In the 7th Intl. Workshop on the Algorithmic Foundations of Robotics

Inferring and Enforcing Relative Constraints in SLAM

Kristopher R. Beevers and Wesley H. Huang

Department of Computer Science, Rensselaer Polytechnic Institute
{beevek,whuang}@cs.rpi.edu

Abstract: Most algorithms for simultaneous localization and mapping (SLAM) do not incorporate prior knowledge of structural or geometrical characteristics of the environment. In some cases, such information is readily available and making some assumptions is reasonable. For example, one can often assume that many walls in an indoor environment are rectilinear. In this paper, we develop a SLAM algorithm that incorporates prior knowledge of relative constraints between landmarks. We describe a "Rao-Blackwellized constraint filter" that infers applicable constraints and efficiently enforces them in a particle filtering framework. We have implemented our approach with rectilinearity constraints. Results from simulated and real-world experiments show the use of constraints leads to consistency improvements and a reduction in the number of particles needed to build maps.

1 Introduction

The simultaneous localization and mapping (SLAM) problem is for a mobile robot to concurrently estimate both a map of its environment and its pose with respect to the map. Most SLAM algorithms make few assumptions about the environment; thus, SLAM does not take advantage of prior information when the environment is known to have specific structural characteristics. For example, a robot designed to operate indoors can often assume its environment is "mostly" rectilinear.

In many cases structural or geometrical assumptions can be represented as information about relative constraints between landmarks in a robot's map, which can be used in inference to determine which landmarks are constrained and the parameters of the constraints. In the rectilinearity example, such a formulation can be used to constrain the walls of a room separately from, say, the boundary of a differently-aligned obstacle in the center of the room.

Given relative constraints between landmarks, they must be enforced. Some previous work has enforced constraints on maps represented using an extended Kalman filter (EKF) [6, 14, 11]. In this paper, we develop techniques to instead enforce constraints in maps represented by a Rao-Blackwellized

particle filter (RBPF). The major difficulty is that RBPF SLAM relies on the conditional independence of landmark estimates given a robot's pose history, but relative constraints introduce correlation between landmarks.

Our approach exploits a property similar to that used in the standard SLAM Rao-Blackwellization: conditioned on values of constrained state variables, unconstrained state variables are independent. We use this fact to incorporate per-particle constraint enforcement into RBPF SLAM. We have also developed a technique to address complications which arise when initializing a constraint between groups of landmarks that are already separately constrained; the technique efficiently recomputes conditional estimates of unconstrained variables when modifying the values of constrained variables.

Incorporating constraints can have a profound effect on the computation required to build maps. A motivating case is the problem of mapping with sparse sensing. In previous work [3], we have shown that particle filtering SLAM is possible with limited sensors such as small arrays of infrared rangefinders, but that many particles are required due to increased measurement uncertainty. By extending sparse sensing SLAM to incorporate constraints, an order-of-magnitude reduction in the number of particles can be achieved.

The paper proceeds as follows. We first discuss previous work on constrained SLAM. Then, in Section 2, we briefly review the general SLAM problem and the ideas behind RBPF, and discuss the assumption of unstructured environments made by most SLAM algorithms. In Section 3 we formalize the idea of SLAM with relative constraints and describe a simple but infeasible approach. We then introduce the Rao-Blackwellized constraint filter: Section 4 describes an RBPF-based algorithm for enforcing constraints, and Section 5 incorporates inference of constraints. Finally, in Section 6 we describe the results of simulated and real-world experiments with a rectilinearity constraint.

1.1 Related work

Most work on SLAM focuses on building maps using very little prior information about the environment, aside from assumptions made in feature extraction and data association. A thorough coverage of much of the state-of-the art in unconstrained SLAM can be found in, e.g., [8].

The problem of inferring when constraints should be applied to a map is largely unexplored. Rodriguez-Losada *et al.* [11] employ a simple thresholding approach to determine which of several types of constraints should be applied.

On the other hand, several researchers have studied the problem of enforcing *a priori* known constraints in SLAM. In particular, Durrant-Whyte [6] and Wen and Durrant-Whyte [14] have enforced constraints in EKF-based SLAM by treating the constraints as zero-uncertainty measurements. More recently, Csorba, Newman and Durrant-Whyte [4, 10] and Deans and Hebert [5] have built maps where the state consists of relationships between landmarks; they apply constraints on the relationships to enforce map consistency. From a consistent relative map an absolute map can be estimated.

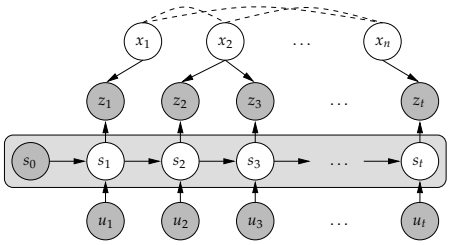


Fig. 1. A Bayes network showing common SLAM model assumptions. Input variables are represented by shaded nodes; the objective of SLAM is to estimate values for the unshaded nodes. Arcs indicate causality or correlation between variables. (Correspondence variables n_t are omitted for clarity — observations are connected directly to the corresponding landmarks.) Correlations between landmarks due to structure in the environment (dashed arcs) are typically ignored in SLAM.

Finally, others have studied general constrained state estimation using the EKF. Simon and Chia [12] derive Kalman updates for linear equality constraints (discussed in detail in Section 3.1) that are equivalent to projecting the unconstrained state onto the constraint surface. In [13], Simon and Simon extend this approach to deal with linear inequality constraints.

