IN: PROCEEDINGS OF THE ECMR 2005, 2ND EUROPEAN CONFERENCE ON MOBILE ROBOTS, ANCONA, PP. 164-169, EDIZIONI SIMPLE SRL

MULTI-SENSOR MONTE-CARLO-LOCALIZATION COMBINING OMNI-VISION AND SONAR RANGE SENSORS

Christof Schroeter, Alexander Koenig, Hans-Joachim-Boehme, Horst-Michael Gross

Department of Neuroinformatics and Cognitive Robotics, Ilmenau Technical University

christof.schroeter@tu-ilmenau.de

ABSTRACT

Nowadays, Monte Carlo Localization (MCL) is a common method for self-localization of a mobile robot under the assumption that a map of the environment is known. Original implementations used range measuring sensors like laser scanners and sonar as well as camera images. Recently, localization approaches using omni-vision systems have been developed with good results. In this paper, we will compare omni-vision-based and sonar-based MCL in terms of localization accuracy in our specific environment, a large home store. We can show that both approaches bear certain weaknesses and that by combining omni-vision and sonar both sensors complement each other, the respective localization errors decrease, and overall accuracy is improved.

1. INTRODUCTION

The ability of self-localization is one of the key requirements for a mobile robot. In order to perform sensible goal-directed autonomous navigation, the robot must have at least a rough estimate of its current position.

Monte-Carlo-Localization is a method for estimating the position of the robot by approximating the probability function for the robot position using a particle filter. The particle distribution is intialized either as a uniform distribution over the entire environment or according to some prior knowledge about the position. With each robot motion, the particles are moved accordingly, introducing some randomness by using a probabilistic motion model. Periodically, observations from some external sensors are used to evaluate the position hypotheses represented by the particles. This is done by generating a weight for each particle which is coding the probability of the current observation at the particle's location and resampling the particles from the distribution that is given by the weighted particles.

MCL and similar apporaches also appear in the literature as particle filters, Condensation algorithms, sampling-importance resampling (SIR) filters, etc. A wide range of sensors have been used for localization with MCL, most commonly laser range sensors and vision sensors (cameras). Dellaert et al. presented results for sonar and laser data in [1], [2]. The same authors used a camera observing the hall ceilings in [3]. Gross et al. presented an omnidirectional camera-based approach for large-scale maze-like environments [4], [5]. Further vision-based approaches are presented in [6], [7] and [8]. [9], [10] examined sensor fusion in localization, fusing laser and vision data and using Kalman filters for state estimation.

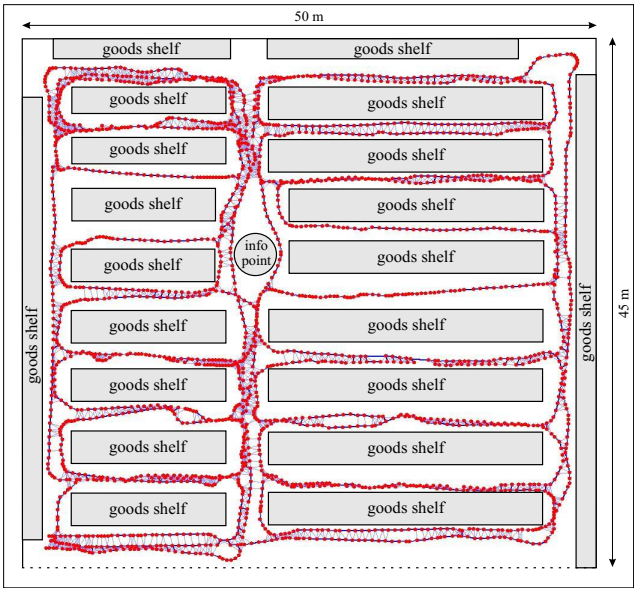

Fig. 1. Learned graph of the operation area, a part of the home store. The size of the area is 50 by 45m^2, the graph consists of about 2000 reference points (red dots) labeled with reference observations and odometric data about the pose of the robot at the moment of node insertion. The total distance traveled to learn this graph was about 1.000 m.

2. VISUAL MCL

In our long-term research project PERSES/SERROKON[1] we are developing an autonomous shopping assistent for a

[1]supported by a grant of the Thuringian Ministry of Science, Research and Art (TMWFK) grant B509-03007 to Horst-Michael Gross

home store to provide guidance and companion functions to customers. The operation area has a size of about 120m by 100m and is characterized by many similar long hallways (Fig. 1). In contrast to typical indoor environments, very few flat surfaces exist, instead store-shelfs with various goods present very finely structured obstacles and walls.

