Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2005-029
AIM-2005-016

April 26, 2005

Simultaneous Localization, Calibration, and Tracking in an ad Hoc Sensor Network

Christopher Taylor, Ali Rahimi, Jonathan Bachrach, and Howard Shrobe

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Abstract

We introduce Simultaneous Localization and Tracking (SLAT), the problem of tracking a target in a sensor network while simultaneously localizing and calibrating the nodes of the network. Our proposed solution, LaSLAT, is a Bayesian filter providing on-line probabilistic estimates of sensor locations and target tracks. It does not require globally accessible beacon signals or accurate ranging between the nodes. When applied to a network of 27 sensor nodes, our algorithm can localize the nodes to within one or two centimeters.

1 Introduction

Many sensor network applications, such as tracking moving targets over large regions, require that the sensor nodes be calibrated and that their locations be known. Due to the scale of the deployment in many applications, it is impractical to rely on careful placement or uniform arrangement of sensor nodes. Because GPS is unavailable indoors and range information between nodes based on radio is often unreliable, automatic localization of nodes is challenging. Furthermore, spatially varying environmental factors preempt a full calibration at the factory and require that some of the sensor parameters be calibrated on the field after deployment. For sensor networks deployed to track moving targets, some authors have suggested localizing the network using a moving target (or *mobile*) [1–3]. This is an attractive solution since it requires no additional hardware on the nodes themselves. However, these methods require that the position of the mobile be known at all times. In this paper, we show how to track the position of an unconstrained mobile while localizing and calibrating the sensor nodes.

This problem, which we label Simultaneous Localization And Tracking (SLAT), is analogous to SLAM, where a robot localizes itself within a map of the environment while concurrently building this map. Over the past decade, SLAM has witnessed a surge in its efficacy and performance due in part to the application of Bayesian methods [4–9]. We adapt some of these methods to the SLAT problem in the form of a Bayesian filter that uses range measurements to update a joint probability distribution over the positions of the nodes, the trajectory of the mobile, and the calibration parameters of the network. To avoid some of the representational and computational complexity of general Bayesian filtering, we use Laplace's method to approximate our state with a Gaussian after incorporating each batch of measurements. Accordingly, we call our algorithm LaSLAT.

LaSLAT inherits many desirable properties from the Bayesian framework. The probabilistic model used in LaSLAT insures that measurement noise is averaged out as more measurements become available, improving localization and tracking accuracy in high-traffic areas. The filtering framework incorporates measurements in small batches, providing online estimates of all locations, calibration parameters, and their uncertainties. This speeds up the convergence of the algorithm and reduces impact on the network. Since uncertainties are known, mobiles can be dispatched on-line to improve localization estimates in regions where localization uncertainty is high. These mobiles may move arbitrarily through the environment, with no constraint on their trajectory or velocity, and multiple mobiles may be used simultaneously to expedite surveying. Ancillary localization information, such as position estimates from GPS, beacons, or radio-based ranging, can also be easily incorporated into this framework. Our algorithm is fast and permits a distributed implementation because it operates on sparse inverse covariance matrices rather than dense covariance matrices themselves. When the user does not specify a coordinate system, LaSLAT recovers locations in a coordinate system that is correct up to a translation, rotation, and possible reflection.

We demonstrate these features by accurately localizing a dense network of 27 nodes to within one or two centimeters. The nodes are wireless Crickets [10] capable of measuring their distance to a moving beacon using a combination of ultrasound and radio pulses. In a larger and sparser network, we localize nodes to within about eight centimeters. In both cases, a measurement bias parameter is accurately calibrated for all nodes. Finally, we present initial results from an experiment in three dimensional localization and tracking.

2 Related Work

The most common sensor network localization algorithms rely on range or connectivity measurements between pairs of nodes [11–16]. When such measurements are available, these methods can supplement LaSLAT by providing a prior or an initial estimate for the location of the sensors (Section 3.5).

Various authors have used mobiles to localize sensor networks [1–3,17], but these methods assume the location of the mobile is known. One exception is [2], which builds a constraint structure as measurements become available. Compared to [2], we employ a very extensible statistical model that allows more realistic measurement models. Our method is most similar to [17], who used an Extended Kalman Filter (EKF) to track an underwater vehicle while localizing sonar beacons capable of measuring their range to the vehicle. We replace the EKF's approximate measurement model with one based on Laplace's method. This provides faster convergence and greater estimation accuracy. We also demonstrate that the Bayesian filtering framework can calibrate the sensor nodes, and that the computation is capable of distributing over the sensor nodes in a straightforward way.

Our solution to SLAT adopts various important refinements to the original Extended Kalman Filter (EKF) formulation of SLAM [9]. LaSLAT processes measurements in small batches and discards variables that are no longer needed, as demonstrated by McLauchlan [18]. Following [6], LaSLAT operates on inverse covariances of Gaussians rather than on covariances themselves to speed up updates and facilitate distributed computation.

3 LaSLAT

To process measurements on-line LaSLAT uses the Bayesian filtering framework. Under this framework, as each batch of measurements becomes available, it is used to update a prior distribution over sensor locations, the mobile trajectory, and various sensing parameters. The resulting posterior distribution is then propagated forward in time using a dynamics model to make it a suitable prior for use with the next batch of measurements.

In LaSLAT, the posterior distribution is approximated with a Gaussian using Laplace's method [19] after incorporating each batch of measurements. Consequently, the amount of state saved between batches is constant with respect to the number of measurements taken in the past. The Gaussian approximation also simplifies propagation with the dynamics model and the incorporation of the next batch of measurements.

3.1 Approximate Bayesian Filtering for SLAT

As the mobile moves through the network, it periodically emits *events* which allow some of the sensors to measure their distances from the mobile.

Let $\mathbf{e}_j$ denote the location of the mobile generating the jth event. The tth batch $\mathbf{e}^t$ is a collection of consecutive events $\mathbf{e}^t = \{\mathbf{e}_m \ldots \mathbf{e}_{m+n}\}$, with $\mathbf{e}^t_j$ denoting the jth event in the tth batch. Each LaSLAT iteration incorporates the measurements from a single batch of events.

