Appearance-Based Mapping using Minimalistic Sensor Models

Paul E. Rybski, Stergios Roumeliotis,
Maria Gini, and Nikolaos Papanikopoulos

Abstract

This paper addresses the problem of localization and map construction by a mobile robot in an indoor environment. Instead of trying to build high-fidelity geometric maps, we focus on constructing topological maps as they are less sensitive to poor odometry estimates and position errors. We propose a modification to the standard SLAM algorithm in which the assumption that the robots can obtain metric distance/bearing information to landmarks is relaxed. Instead, the robot registers a distinctive sensor "signature", based on its current location, which is used to match robot positions. In our formulation of this non-linear estimation problem, we infer implicit position measurements from an image recognition algorithm. We propose a method for incrementally building topological maps for a robot which uses a panoramic camera to obtain images at various locations along its path and uses the features it tracks in the images to update the topological map. The method is very general and does not require the environment to have uniquely distinctive features. Two algorithms are implemented to address this problem. The Iterated form of the Extended Kalman Filter (IEKF) and a batch-processed linearized ML estimator are compared under various odometric noise models.

1 Introduction

Solving the Simultaneous Localization and Mapping (SLAM) problem for small, resource-limited robots means doing so without the aid of good odometric estimates and accurate metric range sensors. This causes a problem for traditional solutions to the SLAM problem which typically require one or both of the above. The motivating factor for this research is the necessity of doing SLAM on custom miniature robots, called Scouts [31] (Fig. 1), that our research group has developed.

We propose a modification to the standard SLAM algorithm in which we relax the assumption that the robots can obtain metric distance and/or bearing information to landmarks. In this approach, we obtain purely qualitative measurements of landmarks where a location "signature" is used to match robot pose locations. Landmarks correspond to sensor readings taken at various (x, y) positions along the path of the robot. This is a divergence from most SLAM approaches where landmarks represent specific objects of a known type in the environment such as edges, corners, and doors.

In this paper, we describe two methods to solve this particular variation of the SLAM problem. The first is an on-line method by which the Iterated form of the Extended Kalman Filter (IEKF) processes all measurements, including both actual odometric and inferred relative positions (cf. section 3.2), and estimates the coordinates of the locations where images were recorded along the trajectory of the robot. In this method, landmarks correspond to images taken at various (x, y) positions of the robot. The second method is a batch-processed linearized ML algorithm which addresses some of the shortcomings of the IEKF method. The IEKF method has the advantage of being able to run in real-time and produce an estimate as the robot navigates around the environment. The ML algorithm has the advantage of having all of the data to process at once. This tends to produce robust estimates as it is capable of handling the nonlinearities in the system in an iterative and more robust fashion (i.e. all Jacobians are computed at each iteration using the new improved state estimate).

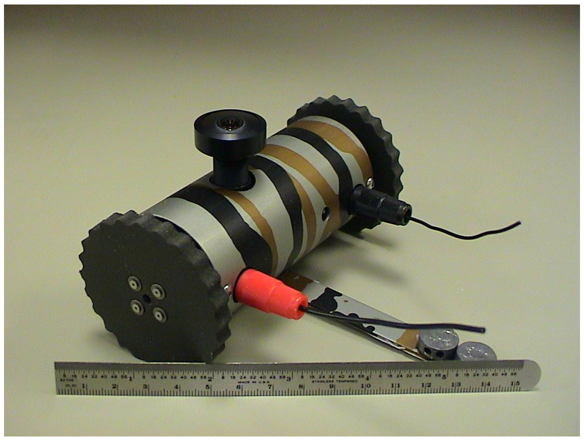


Figure 1: Scouts, due to their small size (11cm long and 4cm diameter), are limited to a monocular camera as their only exteroceptive sensor. Their limited on-board computing capabilities also make them totally dependent on a wireless proxy-processing scheme in which off-board run the software necessary to handle behavior control as well as the processing of the robot's video data. The Scout used in this work is fitted with an upward-facing Omnitech 190° fisheye lens. The lens provides 360° horizontal field of view around the robot, effectively functioning as an omnicamera. The robot is 11 cm long and 4 cm in diameter.

This remainder of this paper is organized as follows: Related work is described next in Section 2. Sections 3 and 4 describes the Extended Kalman Filter estimator and Batch Maximum Likelihood estimators, respectively. Experimental results are shown in Section 5 and the paper is summarized in Section 6.

2 Related Work

The Extended Kalman Filter has been used for localizing [15] and performing SLAM [35] on mobile robots for at least a decade. Our approach differs from traditional EKF estimators in that we do not have the ability of resolving specific geometric information about the landmarks we observe in our environment. Instead, the landmark positions are explicitly coupled to the position of the robot.

In previous implementations of SLAM algorithms, it is frequently assumed that the robot is able to measure its relative position with respect to features/landmarks [4] [22] or obstacles [38] in the area that it navigates. This implies that the robot carries a distance measuring sensor such as a sonar or a laser scanner. The algorithms described in this work are designed for robots that have no such sensor modality.

Bayesian methods have also been used for mobile robot localization (such as Markov Localization) and mapping [38] where the modes of arbitrary robot pose distributions are represented in a discretized grid. Statistical methods such as Monte Carlo localization [39] use sampling techniques to more quickly estimate the distribution of possible robot poses. Most recently, a method of factoring complex joint probability distributions, known as Rao-Blackwellization [21], has been employed for stochastic robot mapping and localization in an algorithm called FastSLAM [19]. Distributed and hierarchical factorizations for the particles in a map have also been proposed [6]. In general, all of these methods typically use very accurate sensors and/or robots with very accurate odometry that allow them to resolve accurate maps over large distances.

In contrast to explicit metric-based methods, more qualitative methods such as topological maps of nodes have been used as well [33] [2]. Of special note is the research into cognitive spatial representations suggested by Ben Kuipers [14] in the Semantic Spatial Hierarchy (SSH) [13] [29]. Locations are explicitly designated by distinctive (but not necessarily unique) sensor signatures. Our work is inspired by the SSH philosophy and attempts to wrap it into a more formal and robust representation using the maximum likelihood techniques. Another closely related area of research is the use of sensor "fingerprints" of places for robot navigation [36]. This approach illustrates an ellegant technique by which a robot can build a map and disambiguate similar locations through the use of a POMDP. In contrast, the approach described in this paper uses a maximum likelihood algorithm and a very rich location sensor signature represenation to help disambiguate similar locations.

Stochastic sampling techniques for searching through the space of stochastic maps have recently been proposed using MCMC [27] as well as a Rao-Blackwellized (factored) Particle Filter (RBPF) [28] techniques. The SSH and other approaches, such as reported in [10], merge metric with topological approaches in order to take the best aspects of both worlds.

Physics-based models that involve spring dynamics have been used quite effectively to find minimum energy states in topological map structures [5] [9]. In previous work [30], we describe an ad-hoc physics-based method that uses spring and mass dynamics to minimize the energy of the topological map. We have had some success with these methods but have found that the parameter choices for the models tend to be very important and that numerically solving for the set of non-linear equations can be unstable. Recently, a method using stochastic gradient descent has been proposed for loop closure in very high-dimensional datasets that appears very promising in both numerical accuracy as well as computational speed [24].

Spatial reasoning algorithms that make use of visual information for landmarks typically fall into two major categories in terms of the features that are extracted. In the first of these two categories, specific features are extracted from each image and are used as a "signature" of the location where that image was taken. In the second category, the entire image is treated as a single high-dimensional feature.

Examples of the first category include [34] [12] [1] where the SIFT [16] feature detector is used to identify "landmarks" in the images that are used as the input to a probabilistic representation of the robot's position. In [41], image signatures captured from an omnidirectional camera are used to construct a topological map of an environment by generating histograms of the RGB and HSV (Hue, Saturation, and Value) components. In [37], visual landmark information is extracted and used as a signature in a formalism called Bayesian Programming (a generalization of Bayesian Networks [20]) for localization of the robot. In [23], visual recognition of landmarks that are used for the identification of loop closure is used to augment a laser-based SLAM approach. Structure From Motion (SFM) algorithms, such as described in [3], compute the correspondences between features extracted from multiple images to estimate the geometric shape of landmarks as well as to estimate the robot's pose. However the applicability of this algorithm is conditioned on the existence of a sufficient number of uniquely identifiable individual features along the trajectory of the robot.

Examples of the second category include [26] where subspace methods are used to map the images to a much lower-dimensional manifold. In [8], a spectral-clustering-like algorithm is proposed which clusters the images to appropriately describe the topology of the map. In practice, our vision system could be replaced by any other kind of sensor modality. The sensor models that we use abstract the specifics of the sensor and create instead a boolean sensor abstraction layer which can report whether the robot has re-visited a location.

