Appearance-based Concurrent Map Building and Localization

J.M. Porta [*] B.J.A. Kröse

IAS Group, Informatics Institute, University of Amsterdam
Kruislaan 403, 1098SJ, Amsterdam, The Netherlands
{porta,krose}@science.uva.nl

Abstract

Appearance-based autonomous robot localization has some advantages over landmark-based localization as, for instance, the simplicity of the processes applied to the sensor readings. The main drawback of appearance-based localization is that it requires a map including images taken at known positions in the environment where the robot is expected to move. In this paper, we describe a concurrent map-building and localization (CML) system developed within the appearance-base robot localization paradigm. This allow us to combine the good features of appearance-base localization without having to deal with its inconveniences.

Key words: Appearance-based localization, concurrent map building and localization.

1 Introduction

Robot localization methods can be divided in two families: methods based on landmark extraction and tracking [8,14,15], and methods based on an appearance modeling of the environment [6,12,13].

A comparison between the two families of localization methods using vision as sensory input can be found in [20], showing that appearance-based methods are more robust to noise, certain type of occlusions, and changes in illumination (when a edge detector is used to pre-process the images) than landmark based-methods. The main drawback of appearance-based methods is that they

[*] Corresponding author: J.M. Porta, *tel* +31 20 5257461, *fax* +31 20 5257490, *e-mail:* porta@science.uva.nl

rely on a data base of position-observations pairs (called the *appearance-map*) and, therefore, localization in only possible in previously mapped areas. The construction of an appearance map is a supervised process that can be quite time-consuming and that is only valid as far as no important modifications of the environment occur. While much work has been done on Concurrent Mapping and Localization (CML) using landmarks [7,9,22], this is not the case within the appearance-based approach. Recent work in this line [19] does not exploit all the potential of the appearance based framework as, for instance, the ability to perform global localization (i.e., the localization without any prior information on the robot's position).

In this paper, we replace the supervised map of the environment used in appearance-based localization by an approximation to it autonomously obtained by the robot. The basic idea we exploit is that, if the robot re-visits an already explored area, it can use the information previously stored to reduce the uncertainty on its position. Additionally, the improvements on the robot's position can be back-propagated to map points stored in previous time slices using trajectory reconstruction techniques. The result is a correction of both the robot's position and the map and, thus, we achieve the objective of concurrently localize and build a correct map of the environment. Similar ideas are exploited in map building based on cyclic trajectories [2,10]. However, these works aim at building a geometric map of the environment and not an appearance-based one.

In next sections, we first describe how to estimate the position of the robot (assuming we have a map). After that, we describe how to extract features from the input images and how to on-line approximate the feature-based map necessary for localization. Finally, we show the results obtained with the new CML system, and we summarize our work and we extract some conclusions out of it.

2 Robot Position Estimation

The probabilistic localization methods aim at improving the estimation of the pose (position and orientation) of the robot at time t, x_t, taking into account the movements of the robot $\{u_1, \ldots, u_t\}$ and the observations of the environment taken by the robot $\{y_1, \ldots, y_t\}$ up to that time[1]. The Markov assumption states that the robot's position can be updated from the previous state probability $p(x_{t-1})$, the last executed action u_t, and the current observation

[1] In our notation, the Markov process goes through the following sequence $x_0 \xrightarrow{u_1} (x_1, y_1) \xrightarrow{u_2} \ldots \xrightarrow{u_t} (x_t, y_t)$.

y_t. Applying Bayes, $p(x_t|u_t, y_t)$ reads to

$$p(x_t|u_t, y_t) \propto p(y_t|x_t)\, p(x_t|u_t), \tag{1}$$

where the probability $p(x_t|u_t)$ can be computed propagating from $p(x_{t-1}|u_{t-1}, y_{t-1})$ using the action model

$$p(x_t|u_t) = \int p(x_t|u_t, x_{t-1})\, p(x_{t-1}|u_{t-1}, y_{t-1})\, dx_{t-1}. \tag{2}$$

Equations 1 and 2 define a recursive system to estimate the position of the robot.