Let $\mathbf{s}_i = \begin{bmatrix}\mathbf{s}^x_i & \mathbf{s}^\theta_i\end{bmatrix}$ represent the unknown parameters of sensor i, with $\mathbf{s}^x_i$ denoting the sensor's position and $\mathbf{s}^\theta_i$ its calibration parameters. Then $\mathbf{s} = \{\mathbf{s}_i\}$ is the set of all sensor parameters. The scalar y^t_{ij} denotes the range measurement between sensor i and the jth event in batch t, with

$\mathbf{y}^t = \{y_{ij}^t\}$ the collection of all range measurements in batch t. For each batch t, $\mathbf{e}^t$ and $\mathbf{s}$ are the unknown values that must be estimated. We aggregate these unknowns into a single variable $\mathbf{x}^t = \begin{bmatrix}\mathbf{e}^t & \mathbf{s}\end{bmatrix}$ for notational simplicity.

The Bayesian filtering framework is a non-linear, non-Gaussian generalization of the Kalman Filter. For each batch t, it computes the posterior distribution over sensor parameters, $\mathbf{s}$, and events locations, $\mathbf{e}^t$, taking into account all range measurements taken so far:

$$p(\mathbf{x}^t|\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^t).$$

In LaSLAT, we wish to update this distribution as range measurements become available, and discard measurements as soon as they have been incorporated. To do this, one can rewrite the distribution in terms of a measurement model and a prior distribution derived from the results of the previous iteration. Rewriting $p(\mathbf{x}^t|\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^t)$ as $p(\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t)$, we get by Bayes rule:

$$\begin{aligned} p(\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t) &\propto p(\mathbf{y}^t, \mathbf{x}^t|\mathbf{y}^{old}) \\ &\propto p(\mathbf{y}^t|\mathbf{x}^t, \mathbf{y}^{old}) p(\mathbf{x}^t|\mathbf{y}^{old}) \\ &\propto p(\mathbf{y}^t|\mathbf{x}^t) p(\mathbf{x}^t|\mathbf{y}^{old}), \end{aligned} \tag{1}$$

where proportionality is with respect to $\mathbf{x}^t$. The final equality follows because when the sensor and mobile locations are known, the past measurements do not provide any additional useful information about the new batch of measurements. The distribution $p(\mathbf{y}^t|\mathbf{x}^t)$ is the measurement model: it reflects the probability of a set of observations given a particular configuration of sensors and event locations (Section 3.2).

The distribution $p(\mathbf{x}^t|\mathbf{y}^{old})$ summarizes all information collected prior to the current batch of measurements, in the form of a prediction of $\mathbf{x}^t$ and an uncertainty measure. It can be computed from the previous estimate, $p(\mathbf{x}^{t-1}|\mathbf{y}^{old})$, by applying a dynamic model:

$$p(\mathbf{x}^t|\mathbf{y}^{old}) = \int_{\mathbf{x}^{t-1}} p(\mathbf{x}^{t-1}|\mathbf{y}^{old}) p(\mathbf{x}^t|\mathbf{x}^{t-1})\, d\mathbf{x}^{t-1}. \tag{2}$$

The distribution $p(\mathbf{x}^t|\mathbf{x}^{t-1})$ models the dynamics of the configuration from one batch to another, by discarding old event locations and predicting the locations of new events (Section 3.4).

When the measurement model $p(\mathbf{y}^t|\mathbf{x}^t)$ is not Gaussian, the updates (1) and (2) become difficult to compute. We handle the non-Gaussianity of the measurement model by after each batch approximating the posterior $p(\mathbf{x}^t|\mathbf{y}^{old})$ with a Gaussian distribution $q(\mathbf{x}^t|\mathbf{y}^{old})$ using Laplace's method (Section 3.3). This Gaussian becomes the basis for the prior distribution for the next batch. q is much simpler to save between batches than the full posterior – in particular, it allows all the old measurements to be discarded. Table 1 summarizes the steps of LaSLAT.

Other approximate Bayesian filters such as the Extended Kalman Filter (EKF) or particle filters could also be used in place of our Laplacian method. The EKF differs from our algorithm because it does not perform a full optimization when incorporating each event. In many cases this is a helpful optimization; however, as we show in section 5, on the SLAT problem it sacrifices accuracy and speed of convergence. Particle filters allow a closer approximation of the posterior distribution, especially when the distribution is multimodal. However, our algorithm seems to perform well in practice while requiring significantly less computation.

1. Observe a new batch of measurements $\mathbf{y}^t$.
2. Represent the posterior $p(\mathbf{x}^t|\mathbf{y}^t, \mathbf{y}^{old})$ in terms of the prior $p(\mathbf{x}^t|\mathbf{y}^{old})$ and the measurement model $p(\mathbf{y}^t|\mathbf{x}^t)$ using Equation (1).
3. Using Newton-Raphson [20], compute curvature at the mode of $p(\mathbf{x}^t|\mathbf{y}^t, \mathbf{y}^{old})$ and use it to construct the approximate posterior $q(\mathbf{x}^t|\mathbf{y}^t, \mathbf{y}^{old})$. This posterior is the estimate for the batch t (Section 3.3).
4. Compute the prediction $p(\mathbf{x}^{t+1}|\mathbf{y}^t, \mathbf{y}^{old})$ using $q(\mathbf{x}^t|\mathbf{y}^t, \mathbf{y}^{old})$ (Section 3.4).
5. Using the prediction as the new prior, return to step 1 to process batch $t+1$.

Table 1: One iteration of LaSLAT. Incorporates batch t and prepares to incorporate batch $t+1$.

3.2 The measurement model

Measurements influence localization and calibration estimates via the measurement model. A measurement model is a probability distribution $p(\mathbf{y}^t|\mathbf{x}^t)$ over a batch of range measurements, given a particular choice of the calibration parameters and positions for the sensors and the mobile. In this paper, we assume that each measurement is a corrupted version of the true distance between the event and the sensor that made the measurement:

$$y_{ij}^t = \|\mathbf{s}_i^x - \mathbf{e}_j^t\| + s_i^\theta + \omega_{ij}^t, \tag{3}$$

where $\|\cdot\|$ indicates the vector 2-norm, giving the Euclidean distance between $\mathbf{s}_i^x$ and $\mathbf{e}_j^t$. ω_{ij}^t is a zero-mean Gaussian random variable with variance σ^2, and $\mathbf{s}_i^\theta$ is a bias parameter that models an unknown shift due to a variable time delay in the ranging algorithm.

