localization method for mobile robots," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.
- [3] F. Donoso-Aguirre, J.-P. Bustos-Salas, M. Torres-Torriti, and A. Gue-salaga, "Mobile robot localization using the hausdorff distance," *Robotica*, vol. 26, no. 2, pp. 129–141, 2008.
- [4] M. Tomoro, "Robust robot localization and map building using a global scan matching method," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2004.
- [5] M. Paz, P. Piniés, J. Neira, and J. Tardos, "Global localization in slam in bilinear time," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2005.
- [6] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," in *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 1999.
- [7] W. Shang, D. Sun, X. Ma, and X. Dai, "A visual based extended monte carlo localization for autonomous mobile robots," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.
- [8] S. Kwon, J. Yang, J. Song, and W. Chung, "Efficiency improvement in monte carlo localization through topological information," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.
- [9] D. Wang, J. Zhao, and S. Kee, "A novel heat kernel based monte carlo localization algorithm," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2004.
- [10] D. Fox, "Kld-sampling: Adaptive particle filter," in *Advances in Neural Information Processing Systems 14 (NIPS)*, 2001.
- [11] C. Kwok, D. Fox, and M. Meila, "Adaptive real-time particle filters for robot localization," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.
- [12] A. Soto, "Self adaptive particle filter," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.
- [13] P. Pfaff, W. Burgard, and D. Fox, "Robust monte-carlo localization using adaptive likelihood models," in *European Robotics Symposium*, 2006.
- [14] D. Yuen and B. MacDonald, "Line-based smc slam method in environments with polygonal obstacles," in *Proc. of the Australasian Conference on Robotics and Automation (ACRA)*, 2003.
- [15] H. Sohn and B. Kim, "A robust localization algorithm for mobile robots with laser range finders," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2005.
- [16] X. M. F. Fang and X. Dai, "Mobile robot localization based on improved model matching in hough space," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.
- [17] V. Nguyen, A. Harati, N. Tomatis, A. Martinelli, and R. Siegwart, "Orthogonal slam: a step toward lightweight indoor autonomous navigation," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.
- [18] V. Nguyen, S. Gachter, A. Martinelli, N. Tomatis, and R. Siegwart, "A comparison of line extraction algorithms using 2d range data for indoor mobile robotics," *Autonomous Robots*, vol. 23, no. 2, pp. 97–111, 2007.
- [19] M. Dubuisson and A. Jain, "A modified hausdorff distance for object matching," in *Proc. of the International Conference on Pattern Recognition*, 1994.
- [20] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.
- [21] A. Aranedo and A. Soto, "Statistical inference in mapping and localization for mobile robots," *Lectures Notes in Artificial Intelligence*, vol. LNAI 3315, pp. 545–554, 2004.