International Journal of Computer Vision 74(3), 303–318, 2007

DOI: 10.1007/s11263-006-0021-0

A Study of the Rao-Blackwellised Particle Filter for Efficient and Accurate Vision-Based SLAM

ROBERT SIM*, PANTELIS ELINAS AND JAMES J. LITTLE
Department of Computer Science, University of British Columbia, 2366 Main Mall, Vancouver, BC, V6T 1Z4
simra@cs.ubc.ca
elinas@cs.ubc.ca
little@cs.ubc.ca

Received November 2, 2005; Accepted December 6, 2006

First online version published in January, 2007

Abstract. With recent advances in real-time implementations of filters for solving the simultaneous localization and mapping (SLAM) problem in the range-sensing domain, attention has shifted to implementing SLAM solutions using vision-based sensing. This paper presents and analyses different models of the Rao-Blackwellised particle filter (RBPF) for vision-based SLAM within a comprehensive application architecture. The main contributions of our work are the introduction of a new robot motion model utilizing structure from motion (SFM) methods and a novel mixture proposal distribution that combines local and global pose estimation. In addition, we compare these under a wide variety of operating modalities, including monocular sensing and the standard odometry-based methods. We also present a detailed study of the RBPF for SLAM, addressing issues in achieving real-time, robust and numerically reliable filter behavior. Finally, we present experimental results illustrating the improved accuracy of our proposed models and the efficiency and scalability of our implementation.

Keywords: vision, slam, robotics, rao-blackwellised particle filters, mixture proposal, feature matching, localization

1. Introduction

Simultaneous localization and mapping (SLAM) is the problem of recursively estimating both the position of a moving sensor and the structure of the environment it senses. SLAM is a core problem in robotics, as it must be solved in order to accurately and effectively explore and navigate through an unknown environment, such that the robot can learn a representation of its surroundings. SLAM is especially difficult to solve because a robot's motions are often noisy and difficult to estimate accurately over time—the error in a position estimate can grow without bound. Likewise, many sensors, including cameras, are noisy and the problem of recovering structure from their output can be ill-posed.

Recent advances in state estimation methods have led to a plethora of approaches to solving the simultaneous localization and mapping (SLAM) problem (Barfoot, 2005; Davison, 2003; Eustice et al., 2005; Folkesson et al., 2005; Leonard and Durrant-Whyte, 1991; Murphy, 1999; Se et al., 2002). However, there are a limited number of *vision-based* solutions that can address real-time mapping, that can scale up to tens of thousands of mapped features, and that can operate robustly. SLAM is particularly challenging when using a vision sensor. The conventional noise models employed for structure recovery are highly non-linear, landmark-based methods are prone to outlier data-associations, and there are several steps of approximation in developing reliable observation models. Under the best of circumstances, the computed probabilistic models can be biased, overconfident, or subject to other numerical challenges. Furthermore, vision-based SLAM is not always observable without additional information, such as is the case for monocular SLAM without odometric or inertial measurements. In light of these challenges, there is a need for a general-purpose architecture for solving SLAM with vision that exploits novel techniques in state estimation for achieving accuracy and robustness

*Correspondence author.

while simultaneously providing maximum flexibility to the user. In this paper we present new methods for achieving robust SLAM solutions and a flexible architecture for SLAM deployment and evaluation.

We consider the family of SLAM solutions based on the Rao-Blackwellised particle filter (RBPF) (Murphy, 1999). RBPF-based SLAM solutions operate by maintaining multiple map hypotheses, each conditioned on a stochastically sampled trajectory through the environment. The complete set of sampled trajectories and inferred maps approximates the probability distribution of maps conditioned on the vehicle's actions and observations, $p(M \mid u^t, z^t)$, where $M = \{m_1, m_2, \ldots, m_n\}$ is the set of maps, each consisting of a set of probability distributions describing landmark positions, $u^t = \{u_1, u_2, \ldots, u_t\}$ are the control inputs to the vehicle (that is, the vehicle's actions), and $z^t = \{z_1, z_2, \ldots, z_t\}$ are the vehicle's observations of the world. RBPFs have been widely and successfully deployed using laser rangefinder sensing modalities (Eliazar and Parr, 2004; Hähnel et al., 2003; Montemerlo et al., 2002). Two core components of robust RBPF SLAM are accurate motion models and a reliable proposal distribution. Our architecture provides a visual ego-motion model that precludes odometric measurement, in addition to the conventional odometry-based motion estimation approach. Furthermore, we present a mixture proposal distribution that facilitates robustness by injecting samples derived from a global pose estimator (Thrun et al., 2000). This approach facilitates robust mapping of large cycles in the environment, even without odometric or inertial measurements.