2 The SLAM problem

The goal of SLAM is to simultaneously estimate both a map M of the environment and the robot's (time-dependent) pose s_t with respect to the map. A number of map representations exist; we focus on landmark-based mapping with $M = \{x_1, x_2, \ldots, x_n\}$, where each landmark x_i is a parameterized geometric object such as a point or a line. In the basic SLAM process, the robot executes a motion and estimates its new pose using odometry. It then takes a sensor reading and extracts geometric features from the raw sensor data. Data association matches features with landmarks in the map, and the map and pose estimates are updated.

SLAM is often posed in a Bayesian filtering formulation where the goal is to estimate a posterior distribution over poses and maps given all of the measurements z^t, commanded motions u^t, and correspondences n^t between features and landmarks [8]. (The superscript notation indicates a set of values $1 \ldots t$ over all time steps.) A Bayes network depicting the standard SLAM model assumptions is shown in Fig. 1. The filter can be written recursively:

$$p(s_t, M | z^t, u^t, n^t) = \\ \eta p(z_t | s_t, x_{n_t}, n_t) \int p(s_t | s_{t-1}, u_t) p(s_{t-1}, M | z^{t-1}, u^{t-1}, n^{t-1}) \, ds_{t-1} \quad (1)$$

where $p(z_t|s_t, x_{n_t}, n_t)$ is the measurement model, $p(s_t|s_{t-1}, u_t)$ models the robot's motion, and η is a normalization constant. In this approach, SLAM is usually done using the extended Kalman filter (EKF).

An alternative is to filter over the entire robot trajectory s^t, i.e.:

$$p(s^t, M|n^t, z^t, u^t) = \eta p(z_t|s_t, x_{n_t}, n_t)\ p(s_t|s_{t-1}, u_t) p(s^{t-1}, M|n^{t-1}, z^{t-1}, u^{t-1}) \tag{2}$$

Under the assumption that the environment is static and that no direct correlations exist between landmarks, this leads to the observation that landmarks are conditionally independent given the robot's trajectory, since correlation between landmarks arises only through robot pose uncertainty [9]. In Fig. 1, the highlighted variables (the robot's trajectory) *d-separate* the landmark variables. Thus, the posterior over trajectories and maps can be factored:

$$p(s^t, M|n^t, z^t, u^t) = p(s^t|n^t, z^t, u^t) \prod_{i=1}^{n} p(x_i|s^t, n^t, z^t) \tag{3}$$

This factorization is known as Rao-Blackwellization. To perform SLAM based on Eqn. 3, the posterior over trajectories can be represented with a particle filter where each particle samples a single trajectory. Associated with a particle are a number of separate small filters (typically EKFs) to analytically estimate each landmark in the particle's map. This approach is known as Rao-Blackwellized particle filtering (RBPF) and is the basis for the well-known FastSLAM algorithm [8].

2.1 Structured environments

Typically, SLAM approaches assume the environment is *unstructured*, i.e., that landmarks are randomly and independently distributed in the workspace. Often this is not the case, as in indoor environments where landmarks are placed methodically. Thus, some correlation exists between landmarks, due not to the robot's pose uncertainty, but rather to the *structure* in the environment. (This is represented by the dotted arcs in Fig. 1).

Correlation between landmarks can arise in many ways, making it difficult to include in the SLAM model. In this paper, we assume that structure in the environment takes on one of a few forms — i.e., that the space of possible (structured) landmark relationships is small and discrete. When this is the case, the model shown in Fig. 2 can be used. Here, arcs indicating correlation between landmarks are parameterized. The parameters $c_{i,j}$ indicate the constraints (or lack thereof) between landmarks x_i and x_j. We perform inference on the constraint parameter space, and then enforce the constraints. In this paper we focus on the pairwise case, but more complicated relationships can in principle be exploited.

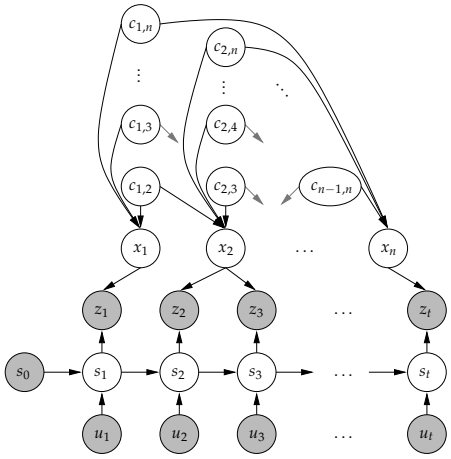


Fig. 2. Bayes network for a SLAM model that incorporates pairwise constraints between landmarks, parameterized by the variables $c_{i,j}$. Inference in the space of relationship parameters can be used to determine correlations between landmark parameters; relative constraints on the landmarks enforce inferred relationships.

3 SLAM with relative constraints

We begin by addressing the issue of efficiently *enforcing* known relative constraints. Parallel to this is the problem of *merging* constraints when new relationships are found between separate groups of already constrained landmarks.

Throughout the rest of the paper we omit time indices for clarity. Variables are vectors unless otherwise noted. We use P_i to represent the covariance of the landmark estimate x_i. We assume that measurements of a landmark are in the parameter space of the landmark (i.e., measurements are of the landmark state). Measurements that do not meet this condition can easily be transformed. Finally, while we present our formulation for a single constraint, the approach can be applied in parallel to several types of constraints.