In contrast to these previous approaches, our approach is especially suited for use on small mobile robots where the computational power and/or the communications bandwidth between sensor and processors is very low. In our particular implementation, we make use of a feature detection and tracking algorithm (KLT [17] [40]) to visually identify a set of sparse distinguishing features in an omnidirectional image captured from the robot's camera. This feature set becomes a "signature" that is used to determine whether the robot has completed a cycle in its path. These intersections become "landmarks" in the robot's map and serve as constraints that help to correct for accumulated odometric error in the robot's estimated trajectory.

3 Extended Kalman Filter Estimator

This section describes how an Extended Kalman filter (EKF) estimator for an appearance-based mapping system can be derived. As described previously, such a system uses an environmental sensor that neither relies on any specific type of features, nor takes distance measurements to landmarks. Such a sensor determines a signature for distinct locations along the robot's path, stores the signature and the estimated pose of the robot at that time instant, and finally retrieves that information once the robot revisits the same area. Determining whether the robot is at a certain location for a second time is the key element for providing positioning updates. By correlating two scenes, a relative position measurement can be inferred and be used to update both the current and previous (at locations visited in the past) pose estimates for the robot.

3.1 Propagation Equations

The Extended Kalman filter uses a model of the robot kinematics to compute an estimate of the robot's position at discrete timesteps. Associated with this state estimate is a covariance matrix which represents the uncertainty in the robot's position estimates over time. Our model of robot motion consists of a 3D pose vector (x, y, ϕ) (2D pose and orientation). Our derivation of the EKF is based on an indirect model where which uses error-state propagation [18]. The relevant details of the propgation equations are provided here for completeness. For further details of the derivation, please refer to [32].

The measured linear (V_m) and angular velocity (ω_m) are used to recursively update the robot's pose at discrete time steps as shown in the following equation:

$$\begin{aligned} \hat{x}_r(k+1) &= \hat{x}_r(k) + V_m(k)\delta t \cos\hat{\phi}_r(k) && (1) \\ \hat{y}_r(k+1) &= \hat{y}_r(k) + V_m(k)\delta t \sin\hat{\phi}_r(k) && (2) \\ \hat{\phi}_r(k+1) &= \hat{\phi}_r(k) + \omega_m(k)\delta t && (3) \end{aligned}$$

The linearized discrete-time *error-state* propagation equation in global coordinates is the difference between the estimated state and the (unknown) true state:

$$\begin{bmatrix} \widetilde{x}_r \\ \widetilde{y}_r \\ \widetilde{\phi}_r \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & 0 & -V_m\delta t \sin\phi_r \\ 0 & 1 & V_m\delta t\cos\phi_r \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \widetilde{x}_r \\ \widetilde{y}_r \\ \widetilde{\phi}_r \end{bmatrix}_{k} + \begin{bmatrix} \delta t\cos\phi_r & 0 \\ \delta t \sin\phi_r & 0 \\ 0 & \delta t \end{bmatrix} \begin{bmatrix} w_v \\ w_\omega \end{bmatrix} \tag{4}$$

or

$$\widetilde{X}_R(k+1) = \Phi_R(k)\widetilde{X}_R(k) + G_R(k)W_R \tag{5}$$

with

$$\Phi_R(k) = \begin{bmatrix} 1 & 0 & -V_m\delta t \sin\phi_r \\ 0 & 1 & V_m\delta t\cos\phi_r \\ 0 & 0 & 1 \end{bmatrix}, \; G_R(k) = \begin{bmatrix} \delta t\cos\phi_r & 0 \\ \delta t \sin\phi_r & 0 \\ 0 & \delta t \end{bmatrix}, \; W_R = \begin{bmatrix} w_v \\ w_\omega \end{bmatrix} \tag{6}$$

With each motion, the state vector is propagated according to these equations. Odometric errors will continuously decrease the quality of the robot's estimate which will increase the robot's position uncertainty. The covariance propagation takes place at time $k+1$ but at this point, the robot has not made a new observation with its sensors since time k. This dual time index is represented by the notation $k+1|k$. The equation for the propagation of the robot's position error covariance matrix at time $k+1|k$ is:

$$\begin{aligned} P_R(k+1|k) &= E[\widetilde{X}(k+1)\widetilde{X}^T(k+1)] \\ &= \Phi_R(k)P_R(k|k)\Phi_R^T(k) + G_R(k)Q_R G_R^T(k) && (7) \end{aligned}$$

The Q_R matrix from the state error covariance propagation in Equation (7) represents the noise covariance of the robot's translational and rotational velocity. For a differentially-driven robotic platform such as the Scout, where linear and rotational velocities are a function of the left v_l and right v_r wheel speeds, i.e. $V_m = \frac{(v_l+v_r)}{2}$, $\omega_m = \frac{(v_l-v_r)}{\alpha}$, this matrix is defined as:

$$Q_R = \begin{bmatrix} \frac{1}{4}(\sigma_{v_l}^2 + \sigma_{v_r}^2) & \frac{1}{2\alpha}(\sigma_{v_l}^2 - \sigma_{v_r}^2) \\ \frac{1}{2\alpha}(\sigma_{v_l}^2 - \sigma_{v_r}^2) & \frac{1}{\alpha^2}(\sigma_{v_l}^2 + \sigma_{v_r}^2) \end{bmatrix} \tag{8}$$

where σ_{v_l} and σ_{v_r} are the standard deviations of the wheel speed noises and α is the length of the wheelbase. As can be seen, the linear and rotational velocities are correlated as long as the standard deviations of the linear and rotational velocity are non-zero and not equal.

The 2D pose (x, y) of the landmarks must also be estimated when mapping and so these quantities must also be integrated into the state vector. Unlike the robot, the coordinates of the landmark locations X_{L_i} do not change over time. Thus the full state vector X contains all of the poses to be estimated of the robot X_R along with all the landmarks X_{L_i}. The error-state propagation and covariance equations are the same form as equations 5 and 7, respectively, but where the state vector and covariance matrix are of dimension $N+1$ where N is the number of landmarks.

3.2 Update Equations

If the robot were to only propagate its state estimates and corresponding covariance using the above equations, the covariance would increase without bounds. To correct for odometric errors and to reduce the uncertainty, the robot must take sensor readings and compare those with the expected ones given the current state estimates.

For appearance-based mapping, a sensor modality is preferred that neither relies on any specific type of features, nor requires distance measurements. Using the robot's sensor, a unique visual signature for distinct locations along the robot's path can be obtained. These signatures are associated with the estimated pose of the robot at that time instant, and can be retrieved once the robot revisits the same area. Determining whether the robot is at a certain location for a second time is the key element for providing positioning updates. By correlating any two scenes, we can infer a relative position measurement and use it to update both the current and previous pose estimates (at locations visited in the past) for the robot. This in effect will produce an accurate map of distinct locations within the area that the robot has explored. In effect, the landmarks that the robot detects explicitly represent the specific locations that the robot has visited.

Every time the robot takes an image of its surroundings, it employs an algorithm to determine whether the sensor reading corresponds to a previously seen locations X_{L_i}, or to a novel location X_{L_j}. We use the above notion of an appearance-based sensor model (more thoroughly described in [30]) and define the sensor reading to be:

$$\begin{aligned} Z_i(k+1) &= 0_{2\times 1} + N_{z_i}(k+1) \\ &= {}^{\mathcal{R}}X_{L_i} + N_{z_i} \end{aligned} \tag{9}$$

where ${}^{\mathcal{R}}X_{L_i}$ is the landmark's state vector in the robot's coordinate system $\mathcal{R}$, and $N_{z_i}(k+1)$ is Gaussian measurement noise. The $0_{2\times 1}$ value is an *inferred* sensor reading which reflects the robot's assertion that its physical location directly corresponds to the sensor reading. That is, the only way the robot could receive this sensor reading is if ${}^{\mathcal{R}}X_R$ is the same location as ${}^{\mathcal{R}}X_{L_i}$. The robot is assumed not to have any other way to measure distances to landmarks and so any erroneous displacement in this reading is captured by the noise term $N_{z_i}(k+1)$.