This paper makes several important contributions. First, we present an application architecture for implementing vision-based SLAM solutions using the RBPF. The focus of our architecture is on robust, real-time mapping of large-scale visual environments (trajectories on the order of 100 m or more in length). As such, we consider the scalability of data structures for achieving real-time SLAM even as the number of landmarks in the map grows to number in the tens or hundreds of thousands, and the number of sampled trajectories numbers in the thousands. Second, as mentioned above, we introduce two new models for updating the RBPF; we derive a motion model using structure from motion methods and later we augment it with a novel mixture proposal distribution that allows for the closing of large loops. These models offer advantages over standard odometry-based models in that they afford a more general SLAM solution that is not dependent on odometric or inertial measurements, and furthermore, they achieve the goal of robust mapping even when the map includes long cycles. Third, and finally, we present extensive experimental results illustrating the functionality of the system, measuring the effectiveness of the new models, and, significantly, providing the first extensive empirical analysis of the data structures commonly used to achieve real-time RBPF-based SLAM. Such an analysis is important for understanding the scaling behavior of the filter under changes in the number of samples in the state estimate and the number of landmarks mapped.

Vision-based options

- Monocular versus stereo sensing
- Vision-based ego-motion estimation

Particle filtering options

- Standard proposal distribution based on robot odometry.
- Vision-based proposal distribution (odometry-free ego-motion)
- A mixture proposal distribution incorporating both ego-motion and a global pose estimator.

State and Hybrid map representations

- 6-DOF pose representation.
- Ability to construct occupancy grids in addition to landmark-based map.

Figure 1. Summary of supported operational modalities.

Our application architecture aims to provide a high degree of functionality and user flexibility. Figure 1 summarizes the main features of the framework. While the architecture can run in conjunction with an exploring robot, a valuable feature of the architecture is its ability to process data off-line, and therefore provide a test-bed for comparing and evaluating new proposal distributions and observation and motion models. Central to the architecture are data structures for facilitating real-time operation and maximizing efficiencies in the RBPF. In particular, we exploit data structures for rapid, robust data association (a key stumbling block for many landmark-based SLAM approaches), and a tree-based data structure, similar to that popularly known as FastSLAM (Montemerlo et al., 2002), for optimizing memory efficiency and facilitating the use of large numbers of particles in the RBPF.

In vision-based feature mapping, the number of landmarks in the map can grow to hundreds of thousands, as feature detectors typically return several hundred feature observations in a single image. Like conventional landmark-based SLAM solutions in the range-sensing domain, data association can be computationally expensive and subject to false positives. However, vision sensing provides the added advantage that local image information associated with a feature can constrain data association, and therefore reduce the rate of false

positives. In order to take advantage of this information, we employ the scale-invariant feature transform (SIFT) descriptor (Lowe, 1999), which has been shown to be very robust for feature correspondence. Matching SIFT features requires finding a nearest neighbour among many thousands of points in a very high dimensional space. To improve performance and facilitate real-time data association, we apply a kd-tree over the space of SIFT features. kd-trees facilitate approximate nearest-neighbor lookups in time logarithmic in the number of visually distinct landmarks (Beis and Lowe, 1997).

In addition to the presentation of new techniques for vision-based SLAM, this paper examines in depth the run-time behavior of efficiently designed RBPFs. Such an analysis is important for understanding the scaling behavior of the filter under changes in the number of samples in the state estimate and the number of landmarks mapped. Specifically, we examine the cost of the map representation for various sample set sizes, observation and resampling costs, and map accuracy under the various operating modalities. Previous work has suggested efficient data structures for map sharing among particles (Montemerlo et al., 2002). In this paper, we study the empirical efficiency of this data structure.

The remainder of this paper is outlined as follows. Section 2 describes related work on SLAM, Section 3 provides a Bayesian formulation of the SLAM problem and Section 4 provides the details of our approach to solving vision-based SLAM with RBPFs. The remaining sections present experimental results and discussion of the results.