In our application, exact position knowledge is not only important for secure and efficient coordination of path planning and motion control of the robot. Since one of the robot's tasks is to guide customers to the goods they are looking for, it should know its relative position to the goods shelfs as accurate as possible to give precise information where to find the desired product.

In [4], [5], we presented a visual localization approach. There, we described a method of splitting an omni-camera image into several slice segments and extracting the red, green and blue mean values of the RGB color space for each segment. Those RGB features are then used as a description of the environment appearance at the respective position and stored in a map. The map is a graph of nodes, covering the two-dimensional environment, where each node contains the features extracted from the omni-camera image at the respective position. The nodes are drawn as points in Fig. 1, connected along the path the robot took during map building. In the localization phase, the same RGB features are determined for the current observation, and for each particle the similarities with nearby nodes are interpolated to calculate the particle weight. A more detailed description of the method and experiments is given in [5].

We argued that because of the specific topology of our operation area with its maze-like structure, visual features should be superior to range-based features, as they produce far less ambiguities. We therefore discarded sonar sensors for MCL and were able to demonstrate that localization is possible using omni-vision only with good results in accuracy and robustness. However, we also experienced that the precision of self-localization strongly depends on the resolution of the underlying graph. In the environment discussed above, this limitation expresses itself particularly in the position component that lies perpendicular to the corridor direction: When the robot moves down a corridor, the changes in the visual observation are far stronger than when just moving closer or further away from a wall. Therefore, to achieve equal localization resolution in both directions, we would need a higher number of reference nodes, particularly many "lanes" in each corridor (see Fig 1: So far, in most hallways the model only consists of one or two "lanes").

3. SONAR MCL

In order to improve the localization, we implemented sonar-based MCL as extension of the vision-based approach. This section will describe MCL using sonar sensors as standalone solution while in the next sections, we will compare the results of both approaches and address the combination of both modalities.

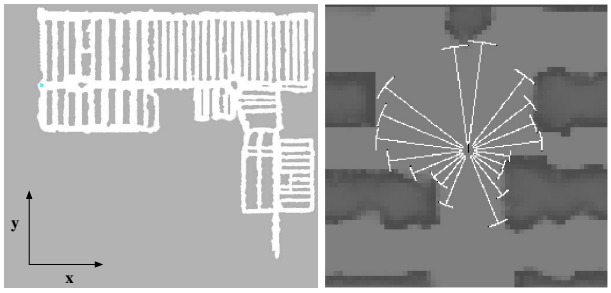

Fig. 2. (Left:) A grid map of the entire home store area (light: free space, dark: obstacles). (Right:) Applying a ray casting algorithm, we can calculate a scan expectation for each position in the map (enlarged view).

As explained above, the base of MCL is the incorporation of sensor observations into the position belief by calculating the probability of an observation provided by the robot sensors, assuming the robot is located at a specific position that is given by a particle. In order to estimate that probability, a map of the environment is required. Furthermore, a sensor model must exist which provides the respective probability distribution.

In our application, the need for an environment map also results from the intention to perform autonomous navigation, which in turn requires the ability of path-planning. To that purpose, the entire area is modelled as a grid map (Fig. 2 left). During a teaching phase, the grid map is built from odometry and sonar range data, using a probabilistic update scheme for each single cell [11]. We employ various odometry correction techniques [12] at that stage in order to ensure the accuracy of the map. The basic consideration here is, that we can put high effort into the one-time process of generating environment and robot models, while the models themselves can be used in low complexity algorithms in the actual application later.

In the MCL context, we can use the grid map as a reference for expected range scans: By applying a ray casting algorithm from a certain position within the map, the expected range measurements in any direction are determined, which correspond to the ranges an ideal sensor would measure (Fig. 2 right). Although the resolution is not very high, as the grid cells in our map have a size of 0.2 by 0.2m, we will see that this is sufficient for pose estimation with MCL.

