Proceedings of the 2006 IEEE International Conference on Mechatronics, Budapest, Hungary, 3-5 July 2006, pp 17-22.

# Omnidirectional Image Matching for Vision-Based Robot Localization

**Mana Saedan[1], Chee Wang Lim[2] and Marcelo H. Ang, Jr[1]**

[1] Department of Mechanical Engineering
National University of Singapore
9 Engineering Dr 1, Singapore 117576
Singapore
*{saedanm, mpeangh}@nus.edu.sg*

[2]Singapore Institute of Manufacturing Technology
Agency for Science, Technology and Research
71 Nanyang Drive, Singapore 638075
Singapore
*cwlim@simtech.a-star.edu.sg*

***Abstract* – The new approach to utilize information from an omnidirectional image is presented in this paper. We describe the integration between the image matching and the Monte-Carlo localization that can be implemented in a large indoor environment. The robot is able to localize with reasonable accuracy despite no metric measurements from the image. Furthermore, the algorithm can recover quickly from localizing a robot at a wrong place.**

## I INTRODUCTION

Omnidirectional vision systems become popular options of vision systems among the vision-based navigation research recently. This may be due to its low cost relative to other vision systems. Moreover, the benefit of having 360° horizontal field of view assists a robot to sense the whole environment in one snapshot regardless of robot's heading directions. Typical usage of the omnidirectional vision in robot navigation involves projecting an original image to either a perspective plane or a cylindrical surface [1, 2, 3]. Although the image projection does not require intensive computations, it is quite inefficient to incorporate this methodology in the online robot navigation. In contrast, we develop the algorithm to utilize an original image from an omnidirectional camera for navigation without having to project them to any two-dimensional surfaces.

Our image matching algorithm for an omnidirectional image is inspired by the image retrieval methods that exploit the local properties of an image. These methods seem to have several advantages; in particular, they are robust to occlusion. The methods normally involve 3 major steps:

1. Local feature point extraction.
2. Descriptor assignment to the feature point.
3. Matching the feature points between the query and the database.

In the first step, pixels in the image are identified based on some peculiar properties. For example, Schimid and Mohr [4] used Harris's corner detection to detect interest points. Bres and Jolion [5] identified interest points based on contrast property of an image. Lowe [6] used the difference of Gaussian functions to detect keypoints. His method is known as the scale-invariant feature transform (SIFT). Loupias et al [7] adopted the wavelet transform to obtain salient points. The second step assigns local properties such as image texture or color to describe those extracted pixels from the first step. The third step compares a query image to database images and identifies images from the database that resemble the query image.

We present details of our image matching algorithm, which is a crucial part in our vision-based localization. Furthermore, we elaborate the integration between the image matching and the Monte-Carlo localization. Our approach does not emphasize the accuracy of the location of a robot. Instead, we are interested in knowing roughly where the robot is currently at. Therefore, we develop the image matching to indicate similarity between the query image and the map database images without obtaining metric information between a robot and an environment.

In recent years, similar appearance-based localization algorithms have been proposed [2, 3, 8]. Among several algorithms, the work by Andreasson et al [8] is closely related to our work. They adopted the SIFT algorithm for matching images and incorporated the image matching with the Monte-Carlo localization. Nonetheless, our approach has some significant differences from their method. In particular, we introduce two levels of image matching that are seamlessly integrated into the localization algorithm. Those two levels of matching help reducing time spent in a matching process especially with a large map database. Furthermore, we develop the localization technique that requires less reference images while maintaining reasonable accuracy. Therefore, our localization can be easily extended to large scale indoor environments.

The paper is organized as follows. In the first section, we explain the methods of extracting feature points from an omnidirectional image. Next, we elaborate two levels of image matching that are performed, i.e., in the global and local levels, respectively. In the third section, we discuss our localization technique with the Monte-Carlo localization. The experiments are shown in the subsequent section.

## II FEATURE EXTRACTION

Some portions of an image that contain key information are obtained to form local features describing the image. The overview of the feature extraction process is depicted in Fig. 1. The feature extraction from the omnidirectional image involves three major steps. Firstly, some salient locations in the image are determined. Next, neighborhood pixels around each salient location are extracted. Finally, a feature descriptor is computed from each neighborhood region. The origin of the image in our implementation is located at the apex of a mirror in the image plane.