3.1 The superlandmark filter

There is an immediate problem with SLAM when the environment is structured: landmark correlations lead to interdependencies that break the factorization utilized in Eqn. 3, which assumes correlation arises only through robot pose uncertainty. We first describe a simple (but ultimately impractical) approach to deal with the correlation, which leads to an improved technique in Section 4. Note that the RBPF factorization still holds for unconstrained landmarks; we rewrite the filter, grouping constrained landmarks. Formally, partition the map into groups:

$$\mathcal{L} = \{\{x_{a_1}, x_{a_2}, \ldots\}, \{x_{b_1}, x_{b_2}, \ldots\}, \{x_c\}, \ldots\} \tag{4}$$

Each group ("superlandmark") $L_i \in \mathcal{L}$ contains landmarks constrained with respect to each other; correlation arises only among landmarks in the same group. We immediately have the following version of the RBPF SLAM filter:

$$p(s^t, M|n^t, z^t, u^t) = p(s^t|n^t, z^t, u^t) \prod_{i=1}^{|\mathcal{L}|} p(L_i|s^t, n^t, z^t) \tag{5}$$

We can still apply a particle filter to estimate the robot's trajectory. Each superlandmark is estimated using an EKF, which accounts for correlation due to constraints since it maintains full covariance information.

There are several ways to enforce constraints on a superlandmark. One approach is to treat the constraints as zero-uncertainty measurements of the constrained landmarks [6, 14, 11]. An alternative is to directly incorporate constrained estimation into the Kalman filter. Simon and Chia [12] have derived a version of the EKF that accounts for equality constraints of the form

$$DL_i = d \tag{6}$$

where L_i represents the superlandmark state with n variables, D is an $s \times n$ constant matrix of full rank, and d is a $s \times 1$ vector; together they encode s constraints. In their approach, the unconstrained EKF estimate is computed and then repaired to account for the constraints. Given the unconstrained state L_i and covariance matrix P_{L_i}, the constrained state and covariance are computed as follows (see [12] for the derivation):

$$\tilde{L}_i \leftarrow L_i - PD^T(DPD^T)^{-1}(DL_i - d) \tag{7}$$

$$\tilde{P}_{L_i} \leftarrow P_{L_i} - P_{L_i}D^T(DP_{L_i}D^T)^{-1}DP_{L_i} \tag{8}$$

i.e., the unconstrained estimate is projected onto the constraint surface.

If a constraint arises between two superlandmarks they are easily merged:

$$L_{ij} \leftarrow \begin{bmatrix} L_i \\ L_j \end{bmatrix} \quad , \quad P_{ij} \leftarrow \begin{bmatrix} P_i & P_i \frac{\partial L_j}{\partial L_i}^T \\ \frac{\partial L_j}{\partial L_i} P_i & P_j \end{bmatrix} \tag{9}$$

Unfortunately, the superlandmark filter is too expensive unless the size of superlandmarks can be bounded by a constant. In the worst case the environment is highly constrained and, in the extreme, the map consists of a single superlandmark. EKF updates for SLAM take at least $O(n^2)$ time and constraint enforcement using Eqns. 7 and 8 requires $O(n^3)$ time for a superlandmark of size n. If the particle filter has N particles, the superlandmark filter requires $O(Nn^3)$ time for a single update. We thus require a better solution.

3.2 Reduced state formulation

A simple improvement can be obtained by noting that maintaining the full state and covariance for each landmark in a superlandmark is unnecessary. Constrained state variables are redundant: given the value of the

variables from one "representative" landmark, the values for the remaining landmarks in a superlandmark are determined. In the rectilinearity example, with landmarks represented as lines parameterized by distance r and angle θ to the origin, a full superlandmark state vector has the form: $[r_1\ \theta_1\ r_2\ \theta_2\ \ldots\ r_n\ \theta_n]^T$. If the $\{\theta_i\}$ are constrained the state can be rewritten as: $[r_1\ \theta_1\ r_2\ g_2(c_{1,2};\theta_1)\ \ldots\ r_n\ g_n(c_{1,n};\theta_1)]^T$. Thus, filtering of the superlandmark need only be done over the reduced state: $[r_1\ r_2\ \ldots\ r_n\ \theta_1]^T$. The function $g_i(c_{j,i};x_{j,\rho})$ with parameters $c_{j,i}$ maps the constrained variables $x_{j,\rho}$ of the representative landmark x_j to values for $x_{i,\rho}$; in the rectilinearity case, $c_{j,i} \in \{0, 90, 180, 270\}$ and $g_i(c_{j,i};\theta_j) = \theta_j - c_{j,i}$. We assume the constraints are invertible: the function $h_i(c_{j,i};x_{i,\rho})$ represents the reverse mapping, e.g., $h_i(c_{j,i};\theta_i) = \theta_i + c_{j,i}$. We sometimes refer to the unconstrained variables of landmark x_i as $x_{i,\overline{\rho}}$.

4 Rao-Blackwellized constraint filter

From the reduced state formulation we see it is easy to separate the map state into constrained variables $M^c = \{x_{1,\rho}, \ldots, x_{n,\rho}\}$, and unconstrained variables $M^f = \{x_{1,\overline{\rho}}, \ldots, x_{n,\overline{\rho}}\}$. By the same reasoning behind Eqn. 3, we factor the SLAM filter as follows:

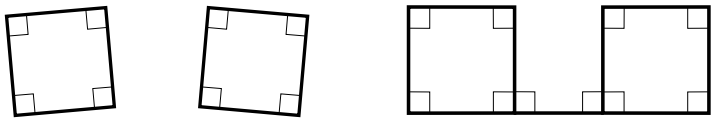
$$p(s^t, M|n^t, z^t, u^t) = p(s^t, M^c|n^t, z^t, u^t) \prod_{i=1}^{|M^f|} p(x_{i,\overline{\rho}}|s^t, M^c, n^t, z^t) \qquad (10)$$

In other words, conditioned on *both* the robot's trajectory and the values of all constrained variables, free variables of separate landmarks are independent.

