Localization for Mobile Robots using Panoramic Vision, Local Features and Particle Filter

Henrik Andreasson*, André Treptow† and Tom Duckett*

* Örebro University
Dept. of Technology
Örebro, Sweden
Email: {henrik.andreasson, tom.duckett}@tech.oru.se

† University of Tübingen
Department of Computer Science WSI-RA
Tübingen, Germany
Email: treptow@informatik.uni-tuebingen.de

***Abstract*— In this paper we present a vision-based approach to self-localization that uses a novel scheme to integrate feature-based matching of panoramic images with Monte Carlo localization. A specially modified version of Lowe's SIFT algorithm is used to match features extracted from local interest points in the image, rather than using global features calculated from the whole image. Experiments conducted in a large, populated indoor environment (up to 5 persons visible) over a period of several months demonstrate the robustness of the approach, including kidnapping and occlusion of up to 90% of the robot's field of view.**

I. Introduction

One of the most basic abilities needed by robots is self-localization ("where am I?"). This can be divided into geometric and topological localization. Geometric localization tries to estimate the position of the robot as accurately as possible, e.g., by calculating a pose estimate (x, y, θ), while topological localization gives a more abstract position estimate, e.g., "I'm in the coffee room". Difficult situations require the robot to be able to relocalize itself from scratch, e.g., when the robot is started (no prior knowledge of its position), or if the robot is lost or "kidnapped" (incorrect prior knowledge). There has been extensive research on using accumulated sensory experience to improve localization performance, including approaches such as Markov localization [7], Particle Filters/Monte-Carlo localization (MCL) [8] or by using a voting scheme as presented in [1].

Panoramic or omni-directional cameras have become popular for self-localization in recent years because of their relatively low cost and large field of view, which makes it possible to create features that are invariant to the robot's orientation, for example, using various colour histograms [1], [2], [3], or Eigenspace models [4]. Approaches that do not use rotationally invariant features create multiple images from the same location by shifting the panoramic view [5] or by rotation [6], which increases the amount of data several times. Another approach is to only take pictures at the same orientation [4], e.g., only when the robot is facing north. Recent work has combined panoramic vision with MCL, including feature matching using the Fourier coefficient [9] and colour histograms [10].

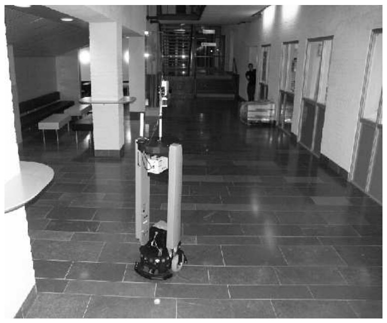
Fig. 1. Robot platform in the test environment.

The main difference in our work from the previous approaches is that we use local features extracted from many small regions of the image rather than global features extracted from the whole image, which makes the method very robust to variations and occlusions, e.g., due to moving persons. We use a version of Lowe's SIFT algorithm [13], which is modified so that stored panoramic images are only recognised from a local area around the corresponding location in the world (Section II). A novel scheme is introduced for combining local feature matching with a Particle Filter for global localization (Section III), which minimizes computational costs as the filter converges. Our experiments were explicitly designed to test the system under a wide variety of conditions, including results in a large populated indoor environment on different days ($\sim$2 month apart) under different lighting conditions (Section IV). The results demonstrate that the robot is able to localize itself from scratch, including experiments in "kidnapping", and that the performance shows a graceful degradation to occlusions (up to 90% of the robot's field of view).

II. Local Feature Matching

To be able to match the current image with the images stored in the database, each image is converted into a set of features. The matching is done by comparing the features in the database with the features created from the current image.