2. Related Work

SLAM is a well studied problem because it is considered one of the fundamental problems in robotics. Many popular solutions are based on the Extended Kalman Filter (EKF) and most recently its inverse, the Extended Information Filter (EIF) (Eustice et al., 2005; Leonard and Durrant-Whyte, 1991; Smith et al., 1990; Thrun et al., 2004). These approaches model the posterior distribution over maps using a unimodal Gaussian distribution. The EKF is known to scale poorly with respect to the number of landmarks in the map and it is unable to handle bad data associations in landmark observations. Recent work has produced information filter approaches that operate in constant time, but in most cases depend on approximations (Paskin, 2003; Thrun et al., 2004) or linear-time iterative state recovery methods (Eustice et al., 2005). We use particle filters in an approach that can model multimodal distributions, whose update cost scales logarithmically with respect to the map size, and that can handle poor data association robustly.

The application of particle filters to SLAM was first studied by Murphy (1999) in the context of learning occupancy grid maps. He was the first to exploit the method of Rao-Blackwellisation to show that a sampling approach can be used to solve this challenging problem. The properties of the Rao-Blackwellised Particle Filter were later studied in detail by Doucet et al. (2000). However RPBF-based SLAM was popularized by Montemerlo et al. (2002), Montemerlo and Thrun (2003) and Montemerlo et al. (2003) that introduced the FastSLAM algorithm for learning landmark-based maps using a laser sensor. Their major contribution was a tree data structure for sharing landmark estimates among particles, significantly reducing the memory requirements of the algorithm and allowing for fast particle propagation. Similarly, the work of Eliazar and Parr (2004) introduced an efficient data structure for a variation of FastSLAM that learns high fidelity occupancy grid maps from laser measurements.

Recently focus is switching to vision as the sensing modality. Se et al. (2002) utilized a stereo camera to learn maps of 3D landmarks identified using SIFT. They use the method of Rao-Blackwellisation to reduce the size of the problem but employ a Kalman filter where Murphy used a particle filter. This is equivalent to block-diagonalizing an EKF (correlations between robot pose and landmarks, and between the landmarks themselves are ignored). They have demonstrated their approach successfully mapping a small laboratory environment. For larger environments, their method is likely to be inconsistent and lead to filter divergence- as it has been demonstrated that a diagonalized (decoupled) EKF for SLAM is overconfident (Dissanayake, 2001). Recently, Barfoot (2005) has extended this work to use a particle filter in place of the Kalman filter but he has only presented results using a single particle in which case it does not differ significantly from the original work of Se et al.

Recent work also addresses the SLAM problem using a single camera. Davison (2003) presents monocular SLAM using a Kalman filter. He uses as landmarks salient image features and he employs a Bayesian approach to estimating their 3D position. Notably, his algorithm runs in real-time but it has only been demonstrated to work well on small trajectories. The vSLAM (Karlsson et al., 2005) approach employs particle filters to localize in a hybrid metric-topological map. They identify landmarks using SIFT but can only localize accurately in locations where nodes of the topological map have been instantiated.

Our work is similar to Se et al. (2002) because we learn the same type of maps and identify landmarks using SIFT. It is also similar to Barfoot (2005) because we employ a particle filter. The chief differences between that work and ours are the real-time performance of our system for numbers of particles up to and above 3000,[1] and its

flexibility in the variety of operating modalities, such as operating without odometry when it is unavailable, or the ability to cope with monocular data. Furthermore, we can demonstrate robust loop-closing capabilities in large environments using a mixture proposal mechanism.

The problem that we are solving is very closely related to the structure from motion problem, wherein given a set of images taken of a scene from unknown positions with an arbitrary amount of overlap, the problem is to recover the position of the camera for each image, as well as the structure of the scene (or a subset of 3D point features extracted from the images) (Hartley and Zisserman, 2000). In fact, we use a SFM-based approach for visual odometry estimation, similar in nature to that developed in Zhang and Shan (2001). The primary difference between SFM and the SLAM problem is that SLAM involves recursive estimation of the camera position and scene structure (usually taking advantage of the spatio-temporal adjacency of images in the sequence), whereas SFM solutions are usually batch methods. Recursive estimation is especially important for a robot that must explore autonomously.