As defined, $p(y_{ij}^t|s_i, e_j^t)$ is a univariate Gaussian with mean $\|\mathbf{s}_i - \mathbf{e}_j^t\| + s_i^\theta$ and variance σ^2. More sophisticated measurement models can be substituted if necessary. For example, using a heavy-tailed distribution such as the student-t in place of the Gaussian would provide automatic attenuation of outlying measurements (such as those caused by echoes). Other calibration parameters could also be included in the measurement model.

Since each measurement y_{ij}^t depends only on the sensor $\mathbf{s}_i$ that took the measurement, and the location $\mathbf{e}_j^t$ of the mobile when it generated the event, the measurement model factorizes according to

$$p(\mathbf{y}^t|\mathbf{x}^t) = \prod_{i,j} p(y_{ij}^t|\mathbf{s}_i, \mathbf{e}_j^t), \tag{4}$$

where the product is over the sensors and the events that they perceived in batch t. Equation (4) provides the measurement model for a batch of measurements. Note that although this distribution is Gaussian in the distances between $\mathbf{y}^t$ and $\mathbf{x}^t$, it is not Gaussian in $\mathbf{y}^t$ and $\mathbf{x}^t$ themselves.

3.3 Incorporating Measurements

The approximate Gaussian posterior $q(\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t)$ can be obtained from the prior distribution $p(\mathbf{x}^t|\mathbf{y}^{old})$ and the measurement model $p(\mathbf{y}^t|\mathbf{x}^t)$ using Laplace's method [19].

To fit an approximate Gaussian distribution $q(x)$ to a distribution $p(x)$, Laplace's method first finds the mode x^* of $p(x)$, then computes the curvature of the negative log posterior at x^*.

$$\Lambda^{-1} = -\frac{\partial^2}{\partial x^2} \log p(x)\Big|_{x=x^*}.$$

The mean and covariance of $q(x)$ are then set to x^* and Λ respectively. Notice that when p is Gaussian, the resulting approximation q is exactly p. For other distributions, the Gaussian q locally matches the behavior of p about its mode.

The mode finding problem can be expressed as:

$$\begin{aligned}
\mathbf{x}^{t*} &= \arg\max_{\mathbf{x}^t} p(\mathbf{x}^t|\mathbf{y}^t, \mathbf{y}_{old}) \\
&= \arg\min_{\mathbf{x}^t} -\log p(\mathbf{y}^t|\mathbf{x}^t) p(\mathbf{x}^t|\mathbf{y}^{old}) \\
&= \arg\min_{\mathbf{s},\mathbf{e}^t} (\mathbf{s}-\mu)^T \Omega (\mathbf{s}-\mu) + \frac{1}{\sigma^2} \sum_{i,j} (\|\mathbf{s}_i^x - \mathbf{e}_j^t\| + s_i^\theta - y_{ij}^t)^2, \qquad (5)
\end{aligned}$$

where $\mu = E\left[\mathbf{x}^t|\mathbf{y}^t, \mathbf{y}^{old}\right]$, and $\Omega = \mathrm{Cov}^{-1}\left[\mathbf{s}|\mathbf{y}^{old}\right]$.

We use the Newton-Raphson iterative optimization algorithm [20] to find the mode $\mathbf{x}^{t*}$, and the curvature $\mathbf{H}$ (see the Appendix). Following Laplace's method, the mean $E\left[\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t\right]$ of q is set to $\mathbf{x}^{t*}$ and its inverse covariance $\mathrm{Cov}^{-1}\left[\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t\right]$ is set to $\mathbf{H}$. Representing q using its inverse covariance allows us to avoid computing the matrix inverse $\mathbf{H}^{-1}$ after adding each measurement, which significantly improves performance and facilitates a distributed implementation of our algorithm (Section 4).

3.4 Dynamics Model

In this paper, we assume the mobiles can move arbitrarily and that sensors do not move over time. When propagating the the posterior $q(\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t)$ forward in time, we need only retain the components that are useful for incorporating the next batch of measurements. Thus, we may remove the estimate of the mobile's trajectory from batch t, but we must incorporate a guess for the mobile's path during batch $t+1$. Therefore, the prediction step of Equation (2) can be written:

$$\begin{aligned}
p(\mathbf{x}^{t+1}|\mathbf{y}^{old}, \mathbf{y}^t) &= p(\mathbf{s}, \mathbf{e}^{t+1}|\mathbf{y}^{old}, \mathbf{y}^t) = p(\mathbf{e}^{t+1}) q(\mathbf{s}|\mathbf{y}^{old}, \mathbf{y}^t) \qquad (6) \\
q(\mathbf{s}|\mathbf{y}^{old}, \mathbf{y}^t) &= \int_{\mathbf{e}^t} q(\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t)\, d\mathbf{e}^t.
\end{aligned}$$

The Gaussian $q(\mathbf{x}^t|\mathbf{y}^{old}, \mathbf{y}^t)$ captures the posterior distribution over sensor locations given all measurements before $\mathbf{y}^t$, and has already been computed by the method of section 3.3. We obtain $q(s|\mathbf{y}^{old}, \mathbf{y}^t)$, by marginalizing out the mobile's trajectory during batch t. The prior $p(\mathbf{e}^{t+1})$ is Gaussian with very broad covariance, indicating that the future trajectory of the mobile is unknown. In some applications, it may be possible to use past trajectories to make better guesses for $\mathbf{e}^{t+1}$. For maximum generality, we will not attempt to do so in this paper, meaning that the mobile is allowed to move arbitrarily between events.