The final feature point from the extraction process consists of a salient location in the image and a descriptor vector.

### *A Salient Location Identification*

We modified the algorithm from [7] to suit our application. The Haar wavelet decomposition is used because of its computational simplicity. The decomposition algorithm is based on the non-standard decomposition procedure reported in [9]. The wavelet decomposition of

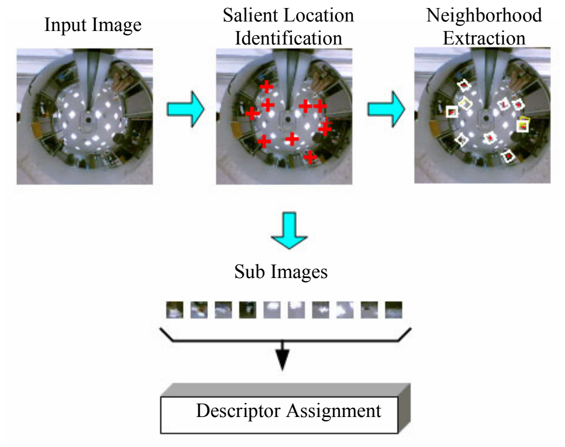

Figure 1
The overview of the feature extraction process from an omnidirectional image. The sub-image are obtain from the neighbourhood region, and the descriptor vector are computed from each sub-image.

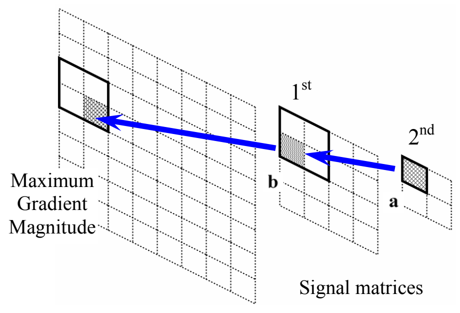

Figure 2
Salient point location from two levels of signal matrices. Every pixel from last signal matrix (on the right most) is tracked to determine the locations of salient point candidates in the original image.

an input image yields three coefficient matrices, namely the Horizontal ($\mathbf{H}$), Vertical ($\mathbf{V}$), Diagonal ($\mathbf{D}$), and the compressed image (to a quarter of the size of the original image). The decomposition can be recursively performed by using the output image from the previous iteration as the input. At each decomposition level, three wavelet coefficient matrices are combined into one signal matrix S:

$$S_j(m,n) = \left|H_j(m,n)\right| + \left|V_j(m,n)\right| + \left|D_j(m,n)\right| \qquad (1)$$

The salient-location-candidates are identified by "tracking" each element of the last signal matrix (last wavelet decomposition) to the untransformed input image. The example of this process is illustrated in Fig. 2. In this example, the signal point from the second level can be viewed as it is produced from four signal points of the first level. Hence, the signal track of point "a" is coming from four points in the top-left corner of the first level of the signal matrix. The highest signal value among those four points is determined, i.e. point "b". The process is repeated using the same analogy in case of more decomposition levels. The exact salient-location-candidate is obtained from the location of an image pixel that has highest gradient magnitude. The signal values along each signal track are also accumulated to indicate the strength of a salient-location-candidate.

The required number of salient locations can be set equal to or less than the number of candidates by sorting the signal strength of each candidate. The candidates that have high signal strength are selected until the number of salient locations is fulfilled. The fixed number of salient locations gives each database image equal opportunity to be matched with a query image in the matching process.

## B Neighborhood Pixels Extraction

The salient locations obtained must be assigned with some unique properties to distinguish them from salient points of other images. Unfortunately, there is no suitable feature description that can distinguish a pixel. Therefore, we extract a region around each salient location to form a sub image. This sub image can be represented by any image properties such as texture or color moment.