3. Simultaneous Localization and Mapping

This paper represents map estimation as the evolution of a Rao-Blackwellised particle filter (Murphy, 1999). In this context, the trajectory and landmark distribution is modeled as a dynamic Bayes network, where trajectories are instantiated as samples, and the landmark distribution can be expressed analytically for each trajectory. At time t, let s_t denote the vehicle pose, m the map learned thus far and $x_t = \{s_t, m\}$ be the complete *state*. Also, let u_t denote a control signal or a measurement of the vehicle's motion from time $t-1$ to time t and z_t be the current observation. The set of observations and controls from time 0 to t are denoted as z^t and u^t respectively. Normally, our goal is to estimate the density

$$p(s_t, m|z^t, u^t) = p(x_t|z^t, u^t). \tag{1}$$

By applying Bayes' rule and making the Markov assumption, the posterior can be estimated recursively using

$$\begin{aligned} p(x_t|z^t, u^t) &= Bel(x_t) \\ &= \eta p(z_t|x_t) \int p(x_t|u_t, x_{t-1}) p(x_{t-1}|z^{t-1}, u^{t-1}) dx_{t-1} \\ &= \eta p(z_t|x_t) \int p(x_t|u_t, x_{t-1}) Bel(x_{t-1}) dx_{t-1} \end{aligned} \tag{2}$$

where η is a normalizing constant. Equation (2) requires that we evaluate integrals for which there is no closed form solution and as such we cannot directly use it for SLAM. However, it has been shown previously that we can avoid integrating if we estimate the density over trajectories instead of just the current pose,

$$p(s^t, m|z^t, u^t) = p(x^t|z^t, u^t) \tag{3}$$

In addition, we can take advantage of the structure in the problem to factor the posterior into the product of simpler distributions (Murphy, 1999)

$$p(s^t, m|z^t, u^t) = p(s^t|z^t, u^t) \prod_k p(m(k)|s^t, z^t, u^t) \tag{4}$$

where $m(k)$ denotes the k-th landmark in the map. We can then use a particle filter to estimate the first factor and a Kalman Filter to estimate the position of each of the k landmarks in the map.

In order to implement the particle filter, we must specify a measurement model, $p(z_t|x_t)$, and a proposal distribution that is often derived from the motion model, $p(x_t|u_t, x_{t-1}, z_t)$. The choice of these distributions is the main difference among all SLAM methods using the particle filter. In the next section, we present a number of different models that we have studied in the context of vision-based SLAM.

4. Solving SLAM with RBPFs

In this section we will describe our system architecture, including map representation, observation models, and proposal distributions.

4.1. Map Representation

In this paper, each state sample $s_{i,t}$ has an associated map $m_i = m(1) \ldots m(K)$, where each $m(k)$ is a 3D Kalman Filter describing a landmark position estimate. A simplistic approach to implementing an RBPF for SLAM would be to store a separate map for each sample, yielding an update and storage complexity of $O(NK)$, where N is the number of samples at each step and K is the number of landmarks. However, Montemerlo et al. introduced in their FastSLAM work a tree-based structure which, on average, refines the update complexity to $O(N \log K)$ and reduces storage requirements by sharing landmark estimates between samples (Montemerlo et al., 2002). Each sample in the filter will share unaltered landmark estimates with other samples (those landmarks that have not been observed since the time the samples became siblings). Each landmark observation results in a landmark being copied and updated but the rest of the map remains unaltered.

We employ a data structure similar to that described in Montemerlo et al. (2002) as a map representation.

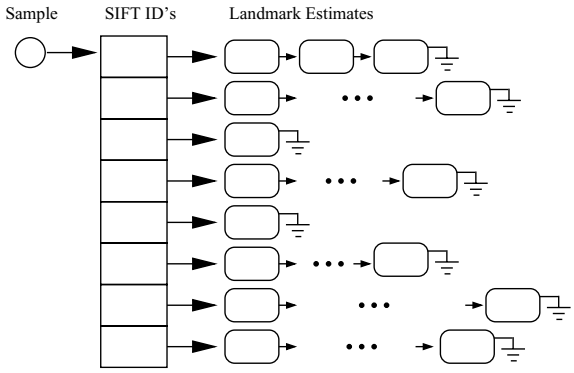

Figure 2. Conceptually, each sample has an associated map, indexed by SIFT descriptor. Each SIFT descriptor might have multiple landmark estimates, each spatially distinct. The physical implementation of this representation is in a reference-counted tree structure similar to that presented in Montemerlo et al. (2002).