The operations of Equation (6) can be carried out numerically by operating on the mean and inverse covariance of $q(\mathbf{x}^t|\mathbf{y}^{old})$. First, partition according to $\mathbf{s}$ and $\mathbf{e}^t$:

$$E\left[\mathbf{x}^t|\mathbf{y}^{old}\right] = \begin{bmatrix} E[\mathbf{s}|\mathbf{y}^{old}] \\ E[\mathbf{e}^t|\mathbf{y}^{old}] \end{bmatrix}$$
$$\mathrm{Cov}^{-1}\left[\mathbf{x}^t|\mathbf{y}^{old}\right] = \begin{bmatrix} \Omega_{\mathbf{s}} & \Omega_{\mathbf{s}\mathbf{e}^t} \\ \Omega_{\mathbf{e}^t\mathbf{s}} & \Omega_{\mathbf{e}^t} \end{bmatrix}.$$

Marginalizing out $\mathbf{e}^t$ produces a distribution $q(\mathbf{s}|\mathbf{y}^{old})$ whose mean is the $\mathbf{s}$ component of the mean of $q(\mathbf{x}^t|\mathbf{y}^{old})$ and whose inverse covariance is

$$\mathrm{Cov}^{-1}\left[\mathbf{s}|\mathbf{y}^{old}\right] = \Omega_{\mathbf{s}} - \Omega_{\mathbf{s}\mathbf{e}^t}\Omega_{\mathbf{e}^t}^{-1}\Omega_{\mathbf{e}^t\mathbf{s}}, \tag{7}$$

The parameters of $p(\mathbf{x}^{t+1}|\mathbf{y}^{old})$ are those of $q(\mathbf{s}|\mathbf{y}^{old})$, augmented by zeros to account for an uninformative prior on $\mathbf{e}^{t+1}$:

$$E\left[\mathbf{x}^{t+1}|\mathbf{y}^{old}\right] = \begin{bmatrix} E\left[\mathbf{s}|\mathbf{y}^{old}\right] \\ \mathbf{0} \end{bmatrix} \tag{8}$$
$$\mathrm{Cov}^{-1}\left[\mathbf{x}^{t+1}|\mathbf{y}^{old}\right] = \begin{bmatrix} \mathrm{Cov}^{-1}\left[\mathbf{s}|\mathbf{y}^{old}\right] & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}. \tag{9}$$

The components of the inverse covariance of $p(\mathbf{x}^{t+1}|\mathbf{y}^{old})$ corresponding to $\mathbf{e}^{t+1}$ are set to $\mathbf{0}$, corresponding to infinite variance, which in turn captures our lack of *a priori* knowledge about the location of the mobiles in the new batch. The mean is arbitrarily set to 0. If some information is known *a priori* about $\mathbf{e}^{t+1}$, then the last $\mathbf{0}$ components of $E\left[\mathbf{x}^{t+1}|\mathbf{y}^{old}\right]$ and the bottom right $\mathbf{0}$ components of $\mathrm{Cov}^{-1}\left[\mathbf{x}^{t+1}|\mathbf{y}^{old}\right]$ can be used to capture that knowledge.

3.5 Prior Information and Initialization

Prior information about the sensor parameters is easy to incorporate into LaSLAT. Such information might be available because the sensors were placed in roughly known positions, or because another less accurate source of localization is available. Calibration in the factory might also supply additional prior information.

If such prior information is available it can be supplied as the prior for incorporating the first batch of measurements. We set the covariance of this prior to $\sigma_0\mathbf{I}$, with σ_0 a large scalar, which makes the prior diffuse. The large covariance allows measurements to override the positions prescribed by the prior, but provides a sensible default when few measurements are available. The mode of this prior (or for subsequent iterations, the mode of $p(\mathbf{s}|\mathbf{y}^{old})$) is also used as the initial iterate for the Newton-Raphson iterations. To obtain the initial iterate an event, we use the average estimated location of the three sensors with the smallest range measurements to the event.

In our experiments, we utilize the radio connectivity of the sensors to obtain prior localization information. The initialization step described by Priyantha et al. [14] provides rough position estimates to serve as a prior before any measurements are introduced. This prior takes the form $p(\mathbf{s}^x) \propto \exp\left[-\frac{1}{2\sigma_0^2}\sum_i \|\mathbf{s}_i^x - x_i^0\|^2\right]$, where x_i^0 is the position predicted by the initialization step of the ith sensor and σ_0 is a large variance.

Because the sensor nodes are nearly identical, we know *a priori* that the variation between their calibration parameters is small. These small variations are due mainly to environmental effects, so sensors that are close together tend to have similar calibration values. We encode this information in a prior of the form $p(\mathbf{s}^\theta) \propto \exp\left[-\frac{1}{2}\sum_{i\sim j}(s_i^\theta - s_j^\theta)^2\right]$, where the summation is over sensors that are in close proximity to each other.

4 Distributing LaSLAT

Though our current implementation sends measurement batches to a central computer, we show here that LaSLAT can be feasibly distributed if desired. LaSLAT consists of two significant computational steps: incorporating measurements and applying the dynamics model. As formulated in this paper, these steps can be performed using only local communication between sensors that have witnessed a common event. Furthermore, the prior $p(\mathbf{x}^t|\mathbf{y}^{old})$ that encodes LaSLAT's state between batches distributes across the network in a straightforward way. We assume that since events are heard only by sensors near the mobile, it is very likely that any two sensors that witness the same event are also within radio communication range.

The prior is completely summarized by a vector of means $\mathbf{x}^{t*}$ and an inverse covariance Ω. The mean vector is trivial to distribute, since each sensor can simply store its own mean. Ω requires more careful consideration, but is also distributable. The symmetric inverse covariance matrix $\Omega = \left[\begin{smallmatrix} \Omega_s & \Omega_{se} \\ \Omega_{es} & \Omega_e \end{smallmatrix}\right]$ defines an undirected graph between sensors and measurements. Two vertices in this graph are connected if their corresponding entry in Ω is non-zero. We say Ω has local connectivity if the corresponding graph only connects sensors that are within measurement range of each other, and connects events only to the sensors that measured the event. Then, if Ω has local connectivity, each row of Ω corresponding to a sensor has about $n_{local} + n_{events}$ non-zero entries, where n_{local} is the sensor's number of one-hop neighbors, and n_{events} is the number of events observed by the sensor in the most recent batch. Each row corresponding to an event has about n_{local} non-zero elements, since only sensors within the neighborhood of the event obtain measurements to it. Locally connected matrices are therefore easy to distribute. Each sensor stores its own rows in Ω, and the rows of some fraction of the events it has witnessed. The amount of data stored by each sensor is $O(n_{local} + n_{events})$. However, for each network, n_{local} and n_{events} are fixed constants. After deployment, therefore, the amount of data stored by a sensor is $O(1)$.