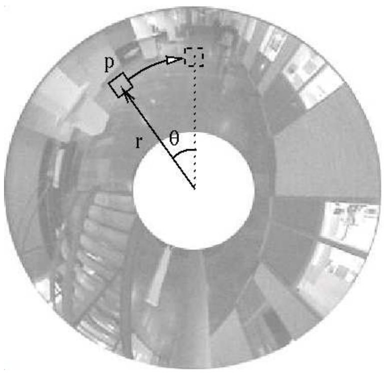

Fig. 2. Creation of the neighbourhood $\mathcal{N}_p$. Before matching, the interest point p and the surrounding pixels are rotated by angle θ to a fixed orientation (facing forwards relative to the robot). This means that matching of two interest points in different images will be independent of the orientation of those points.

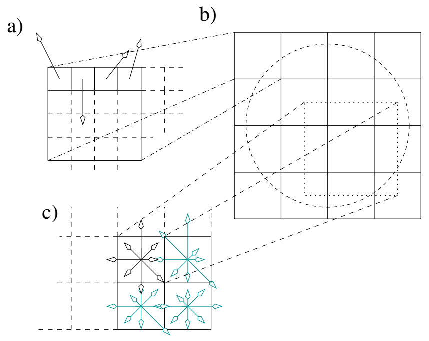

Fig. 3. Creation of MSIFT features. a): Rotation and magnitude for each pixel. The circle in b) represents the Gaussian weighting function. Note that the histograms in c) are created by bilinear interpolation from surrounding pixels represented as a dashed square in b).

A. Selection of Interest Points

To select interest points, a neighbourhood $\mathcal{N}$ of 3x3 pixels is selected around each pixel in the image. The derivatives D_x and D_y are calculated with a Sobel operator for all pixels in the block $\mathcal{N}$. For each pixel the minimum eigenvalue λ is calculated for matrix $\mathcal{A}$ where

$$\mathcal{A} = \begin{bmatrix} \sum D_{x_{i,j}}^2 & \sum D_{x_{i,j}} D_{y_{i,j}} \\ \sum D_{x_{i,j}} D_{y_{i,j}} & \sum D_{y_{i,j}}^2 \end{bmatrix} \qquad (1)$$

and $\sum$ is performed over the neighbourhood of $\mathcal{N}$. The pixels with the highest values of λ are then selected by thresholding. If points are closer than a distance of 5 pixels the weakest point is removed. For further details see [11], or the function `cvGoodFeaturesToTrack` in the OpenCV library [12].

To get rotationally invariant features, the neighbourhood $\mathcal{N}_p$ is created by rotating the surrounding pixels the same angle θ as the interest point p is rotated, see Fig. 2. This is done using bilinear interpolation. By doing this $\mathcal{N}_p$ will be independent of the rotation θ.

In our experiments, 100 interest points were selected for each panoramic image. The diameter of the panoramic view is approximately 576 pixels. For each interest point, the radius r and orientation θ are also stored (Fig. 2) for subsequent processing described in Section III-B.

B. Modified SIFT algorithm (MSIFT)

SIFT was developed by Lowe [13] for local feature generation in object recognition applications. The features are invariant to image translation, scaling and rotation, and partially invariant to illumination changes and affine or 3D projection. These properties make SIFT very suitable for mobile robots because landmarks can be observed from different angles, distances or illumination [14].

However, for the purpose of self localization, we actually do not want full invariance to translation and scale: we would like view matching to be successful only in the vicinity of the location where the original image was recorded in the database. Therefore, we make a number of simplifications to the basic algorithm, described as follows.

To ensure rotational invariance, each interest point is first rotated to the same global orientation as shown in Fig. 2, as described above. (In the full SIFT algorithm, a canonical orientation is stored for each interest point.)

Around each interest point, a sub-window of 32×32 pixels is selected. Image gradients are then calculated for each pixel in the sub-window. This is implemented by first calculating the matrix of derivatives in $x-$ and $y-$ directions, and then converting to polar coordinates. The result is a 32×32 matrix giving the gradient magnitude and orientation for each pixel in the sub-window (Fig. 3a).