To compute the integral in equation 2 we have to take some assumption on the representation of $p(x_{t-1}|u_{t-1}, y_{t-1})$. Sometimes this probability is represented as a Gaussian[14], but is is a rather restrictive assumption on the shape of $p(x_{t-1}|u_{t-1}, y_{t-1})$. When a probabilistic occupancy grid [3,22] or a particle filter [17,24] is used, we can represent probabilities with any shape. However, occupancy grids and particle filters are computationally expensive in memory and in execution time per time slice.

In our work, we use a Gaussian Mixture (GM) X_t with parameters $\{x_t^i, \Sigma_t^i, w_t^i\}$, $i \in [1, N]$ to represent the position of the robot since, as noted by many authors [1,5,11] GM provide a good trade off between flexibility and efficiency. Thus

$$p(x_t|u_t, y_t) \propto \sum_{i=1}^{N} w_t^i\, \phi(x_t|x_t^i, \Sigma_t^i),$$

with $\phi(x_t|x_t^i, \Sigma_t^i)$ a Gaussian centered at x_t^i and with covariance matrix Σ_t^i. The weight w_t^i ($0 < w_t^i \leq 1$) provides information on the certainty of the hypothesis represented by the corresponding Gaussian.

The motion of the robot is modeled as $x_t = f(x_{t-1}, u_t, v_t)$, with v_t a Gaussian noise with zero mean and covariance Q. Thus, using a linear approximation, we can express $p(x_t|u_t)$ in equation 2 as a GM $X_{u_t} = \{x_{u_t}^i, \Sigma_{u_t}^i, w_t^i\}$, $i \in [1, N]$ applying f to the elements on X_{t-1}

$$\begin{aligned} x_{u_t}^i &= f(x_{t-1}^i, u_t), \\ \Sigma_{u_t}^i &= F \Sigma_{t-1}^i F^\top + G Q G^\top, \end{aligned} \tag{3}$$

with F the Jacobian of f with respect to x_{t-1}^i, and G the Jacobian of f with respect to v_{t-1}.

After we have $p(x_t|u_t)$, we need to integrate the information provided by the sensor readings (see equation 1). At this point, we assume we have an appearance map of the environment from which we can define $p(y_t|x_t)$. Using the reasoning presented by Vlassis *et al* in [24]-sec 5.2, this probability can be represented using a GM with parameters $X_{y_t} = \{(x_{y_t}^j, \Sigma_{y_t}^j, w_{y_t}^j)\}$, $j \in [1, N']$.

In the next section, we describe how to create and update the set X_{y_t}. If X_{y_t} has no components ($N' = 0$), the estimation on the robot's position obtained applying equation 3 can not be improved and we have $X_t = X_{u_t}$. If $N' > 0$, we have to fuse the Gaussian functions in X_{u_t} with those in X_{y_t}. The direct application of equation 1 amounts to multiply each one of the elements in X_{u_t} with those in X_{y_t}. This would produce a quadratic ($N \times N'$) number of hypotheses. Too keep the number of hypotheses under a reasonable limit, we will only associate elements of X_{u_t} and X_{y_t} that are alternative approximations of the same positioning hypothesis. This arises the problem of *data association*: to determine which elements on X_{y_t} and on X_{u_t} to be combined. We perform the data association using an innovation-based criterion. For each couple (i, j) with $(x^i_{u_t}, \Sigma^i_{u_t}, w^i_t) \in X_{u_t}$ and $(x^j_{y_t}, \Sigma^j_{y_t}, w^j_{y_t}) \in X_{y_t}$ we compute the innovation as

$$\begin{aligned} \upsilon_{i,j} &= x^i_{u_t} - x^j_{y_t} \\ S_{i,j} &= \Sigma^i_{u_t} + \Sigma^j_{y_t}, \end{aligned}$$

and we assume that hypotheses on the robot position i and sensor reading j match if the following condition holds

$$\upsilon_{i,j} S^{-1}_{i,j} \upsilon^\top_{i,j} \leq \gamma, \tag{4}$$

with γ a user-defined threshold.