Eqn. 10 suggests that we can use a particle filter to estimate both the robot trajectory and the values of the constrained variables. We can then use separate small filters to estimate the unconstrained variables conditioned on sampled values of the constrained variables. The estimation of representative values for the constrained variables is accounted for in the particle filter resampling process, where particles are weighted by data association likelihood.

4.1 Particlization of landmark variables

We first discuss initialization of constraints between previously unconstrained landmarks. Given a set $\mathcal{R} = \{x_1, x_2, \ldots, x_n\}$ of landmarks to be constrained, along with constraint parameters $c_{1,i}$ for each $x_i \in \mathcal{R}, i = 2 \ldots n$ (i.e., with x_1 as the "representative" landmark — see Section 3.2), we form a superlandmark from $\mathcal{R}$. Then, we perform a *particlization procedure*, sampling the constrained variables from the reduced state of the superlandmark. Conditioning of the unconstrained variables of every landmark in the superlandmark is performed using the sampled values. We are left with an EKF for each landmark that estimates only the values of unconstrained state variables.



(a) (b)

Fig. 3. Merging groups of constrained landmarks. (a) Two constrained groups of landmarks. (b) After finding a new landmark constrained with respect to both groups, all landmarks are constrained together.

In selecting values of the constrained variables on which to condition, we should take into account all available information, i.e., the estimates of the constrained variables from each landmark. We compute the maximum likelihood estimate of the constrained variables:

$$P_{\hat{\rho}} \leftarrow \left(\sum_{x_j \in \mathcal{R}} P_{j,\rho}^{-1} \right)^{-1} \quad , \quad \hat{\rho} \leftarrow P_{\hat{\rho}}^{-1} \left(\sum_{x_j \in \mathcal{R}} h_j(c_{1,j}; x_{j,\rho}) P_{j,\rho}^{-1} \right) \tag{11}$$

To choose values for ρ, we can either sample, e.g., according to $\mathcal{N}(\hat{\rho}, P_{\hat{\rho}})$; or we can simply pick $\hat{\rho}$, which is the approach we take in our implementation.

Once values of constrained variables are selected, we condition the unconstrained variables on the selected values. To condition x_i with covariance P_i on values for $x_{i,\rho}$, we first partition the state and covariance:

$$x_i = [x_{i,\overline{\rho}} \;\; x_{i,\rho}]^T \quad , \quad P_i = \begin{bmatrix} P_{i,\overline{\rho}} & P_{i,\overline{\rho}\rho} \\ P_{i,\rho\overline{\rho}} & P_{i,\rho} \end{bmatrix} \tag{12}$$

Then given $x_{i,\rho} = g_i(c_{1,i}; \hat{\rho})$ and since landmark state is estimated by an EKF, the standard procedure for conditioning the Normal distribution yields:

$$\tilde{x}_{i,\overline{\rho}} \leftarrow x_{i,\overline{\rho}} + P_{i,\overline{\rho}\rho} P_{i,\rho}^{-1} (g_i(c_{1,i}; \hat{\rho}) - x_{i,\rho}) \tag{13}$$

$$\tilde{P}_{i,\overline{\rho}} \leftarrow P_{i,\overline{\rho}} - P_{i,\overline{\rho}\rho} P_{i,\rho}^{-1} P_{i,\overline{\rho}\rho}^T \tag{14}$$

For purposes of data association it is convenient to retain the full state and covariance, in which case $\tilde{x}_{i,\rho} = g_i(c_{1,i}; \hat{\rho})$ and $\tilde{P}_{i,\rho} = \tilde{P}_{i,\overline{\rho}\rho} = \tilde{P}_{i,\rho\overline{\rho}} = [\mathbf{0}]$.

4.2 Reconditioning

Particlization is straightforward if none of the landmarks is already constrained. This is not the case when a new landmark is added to a superlandmark or when merging several constrained superlandmarks. Since the values of unconstrained state variables are already conditioned on values of the constrained variables, we cannot change constrained variables without invalidating the conditioning. Such a situation is depicted in Fig. 3.

One solution is to "rewind" the process to the point when the landmarks were first constrained and then "replay" all of the measurements of the landmarks, conditioning on the new values of the constrained variables. This is clearly infeasible. However, we can achieve an equivalent result efficiently because the order in which measurements are applied is irrelevant. Applying k measurements to the landmark state is equivalent to merging $k+1$ Gaussians. Thus, we can "accumulate" all of the measurements in a single Gaussian and apply this instead, in unit time.

From this, we obtain the following reconditioning approach:

1. Upon first constraining a landmark x_i, store its pre-particlization unconstrained state $\beta_i = x_i, \Lambda_i = P_i$, initialize the "measurement accumulator" $\mathcal{Z}_i = [\mathbf{0}], \mathcal{Q}_i = [\infty]$, and particlize the landmark.
2. For a measurement z with covariance R of the constrained landmark update both the conditional state and the measurement accumulator:

$$x_i \leftarrow x_i + P_i(P_i + R)^{-1}(z - x_i) \tag{15}$$
$$P_i \leftarrow P_i - P_i(P_i + R)^{-1}P_i^T \tag{16}$$
$$\mathcal{Z}_i \leftarrow \mathcal{Z}_i + \mathcal{Q}_i(\mathcal{Q}_i + R)^{-1}(z - \mathcal{Z}_i) \tag{17}$$
$$\mathcal{Q}_i \leftarrow \mathcal{Q}_i - \mathcal{Q}_i(\mathcal{Q}_i + R)^{-1}\mathcal{Q}_i^T \tag{18}$$

3. When instantiating a new constraint on x_i, recondition x_i on the new constrained variable values by rewinding the landmark state ($x_i = \beta_i$, $P_i = \Lambda_i$), computing the conditional distribution $\tilde{x}_i, \tilde{P}_i$ of the state (Eqns. 13-14), and replaying the measurements since particlization with:

$$x_i \leftarrow \tilde{x}_i + \tilde{P}_i(\tilde{P}_i + \mathcal{Q}_i)^{-1}(\mathcal{Z}_i - \tilde{x}_i) \tag{19}$$
$$P_i \leftarrow \tilde{P}_i - \tilde{P}_i(\tilde{P}_i + \mathcal{Q}_i)^{-1}\tilde{P}_i^T \tag{20}$$

The reconditioning technique can be extended to handle multiple types of constraints simultaneously by separately storing the pre-particlization state and accumulated measurements for each constraint. Only completely unconstrained state variables should be stored at constraint initialization, and only the measurements of those variables need be accumulated.