Our localization approach determines the location of a robot when it travels near the place that a reference image was taken. Hence, the current query image is different from the corresponding reference image by certain rotation angles. The neighborhood region (sub-image) around each salient location is extracted to make it rotationally invariant using the coordinates transformation in (2). As a result, the sub-images of the same salient point extracted from this method are having the same appearance regardless of rotational deviations from an original image.

$$\begin{aligned} x_n &= x_f + (u - M/2)\cos(\alpha_f) \\ y_n &= y_f + (v - M/2)\sin(\alpha_f) \\ I_n(u,v) &= 0.25[I(x_n,y_n) + I(x_n+1,y_n) + \\ &\qquad I(x_n,y_n+1) + I(x_n+1,y_n+1)] \end{aligned} \qquad (2)$$

where $(u,v)$ are the transformed (new) coordinates, $(x_n,y_n)$ are untransformed coordinates in the original image, and $(x_f,y_f)$ is the coordinate of the salient point. $\alpha_f = tan^{-1}\left(y_f/x_f\right)$. $\mathbf{I}_n$ is the sub-image obtained from the original image.

## C Feature Description

We adopted the idea from [6] which creates the feature description from the histogram of the image intensity gradient magnitudes. However, we choose to construct a descriptor based on the histogram of image color gradients instead. The advantage of using the color components is that they are more robust to illumination changes in images. In particular, we calculate the gradient on the YUV color components. The gradients of 3 color components are computed individually for each sub-image.

The histogram of each color component is created by accumulating the corresponding gradient magnitude according to the gradient direction. In our application, we choose to accumulate the gradient magnitude into 8 direction bins covering $0$ to $2\pi$. Finally, each histogram is normalized; the feature descriptor vector is formed by concatenating each normalized histogram together. Thus, the descriptor vector has 24 elements.

## III Image Matching

Our appearance-based localization relies on the ability to assess the degree of similarity between a query image and reference images in the map database. Consequently, we implement two levels of image matching: Global Matching and Local Matching.

In the following sections, each image is extracted a fixed number of feature points, i.e. 80 feature points. The feature points are derived from the salient locations described in previous section. Each feature point consists of the gradient descriptor vector (24 elements) and the position of the salient location in the image.

### *A Global Matching*

The map database is a collection of feature points obtained from reference image. This database generally contains large number of feature points, so matching the query image (feature points) exhaustively with the database is computationally demanding. Therefore, we embrace the tree search algorithm from [10] to speed the global matching process. The search algorithm requires a construction of the KD-tree from the feature descriptor vectors of the database. The algorithm, in brief, approximates the nearest neighborhood area from a large portion of the tree, and then does finer searching for feature-descriptor-candidates of the database that are closer to the query feature descriptor on a smaller portion. The search duration is specified by the number of maximum feature-descriptor-candidates to be retrieved.

A fixed number of feature-descriptor-candidates are obtained after submitting each descriptor vector of a query image to the global matching. The Euclidean distances between the query feature vector and its corresponding candidates are computed, and the smallest distance is identified. The cut-off threshold is set at 1.25 times the smallest distance. Matched candidates that have distances smaller than the cut-off threshold are marked, and their corresponding distances are also recorded. The process is repeated with different query descriptor vector. If there is a candidate feature point matches with more than one query point, only the pair that has smallest distance is kept while other pairs are dismissed. Reference images are sorted according to their number of matched pair, and the reference images that have matched-pairs higher than 85% of the highest matched pairs are selected.

We assume that the query image and the corresponding reference image are taken at the same position but may be at different angles. Therefore, each initial matched image is examined further to eliminate outliers through the rotation constraint in (3). The rotation angles from pairs of feature points are grouped into 8 angle bins covering $0$ to $2\pi$. The bin that has highest number of match pairs is obtained, and the number of matched pairs in the particular bin is considered as the inliers.

$$\phi(q,d) = \tan^{-1}\left(\frac{x_d y_q - x_q y_d}{x_d x_q - y_d y_q}\right) \qquad (3)$$

where $(x_q, y_q)$ and $(x_d, y_d)$ are the coordinates of the query feature point and the coordinate of the database feature point, respectively.

The maximum number of matched pairs among the matched reference images after applied (3) is the *global score* for the particular query image. The outputs of the global matching are those matched reference images and the matched pairs of the top reference image.