It remains to be shown that the LaSLAT computations require only local communication and retain local connectivity in Ω. The latter is easily confirmed by induction on Ω_s. The initial prior has $\Omega_s = \sigma_0 I$, which is diagonal and therefore locally connected. As we show in the appendix, the measurement incorporation and Gaussian approximation steps do not change the connectivity of Ω. The connectivity of Ω_s only changes when events are marginalized out of the Gaussian prediction by equation (6). It can be verified, however, that $-\Omega_{se}\Omega_e^{-1}\Omega_{es}$ term added to Ω_s only affects elements whose corresponding sensors observed an event in common during the most recent batch. As a result, Ω_s retains local connectivity during LaSLAT operations.

Finally, we will consider the time complexity of LaSLAT operations and show that LaSLAT requires only local communication between sensors. The two significant operations performed during each LaSLAT batch are Newton-Raphson iterations (12) and event marginalization (6).

Each Newton-Raphson iteration has two parts. First a matrix and vector must be computed

based on equation (12). Then, a least squares optimization must be performed. The matrix and vector can be computed locally, since they require only the parts of Ω and $\mathbf{x}^{t*}$ that are found locally and the measurements to any local events. The communication and time costs for each sensor are proportional to $n_{events} * n_{local}$, which is required to broadcast new measurements to neighboring sensors. Once the matrix and vector are computed, the least squares optimization can be performed using Gauss-Seidel iterations [21]. Gauss-Seidel is guaranteed to converge when solving symmetric positive definite systems of equations like ours. Each iteration of Gauss-Seidel requires $O(n_{local})$ computation and $O(1)$ radio messages per sensor and event. In practice, we find that Gauss-Seidel converges in 3-4 iterations for our systems, since LaSLAT does not require high precision convergence. Gauss-Seidel does require that sensors perform their processing in a consistent order, which diminishes the potential parallelization of the least squares computation. However, with a constant bound on the number of Gauss-Seidel iterations, the total time required for each Newton-Raphson iteration is $O(n * n_{local})$, where n is the number of sensors and events whose parameters are updated by the iteration. Note that this is not a tight upper bound: as the sensor parameters begin to converge, many parameters will not need to be updated every iteration. This increases the amount of parallelism that can be exploited, allowing the total running time to approach $O(n_{local})$, the amount of time required to simply locate the newest events.

Marginalization is distributable, since each sensor row is updated only on behalf of local events. Since Ω_{se} and Ω_{es} are sparse, and Ω_e is diagonal, the total time required is only $O(n_{local} * n_{events})$ per sensor. All the computations can occur in parallel.

As we have observed, n_{local} and n_{events} are user-defined constants for each network. This leads us to believe that a distributed implementation of LaSLAT is feasible and would perform tolerably. We have not yet had the opportunity to test our distributed algorithm on a real network – we hope to do so in the future. In our results, we perform all our computations on a centralized computer. Many of the sparse matrix optimizations described in this section yield substantial performance improvements in a centralized context as well.

5 Results

Our experiments use the Cricket ranging system [10]. Sensor Crickets are placed on the floor, and one Cricket is attached to a mobile. The mobile Cricket emits an event (a radio and ultra-sound pulse) every second. The difference in arrival time of these two signals to a sensor is proportional to the distance between the sensor and mobile. Crickets compute ranges from these arrival times. No range measurements between the Crickets on the floor were collected. The measurements are transmitted to a desktop machine, which processes them in batches using LaSLAT. The ultra-sound sensor on a Cricket occupies a 1 cm by 2 cm area on the circuit board, so it difficult to estimate the ground truth location of a Cricket beyond that accuracy.

Our first experiment uses the same setup as [15]. Six sensor crickets were placed around a rectangular enclosure 2.1 meters by 1.6 meters. A ROOMBA robotic vacuum cleaner with the mobile Cricket attached was allowed to move freely within the enclosure, generating about 250 events. See Figure 1(a).

Most events were measured by all 6 sensors. An initial localization guess was obtained from radio connectivity information using the initialization routine of [14]. The resulting average localization

(a) (b)

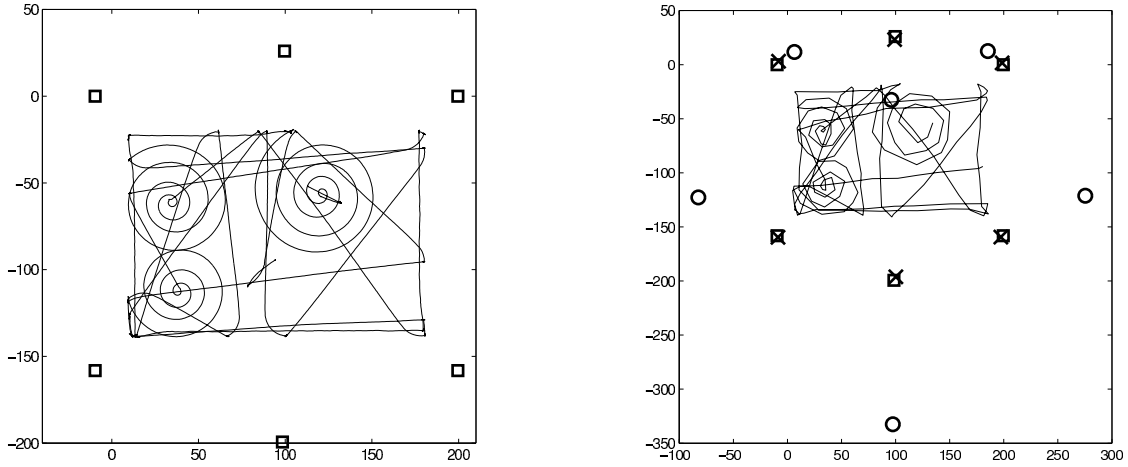
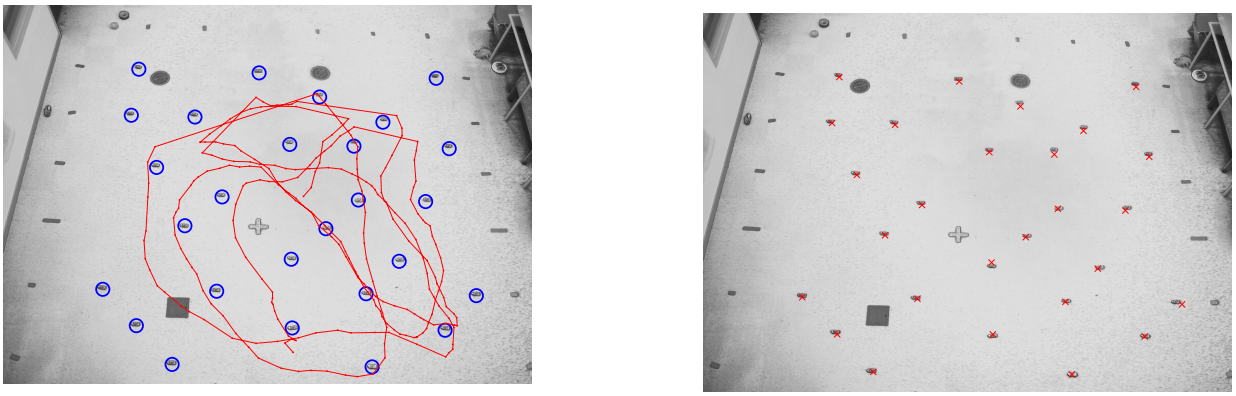