4.3 Discussion

A potential issue with our approach is that reconditioning neither re-evaluates data associations nor modifies the trajectory of a particle. In practice we have observed that the effect on map estimation is negligible.

Computationally, the constrained RBPF approach is a significant improvement over the superlandmark filter, requiring only $O(Nn)$ time per update.[1]

[1] We note that while the data structures that enable $O(N \log n)$ time updates for FastSLAM [8] can still be applied, they do not improve the complexity of constrained RBPF since the reconditioning step is worst-case linear in n.

At first it appears that more particles may be necessary since representative values of constrained variables are now estimated by the particle filter. However, incorporating constraints often leads to a significant reduction in required particles by reducing the degrees of freedom in the map. In a highly constrained environment, particles only need to filter a few constrained variables using the reduced state, and the EKFs for unconstrained variables are smaller since they filter only over the unconstrained state. By applying strong constraint priors where appropriate, the number of particles required to build maps is often reduced by an order of magnitude, as can be seen in Section 6.

4.4 Inequality constraints

So far we have only considered equality constraints, whereas many useful constraints are inequalities. For example, we might specify a prior on corridor width: two parallel walls should be at least a certain distance apart. In [13], the authors apply inequality constraints to an EKF using an active set approach. At each time step, the applicable constraints are tested. If a required inequality is violated, an equality constraint is applied, projecting the unconstrained state onto the boundary of the constraint region.

While this approach appears to have some potential problems (e.g., it ignores the landmark PDF over the unconstrained half-hyperplane in parameter space), a similar technique can be incorporated into the Rao-Blackwellized constraint filter. After updating a landmark, applicable inequality constraints are tested. Constraints that are violated are enforced using the techniques described in Section 4. The unconstrained state is accessible via the measurement accumulator, so if the inequality is later satisfied, the parameters can be "de-particlized" by switching back to the unconstrained estimate.

5 Inference of constraints

We now address the problem of deducing the relationships between landmarks, i.e., deciding when a constraint should be applied. A simple approach is to just examine the unconstrained landmark estimates. In the rectilinearity case, we can easily compute the estimated angle between two landmarks. If this angle is "close enough" to one of $0°, 90°, 180°$, or $270°$, the constraint is applied to the landmarks. (A similar approach is used by Rodriguez-Losada *et al.* [11].) However, this technique ignores the confidence in the landmark estimates.

We instead compute a PMF over the space $\mathcal{C}$ of pairwise constraint parameters; the PMF incorporates the landmark PDFs. In the rectilinearity example, $\mathcal{C} = \{0, 90, 180, 270, \star\}$, where $\star$ is used to indicate that landmarks are unconstrained. Given a PMF over $\mathcal{C}$, we sample constraint parameters for each particle to do inference of constraints. Particles with incorrectly constrained landmarks will yield poor data associations and be resampled.

We compute the PMF of the "relationship" of landmarks x_i and x_j using:

$$p(c_{i,j}) = \int p(x_{i,\rho}) \int_{h_j(c_{i,j};x_{j,\rho})-\delta}^{h_j(c_{i,j};x_{j,\rho})+\delta} p(x_{j,\rho})\, dx_{j,\rho}\, dx_{i,\rho} \tag{21}$$

for all $c_{i,j} \in \mathcal{C} \setminus \star$. Then, $p(\star) = 1 - \sum_{c_{i,j} \in \mathcal{C} \setminus \star} p(c_{i,j})$. The parameter δ encodes "prior information" about the environment: the larger the value of δ, the more liberally we apply constraints. A benefit of this approach is that the integrals can be computed efficiently from standard approximations to the Normal CDF since the landmarks are estimated by EKFs.

In the rectilinearity case, given orientation estimates described by the PDFs $p(\theta_i)$ and $p(\theta_j)$, for $c_{i,j} \in \{0, 90, 180, 270\}$, we have:

$$p(c_{i,j}) = \int_{-\infty}^{\infty} p(\theta_i) \int_{\theta_i + c_{i,j} - \delta}^{\theta_i + c_{i,j} + \delta} p(\theta_j)\, d\theta_j\, d\theta_i \tag{22}$$

which gives a valid PMF as long as $\delta \leq 45°$.

6 Results

We have now described the complete approach for implementing constrained RBPF SLAM. Algorithm 1 gives pseudocode for initializing a landmark x_{n+1} given the current set of superlandmarks $\mathcal{L}$. Algorithm 2 shows how to update a (possibly constrained) landmark given a measurement of its state. The algorithms simply collect the steps described in detail in Sections 4 and 5.

We have implemented the Rao-Blackwellized constraint filter for the rectilinearity constraint described earlier, on top of our algorithm for RBPF SLAM with sparse sensing [3], which extracts features using data from multiple poses. Because of the sparseness of the sensor data, unconstrained SLAM typically requires many particles to deal with high uncertainty. We performed several experiments, using both simulated and real data, which show that incorporating prior knowledge and enforcing constraints leads to a significant improvement in the resulting maps and a reduction in estimation error.