Now, in order to calculate the probability of the real sensor measurements at that position, we need a sensor model (also called observation model) that describes, for a given expected range, the distribution of real range measurements. While the B21 has 24 sonar sensors placed around the enclosure, our model only describes the distribution of one

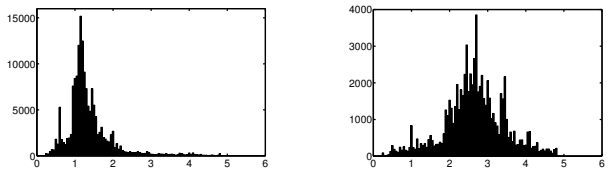

Fig. 3. (Left:) Measurement distribution for an expectation of 1.0m. (Right:) Measurement distribution for an expectation of 2.5m. It is obviously, the variance of these sonar sensors is higher for long ranges.

range for one expectation and is applied to each of the sensors, implicitly assuming that the behaviour of all sensors is identical.

The sensor model is generated from real sensor data in the following way: During the map building as well as other experiments, we usually store all sensor data in a log file. By analyzing the log file, we can assign to each scan the respective robot position it was taken from. With the known grid map, the expected range for each sensor is calculated using the ray casting algorithm mentioned above. The expected/real scan pairs are stored in a table. To this purpose, the range interval 0 - 5 meters (beyond 5 meters scans are too unreliable, hence not used for localization at all) is discretized in segments of 0.05 meters width. Each table row refers to a reference scan range, each column to a real scan range. For each reference/scan pair, the corresponding cell is increased by 1. That way, each table row represents the absolute frequency of real scan ranges for a specific range expectation (Fig. 3). A probability table is generated afterwards by normalizing the rows, dividing by the absolute row sum. Since the model was generated from a large number of scans all over the application area, it is a good approximation of the sonar sensor behaviour in that specific home store environment.

An alternative here would be the approximation of the sensor model by a parametric model, e.g. a mixture of gaussians or a related function. However, that would raise the question what function suits the model best and what are the respective parameters, with the risk of discarding relevant information due to a too simple model. Instead, the table model contains all relevant information and is reasonably small in terms of memory requirement.

We can now use the sensor model to determine a weight for each particle in the observation update step. To this purpose, an expected observation is calculated at the particle's position. Then, for each sonar sensor the respective probability is found in the normalized model table by reading the value in the row corresponding to the expectation and the column corresponding to the actual range measurement. This provides 24 individual probabilities. In order to assign the particle a weight, those 24 probabilities must be merged into one value. One could argue that the individual range readings are mutually independent and, therefore, the overall probability is the product of all values. However, we found that this overemphasizes outliers and can lead to numerical problems with small particle numbers, as the range of particle weights often becomes very large (many orders of magnitude). Instead, averaging over the individual probabilities to calculate the final particle weight leads to much better robustness in the localization. We are aware that this is a deviation from theory, but the improved results legitimate this procedure in our opinion.

Because the calculation of expected reference scans by ray casting in the grid map is quite complex in terms of computation time and could possibly prevent real-time operation with high particle numbers, one more slight modification is needed: Instead of computing the expected observation anew for each particle in each step, the range measurements are pre-calculated for each valid position in the map. Of course this is only possible at a discrete resolution. We found a resolution of 0.1 meters manageable in terms of memory while not significantly affecting the localization results. Storing all expectations for the area presented here results in 200 MB memory allocation. However, for the more relevant task of position tracking (section 4) it is suitable to only keep scans in a local vicinity of the current estimation, that way a few scans must be calculated in each observation update, but the majority of particles can still operate on the cache.

To prove the correctness of the sensor model and the weight calculation, an empirical global localization experiment was conducted. Figure 4 shows the progress of localization based on 20,000 samples. Initially, the samples are uniformely distributed over the entire area. As the robot moves and gathers observations, the samples concentrate in certain hallways and eventually converge to the true robot position.

It should be noted that with sonar sensors, global localization needs a denser initial particle distribution than with visual features. This results from the fact that for range sensors, the observation changes faster with small position changes, while in the camera image features are still visible from a far distance, so the actual change in the observation is smaller from one position to the next one. On one hand, this leads to a higher theoretical position resolution with sonar, on the other hand it also means particles must not be too wide-spread to avoid missing the globally best matching position and converging to a sub-optimum.