The magnitude and orientation values are then accumulated into 16 orientation histograms summarising the contents of the whole sub-window (these histograms are the so-called "keypoint descriptors" or feature vector used for matching interest points). Each histogram has 8 bins, i.e., the orientation information is discretised into 8 possible values. Each bin accumulates the total weighted magnitude information for a particular orientation. A Gaussian function is used to weight the gradient magnitude values in order to avoid sudden changes in the histograms with small changes in the position of the window (Fig. 3b). To avoid boundary effects when gradients are changing from one histogram area to another, each pixel gradient is therefore stored in the four closest histograms using bilinear interpolation (Fig. 3c). (In the full SIFT algorithm, a Gaussian kernel is convolved at multiple resolutions in a pyramid of scaled images [15] to detect interest points at different scales, whereas in our approach we only find interest points in one resolution, since we do not require scale invariance.)

Matching of features in two images in then performed using squared Euclidean distance between the two corresponding histograms. In the full SIFT algorithm, the histograms are first normalised to have unit length. However, we found that localization performance was improved by omitting the normalisation step.

III. Monte Carlo Localization

The use of Monte Carlo methods or Particle Filters [16] became quite popular in the last years to estimate the state of a system at a certain time based on the current and past measurements. The probability $p(X_t|Z_t)$ of a system being in the state X_t given a history of measurements $Z_t = \{z_0, ..., z_t\}$ is approximated by a set of N weighted particles:

$$S_t = \{x_t^{(i)}, \pi_t^{(i)}\},\ i = 1...N. \tag{2}$$

Each particle $x_t^{(i)}$ describes a possible state weighted with $\pi_t^{(i)}$ which is proportional to the likelihood that the system is in this state. Particle Filtering consists of three main steps:

1) Create new particle set S_{t+1} set by resampling from old particle set S_t based on particle weights $\pi_t^{(i)}, i = 1...N$
2) Predict particle states based on dynamic model $p(x_{t+1}^{(i)}|x_t^{(i)}, u_t)$ with odometry u_t, $i = 1...N$
3) Calculate new weights by application of the measurement model: $\pi_{t+1}^{(i)} \propto p(z_{t+1}|X_{t+1} = x_{t+1}^{(i)}), i = 1...N.$

The estimate of the system state at time t is the weighted mean over all particle states:

$$\hat{X}_t = E(S_t) = \sum_{i=1}^{N} \pi_t^{(i)} x_t^{(i)}. \tag{3}$$

The weights $\pi_t^{(i)}$ are normalized so that $\sum_{i=1}^{N} \pi_t^{(i)} = 1$. In our case the state is described by a three dimensional vector $x_t = (x_W, y_W, \theta)_t$ containing the position of the robot (x_W, y_W) and the orientation θ. The x_W and y_W coordinates are initialized randomly within a radius of one meter around a randomly selected database position. The orientation θ is estimated from the closest database position as described in III-B with an added normal distribution with a standard deviation of $\pi/8$ radians. The prediction and measurement of particles are described in the following sections. In the experiments, a total of 500 particles were used and 10% of these were randomly reinitalized to enable relocalization.

A. Dynamic Model

All state variables $x_t^{(i)} = (x_W, y_W, \theta)_t$ are updated from the odometry readings u_t from the robot. To cope with inaccuracy (Fig. 6 shows the odometry from $Run1$ and $Run2$), the odometry values are added with normal distribution. The angle is added with standard deviation of 0.1 radians and the translation with a standard deviation of 2% of the translation length.

B. Measurement Model

To calculate the weight of particles only the database location that is closest to the current particle is used (see also Fig. IV-A). This means that the computation time will decrease as the particles converge around the true location of the robot, since fewer of the features in the database will need to be matched to the current image. Fig. 4 shows the decreasing number of matched database locations against distance travelled after initialization of the Particle Filter.

