Indoor Map-Constrained Robot Localization Using Incidental Radio Observations

Joydeep Biswas
The Robotics Institute
Carnegie Mellon University
Pittsburgh PA 15213
Email: joydeepb@cs.cmu.edu

Manuela Veloso
Computer Science Department
Carnegie Mellon University
Pittsburgh PA 15213
Email: veloso@cs.cmu.edu

***Abstract*— We are interested in equipping an indoor mobile robot with a robust localization algorithm. We assume that the robot is performing some task that requires it to continuously move, and localization needs to be incidentally achieved. A few other known efforts, e.g., a museum-guide robot, a service robot, or a soccer robot, have addressed similar indoor incidental localization and navigation problems, but have mostly relied on visual-based landmark perception. Such tasks can include frequent stoppages that can play an important role in reducing localization uncertainty. In this work, we assume that the environment includes static radio nodes, which can be queried for localization landmark information. We introduce a representation of the indoor map in terms of both rigid map constraints, e.g., corridor width and connectedness, and as a set of sampled location points with signal strengths defining the a priori map knowledge. The dynamic radio signal strength from the sensor network nodes, along with the discrete map points is used to generate a continuous localization probability distribution. This probability distribution, combined with the motion data of the robot and the rigid map constraints reults in a novel indoor localization algorithm. Two novel approaches in our algorithm compared to other algorithms are the method of generation of an estimate of the perceptual model that is continuous in space, and the use of rigid map constraints for weighting location beliefs. We demonstrate the algorithm in an autonomously moving robot that interacts with a real sensor network in our building.**

I. INTRODUCTION

An autonomous robot needs to be location aware in order to meaningfully execute its tasks. Numerous approaches address the localization problem with varying degrees of success.

Most robots operating outdoors can tolerate localization errors on the order of one meter, generally using GPS readings. Very few devices can consistently receive GPS signals indoors with appropriate accuracy.

A variety of approaches to indoor localization have been developed that rely on the detection of features by the robot's sensors, and estimation of the range and orientation with respect to known landmarks. The landmarks chosen can be artificial, like colour-coded markers, or pre-existing objects like doorways and light fixtures.

Some indoor localization algorithms use optical sensors including cameras, laser scanners, and infrared sensors, but such approaches do not perform as well in crowded indoor environments, where people, robots and other mobile obstacles may obstruct the view of the sensors.

Algorithms that attempt to perform Simultaneous Localization and Mapping (SLAM) commonly use particle filters for representing the location hypotheses of the robot. However, the hypotheses are permitted to evolve irrespective of map based constraints.

In this work, we use radio frequency (RF) beacons, as well as wireless networks to provide indicative range information using received signal strength. This approach is not affected by changes in the visual appearance of the environment, although it is still susceptible to local obstacles like people and other robots.

Our approach at tackling the localization problem is motivated by the development of an indoor visitor companion robot. The environment in which the robot is to operate is an office building with a series of interconnected hallways, which can be traversed by the robot. As in any busy office building, the hallways have people and other robots involved in their own tasks. There will also be obstructions like carts, tables, chairs and boxes left in the hallway temporarily. The robot should be free to move about, performing its own tasks without having to stop to localize itself. This implies that any sensory inputs that assist in the localization of the robot should be incidental, without requiring the robot to stop for readings just to obtain a localization estimate.

In this paper, we describe an algorithm based on Markov localization for localizing a robot on a known map using RF signal strength from the nodes of a sensor network. In the derivation of the sensor prediction model, we assume no knowledge of the propagation characteristics of the RF signals.