### *B Local Matching*

The local matching performs similar steps as the global matching except that the local matching compares the query image with a single reference image. Each feature descriptor of the query image is matched with all feature descriptors from the reference image. The pair that has minimum Euclidean distance is recorded as the initial matched pair. If there is more than one query descriptor that matches with the same descriptor of the reference image, only the pair with smallest Euclidean distance is recorded. After all feature descriptors of the query image are matched, the outliers are being eliminated by (3). The outliers are regulated by the same method as the global matching. The number of inliers is the *local similarity score* between the query image and the corresponding reference image. The rotation angle is determined by averaging the rotational angles of those inliers.

## IV Appearance-Based Localization

The result from image matching is very noisy due to several possible causes such as occlusion, changing in lighting condition, or a query image is taken at a location not perfectly at the same location where the reference image was taken. As a result, using only image matching alone to localize a robot almost always does not give consistent results. We, therefore, take uncertainties into account by incorporating a probabilistic model into our localization problem. In particular, we implement the Monte-Carlo localization with our image matching algorithm.

The Monte-Carlo localization (MCL) represents the belief of a robot state by a set of random samples. Each sample comprises the dynamic model of a robot and the importance weight factor. The MCL algorithm is briefly explained in the following steps:

1. Prediction: The samples are drawn according to the transition model of the robot's dynamics given the action executed since the previous robot state.
2. Importance weight update: The new observation is used to update the importance weight of each particle. The weight updating is based on the observation model. We use the local matching to evaluate importance weights of the particles.
3. Re-sampling: Particles are re-sampled according to their updated weights; this is to prevent degeneration problem in the particle filter.

Our localization determines the location of a robot relative to the "place" where the reference image was taken. A hybrid map between topological and metrical maps is constructed from a collection of images taken from various locations. The map is represented by the graph where nodes are encoded with image feature data, and edges indicate the feasible path between nodes. In addition, the edge metrical information is extracted from a robot odometry reading.

### *A Robot Dynamic Model*

The robot state in our system is modeled as

$$\xi(t) = \begin{bmatrix} n & x_r & y_r & \theta_r \end{bmatrix} \qquad (4)$$

where $n$ is the reference node, and $(x_r, y_r, \theta_r)$ is the relative position of a robot with respect to the coordinate frame of the node.

The state of each particle is updated based on the reading of robot odometry. To simulate motion noises, each particle is subjected to uniform random motion errors. In particular, the noise is added to the state of each particle; noise magnitude is up to $\pm 10\%$ of the measured motion.

*B Observation Model*

The characteristic of the local image matching is observed by (local) matching several query images taken from various distance away from the reference image. The prominent characteristic of our local similarity score is that more than 80% of the query images taken within 20 cm range from the corresponding reference image have similarity scores above 15, whereas the similarity scores of the query images taken farther from this area have very wide fluctuations. Therefore, we develop the localization technique to perform (actual) weight updating only when any particle is within 20 cm radius from its reference node.

When a robot is in the sensitive area (within 20 cm range) of any map node, the similarity score between the query image and the reference image of the particular map node is likely to be much higher than the score of other map nodes. Therefore, particles in the true reference node are updated with higher importance weight than the others. Moreover, rotational angle from local matching is added in the importance weight updating to decrease matching ambiguities, i.e. other reference nodes have high similarity score as the true reference node. The false reference node is often to have a random rotational angle with the query image even though their similarity score is high. The weight updating function is given by:

$$w^i(t) = \exp\left( -a\left(S(q,n^i) - 15\right) - b\left|\theta_r^i - \phi(q,n^i)\right| \right) \quad (5)$$

where $S(q,n^i)$ is the local similarity score between the query node $q$ and the node $n^i$. The similarity score is being clipped at 15 if it is more than 15. $\phi(q,n^i)$ is the rotational angle between the query image and the reference image. $a$ and $b$ are the arbitrary positive constants. The weights of samples outside the range are calculated from the last actual updated weight with a linear discounting factor proportional to the distance from the reference node.