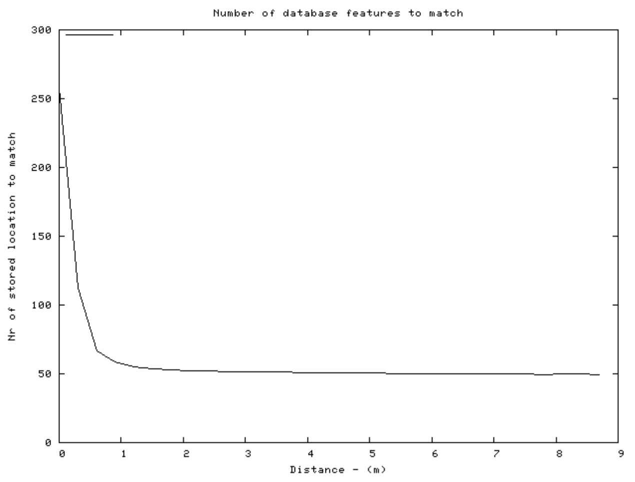

Fig. 4. Number of database features to match against distance travelled for $Run1$ with 50% occlusion.

The weight is based on the number of interest points that matches between the current and the database image features ($match_i$). A candidate interest point match is considered if the lowest match value, calculated from the squared Euclidian distance between the histograms, M_1 is at least 0.6 times smaller than the next lowest match value M_2, the factor that was found empirically and also used in [3]. This guarantees that the interest point match will be significantly better compared to the other possibilities, see also Fig. 5. No interest point is allowed to be matched against more than one other interest point. If an interest point has more then one candidate match, the match which has the lowest match value among the candidate matches is selected.

To reduce the impact of false matches, the position of the matched interest points angles and radius are compared, as well as the homogeneity of the match distribution. To be able to compare the angles θ_p and $\theta_{p'}$ of two matched interest points p and p' (see Fig. 2), the rotation ϕ of the robot between the matched images has to be estimated. ϕ is estimated by creating a 32 bin histogram from $\theta_p - \theta_{p'}$ using all the matched interest points. ϕ is then calculated from the highest bin in the histogram, i.e., the robot orientation can be estimated with a precision of $360/32$ degrees.

For each interest point match where $|\theta_p - \theta_{p'} - \phi| < 0.2$ one extra point is added to the match score ($match_i$). Another point is added to $match_i$ if $|r_p - r_{p'}| < 10$, where the radius is calculated in pixels. Finally $match_i$ is multiplied by 1.5 if more then 60% of the match points pass the angle criteria stated above, in order to award homogeneity. These factors were found empirically.

All particles that are closest to the same database point will have the same match value. To avoid drifting of the particles away from the database positions, a weighting function is used. The final weighted match value $match_i$ is used to calculated the weight by $\pi_t^{(i)} = match_i^3$, in order to reward higher match values with out any need for normalization.

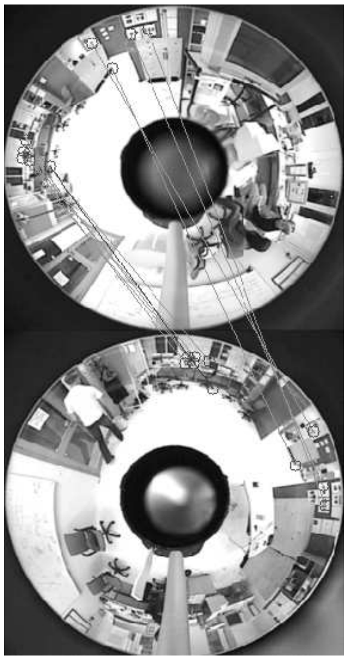

Fig. 5. Matching an image (above) against the database (below).

Finally to increase the inertia of the particles to overcome sections with few matches, only matches that generates high weights compared to previous ones are used. This is done by using a 'forgetting factor' f_{forget}. The new weight is used only if $\pi_{t+1}^{(i)} > \pi_t^{(i)} \cdot f_{forget}$, otherwise $\pi_t^{(i)} \cdot f_{forget}$ is used as the new weight. In the experiments $f_{forget} = 0.9$.