Our algorithm for localization of the robot consists of two stages. In the first stage, the *Map Learning Phase*, we introduce a representation of the map using a set of vertices V, and a set of edges E connecting them. The vertices are distributed in space according to a given distribution, e.g. uniformly at 1m intervals. Fig. 1 shows an example of the graph representation of a section of a map. In the figure, the vertices v_1, v_2, v_3 and v_4 are shown connected by edges (dashed lines). Three RF nodes, n_1, n_2 and n_3 are also shown. To complete the *Map Learning Phase*, the robot is manually driven around, and at every vertex, the robot is

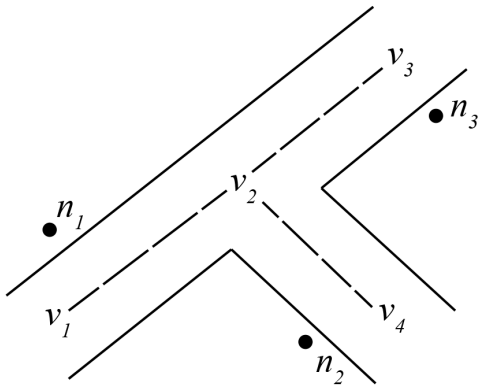

Fig. 1. An example of the graph representation of a section of the map

halted and allowed to collect signal strength data from the accessible nodes of the radio network. The Map Learning Phase needs to be performed only once in the environment where the robot is to operate.

The second stage, the *Run-time Phase*, is performed online on the robot. The robot is free to perform its various tasks, while the localization algorithm continuously updates its location belief, requiring the following data:

1) Data collected and programmed during the Map Learning Phase. This includes the layout of the map and the test measurements taken.
2) Odometry data from the robot. This includes the motion of the robot under its own, or external control.
3) Signal strength measurements from the sensor nodes that are accessible from the robot's present location.

Given the above sets of data, our localization algorithm provides a sample-based probability distribution of the location of the robot on the map, as well as the most probable location of the robot. This is described in detail in Section IV-A.

Since this approach uses no visual sensors, the presence of obstacles has no effect on the algorithm. Furthermore, no RF signal propagation model is necessary as the algorithm is only dependent on the collected samples in the Map Learning Phase. In addition, empirical results show that the rigid map constraints enable the algorithm's estimate to converge to the correct location rapidly.

II. RELATED WORK

Robot soccer is an application where researchers have had to address the problem of mobile robot localization with concurrent tasks. In the Standard Platform League, the Sony AIBO four-legged robots perform localization on a known game field using visual data gathered by a low-resolution camera. There are several colour-coded markers around the field that the robots could use as landmarks. Lenser and Veloso [1] introduced an algorithm called *Sensor Resetting Localization* to maintain an estimate of the robot on the playing field, as well as the capability to "reset" the location estimates when sensory data significantly contradict the current estimates. Of similar nature is the work of Kwok and Fox [2], which involves the use of Rao-Blackwellised particle filters to track a moving target (the ball). The use of map information is shown to provide a decrease in the occurrences of the ball getting lost. In both these efforts, however, the robot would have to stop and look around for visual markers occasionally when lost.

Variations of feature based localization have been developed and tested on a number of robots, including Robox [7], [4], which was extensively tested during the Swiss National Exposition Expo.02. Localization based on landmark detection involves looking for landmarks using the robot's sensory inputs, and comparing with known models of the landmark to estimate robot location [9], [10]. Significant work has also been done in the area of autonomous "learning" of landmarks [11], [12]. In this scheme, the robot is free to learn features and landmarks in the environment that are best suited for localization. Markov Localization is yet another popular localization technique that has been employed by the interactive museum tour-guide robot, Rhino [5], [6]. Another implementation of the Markov localization technique using a metric discretization of the state space [3] was evaluated using the Rhino [5] and Minerva [8] robots. The Xavier robot [13], [14], uses partially observable Markov decision processes to maintain a belief of its position, as required by its navigation algorithm.

III. MARKOV LOCALIZATION AND THE PERCEPTUAL MODEL ESTIMATION

A. A Priori Data

Our algorithm requires a set of a priori data, which is gathered during the *Map Learning Phase*, as described in Section IV-A. This data consists of a graph-based representation of the map, and RF test data.