Figure 1: (a) Small network setup. Six sensors (squares) are arranged around a rectangular enclosure. A camera captured the ground truth trajectory of the ROOMBA. The ROOMBA followed the trajectory depicted. (b) Recovered trajectory and sensor positions. Circles are guesses of initial sensor locations obtained from radio connectivity. LaSLAT processed measurements in batches of 30, and recovered sensor locations depicted by crosses. The trajectory is also correctly recovered. LaSLAT improves considerably on the initial localization guess obtained from connectivity. After a global rotation and translation, the average localization error for the sensors was 1.8 cm, which is within the error tolerance of the ground truth.



(a) (b)

Figure 2: **(a)** Sensor locations and mobile trajectory for the large network. Blue circles outline each of the 27 sensor nodes. Markers on the trajectory depict the location of events. 250 of the 1500 events are shown, with consecutive events connected by a line. The mobile was offset from the ground plane and could pass over nodes. To generate this figure, a homography that accounts for the camera transformation was used to project real-world coordinates to image coordinates. **(b)** Final LaSLAT localization result, with batch size of 10. Crosses show estimated sensor locations. These are correctly estimated to fall on the corresponding sensor. Average localization is 1.9 cm.

error in this initial guess was 66 cm. This initial guess was used as a prior and an initial iterate for LaSLAT. LaSLAT incorporated range measurements in batches of 30. Each mode finding operation required an average of only 2.8 Newton-Raphson steps. Figure 1(b) shows the estimated sensor localizations and trajectory. Since the output had an arbitrary rotation and translation, it was rigidly aligned to fit the rotation and origin of the ground truth using the algorithm described in [22]. The final localization error was 1.8 cm, averaged over the sensor nodes. This is within the error tolerance of the ground truth.

Our second experiment involved a larger network with 27 Cricket sensor nodes deployed in a 7 m by 7 m room. Whereas in the previous experiment the nodes were on the perimeter of the ROOMBA mobile's trajectory, in this experiment, we manually pushed a mobile through the network, generating about 1500 events. Figure 2(a) shows the location of the sensors and part of the trajectory of the mobile projected on a top view picture of the setup.

Each event was heard by about 10 sensors. Figures 3(a)-(d) show localization and tracking output as event batches are processed, along with the ground truth and estimated mobile trajectories for that batch. Error ellipses show unit standard deviation contours for each sensor node. Nodes have high uncertainty at early stages, but when the mobile passes near a node, its error ellipse shrinks appropriately. In this experiment, a measurement bias of about 23 cm was computed for each sensor node. Figure 2(b) shows the final localization of the nodes, reprojected on the picture of the setup. This experiment used batches of 10 measurements and produced a final localization error of 1.9 cm. Since the ground truth is only accurate to a few centimeters, localization performance is

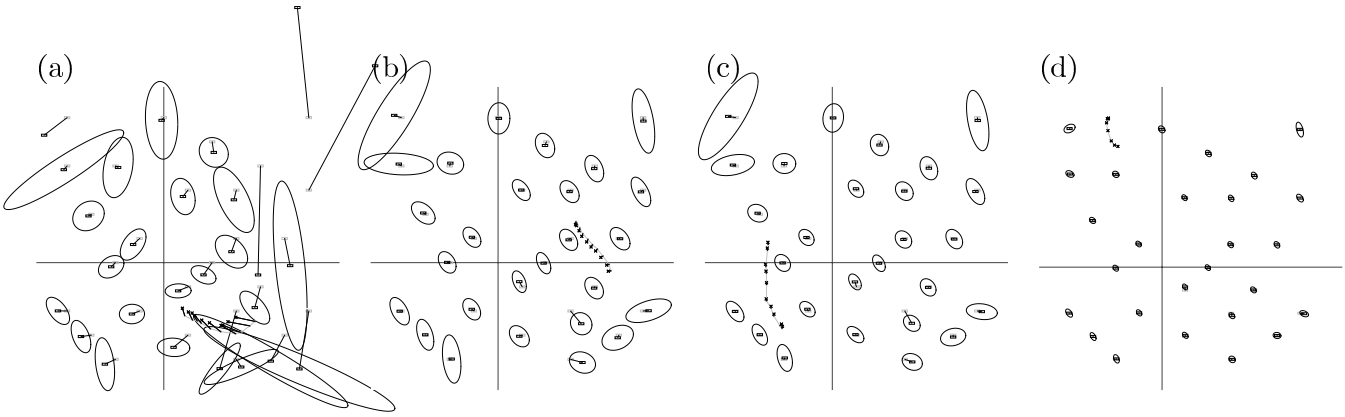


Figure 3: The output of LaSLAT after incorporating (a) 50, (b) 120, (c) 160, and (d) 1510 events. The batch size was 10. Recovered mobile trajectory (crosses) and ground truth mobile trajectory (solid line) for the latest batch are connected by a line to show correspondences. Estimated mobile locations (dark rectangles) and the ground truth mobile locations (light rectangles) are also connected with a line to show correspondence. Error ellipses shrink as more data becomes available. Between events 120 and 160 (subfigures (b) and (c)), the mobile swept around the bottom of the network, and the error ellipses and localization error diminished for those sensors. As sensors became better localized, tracking improved.

best examined visually via Figure 2(b).