4. COMPARISON OF VISUAL AND SONAR MCL

In this section, we will compare vision-based and sonar-based MCL in terms of localization accuracy. To obtain comparable results, both algorithms were investigated on the same pre-recorded data. To this purpose, we recorded

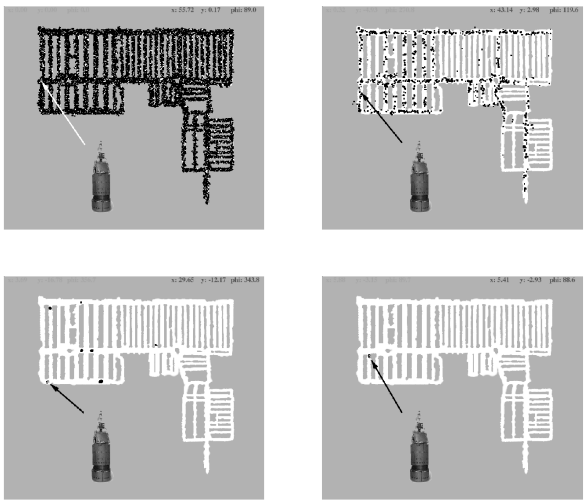

Fig. 4. Global localization with 20,000 samples: the MCL samples are drawn as black points. Initially, samples are distributed uniformly over the entire map area (top left). After the robot has obtained a few observations, samples only remain in the wider hallways (top right). When the robot turns around the corner, only few positions remain as valid hypotheses (bottom left). After a drive distance of about 30m, the robot has resolved all ambiguities and found the correct position as the only possible estimation (bottom right).

a run with a length of approximately 200 meters. By manually marking the robot position at specific reference points and interpolating between these for odometry correction, the absolute robot position (ground truth) is known very well along the entire path. For the visual localization a map was built from additional data that were also recorded at the same day. The sonar MCL used a grid map that was built from different data recorded several months earlier. Since the coarse structure of the environment remains the same except for some small shelves and movable objects, the changes in the grid map usually are much less significant than the changes in the camera view over time.

In contrast to the global localization experiment from the previous section, here we only tested position tracking. This is a simpler but in practice more relevant task, as the robot usually won't just be deployed anywhere, but will have at least some approximate knowledge about its startup position, like a "homezone". Therefore, the MCL particles were initialized as a Gaussian distribution around the known starting point. 1,000 particles were used in the the experiments described in this section. The interval of observation updates was the same for sonar-based and visual MCL.

Along the test path the estimated position was compared to the known reference positions at intervals of about 0.7 meters, resulting in 290 individual position estimations. Because of the partially random nature of Monte Carlo Localization, we executed 10 runs over the same data to receive significant results. The absolute position error μ for sonar-based MCL and vision-based MCL is shown in table 1. Obviously, both methods perform well and the difference in accuracy is quite small. However, we also examined the position error in x and y direction individually: As explained above, the home store environment mainly consists of many parallel hallways. These hallways are all aligned in a certain direction, which is defined as y direction in our coordinate system (Fig. 2), particularly in the top left part where we conducted the MCL experiments described here. From these explanations follows that a position difference in y direction will mostly mean a longitudinal motion along a hallway, while a position difference in x direction corresponds to a lateral motion between the confining walls or shelves, across the corridor.

	Sonar			Visual		
	μ	σ	max	μ	σ	max
absolute error	0.30	0.18	1.06	0.39	0.25	1.18
dx error	0.17	0.15	1.01	0.32	0.25	1.15
dy error	0.22	0.18	0.87	0.16	0.14	0.83

Table 1. Results of sonar-based and visual MCL on the same recorded data. The table shows the difference between the estimated position and the reference position. The first column for both "Sonar" and "Visual" contains the average error over all reference points, the second column is the standard deviation and the third column is the maximum error over the entire test path.

By splitting up the absolute position error into an x (lateral) and y (longitudinal) component, we are able to better evaluate the two localization methods: for the sonar-based approach, the error in y direction is slightly higher than the error in x direction. In contrast to that, for the vision-based method the error in x direction is two times higher than the error in y direction. A more detailed illustration of the localization errors is presented in Fig. 5. These results allow interpretation that with visual localization the robot is able to determine its position along the corridor very well, but it has difficulties finding out if it is close to a wall or in the middle of the corridor. With sonars, on the other hand, of course it is far easier to estimate the distance to the lateral walls, but the position along the corridor is less well defined, as the ranges that are measured by the sensors do not change much over the length of a hallway, providing little information about the true position.