1) Parametric Representation of the Map: The map is represented as a directed graph, with a set of vertices V and a set of edges E to connect them. Nv is the number of vertices in V, and Ne is the number of edges in E. For every edge e_i in E, the direction of the edge is from $v_{i1}$to v_{i2}, where $v_{i1}, v_{i2} \in V$. For every edge e_i in E, the width of the corridor in that section is denoted by $width_i$.

2) RF Test Data: For each of the vertices v_i ($v_i \in V$), the expected RF signal strengths from the sensor nodes need to be measured. This comprises the RF test data. The mean response of the sensor nodes when queried from vertex v_i is represented by a row vector $R_i = [r_{i1} \dots r_{iNn}]$ where r_{ij} is the mean signal strength of sensor node j when queried from vertex v_i. Nn is the number of sensor nodes. Once the RF data collection is complete, the matrix $\mathbf{R} = [R_1^T \dots R_{Nv}^T]^T$ represents the expected signal strength responses from every sensor node, for every vertex v_i.

B. Markov Localization using recursive Bayesian Update

As derived in [15], the recursive Bayesian update for a location belief $Bel(x_t)$ for location x at time step t is given

by:

$$Bel(x_t) = \eta p(y_t|x_t) \int p(x_t|x_{t-1}, u_{t-1}) Bel(x_{t-1}) dx_{t-1} \quad (1)$$

Here, η is a normalization constant, y_t the sensor reading at time step t, and u_{t-1} the odometry data measured between time steps $t-1$ and t. The term $p(x_t|x_{t-1}, u_{t-1})$ is obtained from the motion model of the robot. The term $p(y_t|x_t)$ is given by the perceptual model - the probability of making a sensory reading y_t as a function of the pose of the robot, x_t. This calculation is central to our algorithm.

C. The Perceptual Model Estimate

During the *Run-Time Phase*, the matrix **R** is known, and the latest RF signal measurement is given by $\mathbf{S} = [S_1 \dots S_{Nn}]$ where S_i is the signal strength received from node n_i. Not all S_i's need exist, since not all nodes are neccesarily within RF range of the robot. Let $error_i$ denote the squared error between the measured signal S and the mean response R_i at node v_i. This error term is defined as the sum of the squared errors between the observed signal strengths and the expected signal strengths. To avoid penalizing dropped signals, the error term is considered only with respect to witnessed S_i terms.

$$error_i = \sum_{j=1}^{Nn} IsObserved(S_j)(r_{ij} - S_j)^2 \quad (2)$$

Here, $IsObserved(S_j) = 1$ if S_j exists, and $IsObserved(S_j) = 0$ if S_j does not exist. This ensures that the absence of an observed signal is not penalized.

Once the $error_i$ term has been computed for every i, it can be used to estimate $P(S|v_i)$, the probability of the reading S being made if the robot was located at vertex v_i. The mapping from $error_i \rightarrow P(S|v_i)$ can be chosen according to the theoretical model of the RF signal propagation. Assuming no prior knowledge of RF signal propagation, there are a number of possible choices for such a mapping. However, all such mappings must assign higher probabilities to vertices with lower $error_i$ values. Thus, the vertex v_i with the smallest value of $error_i$ would be the nearest to the actual location of the robot, and should be assigned the highest value for $P(S|v_i)$. The specific mapping that we use is given by:

$$\begin{aligned} \hat{P}(S|v_i) &= \gamma(\max(error_1 \dots error_{Nv}) \\ &\quad - error_i + \epsilon + \mathcal{N}(0, \sigma_s)) \end{aligned} \quad (3)$$

Here, γ is a normalization constant that ensures that the probabilities of all sensor measurements sum to 1.

$$\int_S P(S \mid v_i) = 1 \quad (4)$$

In the estimate of the perceptual model, the Gaussian noise term $\mathcal{N}(0, \sigma_s)$ is added to emulate sensor noise with zero mean and variance σ_s. ϵ is a small number that is added to each term to ensure that the probability function does not got to zero anywhere. This is done to ensure that the weight of the belief does not go to zero in a single step at any location. From this mapping (eq. 3) it can be seen that the vertex with the largest $error_i$ value is assigned the smallest probability value, while the vertex with the smallest value of $error_i$ is assigned the highest probability.