IV. Experiments

The localization system consists of a database of features where each set of features represents one location (place in a topological map). The features were calculated from images taken at the known positions. To obtain these positions and the ground truth data for performance evaluation, a SLAM implementation was applied using the technique described in [17]. A total of 603 images were collected covering an area of approximately 60×55 meters, as shown in Fig. 6. New laser scans and images were recorded if the rotation since the previous image exceeded 15 degrees or when the translation exceeded 0.5 meters. For each image the corresponding pose estimate from the SLAM algorithm was stored.

A. Building the database

Since all the features used are rotationally invariant, it is only necessary to use images with different location and not orientation, i.e., when the robot is travelling back and forward along a corridor it is sufficient to save the data in one direction. The building of the database starts after the run is completed and optimised with SLAM. The images are used in the same order as they were taken. An image is added to the database if the metric distance to the nearest stored image exceeds a threshold T. In this paper, a value of $T = 0.4$ meters was used. For each image that should be a part of the database, a feature set are calculated and stored for the 100 strongest interest points, as described in Section II.

The method were evaluated using a set of test runs that overlap with the area covered by the database, as shown in Fig. 6.

B. Evaluation Method

To calculate performance, the Euclidian distance between positions of the test image and the median value of the 90% of the particles which had the highest fitness value are used to be more robust towards outliers. The database map and the different run maps were manually fitted and 'placed' on top of each other. To get more evaluation data, each dataset was used multiple times by dividing it into smaller runs. The new runs contained 30 images each covering approximate 9 meters where each run has a different starting position. To test the robustness additional levels of occlusion were simulated by removing interest points. The occlusion percentage indicates the proportion of the current image (field of view) where interest points were deleted. For the global localization problem the particles were reinitialized after each completed run (see Fig. 7). To evaluate the kidnapped robot scenario a randomly selected run was used to accumulate knowledge before the robot was 'virtually' moved by randomly selecting another run (see Fig. 8).

C. Results

Run_2 is from a corridor, see Fig. 6, which contains a lot of similar features, e.g., doors to office rooms, and a lack of furniture or objects. Difficulties were also from that a lot of different features were detected if doors were open or closed. This could explain the lower performance compared to Run_1 and Run_4 which mostly take place in the student area and the labs. The student area contains more different features such as staircases and art. The labs are more cluttered with various objects, which tend to move around over time. Run_3 passes parts of the Ph.D. corridor, the coffee room and the secretary offices.

Each run was recorded at a different time compared to the database. Run_1 was taken 2 days before the database, both Run_2 and Run_3 were taken 56 days after while Run_4 was taken 64 days after. Run_4 and most of Run_3 were collected with the robot driving in the opposite direction compared to the database, see Fig. 6.

For $MSIFT$ 16 histograms with 8 directions gave 128 numbers for one feature. To match two images using 100 features takes ∼0.02 second with a 2GHz Pentium.

V. Conclusion

By using experiments with data collected on different days over a period of several months, it has been shown that even if the room has gone through some changes regarding location of furniture, objects and persons, servere occlusion it is possible to extract good position estimates. Future work could be to create a less heuristic method for the matching by using epipolar geometry or similar. Then it would be possible to get more accurate position estimate to initalize the particles.