The inferred and estimated sensor readings are:

$$Z_i = {}^{\mathcal{G}}_{\mathcal{R}}C^T(\phi_r)\left(X_{L_i} - p_R\right) + N_{z_i} \tag{10}$$

$$\hat{Z}_i = {}^{\mathcal{G}}_{\mathcal{R}}C^T(\hat{\phi}_r)\left(\hat{X}_{L_i} - \hat{p}_R\right) \tag{11}$$

where $p_R = [x_r \ y_r]^T$, $\hat{p}_R = [\hat{x}_r \ \hat{y}_r]^T$ and

$${}^{\mathcal{G}}_{\mathcal{R}}C(\phi_r) = \begin{bmatrix} \cos\phi_r & -\sin\phi_r \\ \sin\phi_r & \cos\phi_r \end{bmatrix} \tag{12}$$

is the rotation matrix that relates the orientation of the frame of reference $\mathcal{R}$ on the robot with the global coordinate frame $\mathcal{G}$. By subtracting the true sensor reading from the estimated sensor reading, the linearized measurement error is computed as:

$$\begin{aligned}
\tilde{Z}_i &= Z_i - \hat{Z}_i \\
&\simeq \begin{bmatrix} -C^T(\hat{\phi}_r) & -C^T(\hat{\phi}_r)J\left(\hat{X}_{L_i} - \hat{p}_R\right) & \vdots & C^T(\hat{\phi}_r) \end{bmatrix} \begin{bmatrix} \tilde{p}_R \\ \tilde{\phi}_r \\ \tilde{X}_{L_i} \end{bmatrix} + N_z \\
&= \begin{bmatrix} H_R & \vdots & H_{L_i} \end{bmatrix} \begin{bmatrix} \tilde{X}_R \\ \tilde{X}_{L_i} \end{bmatrix} + N_{z_i} \qquad (13)
\end{aligned}$$

Adding entries for all of the variables, the full equation is expressed as :

$$\begin{aligned}
\tilde{Z}_i &= \begin{bmatrix} H_{R_i} & 0 & \dots & 0 & H_{L_i} & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \tilde{X}_R \\ \tilde{X}_{L_1} \\ \vdots \\ \tilde{X}_{L_{i-1}} \\ \tilde{X}_{L_i} \\ \tilde{X}_{L_{i+1}} \\ \vdots \\ \tilde{X}_{L_N} \end{bmatrix} + N_{z_i} \\
&= H_i \tilde{X} + N_{z_i} \qquad (14)
\end{aligned}$$

The H_i matrix is used to update the state estimate for the pose of the robot X_R and the positions of the landmarks X_{L_i} every time an image is recorded. The remaining update equations are:

$$\begin{aligned}
r(k+1) &= Z(k+1) - \hat{Z}(k+1) && (15) \\
S(k+1) &= H(k+1)P(k+1|k)H^T(k+1) + R(k+1) && (16) \\
K(k+1) &= P(k+1|k)H^T(k+1)S^{-1}(k+1) && (17) \\
\hat{X}_(k+1|k+1) &= \hat{X}(k+1|k) + K(k+1)r(k+1) && (18) \\
P(k+1|k+1) &= P(k+1|k) - K(k+1)S(k+1)K^T(k+1) && (19)
\end{aligned}$$

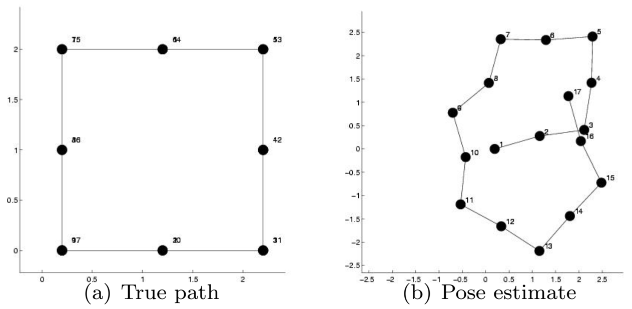
The difference between the measured and estimated sensor reading in Equation (15) is called the residual. The covariance matrix of the residual is shown in Equation (16). These two values are used to compute the Kalman gain (which affects how much to change the state vector based on the correction required) in Equation (17) which is used to update the state vector and state covariance in Equations (18) and (19), respectively.

3.2.1 Iterative Extended Kalman Filter

Since the accuracy of this update depends on the accuracy of the linearization, we employ the Iterated form of the Extended Kalman Filter (IEKF) [7], [18]. First, the IEKF linearizes the measurement equation Equation (13) around the current estimate $X(k+1|k)$ of the state and calculates the updated state estimate $\hat{X}(k+1|k+1)$ using Equations (15), (16), (17), (18). Then, the filter resets $X(k+1|k)$ to this updated value and the same process is repeated until it converges (the rate of change in the state estimate drops below a preset threshold). The state covariance $P(k+1|k)$ is *not* updated with Equation (19) until after the state estimate has converged.

3.3 Simulation Experiment

This method was tested on a simulated Scout robot. The standard deviation of the estimated wheel encoder error was 1.4 cm/s. The true path of the simulated robot is shown in Figure 2(a) as a square that is traversed twice. Sensor snapshots are taken roughly every 0.5 m as the robot traverses the path. The first time around the loop, the robot is essentially in an exploration mode. Each landmark that it observes is unique and thus, it adds the estimated positions of those landmarks (i.e. robot positions where the images were taken) directly to the state vector. Since the robot has no other information on those landmarks, the first sighting is the best information available. The second time around the loop, the robot re-discovers the landmarks that it saw on its first pass. If the Kalman update procedure is used, the odometric error in the robot's position will be reduced. In addition to correcting the current position estimate, each of the previous landmark positions will also be corrected. If no update step is done, as shown in Figure 2(b), the robot's path estimate will be very poor and multiple positions will exist for the same landmark.



(a) True path (b) Pose estimate

Figure 2: True and estimated (corrupted by odometric error) paths for the simulation experiments. The path starts from the lower left, moves counter-clockwise, and is traversed twice. Sensor readings are taken at the corners of the square and at the midpoints of each path leg. The scale is in meters.

As the robot moves through the environment without any sensor updates, the certainty in its odometric estimate becomes increasingly worse. The covariance of the robot's position with no landmark corrections is shown in Figure 3.

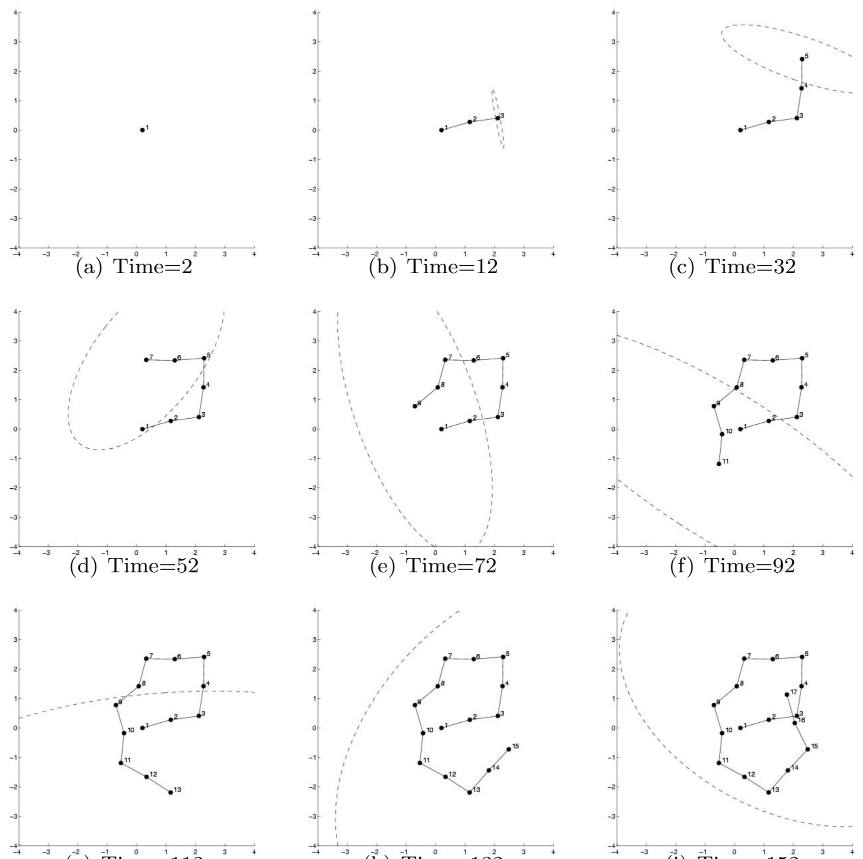
In contrast, Figure 4 shows the landmark positions and position uncertainty of each location after correction by correlating the robot's position with the sensor readings. After the initial path around the cycle, the first subfigure (timestep 71) shows the large uncertainty accumulated in the robot's position. At timestep 72, the first update step is done and the uncertainty is greatly diminished. This is mostly due to the small covariance of the sensor reading vs. the large covariance of the robot's odometric propagation. After propagating to timestep 86, the error covariance of the robot's path estimate (shown as a dashed line) has generated a somewhat substantial error. This error is once again diminished in timestep 87 when another previously-seen landmark is observed.

Figure 5 illustrates how the estimated landmark positions are improved by using the IEKF and how the sensor residual (the error between estimated and true sensor reading) improves with respect to the 3σ upper and lower bounds of the residual covariance estimates.

4 Batch Maximum Likelihood Estimator

The EKF is a recursive "real-time" estimator which processes each sensor reading as it arrives. An alternative approach is to wait until all of the sensor readings have arrived and then process all of the data at once. This section describes how to formulate such a batch maximum likelihood (ML) estimator as a summation of cost functions that must be minimized.