The calculated estimate of $P(S|v_i)$, $\hat{P}(S|v_i)$, is limited to the discrete locations of the vertices v_i on the map. In order to update the probabilities of location beliefs at arbitrary locations on the map, the complete perceptual model $\hat{P}(S|x)$ is estimated by linear intorpolation between adjacent vertices on the map.

With this estimate of the perceptual model $\hat{P}(S|x)$ and the motion model of the robot $p(x_t|x_{t-1}, u_{t-1})$, the localization belief of the robot can be recursively updated.

IV. IMPLEMENTATION

The algorithm for localization consists of two stages. The first stage is the *Map Learning Phase*, where the a priori data required by the algorithm is collected. The second stage, the *Run-Time Phase* is performed online while the robot is operational.

A. Map Learning Phase

The first step in the Map Learning Phase is to define the map that is traversable by the robot. After this, the RF signal strength data for every vertex on the map is collected.

1) Defining the Parameters of the Map: The map is represented as a directed graph, as described in Section III-A.1. The vertices v of V are placed along the corridors traversable by the robot. The vertex spacing can be reduced for a finer representation or increased for a coarser representation. For every edge e_i in E, the width of the corridor in that section is denoted by $width_i$. The locations of the vertices, lengths of the edges, and the widths of the corridors are taken from the architectural plans of the building.

2) Collection of RF data: Once the representation of the map is defined, the robot collects sensor data at each of the vertex locations v_i ($v_i \in V$). At each location, the robot is allowed to continuously send out ping requests to all sensor nodes in range, and it gathers the signal strengths of all these nodes. This is used to populate the matrix **R** as described in Section III-A.2. Signal strengths are recorded in dBm. If a node n_j is inaccessible from vertex v_i then the corresponding term r_{ij} is set to -46dBm, which is the sensitivity threshold of the radio receiver that we used.

B. Run-Time : Representation of the Location Hypotheses

The multiple hypotheses of the robot location are sampled and represented by *particles* p_i. The number of particles is denoted by Np. Each particle p_i has the following properties:

- ep_i, the edge that the particle is associated with.
- $v1p_i$, the first vertex of edge ep_i.
- $v2p_i$, the second vertex of edge ep_i.
- d_i, the projected location of the particle on the edge.
- o_i, the offset of the location of the particle from the edge.

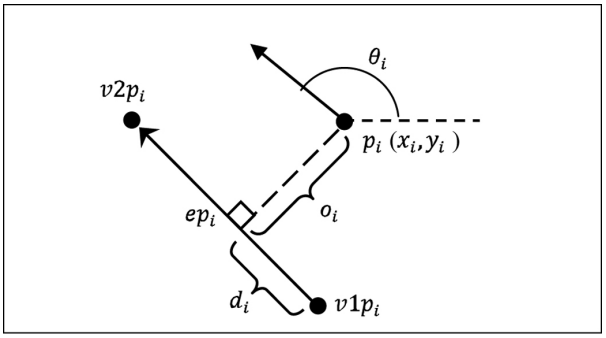

Fig. 2. Properties of a particle

- x_i, y_i the Cartesian location of the particle on the map with respect to a global reference frame.
- θ_i, the orientation of the particle with respect to the global reference frame.
- w_i, the normalized weight assigned to the particle.
- wc_i, the map constrained weight assigned to the particle. This weight is calculated in the *Constrain* step of the Run-Time phase, as described in Section IV-C.3.

The properties of the particle are graphically illustrated in Fig. 2.