5. COMBINATION OF VISUAL AND SONAR MCL

Now that we know that sonar localization achieves higher accuracies in lateral direction while visual localization leads

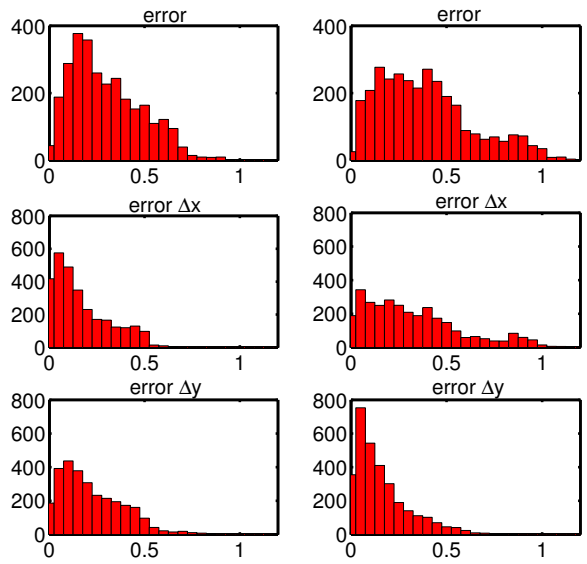

Fig. 5. Error histograms for sonar-based MCL (left) and visual MCL (right). The plots show the histograms for absolute position error (top), position error in x direction (middle) and position error in y direction (bottom) over the entire robot test path (all test points on the path). The histograms give a more precise impression of the error distribution than the values in table 1. In each plot, the total number of points is the same. A stronger concentration on the left side means more small errors, therefore a smaller average error. In contrast, plots stretching out to the right contain more high errors, which corresponds to a higher average (expected) position error. Comparing the left and right plots, it is clearly visible that sonar localization is more accurate in x (corridor lateral) direction, while visual localization behaves better in y (longitudinal) direction.

to higher accuracy in longitudinal direction, an obvious next step is to combine both sensors in order to benefit from the advantages of both, resulting in high position accuracy in both directions.

Therefore, we must find a way to integrate the position distributions estimated by the sonar-based and vision-based MCL into one common distribution. Luckily, this is very easy with particle filters: Each of the methods calculates a fit value for each particle in the observation update step, which is then used as weight in the resampling process. Instead of using just one fit value, we can now multiply the values from sonar and vision and use the product as resampling weight. Since each fit value represents the (relative) probability of the observation at the respective position, and sonar and vision observations are mutually independent, the product of the fit values effectively is the (relative) probability of the combination of both observations. Furthermore, with multiplication (in contrast to e.g. averaging) there is no need to pay attention to the relative magnitude of the individual fit values.

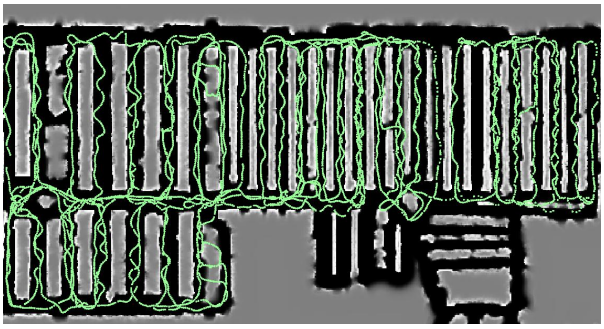

Fig. 6. An example of a grid map overlayed with a visual map. Only a part of the entire area is displayed here.

Overall, the changes from MCL with one sensor to multi-sensor MCL are marginal: instead of one map there are now two maps (which share a common reference coordinate system, see Fig 6). The motion model and the motion update step remain unchanged, as well as the particle initialization. In the observation update, the fit values for both observations are calculated and multiplied. The resulting weight is used in the resampling step in the usual way. Actually, it is possible to integrate an arbitrary number of sensors, as long as a map (environment model) exists for each sensor with a common reference frame.

We repeated the experiment from section 4 with the Multi-Sensor MCL, again with 1000 particles. The results are shown in table 2 and in Fig. 7. It is clearly visible that not only the absolute error is 50% smaller than with only one sensor, but also in x as well as in y direction the error is smaller than the best single sensor method for each direction. This confirms our expectation that with the combination of the two sensor modalities, the resulting localization method should benefit from the strengths of both and suppress the individual weaknesses.

	Vision + Sonar		
	μ	σ	max
absolute error	0.24	0.17	0.83
dx error	0.16	0.14	0.60
dy error	0.15	0.13	0.77

Table 2. Results of the combination of sonar-based and vision-based MCL. The average error in x as well as in y direction is smaller than the minimum of sonar and visual MCL alone (see Table 1).