6.1 Simulated data

We first used a simple kinematic simulator based on an RWI MagellanPro robot to collect data from a small simulated environment with two groups of rectilinear features. The goal was to test the algorithm's capability to infer the existence of constraints between landmarks. Only the five range sensors at $0°, 45°, 90°, 135°$, and $180°$ were used (i.e., ). Noise was introduced by perturbing measurements and motions in proportion to their magnitude. For a laser measurement of range r, $\sigma_r = 0.01r$; for a motion consisting of a translation d and rotation ϕ, the robot's orientation was perturbed with $\sigma_\theta = 0.03d + 0.08\phi$, and its position with $\sigma_x = \sigma_y = 0.05d$.

Algorithm 1 INITIALIZE-LANDMARK$(x_{n+1}, P_{n+1}, \mathcal{L})$

$\beta_{n+1} \leftarrow x_{n+1}$; $\Lambda_{n+1} = P_{n+1}$ // *initialize backup state*
$\mathcal{Z}_{n+1} \leftarrow [\mathbf{0}]$; $\mathcal{Q}_{n+1} \leftarrow [\infty]$ // *initialize measurement accumulator*
$\mathcal{R} \leftarrow \{\}$ // *initialize constraint set*
for all $L_i \in \mathcal{L}$ **do** // *previously constrained groups*
  $c_{n+1,j} \sim p(c_{n+1,j}), \ \forall x_j \in L_i$ // *draw constraint parameters*
  if $\exists x_j \in L_i$ such that $c_{n+1,j} \neq \star$ **then** // *constrained?*
    for all $x_j \in L_i$ **do**
      $\mathcal{R} \leftarrow \mathcal{R} \cup \{x_j\}$ // *add x_j to constraint set*
    $\mathcal{L} \leftarrow \mathcal{L} \setminus L_i$ // *remove old superlandmark*
if $\mathcal{R} = \emptyset$ **then**
  return // *no constraints on x_{n+1}*
$\mathcal{R} \leftarrow \mathcal{R} \cup \{x_{n+1}\}$ // *add new landmark to constraint set*
$\mathcal{L} \leftarrow \mathcal{L} \cup \{\mathcal{R}\}$ // *add new superlandmark*
for all $x_j \in \mathcal{R}$ **do** // *for all constrained landmarks*
  $\hat{x}_j \leftarrow \beta_j + \Lambda_j \mathcal{Q}_j^{-1}(\mathcal{Z}_j - \beta_j)$ // *compute unconstrained state estimate*
  $\hat{P}_j \leftarrow \Lambda_j - \Lambda_j \mathcal{Q}_j^{-1} \Lambda_j^T$ // *compute unconstrained covariance*
$P_{\hat{\rho}} \leftarrow \left(\sum_{x_j \in \mathcal{R}} P_{j,\rho}^{-1}\right)^{-1}$ // *covariance of ML estimate of ρ*
$\hat{\rho} \leftarrow P_{\hat{\rho}}^{-1} \left(\sum_{x_j \in \mathcal{R}} h_j(c_{n+1,j}; x_{j,\rho}) P_{j,\rho}^{-1}\right)$ // *ML estimate of ρ*
for all $x_j \in \mathcal{R}$ **do** // *for all constrained landmarks*
  $x_j \leftarrow \beta_j$; $P_j \leftarrow \Lambda_j$ // *"rewind" state to pre-particlized version*
  $x_{j,\overline{\rho}} \leftarrow x_{j,\overline{\rho}} + P_{j,\overline{\rho}\rho} P_{j,\rho}^{-1} \left(g_j(c_{n+1,j}; \hat{\rho}) - x_{j,\rho}\right)$ // *conditional mean given ρ*
  $P_{j,\overline{\rho}} \leftarrow P_{j,\overline{\rho}} - P_{j,\overline{\rho}\rho} P_{j,\rho}^{-1} P_{j,\overline{\rho}\rho}^T$ // *conditional covariance*
  $x_{j,\rho} \leftarrow g_j(c_{n+1,j}; \hat{\rho})$; $P_{j,\rho} \leftarrow [\mathbf{0}]$; $P_{j,\overline{\rho}\rho} \leftarrow [\mathbf{0}]$ // *fix constrained variables*
  $x_j \leftarrow x_j + P_j(P_j + \mathcal{Q}_j)^{-1}(\mathcal{Z}_j - x_j)$ // *"replay" meas. since particlization*
  $P_j \leftarrow P_j - P_j(P_j + \mathcal{Q}_j)^{-1} P_j^T$

Algorithm 2 UPDATE-LANDMARK(x_j, P_j, z, R)

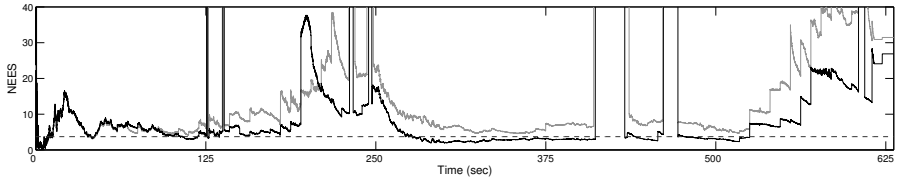
$x_j \leftarrow x_j + P_j(P_j + R)^{-1}(z - x_j)$ // *update state*
$P_j \leftarrow P_j - P_j(P_j + R)^{-1} P_j^T$ // *update covariance*
if $\exists L \in \mathcal{L}, x_k \in L$ such that $x_j \in L$ and $x_j \neq x_k$ **then** // *is x_j constrained?*
  $\mathcal{Z}_j \leftarrow \mathcal{Z}_j + \mathcal{Q}_j(\mathcal{Q}_j + R)^{-1}(z - \mathcal{Z}_j)$ // *update measurement accumulator*
  $\mathcal{Q}_j \leftarrow \mathcal{Q}_j - \mathcal{Q}_j(\mathcal{Q}_j + R)^{-1} \mathcal{Q}_j^T$ // *update accumulator covariance*
else // *not constrained*
  $\beta_j \leftarrow x_j$; $\Lambda_j \leftarrow P_j$ // *update backup state/covariance*