C. Run-Time : Updating the Location Hypotheses

The location hypotheses are updated iteratively when new data is available. The four steps involved in the update are the *Predict* step, the *Update* step, the *Constrain* step and the *Resample* step. The Predict step is executed whenever new odometry data from the robot is available, and updates the positions and orientations of the particles. The Update step is executed when new RF signal data is available, and updates the weights of the particles. The predicted perceptual model based on the latest signal measurements, as described in Section III-C, is used for this step. The Constrain step is executed whenever the Predict step is executed, and updates the weights and edge associations of the particles based on map data and constraints. The Resample step is used to resample the particles when their weightage drops below a threshold.

1) Predict: Given new odometry data of the motion of the robot, for each particle p_i, its properties θ_i,x_i,y_i,d_i,o_i are updated using the motion model of the robot, as described in [15]. Algorithm 1 demonstrates this implementation. In the motion model of the robot, $\mathcal{N}(0, \sigma_\theta)$ and $\mathcal{N}(0, \sigma_u)$ are the Gaussian error terms for robot orientation and distance traversed, respectively.

2) Update: To update the weights of the particles based on the latest RF signal readings, the estimated perceptual model as described in Section III-C is used. Once $P(v_i|S)$ has been computed, the probability distribution of being at any point on the map can be estimated by linear interpolation between adjacent vertices. Thus, the weight of each particle is updated by multiplying its previous value with a linear combination of the probabilities of being at the ends of the edge that the particle is associated with. Algorithm 2 implements this.

3) Constrain: Following the Update step, the edge association of each particle is re-evaluated, and the map constrained weights computed. For this, particles which are within a threshold $thresh$ of the ends of the edge are projected onto the neighboring edges to determine which among them would be best associated with the particle. This association is determined based on the offset from the edge, and the difference between the orientation of the particle and the orientation of the edge. Furthermore, the weights of the particles which have an offset from their associated edge greater than the width of the edge, are attenuated with a gaussian falloff. Algorithm 3 illustrates the implementation of this step.

4) Resample: Particles with wc_i values less than a threshold value are removed, and replaced by a new sampling.

Algorithm 1 Predict.

```
let u = distance traversed by robot
let dθ = rotation of robot
for i = 1 to Np do
    θ_i ← θ_i + dθ +N(0, σ_θ)
    u_i ← u+N(0, σ_u)
    x_i ← x_i + u_i cos(θ_i)
    y_i ← y_i + u_i sin(θ_i)
    recalculate d_i and o_i
end for
```

Algorithm 2 Update particle weights.

```
for i = 1 to Np do
    fr ← d_i/(length of edge ep_i)
    w_i ← w_i * (fr * P(v_2|S) + (1 − fr) * P(v_1|S))
end for
```

Algorithm 3 Apply map constraints.

```
for i = 1 to Np do
    fr ← d_i/(length of edge ep_i)
    if fr > 1 − thresh or fr < thresh then
        Find edge e best associated with p_i
        if e! = ep_i then
            ep_i ← e
            Calculate new d_i
            o_i ← 0
        end if
    end if
    width_i ← width of edge ep_i
    if abs(o_i) > width_i then
        wc_i ← w_i * exp(−(abs(o_i) − width_i/2)^2)
    else
        wc_i = w_i
    end if
end for
Renormalize all weights wc_i
```

This sampling is drawn from the map at location x with probability $P(S|x)$ based on the last RF signal measurement S.

D. Inference of Location

Given the set of particles p_i, the best location hypothesis of the robot is obtained by taking the mean of the location of the particles within a distance threshold from the particle with the highest wc_i value, and projecting this mean onto the map.

V. EXPERIMENTAL RESULTS

The localization algorithm presented here was tested on an actual robot in an office building that was equipped with an RF sensor network. The robot used for the experiments was a custom built platform for a visitor companion robot, with an omnidirectional drive mechanism, a LADAR sensor for obstacle detection, and an onboard notebook computer. The number of particles was set to 4000. The predict, update, contrain and resample iteration cycles take less than 0.25s to compute on the onboard 1.73GHz Pentium M processor. RF signal strengths are polled by the robot every two seconds. Fig. 3 shows the layout of the traversable map with the sensor nodes marked with dots.