If there is a match, the update of positioning hypothesis i with the sensor information j is done using the *Covariance Intersection* rule [23], since it permits estimation even with unmodeled correlations in the sensor readings. This rule updates the covariance matrix and the average as

$$\begin{aligned} (\Sigma^i_t)^{-1} &= (1-\omega)(\Sigma^i_{u_t})^{-1} + \omega(\Sigma^j_{y_t})^{-1} \\ x^i_t &= \Sigma^i_t[(1-\omega)(\Sigma^i_{u_t})^{-1} x^i_{u_t} + \omega(\Sigma^j_{y_t})^{-1} x^j_{y_t}], \end{aligned} \tag{5}$$

with $\omega = |\Sigma^i_{u_t}|/(|\Sigma^i_{u_t}| + |\Sigma^j_{y_t}|)$.

Hypotheses on the state not matched with any Gaussian on X_{y_t} are just keep without any modification, but the weight update described below. Sensor components not matched with any state hypothesis have to be introduced as new hypotheses on X_t. This allow us to deal with the kidnap problem.

The confidence on each hypothesis in X_t is represented by the corresponding weight w^i_t. Following [21], we increase w^i_t for the hypothesis that are properly matched in the *data association* process

$$w^i_{t+1} = w^i_t + \alpha(1 - w^i_t), \tag{6}$$

with α a learning rate in the range $[0, 1]$. For the not matched hypothesis, the confidence is decreased as

$$w_{t+1}^i = (1-\alpha)w_t^i. \tag{7}$$

Hypotheses with a weight below a given threshold (0.01 in our test) are removed from X_t.

3 Image Feature Extraction

Our sensory input for localization are images taken by the camera mounted on the robot. A problem with images is their high dimensionality, resulting in large storage requirements and high computational demands. For this reason, in appearance-base modeling [16] images z are compressed to few features y using a linear projection, $y = W\ z$. The projection matrix W is obtained by Principal Component Analysis on a supervised training set. However, in our case, we don't have a training set. For this reason, we use a standard linear compression technique: the *discrete cosine transform* (DCT) [4,18]. We select this transform since, PCA on natural images approaches the discrete cosine transform in the limit. The DCT computes features using a $d \times n$ projection matrix W defined as

$$W_{j,k} = z_k\ \cos\left(\frac{\pi}{n} j(k-1/2)\right) \tag{8}$$

with j the number of the feature to be extracted, z_k the k-th pixel of image z (considering the image as a vector) and n the total number of pixels in the image. For a given image we compute a set of d features (d is typically around 10).

4 Environment Mapping

The manifold of features y can be seen as a function of the pose of the robot x, $y = g(x)$. The objective of a appearance-based mapping is to approximate g^{-1} since this gives us information about the possible positions of the robot given a set of features.

Using the localization system introduced previous section, at a given time, we have a pair (X_t, y_t) with y_t a set of features and X_t the estimation of the position of the robot. We can use the sequence of such pairs obtained as the robot moves as a first approximation to the map needed for localization. Thus, our map can be initially defined as $M = \{(X_{y_t}, y_t)\}$, with $X_{y_t} = X_t$.

As explained in Section 2, when the robot re-observes a given set of features y_t, we can use X_{y_t} to improve the location of the robot. Due to noise a given observation is never *exactly* re-observed. So, we considerer that two observations y and y' are equivalent if

$$\|y - y'\| < \delta. \tag{9}$$

Thus, if y $(\pm\delta)$ is re-observed, we can use an old estimation on the robot's position to improve the current one. However, we can also use the information the other way around, we can improve X_{y_t} using the additional information provided by the current X_{u_t}. What we need to do is to introduce new information into a given GM. Therefore, we can improve X_{y_t} using the procedure described in Section 2, but with the roles swapped: in previous section we update the position of the robot using the sensor information and now we adjust the map using the information provided by the position of the robot. So, we have to swap X_{u_t} with X_{y_t}, x_{u_t} with x_{y_t}, and i with j. The only difference is that we assume the environment to be static and, thus X_{y_t} is not updated by any action model.