Fig. 4 shows the results of RBPF SLAM with a rectilinearity prior (as described in Section 5, with $\delta = \frac{\pi}{10}$). The filter contained 20 particles and recovered the correct relative constraints. The edges of the the inner "box" were constrained, and the edges of the boundary were separately constrained.

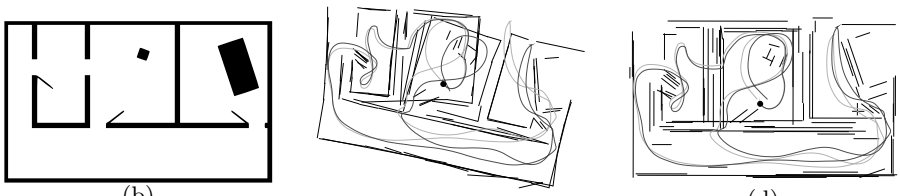
A separate experiment compared the consistency of the rectilinearity-constrained filter and the unconstrained filter (all other filter parameters were kept identical, including number of particles). A filter is inconsistent if it significantly underestimates its own error. It has been shown that RBPF SLAM is

(a) (b)

Fig. 4. (a) Simulated environment (ground truth). (b) Results of applying constrained SLAM. The dark curved line is the trajectory estimate, the light curved line is the ground truth trajectory, and the dot is the starting pose. The landmarks on the boundary form one constrained group; those in the interior form the other.



(a)



(b) (c) (d)

Fig. 5. (a) Normalized estimation error squared (NEES) of the robot's estimated pose with respect to the ground truth, computed over 50 Monte Carlo trials for the environment in (b). The gray plot is the error for standard (unconstrained) RBPF SLAM. The black plot is the error for our algorithm with rectilinearity constraints. Error significantly above the dashed line indicates an optimistic (inconsistent) filter. Our approach is less optimistic. (Sharp spikes correspond to degeneracies due to resampling upon loop closure.) (c) A typical map produced by unconstrained sparse sensing SLAM. (d) A typical rectilinearity-constrained map.

generally inconsistent [1]; our experiments indicate that using prior knowledge and enforcing constraints improves (but does not guarantee) consistency.

Fig. 5 depicts the consistency analysis. The ground truth trajectory from the simulation was used to compute the normalized estimation error squared (NEES) [2, 1] of the robot's trajectory estimate. For ground truth pose s_t and estimate $\hat{s}_t$ with covariance $\hat{P}_{s_t}$ (estimated from the weighted particles assuming they are approximately normally distributed), the NEES is $(s_t - \hat{s}_t)\hat{P}_{s_t}^{-1}(s_t - \hat{s}_t)^T$. For more details of how NEES can be used to examine

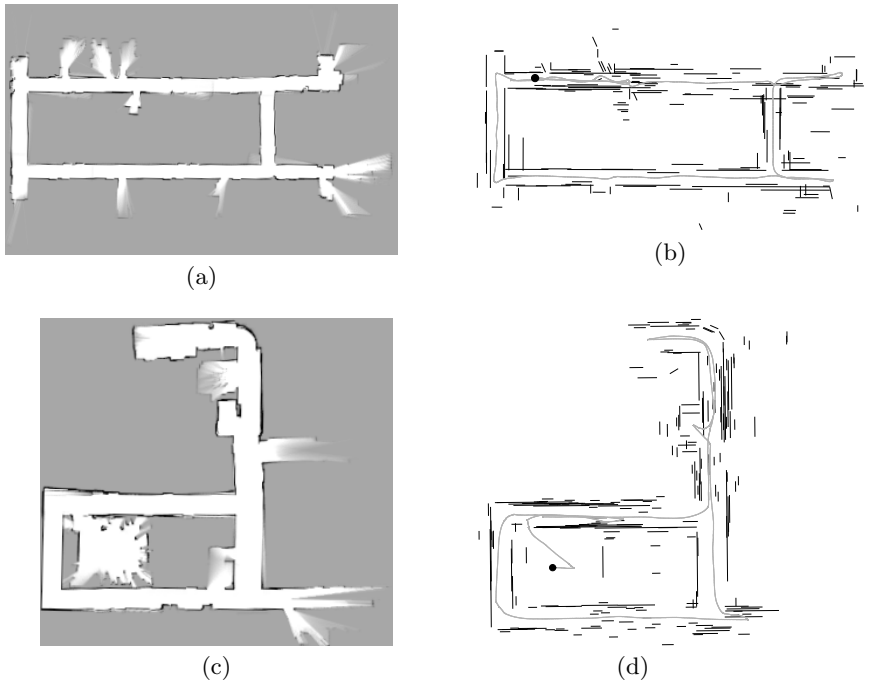


Fig. 6. (a) and (b) show the USC SAL Building, second floor (dataset courtesy of Andrew Howard). (c) and (d) show Newell-Simon Hall Level A at CMU (dataset courtesy of Nicholas Roy). (a) and (c) Occupancy data for the corrected trajectories (generated using the full laser data for clarity). (b) and (d) The estimated landmark maps (black) and trajectories (gray).