To illustrate some features of the algorithm, we present in Fig. 6 some snapshots of a test run of the robot along a subsection of the map. The robot was pre-programmed to follow a pre-specified path (ground truth shown in Fig. 4). The motion primitives to implement these commands utilized sensory input from an on-board laser range-finder for obstacle avoidance and aligning with the walls.

The evolution of the location hypotheses, as represented by the cloud of particles on the map, is shown in Fig. 6. Initially, the localization algorithm is started with no prior knowledge of its location, but with a known orientation. The uniformly distributed particle cloud in Fig. 6a indicates the poor localization knowledge. The actual position of the robot is depicted by the cross. As the robot gathers RF signal data while still stationary, the particle cloud shrinks to one corridor, and then one section of the corridor (Fig. 6b and Fig. 6c, respectively). As the robot is driven forward, the particle cloud similarly translates on the map, while the application of the map constraints eliminates the particles which move out of the corridor boundaries. This results in the particle cloud growing smaller and denser, as is seen in Fig. 6d. As the robot starts turning left and moving down the connecting corridor, particles in those locations of the map where such a turn is not possible get weighted less and less until they are completely ruled out (Fig. 6e, Fig. 6f). The resulting cloud of particles is clustered close around the actual location of the robot (Fig. 6g), and remains well localized even after the second left turn (Fig. 6h).

For comparison, the algorithm was simultaneously tested under the same circumstances, without applying the map constraints of the Constrain step (Section 3.3). This is equivalent to assuming that the corridors have infinite width, so particles never"fall off" the corridors. The same test run was performed a number of times with and without map constraints, and the time progression of the weighted location errors were recorded. The mean response of all these runs, with and without map constraints, is plotted in Fig. 5. At the beginning of the runs, both the map constrained and the unconstrained algorithms have a similar response. As the robot starts taking the first turn (around $t = 55s$) however, the map constrained algorithm decreases the weights of the particles for which such a turn is not possible. This leads to a sudden reduction in the mean error of the particles for the map constrained algorithm. The unconstrained algorithm continues to suffer from large errors until successive RF signal measurements finally reduce the distribution of its particles. The second turn around $t = 90s$ again causes an increase in the error for the unconstrained algorithm. The error in the localization for the map constrained case remains around 0.5m after convergence. Thus it can be seen that the algorithm with the map constraints leads to a faster convergence than for the unconstrained algorithm.

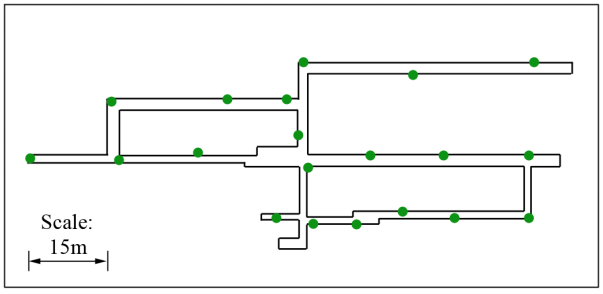

Fig. 3. Layout of traversible map, with sensor nodes marked in green.

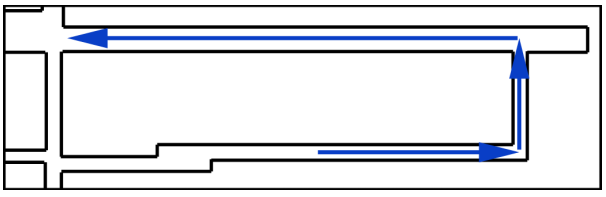

Fig. 4. Ground truth of path traversed by robot (blue line).