Two separate cost functions must be defined. The first cost function represents the odometric estimate of the robot's pose and is described as:



(a) Time=2 (b) Time=12 (c) Time=32

(d) Time=52 (e) Time=72 (f) Time=92

(g) Time=112 (h) Time=132 (i) Time=152

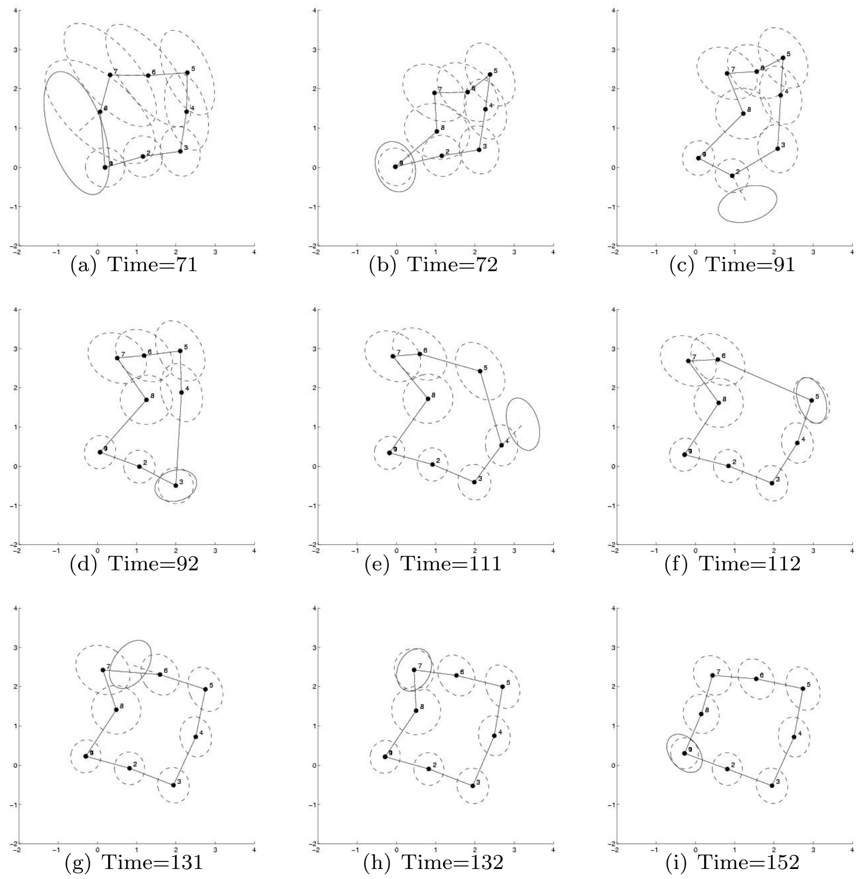
Figure 3: Propagation of uncertainty as the robot traverses its environment. No Kalman update step is done and so multiple positions exist for each landmark measurement and the position estimate becomes progressively worse with each step. Each subfigure represents the location where the robot has taken a sensor reading. The 3σ region of uncertainty is shown surrounding the robot's estimated position.

$$(y_i - h_{y_i}(X))^T P_i^{-1} (y_i - h_{y_i}(X)) \tag{20}$$

where y_i is a vector that describes the measured displacement between the previous position measured at time $i-1$ and the current position measured at time i. The function $h_{y_i}(X)$ computes the predicted displacement of the robot given the current state vector from time $i-1$ to time i. The covariance of this measurement is P_i.

As described in the previous section, the state vector of a maximum likelihood estimator consists of all of the necessary parameters to solve for. For the case of a mobile robot moving on a 2D surface, the variables represent individual locations to which the robot has traveled. In the previous section, this was the set of sensor readings S. It was also assumed that the robot only traveled to D distinct locations and that $|D| < |S|$. This assumption is not quite true since while the robot may have traveled near the same location several times, those exact positions of the robot were not completely identical. That means that simply merging the nodes as was done previously will not provide the most accurate estimate. Thus, S and D will have the same number of elements and the cost function associated with the sensor readings is:

$$(z_i - h_{z_i}(X))^T R_i^{-1} (z_i - h_{z_i}(X)) \tag{21}$$


(a) Time=71 (b) Time=72 (c) Time=91

(d) Time=92 (e) Time=111 (f) Time=112

(g) Time=131 (h) Time=132 (i) Time=152

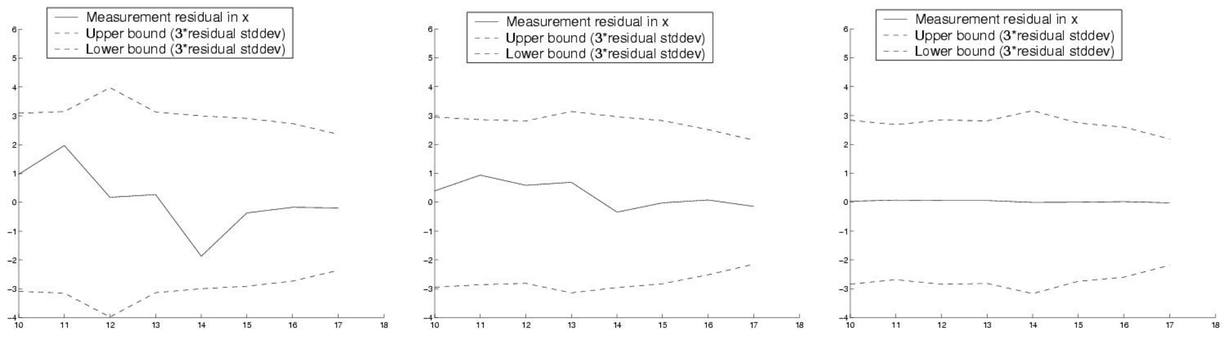
Figure 4: Propagation of uncertainty as the robot traverses its environment with Kalman update correction. Each pair of images shows the estimated position of the robot with uncertainty the timestep before and after the sensor reading was taken and the landmark positions were correlated. The estimated path of the robot just before the update is drawn with a dashed ellipse. The 3σ region of uncertainty is shown surrounding the robot's estimated position as a solid ellipse.

In the sensor cost function, z_i is a vector that describes the measured displacement between a position measured previously at time j (not limited to time $i-1$) and the current position at time i. Using the notion of the appearance-based sensor, the value of z_i will always be 0 since the landmarks correspond directly to the positions of the robot. The function $h_{z_i}(X)$ computes the predicted displacement of the robot given the current state vector from the previously-seen location at time j to the current time i. The covariance of this measurement is R_i.

As the robot discovers new landmarks, it adds their positions to the state vector and marks those variables as the locations of the original sightings. When the robot re-discovers a landmark, it also adds this position to its state vector, but flags it as previously-seen. The sensor cost function in Equation (21) always compares the current measured position of a landmark against the first discovered position of that landmark.

Combining the motion and sensor cost functions (Equations (20) and (21)), the complete cost function is:

$$\sum_i (y_i - h_{y_i}(X))^T P_i^{-1} (y_i - h_{y_i}(X)) + \sum_j (z_j - h_{z_j}(X))^T R_j^{-1} (z_j - h_{z_j}(X))$$



1 iteration 2 iterations 4 iterations

Figure 5: The effect of different numbers of iterations in the update step of the IEKF. The plots show the sensor residual $r = z - \hat{z}$ and the 3σ upper and lower bounds of the residual covariance S. The plots are of the x position of the robot. The y position (not shown) has similar characteristics. These residuals are all for landmark positions that have been visited a second time.

The number of motion cost function terms is the number of sensor readings minus one, $|S| - 1$. The number of sensor cost function terms corresponds to the number of non-unique landmarks the robot has identified.

4.1 Linearized Estimator

The non-linear nature of this problem, introduced by the need to handle the rotational component of the robot, means that finding the best solution can be analytically and computationally challenging. The method that we use for finding the minimum solution is to linearize the system with a first-order linear approximation such as a Taylor series expansion. Thus, the sensor and motion measurement functions take the form:

$$\begin{aligned} h(X) &\simeq h(\hat{X}) + \nabla_X h(X)\Big|_{X=\hat{X}} (X - \hat{X}) \\ &\simeq h(\hat{X}) + H(X - \hat{X}) \end{aligned} \tag{22}$$

where X is the true (unknown) state vector, $\hat{X}$ is the robot's estimate of the state vector, and H is the Jacobian of the cost function h. Expanding this equation for each of the cost functions and taking the first derivative to solve for its minimum, a recursive formulation of the estimator can be found which is quadratic in X. To minimize the function with respect to X, the first derivative is taken and the equations are set to 0. This results in the following equation:

$$X = \hat{X} + \left(\sum_{i=1}^{n-1} H_{y_i}^T P_i^{-1} H_{y_i} + \sum_{i=1}^{n} H_{z_i}^T R_i^{-1} H_{z_i} \right)^{-1} \left(\sum_{i=1}^{n-1} H_{y_i} P_i^{-1} \left(y_i - h_{y_i}(\hat{X}) \right) + \sum_{i=1}^{n} H_{z_i} R_i^{-1} \left(z_i - h_{z_i}(\hat{X}) \right) \right) \tag{23}$$

where the $\hat{X}$ on the right-hand side of the equation is the initial estimate of the system (see [32] for more details about the derivation).