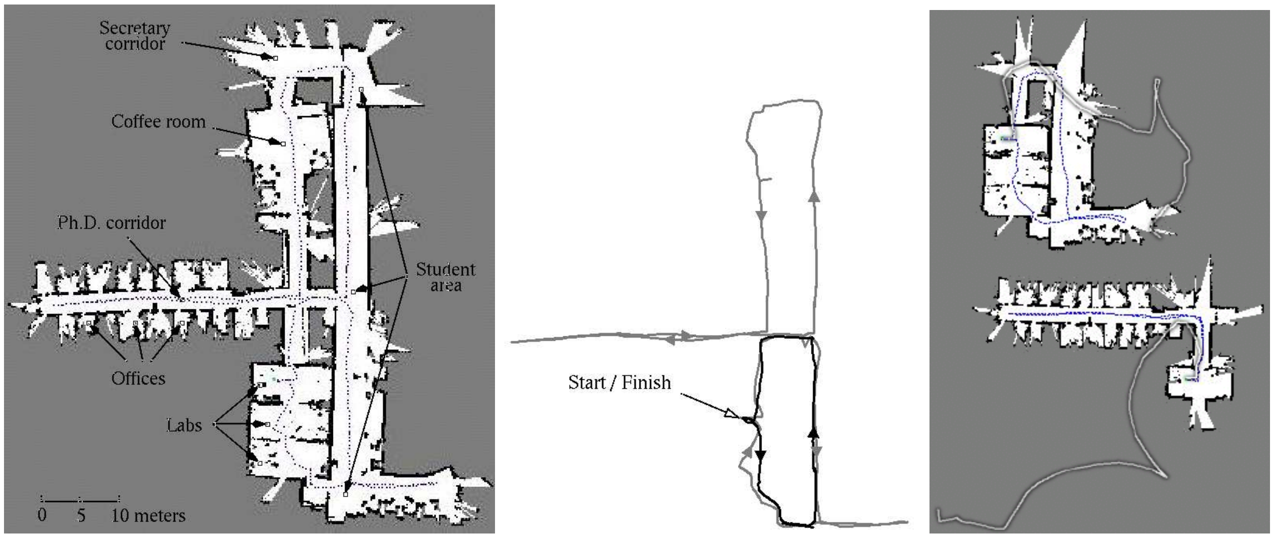

Fig. 6. Left: area covered by the database. Middle: ground truth information from SLAM, Run_4 (black) and the database (gray). Right: Two of the test sequences with ground truth and raw odometry data, Run_1 (above) and Run_2 (below).