Conceptually, each particle has an associated set of landmark estimates, each described by Gaussian distributions. However, unlike the anonymous features used in the previous work, vision provides additional information about the image features—we use SIFT descriptors for correspondence, enabling us to improve the quality of data association. In this formulation, each sample maintains a list of SIFT IDs, and these IDs in turn point to a linked list of one or more 3D landmark estimates (Fig. 2). Note that one SIFT ID can point to multiple landmarks—landmarks that have similar appearance but are spatially distinct. The implementation of the map structure is identical to that of the original reference-counted FastSLAM binary search tree, with the exception that indexing a landmark by SIFT ID returns a linked list of the landmarks associated with that ID. Individual landmark estimates are represented as 3D Gaussian distributions using the Extended Kalman Filter.

4.2. *Observation Model and Data Association*

Figure 3 and Algorithm 1 summarize the observation update process. We select points of interest using the difference of Gaussian detector described in Lowe (1999), and locate stereo matches in the left and right images by matching SIFT keys. Our framework allows for SIFT feature representations with either 36 or 128 dimensions (trading off memory consumption versus resolution in the matching domain). In the stereo case, we perform a straightforward linear search of the keys in the left image for the best match to each key in the right, subject to epipolar constraints, producing an observation:

$$\mathbf{z} = [u\ v\ d] \tag{5}$$

where $[u\ v]$ is the pixel position of the key in the right camera, and d is the observed disparity between the keys in the left and right camera. Observations are assumed to have a diagonal covariance C_z matrix with noise parameters σ_{uv}^2 for u and v, and σ_d^2 for the disparity. A typical value for σ_{uv} is 5 pixels, and 0.5 pixels for σ_d.

Algorithm 1 Observation update procedure

$F :=$ Extract features $f = \{k, p\}$ from image. $\{k =$ SIFT key, $p =$ feature position.$\}$
for all features f in F
  $id :=$ kd_tree_lookup$(f.k)$ {Index into kd-tree.}
  for all Samples s
    List $L := s$.map_lookup(id)
    Find most likely landmark estimate l in L, given $f.p$ {Maximizing observation likelihood.}
    Copy l if necessary {If shared with other samples.}
    Update l with $f.p$ using Kalman Filter update.
    Update w_t for s according to observation likelihood.
  end for
end for

From Eq. (5), the 3D position of the landmark can be computed according to the well-known stereo

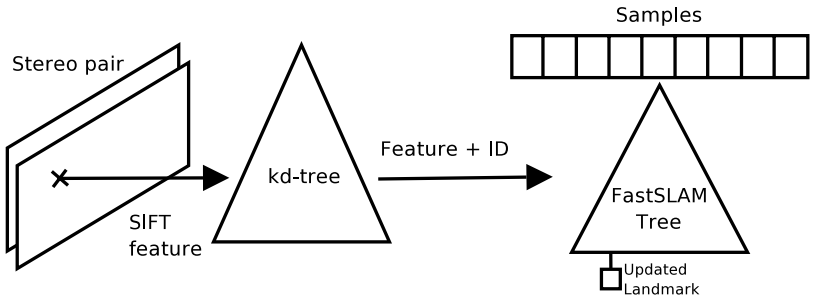

Figure 3. Observation update (refer to text for details).

equations:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} uB/d \\ vB/d \\ fB/d \end{bmatrix} \tag{6}$$

where f is the focal length of the camera and B is the base-line of the stereo head. The covariance of the landmark is then approximated as $\nabla h C_z \nabla h^T$, where ∇h is the Jacobian of Eq. (6) with respect to $\mathbf{z}$ (Faugeras, 1993).

Once landmark observations are extracted from the stereo pair,[2] the landmark estimates must be updated for the individual samples. To efficiently store and access what can quickly become a large number of SIFT keys we use a kd-tree. The kd-tree facilitates nearest-neighbor matching in time logarithmic in the size of the tree, and has been demonstrated to be reliable for object recognition tasks (Beis and Lowe, 1997). The disadvantage of using a kd-tree is that it can sometimes produce not the nearest match but a close match. We maintain a single tree for the sensor and associate an arbitrary integer ID with each SIFT identifier we add. New keys are considered to be *candidate keys* and are not passed as an observation to the particle filter until they have been observed for a sufficient number of frames.

Each particle's map is indexed by a set of IDs associated with SIFT descriptors and each node contains a linked list of 3D landmarks sharing that descriptor. Multiple data associations can be entertained by the filter because each particle determines the specific landmark to which an observation corresponds. A sample's weight is updated for a given landmark observation according to Eq. (10) by first selecting from the linked list for the matched landmark ID the landmark estimate that is most likely to have generated the observed point. If the observation deviance exceeds a particular threshold, or if no previous landmark estimates exist for a particular SIFT ID, a new landmark estimate is instantiated using the observation as the initial estimate.