The first value of this estimate can be obtained from the robot's raw odometry, if no other estimate is available. This is a recursive form where the result from the left-hand side of the equation is plugged back into the equation on the right-hand side. This first-order linear approximation of the measurement function is only valid for small errors in the estimate of X. As the equations are iterated, the state estimate will continue updating until it converges to a stable solution.

The derivations of the cost functions for the odometry propagation and place sensor measurements are described below:

4.1.1 Odometry Propagation Measurement

The measurement function for the displacement estimates between subsequent nodes based on their odometry is defined as:

$$h_{y_i}(X) = \begin{bmatrix} {}^{\mathcal{G}}_{\mathcal{R}}C^T(\phi_r) & 0_{2x1} \\ 0_{1x2} & 1 \end{bmatrix} \left(X_{L_i} - X_{L_{i-1}}\right) \tag{24}$$

where $X_{L_i} = [x_i \; y_i \; \phi_i]^T$ and $X_{L_{i-1}} = [x_{i-1} \; y_{i-1} \; \phi_{i-1}]^T$ are the positions of the robot at time i and $i-1$, respectively, and ${}^{\mathcal{G}}_{\mathcal{R}}C(\phi_r)$ is the same as equation 12.

The first-order Taylor approximations of the odometry measurement function is defined as:

$$\begin{aligned} \tilde{y}_i &= \begin{bmatrix} -C^T(\hat{\phi}_{L_{i-1}}) & -C^T(\hat{\phi}_{L_{i-1}})J\left(\hat{X}_{L_i} - \hat{X}_{L_{i-1}}\right) & \vdots & C^T(\hat{\phi}_{L_{i-1}}) & 0_{2x1} \end{bmatrix} \begin{bmatrix} \tilde{X}_{L_{i-1}} \\ \tilde{X}_{L_i} \end{bmatrix} \\ &= \begin{bmatrix} H_{L_{i-1}} & \vdots & H_{L_i} \end{bmatrix} \begin{bmatrix} \tilde{X}_{L_{i-1}} \\ \tilde{X}_{L_i} \end{bmatrix} \\ &= H_{y_i} \begin{bmatrix} \tilde{X}_{L_{i-1}} \\ \tilde{X}_{L_i} \end{bmatrix} \end{aligned} \tag{25}$$

$\tilde{X}$ is the error in the state estimate $\hat{X}$). These expressions for the error terms are only important for calculating the Jacobian and are not used for any other part of the estimator.

4.1.2 Place Sensor Measurement

The measurement function for the estimated distance between two nodes based on the virtual place sensor's reading that are on the same location is defined as:

$$h_{z_i}(X) = \left(X_{p_i} - X_{p_j}\right) \tag{26}$$

where $X_{p_i} = [x_i \; y_i]^T$ and $X_{p_j} = [x_j \; y_j]^T$ are the global 2D poses of the robot's position (orientation is not considered). Orientations of these landmarks are not tracked as some sensor modalities may not have an orientation associated with their readings.

Likewise, the first-order Taylor approximations of the place sensor is defined as:

$$\tilde{z}_i = \begin{bmatrix} -1 & 0 & \vdots & 1 & 0 \\ 0 & -1 & \vdots & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{X}_{p_j} \\ \tilde{X}_{p_i} \end{bmatrix}$$

$$= \begin{bmatrix} H_{p_j} & \vdots & H_{p_i} \end{bmatrix} \begin{bmatrix} \tilde{X}_{p_j} \\ \tilde{X}_{p_i} \end{bmatrix} \tag{27}$$

$$= H_{z_i} \begin{bmatrix} \tilde{X}_{p_j} \\ \tilde{X}_{p_i} \end{bmatrix} \tag{28}$$

4.2 Simulation Results

This estimator was run on the simulated data shown in Figure 6(b). Figure 7 shows plots of the covariance matrices associated with each of the individual odometry readings at each of the locations where sensor readings were taken. Each odometric reading is considered to be independent of each other and thus, the covariance matrices are only defined between a single pair of sensor readings. Because of the nonlinearities in the system, this ML algorithm must be iterated several times until convergence. The convergence of this algorithm also depends greatly upon the initial positions of the nodes.

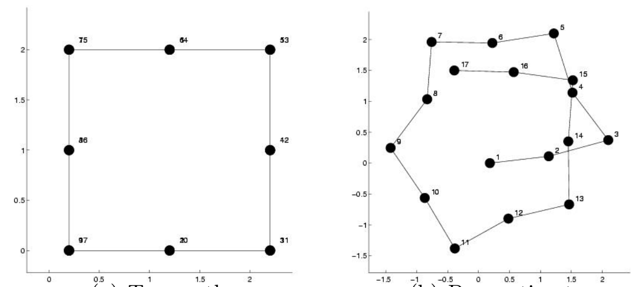

(a) True path (b) Pose estimate

Figure 6: True and estimated (corrupted by odometric error) paths for the simulation experiments. The path starts from the lower left, moves counter-clockwise, and is traversed twice. Sensor readings are taken at the corners of the square and at the midpoints of each path leg. The scale is in meters.

Figure 8 illustrates the multi-step process of how the linearized ML estimator converges to a solution. The uncorrected odometric readings are used as the initial estimate for the state vector. The iterative process was stopped when the average landmark update per iteration dropped below 0.001 m. In this experiment, only four iterations of the algorithm were necessary before the stopping condition was reached. Because the algorithm is a closed-form solution, the computational complexity is based on the time required to invert the covariance matrix. This is order $O(n^3)$ where n is the number of landmarks.

5 Experimental Results

The miniature Scout robots were used as the experimental platform for this work. Scouts, such as the one shown in Figure 1, are differentially-driven robots 11cm long and 4cm in diameter. Because of the small size of the Scout, a camera is the *only* extereoceptive sensor that is used. Video data is transmitted from the robot to an off-board workstation for processing as the robot's on-board computers are insufficient to process its own video stream. All of the algorithms described in this paper are executed on the off-board computers and operate on the video data stream transmitted from the robot.

For these experiments, a Scout robot was teleoperated around an environment (in order to collect ground truth) and image data was captured from its camera. In their original design, the Scouts were equipped with forward-facing cameras with a 65° field of view. For this work the Scout has been equipped with an upward-facing 190° vertical/360° horizontal field of view lens from Omnitech Robotics [25]. An example image taken from this camera and the corresponding de-warped image is shown in Figure 9.

In order to compute a signature for each location visited, a set of features must be identified and extracted from the image. However, in the most general case, the robot will be required to explore a completely unknown environment and as such, a specific feature detection algorithm chosen ahead of time could fail to find a distinctive set of features.

For this work, the Lucas-Kanade-Tomasi (KLT) feature tracking algorithm is used to compare images to determine the degree of match. The KLT algorithm consists of a registration algorithm that makes it possible to find the best match between two images [17] as well as a feature selection rule which is optimal for the associated tracker under pure translation between subsequent images [40]. An implementation of the KLT algorithm[1] is used to identify and track features between successive images as a method for determining the match between two images. KLT features are selected from each of the images and are tracked from one image to the next taking into account a small amount of translation for each of the features. The degree of match is the number of features successfully tracked from one image to the next. A total of 100 features are selected from each image and used for comparison. To take into account the possibility that two panoramic images might correspond to the same location but differ only in the orientation of the robot, the test image is

[1]Originally developed by Stan Birchfield at Stanford University [11].

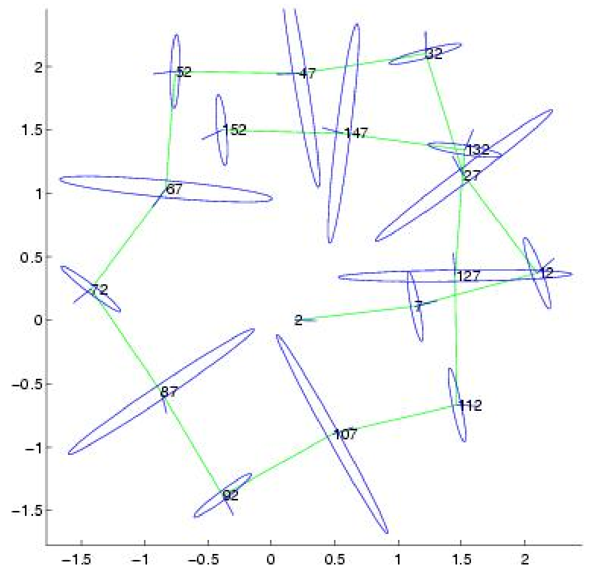


Figure 7: Uncertainty ellipses for each of the independent odometric readings used by the linearized maximum likelihood estimator.