At a given moment, the robot is at a single position. So, when the state X_{u_t} includes more than one hypothesis, we are uncertain about the robot's location and, consequently, any map update using X_{u_t} will include incorrect map modifications that we will need to undo later on. To avoid this problem, we use a conservative strategy in the map update: when X_{u_t} is not a single hypothesis (i.e., a single Gaussian) no map update is done. If X_{u_t} is a set of Gaussian functions (or a uniform distribution) and, due to the state update (equation 5) or the weight update (equation 7), it becomes a single Gaussian, we use the path followed by the robot and the backward action model to add new points to the map corresponding to the time slices where the robot was unsure about its location (and that now we now can unambiguously determine). Additionally, if X_t is a single Gaussian and the state update results in a reduction of its covariance, we backpropagate this error reduction to previous map points. This backpropagation is performed applying the action model back along the path followed by the robot and using the normal state update procedure, while the update reduces the covariance of the map points. The result of these two backpropagation process is to add new points to the map (i.e., to extend the map) and to improve the estimation of previously stored map points (i.e., to improve the quality of the map).

The mapping strategy just outlined allow us to build an appearance-based map of a given environment along the paths followed by the robot. When the map M is empty, we need to now the initial position of the robot, but when M has some elements, global localization is possible. To perform global localization we have to initialize the system with X_0 containing a uniform distribution. As soon as the robot observes features already stored in the map (i.e., it visits an already explored area) new hypotheses are added to X_t using the data stored in the corresponding map points. Eventually, one of these new hypotheses

```
Multi Hypotheses CML(M):
  Input: M, the map.
      If M is empty $X \leftarrow \{(x = 0, \Sigma = 0, w = 1)\}$.
      else $X \leftarrow$ Uniform distribution.
Do forever:
  Update $X$ according to the action model (Eqs. 3).
  Get an image $z$.
  Compute the features $y$ of $z$ using the DCT (Eq. 8).
  $(y, Y) \leftarrow arg\min_{\forall (y',Y') \in M} \|y - y'\|$
  if $\|y - y'\| > \delta$ then
      $M \leftarrow M \cup \{X, y\}$
  else
      Associate elements in $X$ and $Y$ (Eq. 4).
      Update $X$:
          For the associated elements use Eqs. 5, 6.
          For the non-associated decrease weight (Eq. 7).
          Remove elements in $X$ with too low weight.
      if $X$ is a single Gaussian ($x$)
          if $x$ associated with $y \in Y$
              Update $y$ using Eqs. 5, 6.
          else
              Add $x$ to $Y$.
          Decrease weight for $y' \in Y, y' \neq y$ (Eq. 7).
          Remove elements in $Y$ with too low weight.
```

Fig. 1. The multi hypothesis tracking, appearance-based CML algorithm.

will becomes relevant enough to be considered as the correct position of the robot. At this point the rest of tentative hypothesis are removed, and the new information on the robot position is backpropagated adding points to the map.

Figure 1 summarizes the CML algorithm we introduce in this paper.

5 Experiments and Results

We first tested the proposed CML system mapping a corridor 25 meters long. We drive the robot using a joystick all along the corridor and back to the origin. Figure 2-A shows the path followed by the robot according to odometric information. As we can see there is a large error in the final position of the robot (about 40 cm in the X axis, 300 cm in the Y, and 20 degrees in orientation). In the figure, each one of the arrows correspond to a robot's pose where a map point is stored.