*C Disbelief Algorithm for Re-localization*

To add the ability for a robot to recover from localization at the wrong place, we introduce the disbelief algorithm to re-initialize the states of some particles. After applying the standard re-sample algorithm, the number of particles at each map node is examined. If one of the map nodes has particles concentrated significantly higher than other map nodes, some particles from that particular node are randomly removed. The removed particles are then re-initialized to new states according to the output nodes from the global matching. However, the disbelief algorithm is activated only when the global similarity score is higher than the "disbelief" threshold. This disbelief threshold indicates whether the robot is currently within the sensitive area of any map node or not. The empirically value of the disbelief threshold is given as a function of number of map nodes $(N)$ as:

$$T_{db} = 5 + 10\exp(-0.05N) \quad (6)$$

## V EXPERIMENTAL RESULTS

The first experiment was conducted to evaluate the accuracy of the global matching. The robot was manually controlled to travel along the corridor of our laboratory building; the size of the building is approximately $100 \times 50$ meter. A total of 304 images were collected to create a database, and 80 feature points were extracted from each reference image. The location of each reference image was also recorded in the database.

Another set of images were taken by controlling the robot to move along the same corridor but in the opposite direction. Forty-one images were used as query images to the global matching function. The accuracy of the matching function was verified by comparing the location of output images obtained from the global matching to the location of the query images. Each trial was regarded as a successful matching when the location of the query image was within 100 cm range from one of the output candidates.

The accuracy of the global matching when varying the number of maximum search feature-point-candidates is depicted in Fig. 3. Moreover, the time spent in matching process is illustrated in Fig. 4. The result in Fig. 3 suggests that the global matching accuracy is above 50% even when a small number of maximum search candidates are applied. Increasing the number of maximum search candidates improves the matching accuracy. However, we do not find any significant improvements when using the maximum search candidates more than 10% of the total feature points in the database.

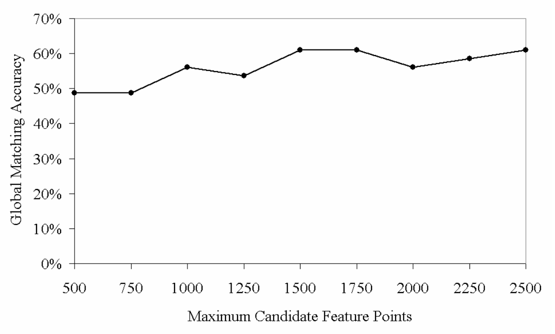

Figure 3
Global matching accuracy vs. maximum searched candidates. The total database feature points are 24320 points.

In Fig. 4, the time spent in the global matching is increasing with the number of maximum search candidates. Nonetheless, large number of maximum search candidates reduces the maximum number of matched reference images; hence, the local matching requires less time to match every output image. As a result, the optimal percentage of the maximum search candidates to the total database feature points is approximately 10%. Consequently, we continued using this ratio in the subsequent experiments.

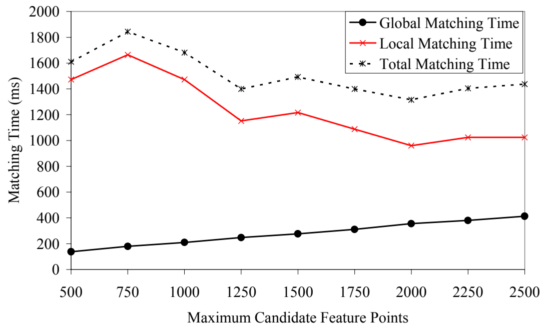

Figure 4
Matching time vs. maximum search candidates.

The second experiment was carried inside our laboratory. The objective of this experiment was to observe the performance of our localization algorithm. The map database was constructed from 72 images taken at about 100 cm interval while the robot moved along the route depicted in Fig. 5. The number of feature point extracted from each image was fixed at 80 points.

Two weeks after the map was created, the localization was performed on-line from the on-board computer; the Intel Pentium M 1.73 GHz notebook PC with 1024 MB memory was placed on top of the robot as shown in Fig. 6. The query images were taken every time the robot had moved by 10 cm or turned by 30 degrees.

Two localization scenarios were conducted. In both scenarios, the robot traveled through all nodes except that it moved in different direction. In particular, the robot traveled from reference node number 1 to 72 in first scenario, whereas the robot traveled in reversed order in the second scenario. The video clips of the second experiment can be downloaded from the following url:

*http://guppy.mpe.nus.edu.sg/~manna*

The video clips show the user interface program while the robot was traveling and localizing. The big green dot in the map shows the estimated reference node of the robot, and the blue dot indicates the true location. The estimated reference node was determined from the map node that particles converged more than 30% of the total particles.