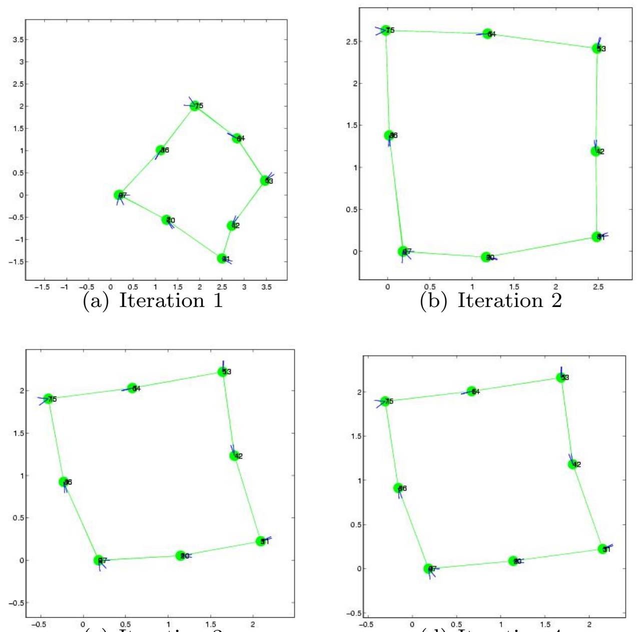
rotated through discrete angles (typically 16) the best match is found. The 16 rotated images are generated and cached when each new image is found. This operation only takes a few seconds per image on a 2.3GHz Intel Pentium-M processor. Figure 10 shows the 100 best features identified in an image and shows how many of those features are successfully tracked to the lower image.

This approach is similar in flavor to [16] in that the image is reduced in resolution for the sake of rapid matching. In that work, a pyramid structure involving several levels of dimensionality reduction is created from each image and different images are matched from the lowest resolution to the highest. In our case, the KLT features serve as a single level of "dimensionality reduction" that is used for matching one image with the next.

It is important to note that no attempt is made to track the features over multiple frames of video. This technique does not attempt to compute structure from motion on this data primarily because the algorithms described in this research will ultimately be run on robots that do not have real-time video processing capability.

While mapping, the mobile robot travels around an unknown area and stores images from its camera. KLT is used to compare images recorded at different locations along the trajectory of the robot. When the received image does not match a previously recorded one, it is assumed that this location is novel and is added to the state vector of landmarks. This constitutes an exploration phase where the robot creates its world model. The rate at which images are collected can either be uniform based on the robot's odometry, or it can be data-driven. In general, sensor readings are only necessary at points where a noticeable change in the number of matched features is detected. When the robot encounters an image which matches one that was previously seen, it considers these features to be the same and corrects its estimate of the landmark position.

The KLT algorithm and omnicamera setup are treated as a "virtual sensor" that returns true or false as to whether the robot has returned to a location that it has visited before. This information is given to the estimators and a relative position measurement $Z = 0_{2\times1} + N_z$ between the current position of the robot and that of the same location visited in the past is inferred. The accuracy of this measurement is inferred by the locus of points (forming an ellipsoid) around a location, with the characteristic that the images recorded at each of them are considered identical by the KLT. The parameters of the ellipse are computed empirically



(a) Iteration 1 (b) Iteration 2

(c) Iteration 3 (d) Iteration 4

Figure 8: Four steps in the convergence of the linearized maximum likelihood estimator. By the fourth step, the estimate has almost converged.

by how far the robot has to move from a particular location before the signature match fails.

Once a set of sensor readings are found to match, the physical (x, y) location of the robot is stored as a landmark in the state vector. This is different from other mapping approaches whereby specific objects in the environment are stored as landmarks. In our approach, the features identified in the image are only used for finding the correlation between images rather than being used to identify the positions of structure in the environment. The visual information is abstracted away to a boolean function which returns whether the robot has returned to the same location. Thus, only the robot's *position* where those sensor signatures line up is used as a landmark in the state vector.

The image matching algorithm is the most computationally expensive part of the mapping process. Finding the 100 best KLT features in an 1507x240 pixel image on a 2.3GHz Intel Pentium-M processor takes approximately 0.7 seconds. Tracking these features between one image and the next takes approximately 0.8 seconds. Because this process is repeated for each of the 16 different rotated images for a given location each new image must be compared against the history. As such, this algorithm will not run in real time for large numbers of stored images. However, due to the proxy-processing nature of the Scout robot, the image processing algorithm can be offloaded to any number of available off-board computers to help speed up the process through parallel processing of the image data.

5.1 Office Environment Experiment

The robot was moved around an environment in a path that intersected itself five times and an image was taken from the camera roughly every 0.3 m. The robot's path is shown in Figure 11.

The KLT algorithm was used to track features between each pair of images in order to find locations where the robot's path crossed itself. Figure 12(a) shows the true path of the robot and the locations where the path crossed itself and landmarks were thus observed. Figure 12(b) shows the estimated path of the robot as computed by the robot's noisy odometry readings. The estimated landmark positions observed during the run are shown as well. This figure does not assume that any sensor updates were made.

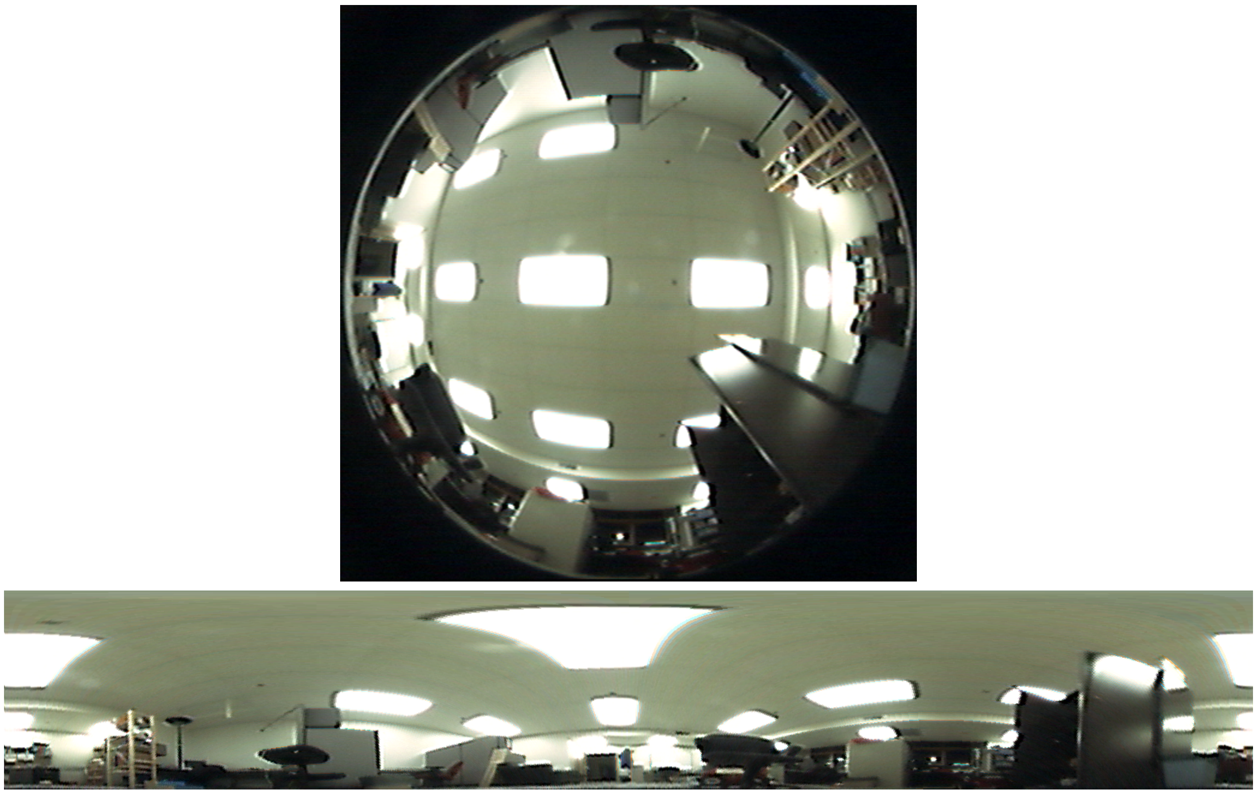


Figure 9: A raw and de-warped image taken from the Omnitech 190° lens.

The different estimators were run on this dataset in order to compare their relative performances. The average Euclidean error between the estimated positions and ground truth is shown in Table 5.1.

Estimator Algorithm	IEKF	Batch ML
Average Euclidean Error	0.171 m	0.092 m

Table 1: Average Euclidean error for the five landmarks generated in for the experiments using images and odometry captured from a real robot.

5.2 Comparison of Estimators with Varying Noise Models

A series of synthetic paths were generated from the above data set and used to test the performance of each of the estimators using different odometric noise models. The simulated odometric noise ranged from a standard deviation of 10 deg / sec to 120 deg / sec in encoder error (in 10 deg increments). A set of 100 robot paths were created for each noise variance setting. For each path, both of the robot's wheel encoders was corrupted by noise drawn from a distribution with the same variance.

Figure 13 shows the results of the different estimators on paths affected by increasing levels of odometric error. The linearized ML estimator had the least amount of error in the placement of the landmarks. The performance of IEKF estimator was equivalent to linearized ML up to an error of around 50 deg / sec but rapidly diminished in accuracy as the odometric errors increased.