We compared the Extended Kalman Filter, which is LaSLAT limited to one Newton-Raphson iteration, to LaSLAT operating with different batch sizes. Figure 4 shows average localization errors as events were processed. The EKF performs best with no batching (batch size = 1). LaSLAT converges faster and also exhibits lower steady state localization error. As batch sizes are increased, so does the rate of convergence of LaSLAT. Batching also improves the final localization error. LaSLAT, with batch sizes of 1, 10 and 40, produced final localization errors of 3 cm, 1.9 cm, and 1.6 cm respectively. On average, LaSLAT took 3 Newton-Raphson iterations to incorporate each batch. The EKF's final localization error was 7.5 cm, which is outside the error tolerance for the ground truth.

We simulated the distributed version of our algorithm described in Section 4 using various batch sizes. In this simulation, we replaced the linear-system solver used in each Newton-Raphson iteration with a solver that uses Gauss-Seidel iteration and hence could be distributed in a straightforward way. Our simulations show that on average, only three to four Gauss-Seidel iterations are sufficient to implement each Newton-Raphson iteration, for a total average of about 9 Gauss-Seidel iterations for processing each batch. Gauss-Seidel based localization and tracking convergence and accuracy was identical to the centralized computations reported in this section.

Figure 5 shows localization results on a larger network (49 sensors) deployed over a larger area (10 m by 17 m). With about 0.3 sensors per square meter, this network is about half as dense as the one shown in Figure 2, which had about 0.5 sensors per square meter. As a result, on average only 5 sensors heard each event, and the localization error was about 7.5 cm. The algorithm also determined a measurement biases of about 20 cm for all nodes. For all batch sizes, the EKF produced an average localization error of about 80 cm, showing that the improvement due to Laplace's method can be very significant.

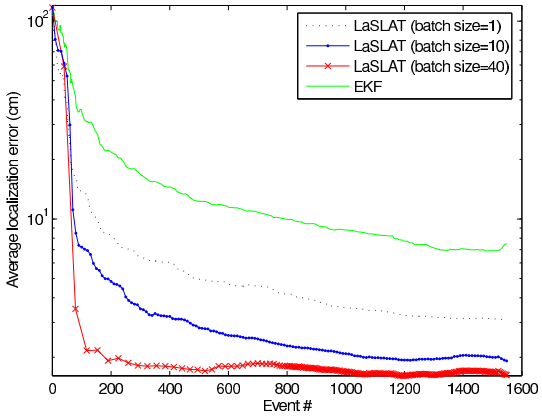


Figure 4: Localization error as a function of the number of events observed for EKF and various batch sizes for LaSLAT. LaSLAT converges more quickly and attains a lower steady state error than the EKF. Furthermore, larger batch sizes improve the convergence rate and the steady state error of LaSLAT.

Figure 5: LaSLAT localization result on a sparser sensor network with 49 nodes in a 10m by 17m environment. Crosses indicate the recovered sensor locations, projected onto the image. The average localization error was 7.5 cm.

Figure 6: Environment used for initial 3d localization experiments. LaSLAT localized sensors to within 7 cm.

5.1 Extending LaSLAT to 3D

It is possible to use LaSLAT to localize and track sensors and mobiles in a three dimensional environment. In our initial experiments, we placed 40 crickets on the floor and walls of a 4 x 6 meter room, which contained all of its normal furniture: tables, chairs, printers, and a refrigerator. The mobile was a special cricket with additional ultrasound transmitters attached so that it could broadcast in all directions. This mobile was carried by hand through the room and moved completely arbitrarily, including changes in speed, loops, and twists.

We made several adjustments to better accomodate the new environment. First, the measurement model was extended to detect and reject range measurement outliers due to ultrasound echoes. In the new model, measurements typically have additive Gaussian noise as in 2d, but are occasionally completely incorrect, which we represent by a uniform distribution. This change in model required us to use an Expectation Maximization (EM) algorithm to optimize equation (1).

Our initial results suggest that LaSLAT is capable of achieving very good results in three dimensions. In the room shown in figure 6, LaSLAT localized sensors to within 7 cm while successfully tracking the path of the mobile in 3d. Much of this error is accounted for by the difficulty of measuring ground truth in this environment.

6 Conclusion and Future Work

We have demonstrated that adapting Bayesian techniques to SLAT results in very accurate localization. Our algorithm is on-line and uses a Bayesian filter to estimate sensor locations and mobile trajectories. We showed experimentally that we can track and localize sensors to within one or two centimeters.

The Bayesian framework provides many other advantages that we hope to demonstrate in the future. The three dimensional extension described in our results section shows great promise, and we look forward to experimenting with it further. Also, different types of measurements such as bearings could be included by suitably modifying the measurement model. By modeling dynamics on the position of sensors, sensor may be allowed to be move over time. Using our framework, we also hope to simultaneously calibrate sensor parameters (such as a time offset parameter when computing ranges from time differences of arrivals on the Crickets), while localizing the sensors and tracking the mobile. Finally, we hope to complete a true distributed implementation of LaSLAT on a real sensor network.

7 Acknowledgments

We are grateful to Michel Goraczko for providing the cricket hardware, David Moore for providing the small cricket dataset and code and assistance towards acquiring the larger cricket dataset. This research is partly supported by DARPA under contract number F33615-01-C-1896, whose program coordinator, Vijay Raghavan, provided helpful discussions.