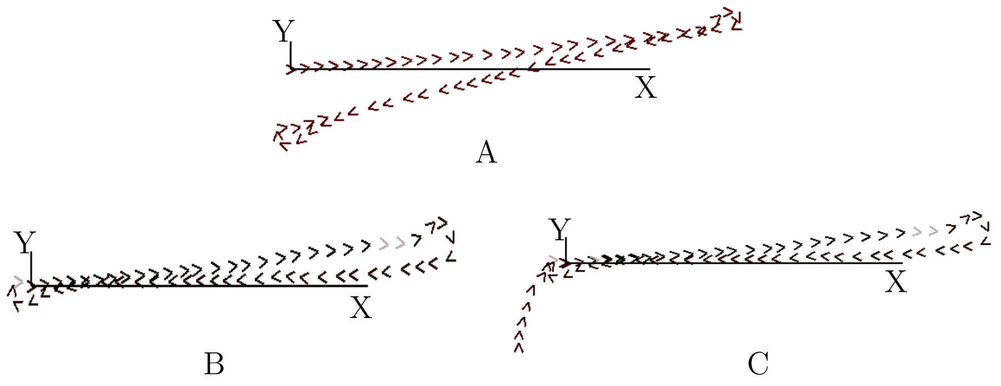

Fig. 2. Path of the robot in a 25 meter corridor. A- Using only odometry, B- The map is corrected when the loop is closed C- Adding new portions the map (to do that the robot performs global localization).

Our CML system detects that the final point in the trajectory is close to the initial one (see equation 9). This coincidence allows a drastic reduction on the uncertainty on the robot's location and this reduction can be back-propagated improving the quality of the map. Figure 2-B show the corrected map points after the loop is closed. We can see that the correction affects mainly to the points close to the end of the trajectory, where the back-propagated information is more certainty that the previously stored one: close to the beginning of the trajectory the initial information is more reliable than the back-propagated one and the map points are not modified. In Figure 2-B, the light grey arrows correspond to poses where there is perceptual aliasing: the *same* set of features are observed from all the positions plotted in red. Perceptual aliasing is one of the reasons why we need to keep track of many hypotheses simultaneously.

Another situation where multiple hypothesis should be considered is when performing global localization: when the robot is started at a unknown position (but in a previously mapped area) or after a kidnap. In the experiment reported in Figure 2-C, we started the robot in a non-mapped area and we drive it to the beginning of the corridor previously mapped. In this case, the robot operates with a uniform distribution about its position. When consistent matches with already existing map points occur, the robot determines its position, the uniform distribution is replaced by a Gaussian defined from the map and new map points along the robot path are added.

Next experiment was aimed to test the performance of the CML system in front of large errors in odometry, We manually drive the robot along a closed circuit, starting the robot at $O = (0, 0)$ pointing to the X positive axis (see Figure 3-left). The first loop is used to initialize the map. In the second iteration, an large error in odometry was artificially induced by slightly lifting and rotating the robot clockwise when it is at position $D = (0, 0.75)$. This makes the information provided by odometry invalid and the position of the robot must be estimated by exploiting the map built in the first loop. Figure 3-right

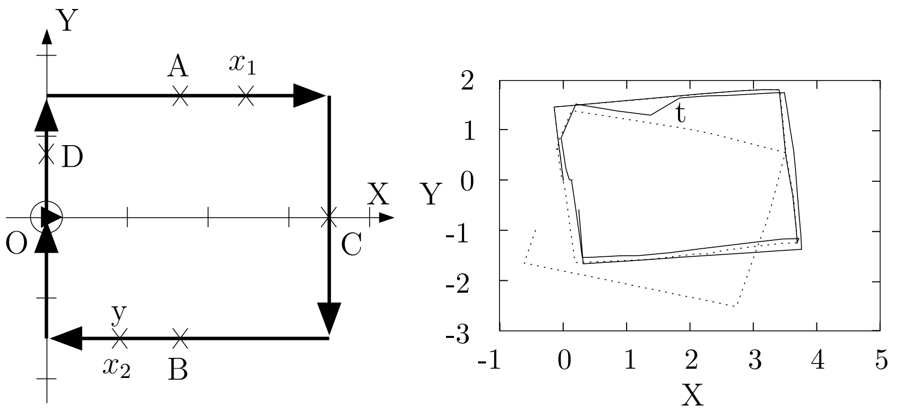

Fig. 3. Left: The circuit used for the kidnap experiment Bottom: Robot's trace according to the CML system (solid line) and according to odometry (dashed line). Units are in meters.