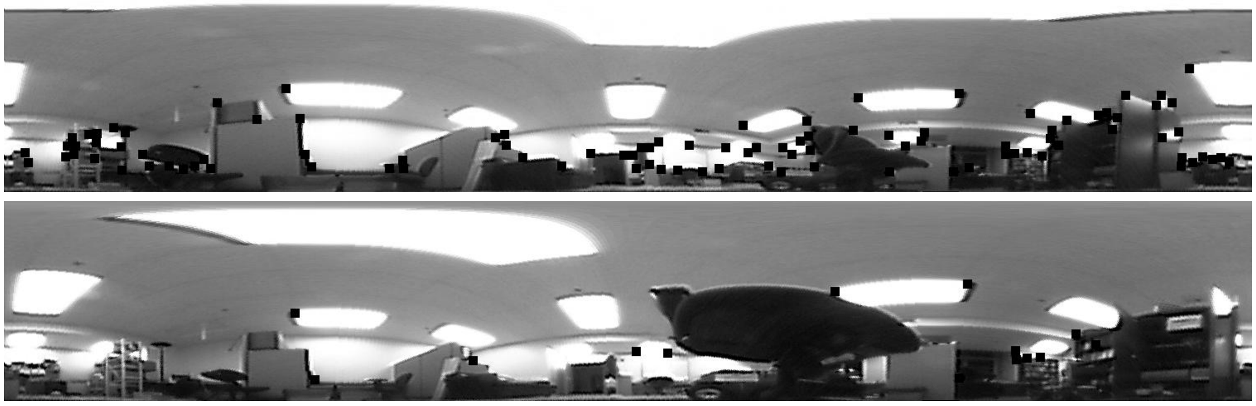


Figure 10: The 100 best features selected by the KLT algorithm in the top image are shown as black squares. The bottom image shows how many features were tracked from the top image to the bottom image (corresponding to a robot translation of approximately 0.6 m.)

5.3 Data Association

In the previous experiments, the office was cluttered enough such that 100 KLT features were sufficient to disambiguate all of the locations where the robot visited. A set of 320 images was taken at 0.3 m intervals in the office environment used for these experiments. Figure 14 shows a plot of the Euclidean distance estimate between each pair of locations as a function of the number of features that the KLT algorithm can track between the respective images. As can be seen, until the number of features tracked drops between 40-50, the likelihood that the two images are within 0.5 m of each other is extremely high. With fewer features, it becomes extremely hard to tell whether a location is the same or not. In this graph, there were no values of matched features of 60 and higher. A match of 100 features would indicate that the the robot was in exactly the same location.

In a feature and texture-rich environment such as an office or a home, we have found that perceptual aliasing is not that much of a problem. Finding and tracking such a large collections of KLT features ensures that each location is unique. However, in environments that do not have unique sets of features, such as in sparse corridors, locations will become more ambiguous and a mechanism for handling improperly matched images will be needed.

6 Summary

Localization and mapping is a challenge for all mobile robots. Existing methods which work well on large robots do not necessarily scale well as the size of the robot decreases. Sensors typically used in mapping algorithms, such as sonars, laser, and stereo camera pairs, are not appropriate for many miniature robots. Additionally, odometric estimates tend to get worse as the robot becomes smaller since its wheels are likely to slip more as well as being severely affected by distortions in the surface that it travels over.

A method for performing localization and map construction with sensor-poor robots has been proposed in which several maximum likelihood-based estimators, such as batch methods and the recursive Kalman filter, have been formulated to relax the assumption that our sensors return metric distance information to landmarks. To accomplish this, a conventional sensor modality is converted into a “virtual sensor” which is used to determine whether the robot has returned to a location that it has visited before. Using this methodology, landmarks are designated by their sensor signatures and indicate locations the robot has visited. The virtual sensor is both the strength and the weakness of the method as it allows correlations to be found between locations that the robot has visited, but global metric information, such as orientation, can be difficult to capture. As shown in the experimental results, the local structure of the landmarks

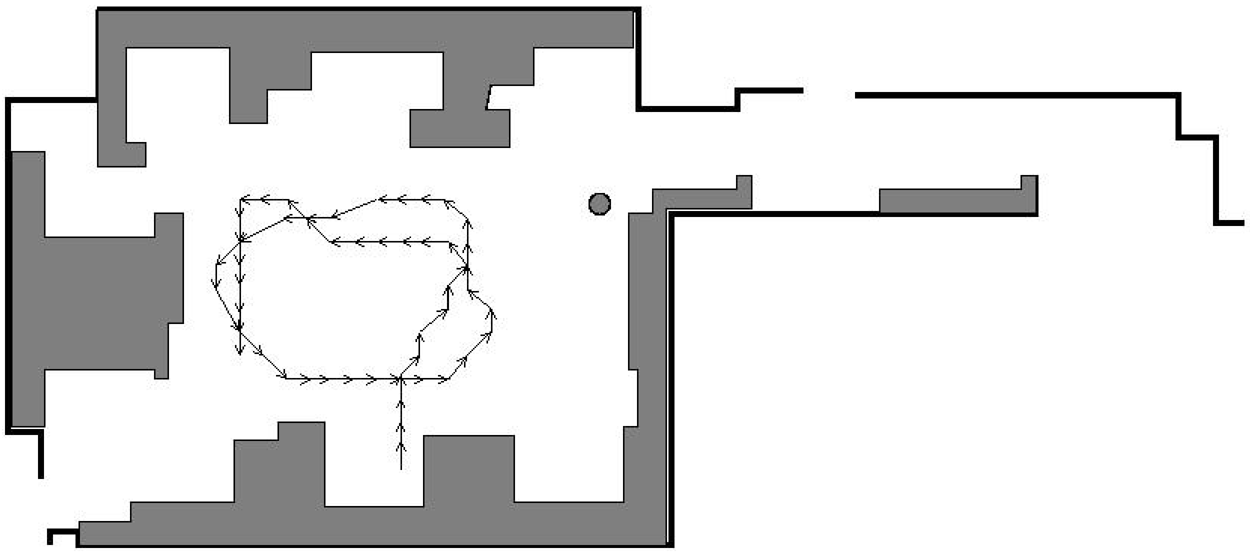
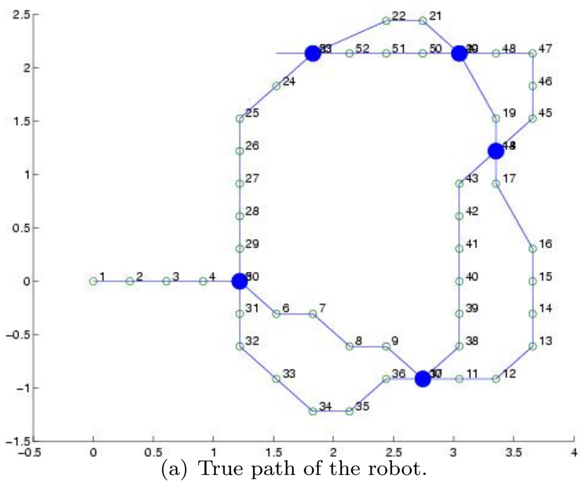
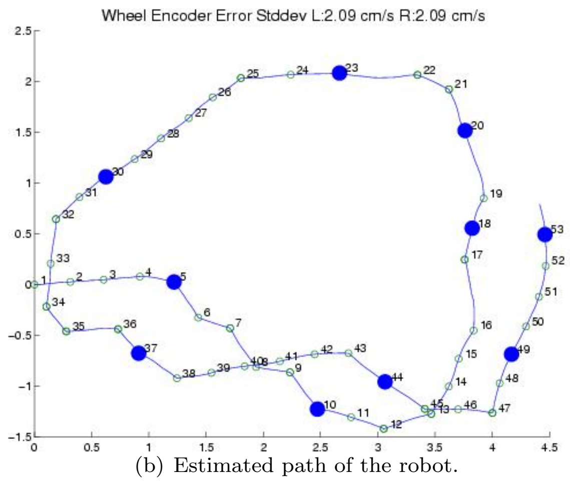


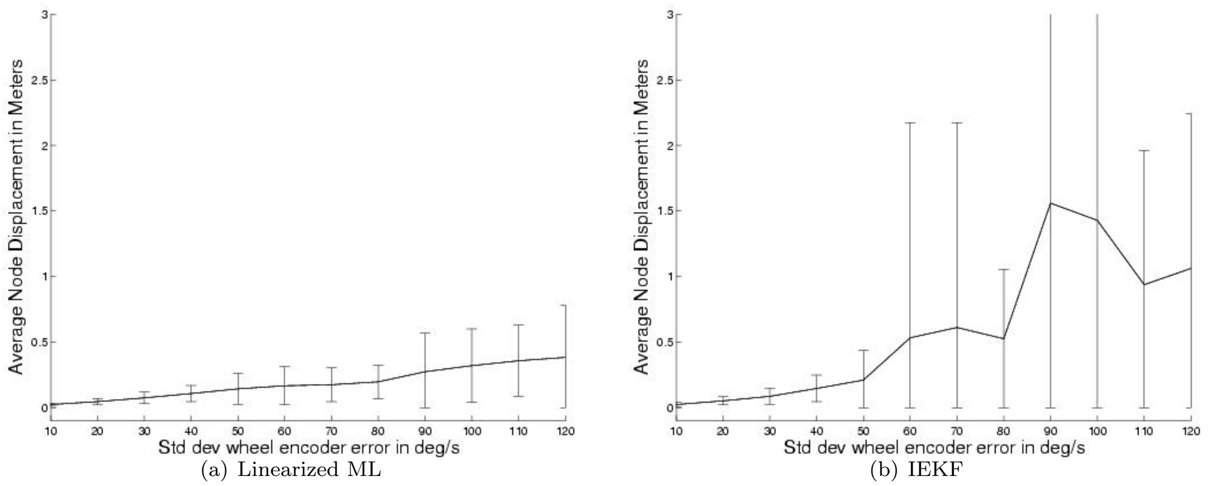
Figure 11: The path of the robot through the office environment.