8 Appendix: Finding a Mode

The update (5) can be rewritten in the form of non-linear least squares. Letting $f_{ij}(\mathbf{x}) = \|\mathbf{s}_i^x - \mathbf{e}_j^t\| + \mathbf{s}_i^\theta$, defining $\mathbf{f}(\mathbf{x})$ as a column vector consisting of all $f_{ij}(\mathbf{x})$, $\Omega_\mathbf{x} = \mathrm{Cov}^{-1}\left[\mathbf{x}^t|\mathbf{y}^{old}\right]$, and $\mu_\mathbf{x} = E\left[\mathbf{x}^t|\mathbf{y}^{old}\right]$, recast (5) as:

$$\arg\min_\mathbf{x} \frac{1}{\sigma^2}\|\mathbf{f}(\mathbf{x}) - \mathbf{y}^t\|^2 + (\mathbf{x}-\mu_\mathbf{x})^T\Omega_\mathbf{x}(\mathbf{x}-\mu_\mathbf{x}) \tag{10}$$

Each iteration of Newton-Raphson maps an iterate $\mathbf{x}^{(t)}$ to the next iterate $\mathbf{x}^{(t+1)}$ by approximating (10) via linearization about $\mathbf{x}^{(t)}$, and optimizing over $\mathbf{x}$:

$$\mathbf{x}^{(t+1)} = \arg\min_\mathbf{x} \frac{1}{\sigma^2}\left\|\nabla\mathbf{f}^{(t)}\mathbf{x} - \mathbf{b}\right\|^2 + (\mathbf{x}-\mu_\mathbf{x})^T\Omega_\mathbf{x}(\mathbf{x}-\mu_\mathbf{x}), \tag{11}$$

where $\nabla\mathbf{f}^{(t)}$ is the derivative of $\mathbf{f}$ with respect to $\mathbf{x}$ at $\mathbf{x}^{(t)}$, and $\mathbf{b} = \nabla\mathbf{f}^{(t)}\mathbf{x}^{(t)} - \mathbf{f}\left(\mathbf{x}^{(t)}\right) - \mathbf{y}^t$.

This is a linear least squares problem in terms of $\mathbf{x}$. Its solution can be found by setting the derivative with respect to $\mathbf{x}$ to zero to obtain a linear problem that can be solved by matrix inversion:

$$\left[\Omega + \frac{1}{\sigma^2}\nabla\mathbf{f}^{(t)\top}\nabla\mathbf{f}^{(t)}\right]\mathbf{x} = \Omega\mu + \frac{1}{\sigma^2}\nabla\mathbf{f}^{(t)\top}\mathbf{b}. \tag{12}$$

Furthermore, differentiating (11) one more time results in $\mathbf{H} = \Omega + \frac{1}{\sigma^2}\nabla\mathbf{f}^{(t)\top}\nabla\mathbf{f}^{(t)}$. Since (11) is an approximation to the negative log posterior (5), $\mathbf{H}$ serves as an approximation to its Hessian at $\mathbf{x}^{(t)}$.

Because the true distance f_{ij} depends only on sensor i and event location j, each row of $\nabla\mathbf{f}^{(t)}$ is made up of zeros, except at locations corresponding to the ith sensor and the jth event. Thus $\nabla\mathbf{f}^{(t)\top}\nabla\mathbf{f}^{(t)}$ has local connectivity. If $\Omega_\mathbf{x} = \mathrm{Cov}^{-1}\left[\mathbf{x}^t|\mathbf{y}^{old}\right]$ has local connectivity, then the updated covariance matrix $\mathrm{Cov}^{-1}\left[\mathbf{x}^t|\mathbf{y}^{old}\right] = \Omega + \nabla\mathbf{f}^{(t)\top}\nabla\mathbf{f}^{(t)}/\sigma^2$ also has local connectivity. Therefore incorporating a batch of measurements preserves local connectivity.

References

[1] P. Pathirana, N. Bulusu, S. Jha, and A. Savkin, "Node localization using mobile robots in delay-tolerant sensor networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 4, Jul/Aug 2005.

[2] A. Galstyan, B. Krishnamachari, K. Lerman, and S. Pattem, "Distributed online localization in sensor networks using a moving target," in *Information Processing In Sensor Networks (IPSN)*, 2004.

[3] V. Cevher and J.H. McClellan, "Sensor array calibration via tracking with the extended kalman filter," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2001, vol. 5, pp. 2817–2820.

[4] J. J. Leonard and P. M. Newman, "Consistent, convergent, and constant-time SLAM," in *IJCAI*, 2003.

[5] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun, "Using EM to learn 3D models of indoor environments with mobile robots," in *IEEE International Conference on Machine Learning (ICML)*, 2001.

[6] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," Submitted for journal publication, April 2003.

[7] A.J. Davison and D.W. Murray, "Simultaneous localization and map-building using active vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 865–880, 2002.

[8] N. Ayache and O. Faugeras, "Maintaining representations of the environment of a mobile robot," *IEEE Tran. Robot. Automat.*, vol. 5, no. 6, pp. 804–819, 1989.

[9] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Uncertainity in Artificial Intelligence*, 1988.

[10] H. Balakrishnan, R. Baliga, D. Curtis, M. Goraczko, A. Miu, N. B. Priyantha, A. Smith, K. Steele, S. Teller, and K. Wang, "Lessons from developing and deploying the cricket indoor location system," Tech. Rep., MIT Computer Science and AI Lab, `http://nms.lcs.mit.edu/projects/cricket/#papers`, 2003.

[11] Shang, Ruml, Zhang, and Fromherz, "Localization from mere connectivity," in *MobiHoc*, 2003.

[12] X. Ji and H. Zha, "Sensor positioning in wireless ad hoc networks using multidimensional scaling," in *Infocom*, 2004.

[13] L. Doherty, L. El Ghaoui, and K. S. J. Pister, "Convex position estimation in wireless sensor networks," in *Proceedings of Infocom 2001*, April 2001.

[14] N. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, "Anchor-free distributed localization in sensor networks," 2003.

[15] David Moore, John Leonard, Daniela Rus, and Seth Teller, "Robust distributed network localization with noisy range measurements," in *Proceedings of ACM Sensys-04*, Nov 2004.

[16] A. Ihler, J. Fisher, R. Moses, and A. Willsky, "Nonparametric belief propagation for self-calibration in sensor networks," in *Information Processing in Sensor Networks (IPSN)*, 2004.

[17] E. Olson, J. J. Leonard, and S. Teller, "Robust range-only beacon localization," in *Proceedings of Autonomous Underwater Vehicles*, 2004.

[18] P. F. McLauchlan, "A batch/recursive algorithm for 3D scene reconstruction," *Conf. Computer Vision and Pattern Recognition*, vol. 2, pp. 738–743, 2000.

[19] A. Gelman, J. Carlin, H. Stern, and D. Rubin., *Bayesian Data Analysis*, Chapman & Hall/CRC, 1995.

[20] G. Golub and C.F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, 1989.

[21] D. P. Bertsekas and J. T. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, 1989.

[22] B. K. P. Horn, "Relative orientation," *International Journal of Computer Vision*, vol. 4, pp. 59–78, Jan. 1989.