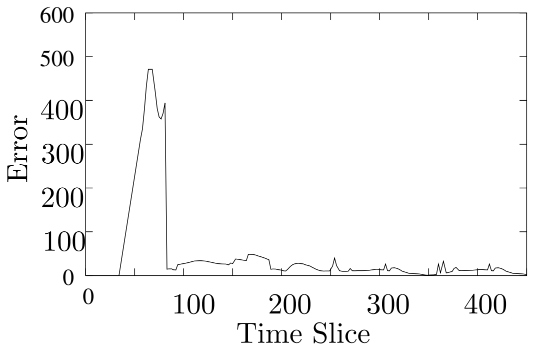

Fig. 4. Error in localization per time slice when the robot is kidnapped at $t = 50$.

shows the path of the robot according to odometry (dotted line) and according to our localization method (solid line). After the introduction of the odometry error at point D, matches of the images taken by the robot and those in the map, result in the introduction of a new hypothesis on the robot's position. This new hypothesis is reinforced as the robot gets more images while the weight of the hypothesis supported by odometry slowly decreases (equation 7). At time t, the odometry related hypothesis is finally abandoned and the one supported by the map becomes the estimation on the robot's position.

In the last experiment, we tested the ability of our system to deal with the kidnap problem, even while the map is in early phases of its construction. We move the robot in the circuit of Figure 3-left. At the first loop, at position A the robot is kidnapped: lifted and displaced to position B and rotated 180°. Thus, according to odometry the robot moves from A to C while it is actually is moving from B to O. Due to the kidnap observations initially assigned to points in the path from A to C are actually occurring in the path from B to O. When the robot gets to O a new hypothesis (the correct one) is introduced into the state. This new hypothesis is reinforced by the observations and, after few time slices, it becomes the most relevant one, the wrong hypothesis is removed and, in the following iterations over the circuit, the map is corrected. We can use an example to show how the map correction works. Initially, the incorrect association $\{(x_1, \Sigma_1, w_1 = 1), y\}$ is stored in the map. In the second loop, this map point is updated to $\{(x_1, \Sigma_1, w_1 = 1), (x_2, \Sigma_2, w_2 = 1), y\}$. In the following iterations, the weight of the association (x_1, y) decreases since it is not observed again while the correct association (x_2, y) is reinforced. After few loops, the association (x_1, y) is eventually removed from the map. Figure 4 shows the error in localization as the robot moves around the circuit. The large errors at time slices $40 - 90$ are caused by the initial kidnap but, as the robot gets to O again, the error is canceled and is maintained low. Every loop around the circuit takes 100 time slices, but the first one that is sorter due to the kidnap.

6 Discussion and Future Work

We have introduced a system that is able to simultaneously build an appearance-map of the environment and to use this map, still under construction, to improve the localization of the robot. The on-line construction and update of the map allow us to overcome the major hurdles of traditional appearance-based localization. First, the robot can operate in previously unknown areas. Second, we can deal with changes in the environment: new observations obtained at already explored positions are added to the map, the old observations at those position are not used any more and they are slowly forgotten. Finally, the way in which the map is built guarantees a uniform sampling of the feature space and not of the geometric space, as it happens in normal appearance-based localization. Sampling uniformly the feature space is essential for achieving a good localization since the sensor model is based on the similarities (i.e., the distances) in that space.

An implicit assumption in our mapping strategy is that the robot moves repetitively through the same areas/paths. However, this is a quite reasonable assumption for service robots moving in relatively small offices or houses, that are the kind of environments in which we plan to use our system.

Observe that the proposed CML approach does not provides an exact position for the robot, but an approximation to it. However, this kind of rough information on the robot position is enough for most tasks, assuming that the low level behaviors of the robot controller are able to deal with local aspects of the environment (obstacles, doors, etc). Due to the way in which we define the map, the map error will be small close to the areas where the map is started and growing for points far away from the origin. Actually, the error for a given map point p is is lower bounded by the error in odometry for a direct displacement from the origin of the map O to p. As a reference, our Nomad Scout robot can map an area of 20×20 meters with an accuracy below 1 meter. Since the error in odometry limits the area that we can map with a given accuracy, we would like to complement our localization system with additional odometric sensors (accelerometers, vision-based motion detection, etc) to determine more accurately the relative displacements of the robot. Another solution to enlarge the mapped area is to perform the CML in contiguous areas and, then, integrate the resulting sub-maps.