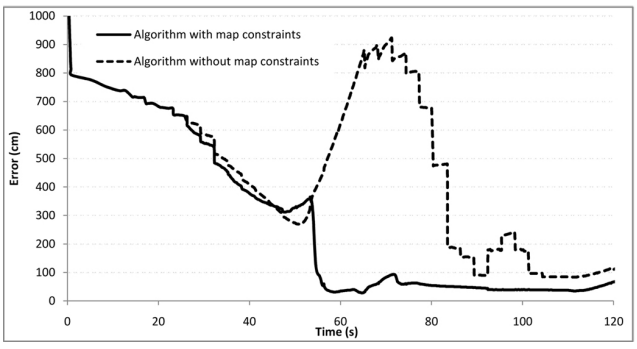

Fig. 5. Time evolution of mean localization errors

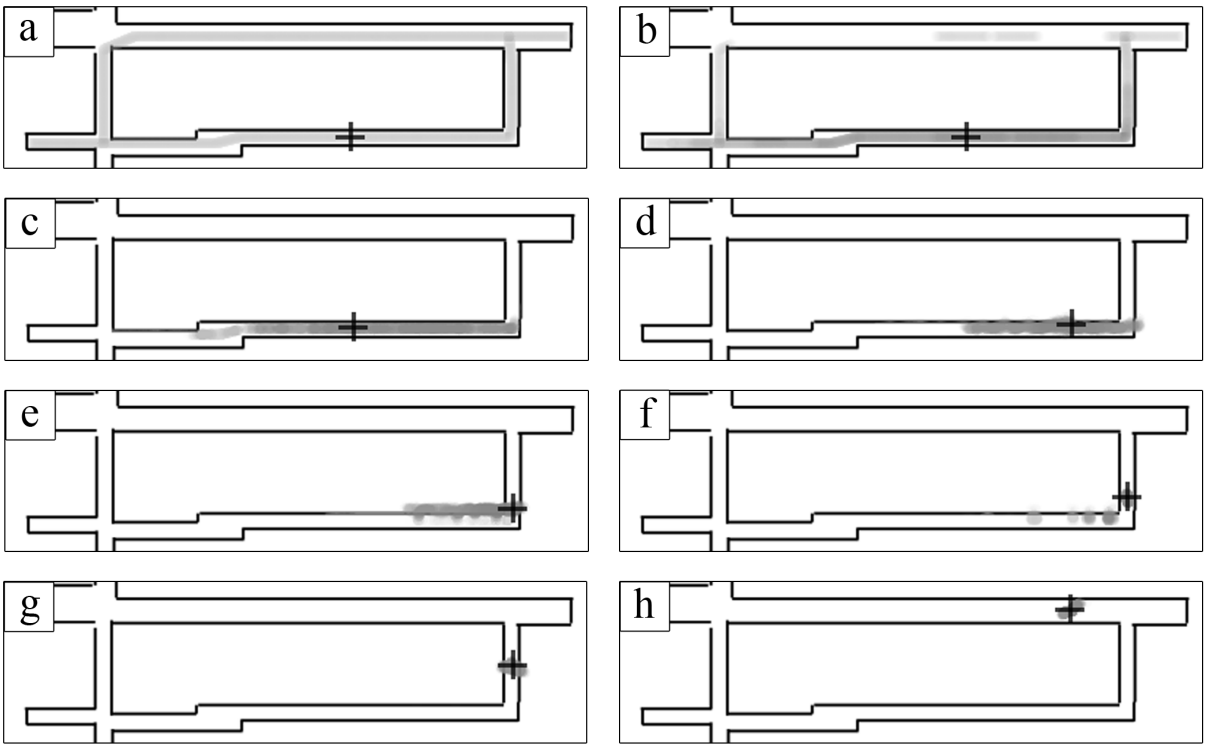

Fig. 6. Time snapshots of test run

VI. CONCLUSION

In this paper, we introduced an algorithm for the localization of an indoor mobile robot on a map as represented by a graph. The data collected during the Map Learning Phase of the algorithm is used to generate a perceptual model for the robot's location hypotheses, and entirely encapsulates the region of the map traversable by the robot. This, along with the application of rigid map constraints, led to our unique localization algorithm. By running the algorithm on our robot, we demonstrated the effectiveness of the algorithm.