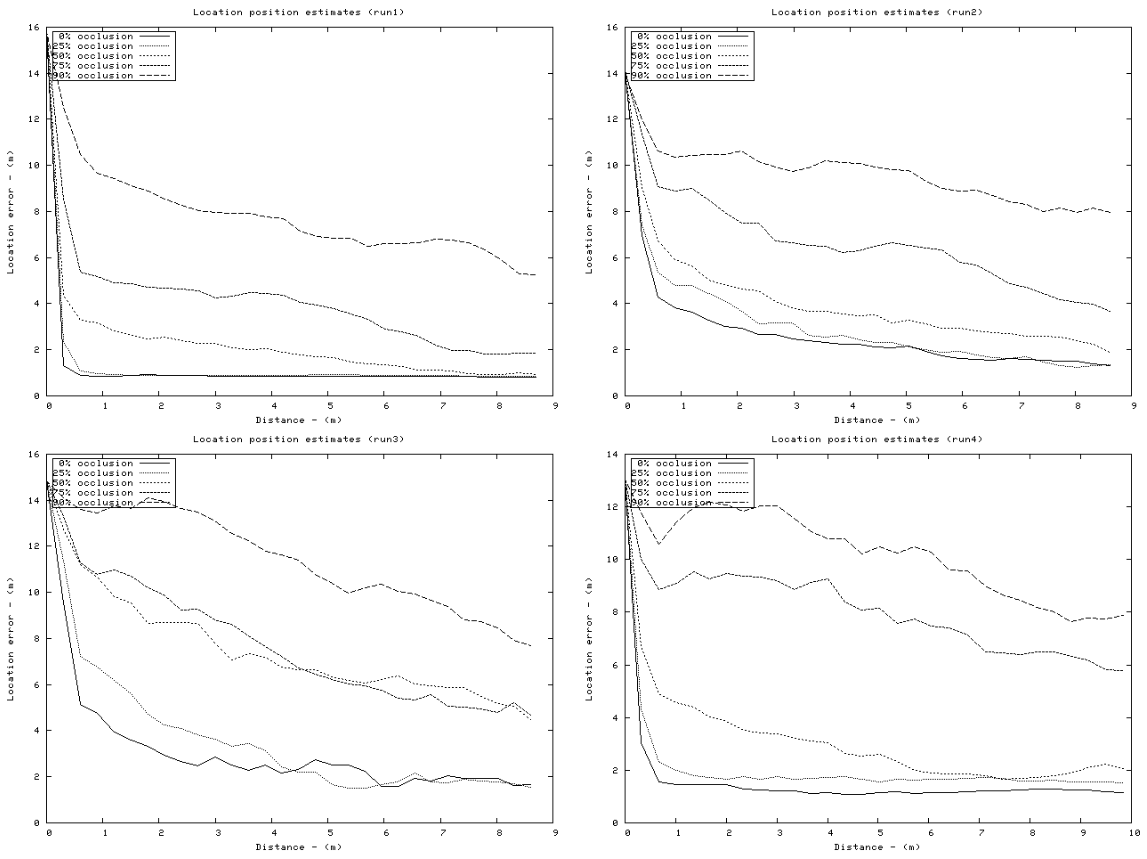

Fig. 7. Localization errors against distance travelled for global localization experiments. Top left: results from $Run1$. Top right: results from $Run2$. Bottom left: results from $Run3$. Bottom right: results from $Run4$.

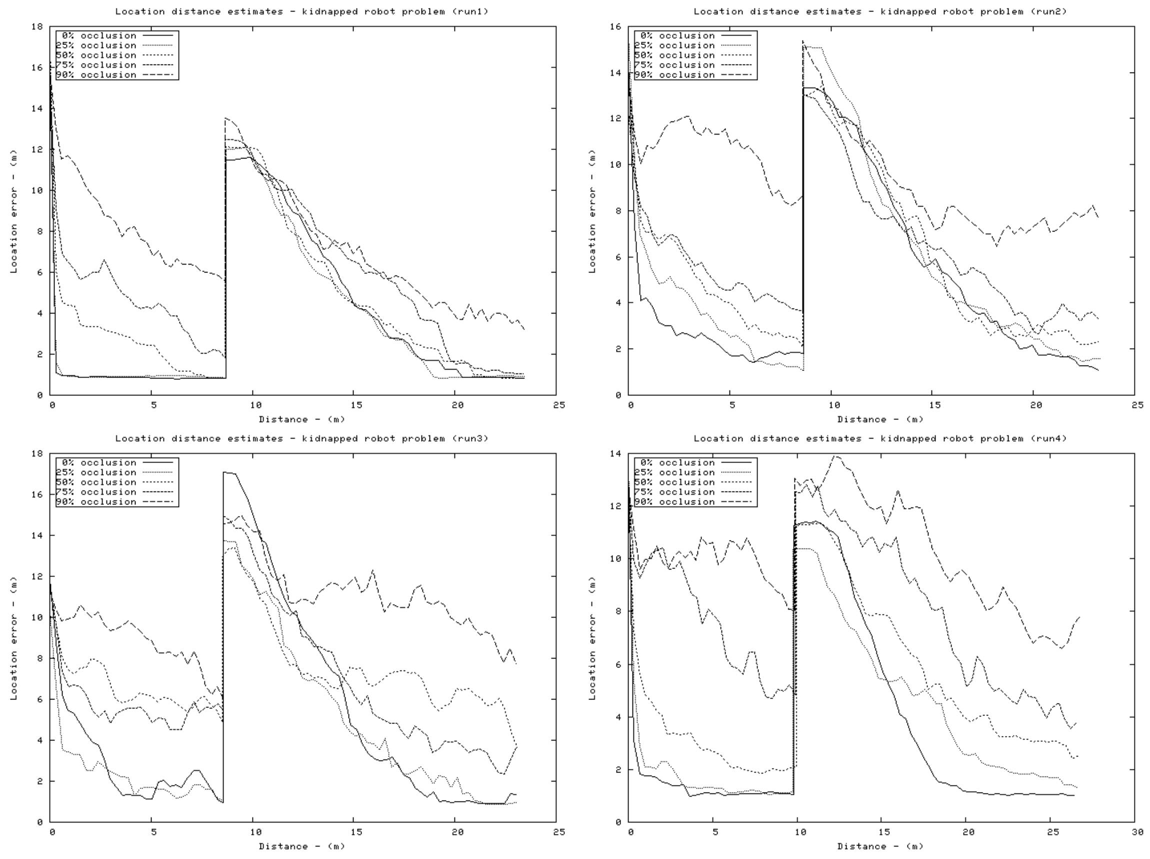

Fig. 8. Localization errors against distance travelled for the kidnapped robot prolem. Top left: results from $Run1$. Top right: results from $Run2$. Bottom left: results from $Run3$. Bottom right: results from $Run4$.