Table 1 summarizes the performance of our localization algorithm. The correct reference node was verified when the actual robot location was within 200 cm, otherwise the reference node was marked as a false reference node. The loss time was count when the estimated reference node has less than 30% of total particles.

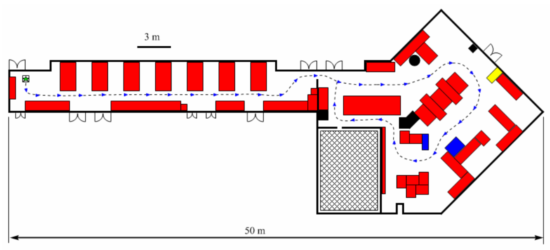

Figure 5
The floor-plan of the laboratory during the experiments

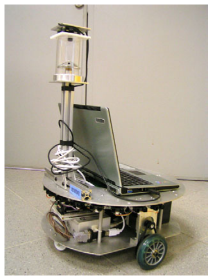

Figure 6
Our robot with the omnidirectional camera

The result in Table 1 shows that our localization algorithm is able to localize the robot within the position error range of 200 cm. However, both scenarios indicate some false localization (the estimated robot location is farther than 200 cm from the true location). This is because our reference node selection method is based on identifying the node with maximum number of particles. Sometimes the true reference node has lower number of particles than other nodes.

Table 1
The performance of the localization algorithm

| Scenario No. | Localization Outcome (100%) | | |
|---|---|---|---|
| | Correct | False | Loss |
| 1 | 79.9 | 16.2 | 3.9 |
| 2 | 66.9 | 28.3 | 4.8 |

The kidnapping problem was demonstrated in the last experiment. Fig. 7 depicts the distribution of particles during the first kidnapping attempt. The result in Fig. 7 has shown that our algorithm has the ability to perform global localization despite the absence of absolute position information in our map. The success in global localization is from global image matching.

## VI Conclusion

We present image matching algorithm based on our modified version of the wavelet-based salient point detection. The matching algorithm has two parts: the global matching selects the reference image-candidates that are similar to the query image, and the local matching determines the similarity score and the rotation angle between a single reference image and the query image. Furthermore, the integration of the image matching with the Monte Carlo localization enables the robot to localize with reasonable accuracy.

Future development is to implement the algorithm for a robot to be able to incrementally build the map from scratch when it is operated in an unknown indoor environment. Moreover, the robot must be able to update the map nodes with latest sensor data.

## References

[1] Y. Mutsumoto, K. Ikeda, M. Inaba, and H. Inoue, "Exploration and navigation in corridor environment based on omni-view sequence," IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000), 2000, pp. 1505–1510.

[2] H.-M. Gross, A. Koenig, C. Schroeter, H.-J. Boehme, "Omnivision-based probabilistic self-localization for a mobile shopping assistant continued," IEEE/RSJ International Conference on Intelligent Robot and Systems (IROS 2003), 2003, pp.1505-1511.

[3] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro, "Image-based Monte-Carlo localization with omnidirectional images," Robotics and Autonomous Systems, vol. 48, no. 1, 2004, pp.17-30.

[4] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 19, no. 5, pp. 530–535, 1997.

[5] S. Bres and J.-M. Jolion, "Multiresolution contrast based detection of interest points," Technical Report, Laboratoire Reconnaissance de Formes et Vision, 1998.

[6] D. G. Lowe, "Distinctive image feature from scale-invariant keypoints," International Journal of Computer Vision, vol. 60, no. 2, 2004.

[7] E. Loupias, N. Sebe, S. Bres, and J.-M. Jolion, "Wavelet-based salient points for image retrieval," in the International Conference on Image Processing, 2000.

[8] H. Andreasson, A. Treptow, and T. Duckett, "Localization for mobile robots using panoramic vision, local features and particle filter," in IEEE International Conference on Robotics and Automation (ICRA 2005), 2005.

[9] E. Stollnitz, T. DeRose, and D. Salesin, "Wavelets for computer graphics: A primer, part 1," IEEE Computer Graphics and Applications, vol. 15, no. 3, pp. 76–84, 1995.