REFERENCES

[1] Scott Lenser and Manuela Veloso, "Sensor resetting localization for poorly modelled mobile robots", *In Proc. IEEE International Conference on Robotics and Automation (ICRA-2000)*, San Francisco, CA, 2000.

[2] Cody Kwok and Dieter Fox, "Map-based Multiple Model Tracking of a Moving Object", *In The International RoboCup Symposium*, Lisbon, Portugal, 2004.

[3] Dieter Fox, Wolfram Burgard and Sebastian Thrun, "Markov Localization for Mobile Robots in Dynamic Environments", *In Journal of Artificial Intelligence Research*, 11:391-427, 1999.

[4] Kai O. Arras, Jos A. Castellanos and Roland Siegwart, "Feature-Based Multi-Hypothesis Localization and Tracking for Mobile Robots Using Geometric Constraints", *In Proc. IEEE International Conference on Robotics and Automation (ICRA-2002)*, Washington, DC, May 2002.

[5] Wolfram Burgard, Armin B. Cremers, Dieter Fox, Dirk Hhnel, Gerhard Lakemeyery, Dirk Schulz,Walter Steiner and Sebastian Thrun, "The Interactive Museum Tour-Guide Robot", *In Proc. of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, 1998.

[6] Wolfram Burgard, Andreas Den, Dieter Fox and Armin B. Cremersm, "Integrating Global Position Estimation and Position Tracking for Mobile Robots: The Dynamic Markov Localization Approach", *In Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98)*, Victoria, BC, 1998.

[7] Roland Siegwart, Kai O. Arras, Samir Bouabdallah, Daniel Burnier, Gilles Froidevaux, Xavier Greppin, Bjrn Jensen, Antoine Lorotte, Laetitia Mayor, Mathieu Meisser, Roland Philippsen, Ralph i, Guy Ramel, Gregoire Terrien and Nicola Tomatis, "Robox at Expo.02: A large-scale installation of personal robots", *In Special issues on Socially Interactive Robots, Robotics and Autonomous Systems*, 42:203-222, 2003.

[8] Sebastian Thrun, Maren Bennewitz, Wolfram Burgard, Armin B. Cremers, Frank Dellaert, Dieter Fox, Dirk Hhnel, Charles Rosenberg, Nicholas Roy, Jamieson Schulte and Dirk Schulz, "MINERVA: A Second-Generation Museum Tour-Guide Robot", *In Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[9] Margrit Betke and Leonid Gurvits, "Mobile Robot Localization Using Landmarks", *In IEEE Transactions On Robotics And Automation*, Vol. 13, No. 2, April 1997.

[10] Robert Sim and Gregory Dudek, "Mobile robot localization from learned landmarks", *In Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems (IROS)*, Victoria, BC, 1998.

[11] Sebastian Thrun, "Bayesian Landmark Learning for Mobile Robot Localization", *In Machine Learning*, 33(1), 1998.

[12] Claus B. Madsen and Claus S. Andersen, "Optimal landmark selection for triangulation of robot position", *In Robotics and Autonomous Systems*, 23(4):277-292, 1998.

[13] Sven Koenig and Reid G. Simmons, "Xavier: a robot navigation architecture based on partially observable markov decision process models", *In Artificial Intelligence and Mobile Robots: Case Studies of Successful Robot Systems*, D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, pages 91-122. AAAI Press//The MIT Press, Menlo Park, CA, 1998.

[14] Reid Simmons, Joaquin Fernandez, Richard Goodwin, Sven Koenig and Joseph O'Sullivan, "Xavier: An Autonomous Mobile Robot on the Web", *In Robotics and Automation Magazine*, IEEE, 1999.

[15] Dieter Fox, Sebastian Thrun, Wolfram Burgard and Frank Dellaert, "Particle Filters for Mobile Robot Localization", *In Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, editors, Springer, 2001.