References

[1] K. O. Arras, J. A. Castellanos, and R. Siegwart. Feature-Based Multi-Hypothesis Localization and Tracking for Mobile Robots Using Geometric Constraints. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1371–1377, 2002.

[2] H. Baltzakis and P. Trahanias. Closing Multiple Loops while Mapping Features in Cyclic Environments. In *Proceedings of the International Conference on Robotics and Intelligent Systems (IROS), Las Vegas, USA*, pages 717–722, 2003.

[3] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the Absolute Position of a Mobile Robot using Position Probability Grids. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 896–901, 1996.

[4] R.J. Clarke. *Digital Compression of Still Images and Video.* Academic Press, 1995.

[5] I.J. Cox and J.J. Leonard. Modeling a Dynamic Environment Using a Multiple Hypothesis Approach. *Artificial Intelligence*, 66(2):311–344, 1994.

[6] J. Crowley, F. Wallner, and B. Schiele. Position Estimation Using Principal Components of Range Data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3121–3128, 1998.

[7] G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Csorba. A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[8] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, 3(3):249–265, 1987.

[9] H.J.S. Feder, J.J. Leonard, and C.M. Smith. Adaptive Mobile Robot Navigation and Mapping. *International Journal of Robotics Research, Special Issue on Field and Service Robotics*, 18(7):650–668, 1999.

[10] J.S. Gutmann and K. Konelige. Incremental Mapping of Large Cyclic Environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA*, pages 318–325, 1999.

[11] P. Jensfelt and S. Kristensen. Active Global Localization for a Mobile Robot Using Multiple Hypothesis Tracking. *IEEE Transactions on Robotics and Automation*, 17(5):748–760, 2001.

[12] M. Jogan and A. Leonardis. Robust Localization using Eigenspace of Spinning-Images. In *Proceedings of the IEEE Workshop on Omnidirectional Vision, Hilton Head Island, South Carolina*, pages 37–44, 2000.

[13] B.J.A. Kröse, N. Vlassis, and R. Bunschoten. Omnidirectional Vision for Appearance-based Robot Localization. *Lecture Notes in Computer Science*, 2238:39–50, 2002.

[14] J.J. Leonard and H.F. Durrant-Whyte. Mobile Robot Localization by Tracking Geometric Beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–382, 1991.

[15] H.P. Moravec. Sensor Fusion in Certainty Grids for Mobile Robots. *AI Magazine*, 9(2):61–74, 1988.

[16] H. Murase and S.K. Nayar. Visual Learning and Recognition of 3-D Objects from Appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[17] M.K. Pitt and N. Shephard. Filtering Via Simulation: Auxiliary Particle Filters. *J. Amer. Statist. Assoc.*, 94(446):590–599, June 1999.

[18] K.P. Rao and P. Yip. *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press, Boston, 1990.

[19] P. E. Rybski, S.J. Roumeliotis, M. Gini, and N. Papanikolopoulos. Appearance-Based Minimalistic Metric SLAM. In *Proceedings of the International Conference on Robotics and Intelligent Systems (IROS), Las Vegas, USA*, pages 194–199, 2003.

[20] R. Sim and G. Dudek. Comparing Image-based Localization Methods. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI), Acapulco, Mexico*, 2003.

[21] B. Terwijn, J.M. Porta, and B.J.A. Kröse. A Particle Filter to Estimate non-Markovian States. In *Proceedings of the 8th International Conference on Intelligent Autonomous Systems (IAS)*, pages 1062–1069, 2004.

[22] S. Thrun, W. Burgard, and D. Fox. A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. *Machine Learning*, 31:29–53, 1998.

[23] J. Uhlmann, S. Julier, and M. Csorba. Nondivergent Simultaneous Map Building and Localization Using Covariance Intersection. In *Proceedings of the SPIE Aerosense Conference, 3087*, 1997.

[24] N. Vlassis, B. Terwijn, and B.J.A. Kröse. Auxiliary Particle Filter Robot Localization from High-Dimensional Sensor Observations. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Washington D.C., USA*, pages 7–12, May 2002.