SLAM filter consistency, see [1]. The experiment used 200 particles for each of 50 Monte Carlo trials, with a robot model similar to the previous simulation.

6.2 Real-world data

Our real-world experiments used data from Radish [7], an online repository of SLAM datasets. Most of the datasets use scanning laser rangefinders. Since our goal is to enable SLAM with limited sensing, we simply discarded most of the data in each scan, keeping only the five range measurements at $0°, 45°, 90°, 135°$, and $180°$. We also restricted the sensor range (see Table 1). We used the same rectilinearity prior as for the simulated examples ($\delta = \frac{\pi}{10}$).

Fig. 6 shows the results of our algorithm for two datasets. The *USC SAL* dataset consists of a primary loop and several small excursions. Most landmarks are constrained, in three separate groups. For the *CMU NSH* experiment, the maximum sensing range was restricted to 3 m, so the large initial loop (bottom) could not be closed until the robot finished exploring the up-

Table 1. Experiment statistics

	USC SAL	CMU NSH
Dimensions	39×20 m^2	25×25 m^2
Particles (constrained)	20	40
Particles (unconstrained)	100	600
Avg. Runtime (constrained, 30 runs)	11.24 s	34.77 s
Avg. Runtime (unconstrained, 30 runs)	32.02 s	268.44 s
Sensing range	5 m	3 m
Path length	122 m	114 m
Num. landmarks	162	219
Constrained groups	3	3

per hallway. Aside from several landmarks in the curved portion of the upper hallway, most landmarks are constrained.

Table 1 gives mapping statistics. Also included is the number of particles required to successfully build an unconstrained map, along with running times for comparison. (The complete results for unconstrained sparse sensing SLAM can be found in [3].) All tests were performed on a P4-1.7 GHz computer with 1 GB RAM. Incorporating constraints enables mapping with many fewer particles — about the same number as needed by many unconstrained SLAM algorithms that use full laser rangefinder information. This leads to significant computational performance increases when constraints are applicable.

One caveat is that the conditioning process is sensitive to the landmark cross-covariance estimates. (The cross-covariances are used in Eqns. 13-14 to compute a "gain" indicating how to change unconstrained variables when conditioning on constrained variables.) Because we use sensors that give very little data for feature extraction, the cross-covariance of $[r \;\; \theta]^T$ features is only approximately estimated. This leads to landmark drift in highly constrained environments since landmarks are frequently reconditioned, as can be seen in, e.g., the upper right corner of the NSH map in Fig. 6(d). Future research will examine alternative feature estimators and map representations (e.g., relative maps [10, 5]) that may alleviate this issue.

7 Conclusions

In this paper we have described a Rao-Blackwellized particle filter for SLAM that exploits prior knowledge of structural or geometrical relationships between landmarks. Relative constraints between landmarks in the map of each particle are automatically inferred based on the estimated landmark state. By partitioning the state into constrained and unconstrained variables, the constrained variables can be sampled by a particle filter. Conditioned on these samples, unconstrained variables are independent and can be estimated by EKFs on a per-particle basis.

We have implemented our approach with rectilinearity constraints and performed experiments on simulated and real-world data. For SLAM with sparse (low spatial resolution) sensing, incorporating constraints significantly reduced the number of particles required for map estimation.

Most of this work has focused on linear equality constraints. While we have described a way to extend the approach to inequality constraints, this remains an area for future work. Also, while constraints clearly help in mapping with limited sensing, they do not significantly improve data association inaccuracies related to sparse sensing, another potential avenue for improvement.

References

1. T. Bailey, J. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *IEEE Intl. Conf. on Robotics and Automation*, pages 424–427, 2006.
2. Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation*. Wiley, New York, 2001.
3. K. R. Beevers and W. H. Huang. SLAM with sparse sensing. In *IEEE Intl. Conf. on Robotics and Automation*, pages 2285–2290, 2006.
4. M. Csorba and H. Durrant-Whyte. New approach to map building using relative position estimates. *SPIE Navigation and Control Technologies for Unmanned Systems II*, 3087(1):115–125, 1997.
5. M. Deans and M. Hebert. Invariant filtering for simultaneous localization and mapping. In *IEEE Intl. Conf. on Robotics and Automation*, pages 1042–1047, 2000.
6. H. Durrant-Whyte. Uncertain geometry in robotics. *IEEE Journal of Robotics and Automation*, 4(1):23–31, 1988.
7. A. Howard and N. Roy. The Robotics Data Set Repository (Radish), 2003.
8. M. Montemerlo. *FastSLAM: a factored solution to the simultaneous localization and mapping problem with unknown data association*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, 2003.
9. K. Murphy. Bayesian map learning in dynamic environments. In *Advances in Neural Information Processing Systems*, volume 12, pages 1015–1021. MIT Press, 2000.
10. P. Newman. *On the structure and solution of the simultaneous localization and mapping problem*. PhD thesis, University of Sydney, Australia, 1999.
11. D. Rodriguez-Losada, F. Matia, A. Jimenez, and R. Galan. Consistency improvement for SLAM – EKF for indoor environments. In *IEEE Intl. Conf. on Robotics and Automation*, pages 418–423, 2006.
12. D. Simon and T. Chia. Kalman filtering with state equality constraints. *IEEE Transactions on Aerospace and Electronic Systems*, 39:128–136, 2002.
13. D. Simon and D. Simon. Aircraft turbofan engine health estimation using constrained Kalman filtering. In *ASME Turbo Expo*, 2003.
14. W. Wen and H. Durrant-Whyte. Model-based multi-sensor data fusion. In *IEEE Intl. Conf. on Robotics and Automation*, pages 1720–1726, 1992.