(a) True path of the robot.



(b) Estimated path of the robot.

Figure 12: Real world experiments in an indoor environment (scale is in meters). Landmarks in the true path occur wherever there is an intersection in the path. Positions in the path are labeled chronologically.



(a) Linearized ML (b) IEKF

Figure 13: Comparison of the means and standard deviations of the two estimators on datasets with varying degrees of encoder error. Standard deviation of errors ranged from 10 deg / sec to 120 deg / sec.

can be recreated, but there can be global misalignments in rotation that can be corrected by incorporating additional information such as the known global position of one of the landmarks. The effectiveness of this algorithm has been illustrated on simulated and real world data.

Experimental results are presented throughout this paper both in simulation and using a miniature mobile robot with an omnicamera in an indoor office environment. As it traverses the environment, the robot's path is reconstructed using the estimators developed in this work and the results are compared. The results demonstrate that both these estimators are capable of reducing the error in the robot estimates of its path even when the odometry is very poor and the only sensory information available is in the form of location signatures. Furthermore, our studies show that the linearized maximum likelihood estimator produces the best results. The Kalman filter is fairly close in estimate quality until the robot's odometric error exceeds a threshold at which point the estimation quality of the Kalman filter decreases significantly.

Acknowledgment

Material based in part upon work supported by the National Science Foundation through grant #EIA-0224363, Microsoft Inc., and the Defense Advanced Research Projects Agency, MTO (Distributed Robotics), ARPA Order No. G155, Program Code No. 8H20, Issued by DARPA/under Contract #MDA972-98-C-0008.

References

[1] Amy Briggs, Yunpeg Li, Daniel Scharstein, and Matt Wilder. Robot navigation using 1d panoramic images. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2679–2685, 2006.

[2] H. Choset and K. Nagatani. Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization. *IEEE Transactions on Robotics and Automation*, 17(2), April 2001.

[3] Frank Dellaert and Ashley Stroupe. Linear 2D localization and mapping for single and multiple robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2002.

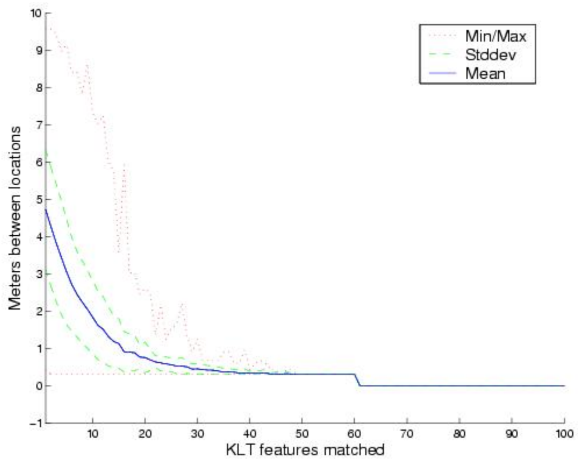


Figure 14: Comparison of the number of features tracked vs. the Euclidean distance between locations where the features were obtained.

[4] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. 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, June 2001.

[5] T. Duckett, S. Marsland, and J. Shapiro. Learning globally consistent maps by relaxation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3841–3846, 2000.

[6] Austin Eliazar and Parr Ronald. Hierarchical linear/constant time slam using particle filters for dense maps. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 339–346. MIT Press, Cambridge, MA, 2006.

[7] A. Gelb. *Applied Optimal Estimation.* MIT Press, 1994.

[8] Greg Grudic and Jane Mulligan. Topological mapping with multiple visual manifolds. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[9] A. Howard, M.J. Matarić, and G.S. Sukhatme. Localization for mobile robot teams using maximum likelihood estimation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, EPFL Switzerland, September 2002.

[10] Margaret E Jefferies, Wenrong Weng, Jesse T Baker, Michael C Cosgrove, and Michael Mayo. A hybrid approach to finding cycles in hybrid maps. In *Australasian Conference on Robotics and Automation*, 2003.

[11] KLT: An implementation of the Kanade-Lucas-Tomasi feature tracker. http://robotics.stanford.edu/~birch/klt/.

[12] Jana Košecká and Fayin Li. Vision based topological markov localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1481–1486, 2004.

[13] Ben Kuipers and Yung-Tai Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8:47–63, 1991.

[14] Ben J. Kuipers. Modeling spatial knowledge. *Cognitive Science*, 2:129–153, 1978.

[15] John J. Leonard and Hugh F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–382, 1991.

[16] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pages 1150–7, 1999.

[17] Bruce D. Lucas and Takeo. Kanade. An iterative image registration technique with an application to stereo vision. *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[18] P. S. Maybeck. *Stochastic Models, Estimation and Control*, volume 141–142 of *Mathematics in Science and Engineering*. Academic Press, 1982.

[19] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003. IJCAI.

[20] Kevin Murphy. *Dynamic Bayesian Networks: representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, July 2002.

[21] Kevin Murphy and Stewart Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York, January 2001.

[22] J. Neira and J. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, December 2001.

[23] P. Newman, D. Cole, and K. Ho. Outdoor slam using visual appearance and laser ranging. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1180–1187, Orlando, Florida, USA, May 2006.

[24] Edwin Olson, John Leonard, and Seth Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2262–2269, 2006.

[25] Omnitech Robotics International, LLC, 2640 South Raritan Circle, Englewood, CO, 80110. *ORIFL190-3: 190 Degree Field of View Fisheye Lens for 1/3" Image Sensor Cameras*, 2002.

[26] J. M. Porta and B.J A. Kröse. Appearance-based concurrent map building and localization. *Robotics and Autonomous Systems*, 54(2):2005, 2005.

[27] Ananth Ranganathan and Frank Dellaert. Data driven MCMC for appearance-based topological mapping. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[28] Ananth Ranganathan and Frank Dellaert. A rao-blackwellized particle filter for topological mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 810–817, 2006.

[29] Emilio Remolina and Benjamin Kuipers. Towards a general theory of topological maps. *Artificial Intelligence*, 152(1):47–104, 2004.

[30] P. Rybski, F. Zacharias, M. Gini, and N. Papanikolopoulos. Using visual features for building and localizing within topological maps of indoor environments. In Srikanta Patnaik, Lakhmi C. Jain, and Spyros G. Tzafestas, editors, *Innovations in Robot Mobility and Control*, volume 8, pages 251–271. Springer-Verlag, 2005.

[31] P. E. Rybski, S. A. Stoeter, M. Gini, D. F. Hougen, and N. Papanikolopoulos. Performance of a distributed robotic system using shared communications channels. *IEEE Transactions on Robotics and Automation*, 22(5):713–727, October 2002.

[32] Paul E. Rybski. *Building Topological Maps using Minimalistic Sensor Models*. PhD thesis, The University of Minnesota, Minneapolis, July 2003.

[33] Hagit Shatkay and Lesie Kaelbling. Learning topological maps with weak local odometric information. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 920–927, San Mateo, CA, 1997. Morgan Kaufmann.

[34] Robert Sim and Gregory Dudek. Learning environmental features for pose estimation. *Image and Vision Computing*, 19(11):733–739, 2001.

[35] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. J. Cox and G. T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, 1990.

[36] A. Tapus and R. Siegwart. A cognitive modeling of space using fingerprints of places for mobile robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1188–1193, Orlando, Florida, U.S.A., May 2006.

[37] Adriana Tapus, Guy Ramel, Luc Dobler, and Roland Siegwart. Topology learning and place recognition using bayesian programming for mobile robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, September 2004.

[38] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998.

[39] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 101:99–141, 2000.

[40] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical report, School of Computer Science, Carnegie Mellon University, April 1991.

[41] Iwan Ulrich and Illah Nourbakhsh. Appearance-based place recognition for topological localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1023–1029, San Francisco, CA, April 2000.