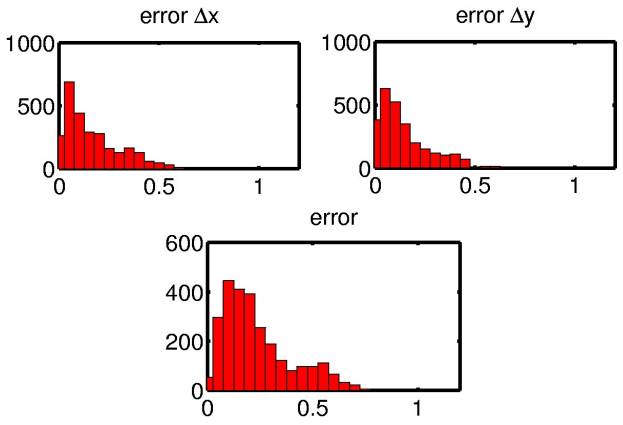

Fig. 7. Error histograms for the combination of sonar-based and vision-based MCL. Clearly, the histogram values are concentrated stronger on the left side than in the individual plots in Fig. 5 indicating smaller localization errors.

6. CONCLUSIONS

We have presented two different MCL approaches, one using omni-camera images for environment observations, the other one using sonar range sensors. We tested and compared both methods in a localization task taken from our home store service scenario and found that both perform reasonably well on their own, with each one having its specific strength and weakness. We then developed a simple and very intuitive approach for combination of omni-camera observations and range measurements. The results show that the overall accuracy can be significantly improved and the sensors complement each other very well, each one contributing its own strengths.

We plan to further expand the scope of our approach by closer examination of situations where the sensors strongly diverge in the localization or even one sensor fails, like high occlusion due to crowded environments or radical lighting changes. Our aim is to identify such situations and temporarily ignore the failing sensor. The inclusion of other low-cost sensors is also considered in order to improve robustness and accuracy of the localization.

7. REFERENCES

[1] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proc. 1999 IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1999.

[2] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," in *Proc. AAAI Natl. Conf. on Artifical Intelligence*, 1999.

[3] F. Dellaert, W. Burgard, D. Fox, and S. Thrun, "Using the condensation algorithm for robust, vision-based mobile robot localization," in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'99)*, 1999, pp. 2588–2596.

[4] H.-M. Gross, A. Koenig, H.-J. Boehme, and C. Schroeter, "Vision-based monte carlo self-localization for a mobile service robot acting as shopping assistant in a home store.," in *Proc. 2002 IEEE/RSJ Intl. Conf. on Intelligent Robots and System*, 2002, pp. 265–262.

[5] H.-M. Gross, A. Koenig, C. Schroeter, and H.-J. Boehme, "Omnivision-based probalistic self-localization for a mobile shopping assistant continued.," in *Proc. 2003 IEEE/RSJ Intl. Conf. on Intelligent Robots and System*, 2003, pp. 1505–1511.

[6] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro, "Image-based monte carlo localization with omnidirectional images," *Robotics and Autonomous Systems*, vol. 48, pp. 17–30, 2004.

[7] M. Jogan and A. Leonardis, "Robust localization using panoramic view-based recognition," in *Proc. 15th Int. Conf. Pattern Recogn. (ICPR'00)*, 2000, pp. 136–139.

[8] N. Winters, J. Gaspar, G. Lacey, and J. Santos-Victor, "Omnidirectional vision for robot navigation," in *IEEE Workshop on Omnidirectional Vision (OMNIVIS'00)*, 2000, pp. 21–28.

[9] J. Neira, J.D. Tardos, J. Horn, and G. Schmidt, "Fusing range and intensity images for mobile robot localization," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 1, pp. 76–84, 1999.

[10] K.O. Arras, N. Tomatis, B.T. Jensen, and R. Siegwart, "Multisensor on-the-fly localization: Precision and reliability for applications," *Robotics & Autonomous Systems*, vol. 34, no. 23, pp. 131 – 143, 2001.

[11] H. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI Magazine*, vol. 9, no. 2, pp. 61–77, 1988.

[12] C. Schroeter, H.-J. Boehme, and H.-M. Gross, "Robust map building for an autonomous robot using low-cost sensors," in *Proc. of the 2004 IEEE Conf. Systems, Man & Cybernetics (SMC2004)*, 2004, pp. 5398 – 5403.