[10] J. S. Beis and D. G. Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," in Conference on Computer Vision and Pattern Recognition, 1997, pp. 1000–1006

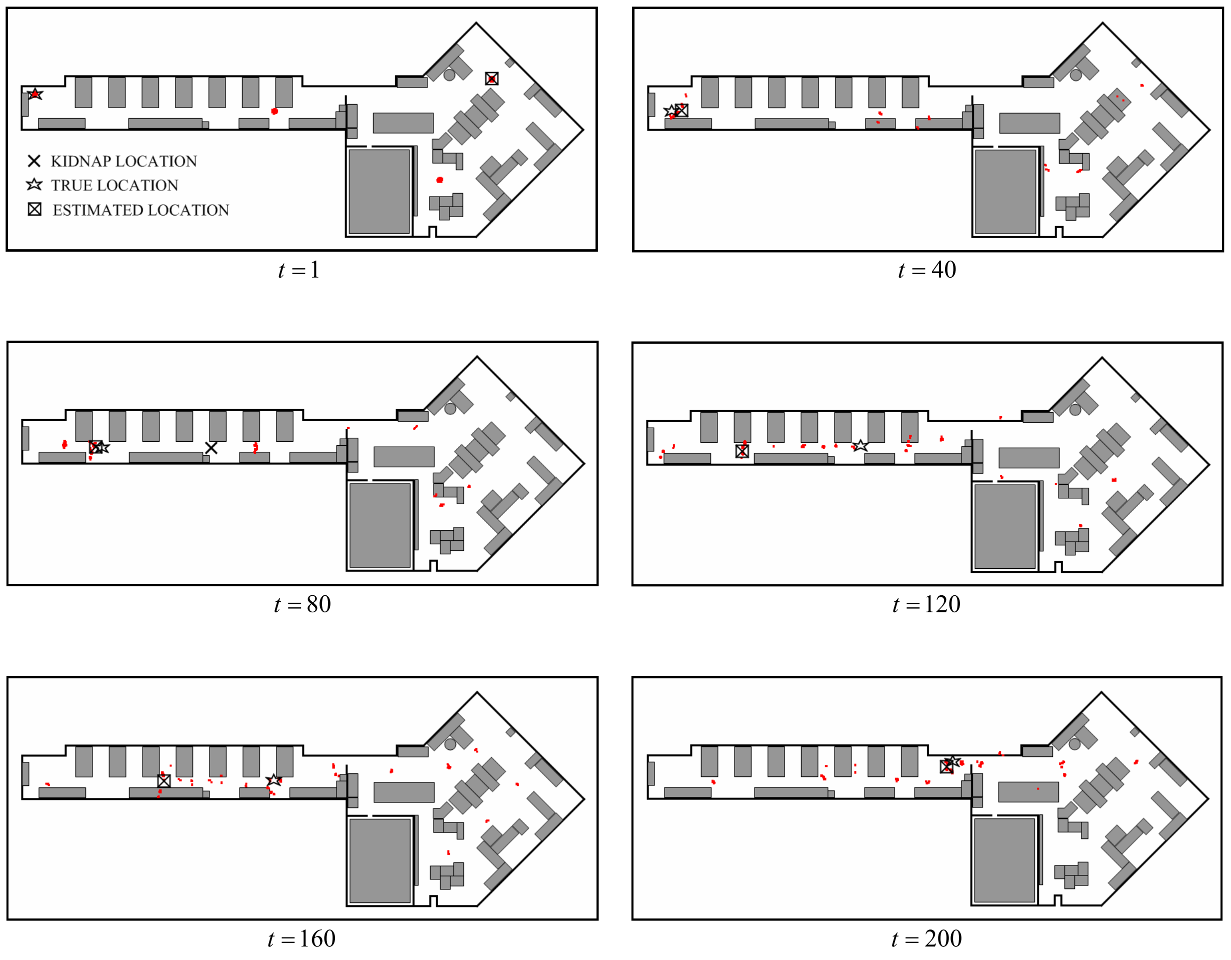

Figure 7
Snapshots of particles distribution during the kidnapping trial. The robot was kidnapped after time steps 80.