ACKNOWLEDGMENT

The authors would like to thank Per Larsson for creating the client/server application used to teleoperate the robot.

REFERENCES

[1] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *Proc. IEEE Int. Conf. Robotics & Automation (ICRA)*, vol. 2, April 2000, pp. 1023 – 1029.

[2] P. Blaer and P. Allen, "Topological mobile robot localization using fast vision techniques," in *Proc. IEEE Int. Conf. Robotics & Automation (ICRA)*. IEEE, 2002, pp. 1031–1036.

[3] J. Gonzalez-Barbosa and S. Lacroix, "Rover localization in natural environments by indexing panoramic images," in *Proceedings of the International Conference on Robotics and Automation*. IEEE, 2002, pp. 1365–1370.

[4] M. Artac, M. Jogan, and A. Leonardis, "Mobile robot localization using an incremental Eigenspace model," in *Proceedings of the International Conference on Robotics and Automation*. IEEE, 2002, pp. 1025–1030.

[5] N. Vlassis, B. Terwijn, and B. Kröse, "Auxiliary particle filter robot localization from high-dimensional sensor observations," in *Proceedings of the International Conference on Robotics and Automation*. IEEE, 2002, pp. 7–12.

[6] M. Jogan and A. Leonardis, "Parametric eigenspace representations of panoramic images," in *ICAR 2001 - Omnidirectional Vision Applied to Robotic Orientation and Nondestructive Testing (NDT)*. IEEE, 2001, pp. 31–36.

[7] D. Fox, W. Burgard, and S. Thrun, "Active Markov localization for mobile robots," *Robotics and Autonomous Systems*, vol. 25, pp. 195–207, 1998.

[8] J. Wolf, W. Burgard, and H. Burkhardt, "Robust vision-based localization for mobile robots using an image retrieval system based on invariant features," in *Proc. IEEE Int. Conf. Robotics & Automation (ICRA)*, 2002.

[9] E. P. E. Menegatti, M. Zoccarato and H. Ishiguro, "Image-based monte-carlo localisation with omnidirectional images," vol. 48, no. 1, 2004, pp. 17–30.

[10] C. S. H.-M. Gross, A. Koening and H.-J. Boehme, "Omnivision-based probabilistic self-localization for a mobile shopping assistant continued," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2003, pp. 1031–1036.

[11] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, June 1994. [Online]. Available: citeseer.nj.nec.com/shi94good.html

[12] G. Bradski, *Open Source Computer Vision Library*, Intel Corporation, 2001.

[13] D. Lowe, "Object recognition from local scale-invariant features," in *Proc. Int. Conf. Computer Vision ICCV, Corfu*, 1999, pp. 1150–1157. [Online]. Available: citeseer.nj.nec.com/lowe99object.html

[14] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *International Journal of Robotics Research*, vol. 21, no. 8, pp. 735–758, 2002.

[15] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*. Pacific Grove, CA, USA: Brooks/Cole Publishing Company, 1999.

[16] A. Doucet, N. de Freitas and N. Gordon, Ed., *Sequential Monte Carlo Methods in Practice*. New York: Springer, 2001.

[17] U. Frese and T. Duckett, "A multigrid approach for accelerating relaxation-based SLAM," in *Proc. IJCAI Workshop on Reasoning with Uncertainty in Robotics (RUR 2003)*, Acapulco, Mexico, 2003, pp. 39–46.