4.3. Monocular SLAM

In addition to stereo sensing, our architecture supports mapping with a monocular camera. Several authors have demonstrated that in order to successfully map an environment with a bearings-only sensor, it is important to maintain several initial landmark hypotheses until a substantially wide baseline is achieved between observations (Davison, 2003; Kwok and Dissanayake, 2004; Solà et al., 2005). We take advantage of the framework's ability to assign multiple landmarks per SIFT ID in order to perform undelayed landmark initialization for accomplishing SLAM using a monocular camera (that is, performing bearings-only SLAM). We employ the initialization method proposed by Solà et al., which aims to accurately represent the full probability distribution of a landmark's position using a geometric progression of Gaussian distributions along the cone projected from the camera's focal point through the pixel location of the observation (Solà et al., 2005).

Specifically, when a landmark is initialized, its probability distribution is a cone with Gaussian cross-section, lying along the ray projected from the focal point of the camera through the pixel corresponding to the observation such that the intersection of the cone with the image plane represents the measurement covariance R. This distribution is approximated by defining a progression of weighted Gaussian distributions such that

$$p(l_t|z_t) = \sum_{1}^{N_g} \Lambda_j N(s_j, \sigma_j) \tag{7}$$

where, as depicted in Fig. 4, $s_j = \beta^{j-1} s_1$, $\sigma_j = \beta^{j-1}\sigma_1$ and N_g, β, s_1, and σ_1 are user-defined constants calibrated according to the scale of the environment being mapped, and Λ_j is the weight assigned to each Gaussian, as described in Solà et al. (2005).

Observation updates in the monocular case are performed by updating all of the individual landmark estimates and recording the observation likelihood for each

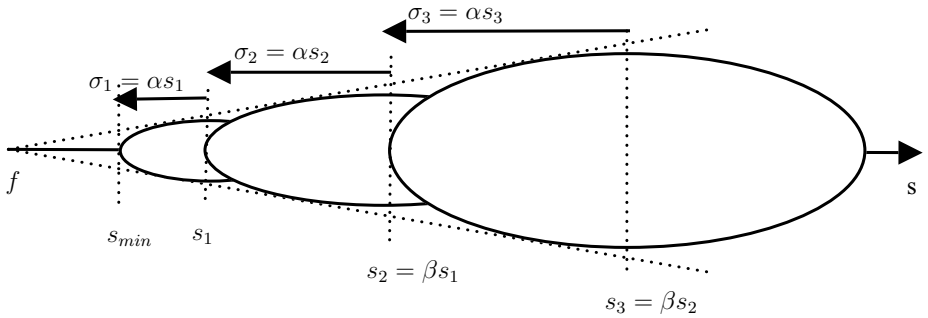

Figure 4. Monocular landmark initialization. The probability distribution for landmark position is described as a geometric progression of Gaussian distributions along the ray from the focal point f of the camera through the observed pixel position in the image plane. Refer to Solà et al. (2005) for further details.

in a vector Λ. The landmark whose observation likelihood is highest is used to weight the sample, and over time landmarks whose Λ_j drops below a threshold are pruned, until a single landmark estimate is determined. Note that for the monocular case, this approach precludes the estimation of multiple distinct landmark estimates (corresponding to distinctly different 3D points) per SIFT ID.

In this paper we will present results for the monocular case, however our current implementation requires that odometry be available to resolve the well-known scale ambiguity in monocular SLAM and structure from motion (SFM) problems.

5. State Representation and Proposal Distributions

We describe samples of the vehicle's pose with the set $s_t = \{T, R\}$, where $T = [x\ y\ z]$ is the robot's position and R is a 3×3 rotation matrix. We have implemented three different proposal mechanisms, two of which depend on the next state model that is either derived from the robot's mechanical odometry or from an estimate of the robot's motion using structure from motion methods. We also study a third mixture proposal that suggests hypotheses from either the motion model or a global localization estimate. We next describe these proposals and define how the particles' weights are computed in each case.

5.1. *Proposal Distributions Derived from a Motion Model*

At each time step, the N pose samples are propagated according to the proposal distribution $q(s_t|s_{t-1}, z_t, m)$. We will refer to the *standard proposal* as the distribution based on the robot's motion model:

$$q(s_t|s_{t-1}, z_t, m) = p(s_t|s_{t-1}, u_t).$$

This model is similar to those used in previous work. Other proposal distributions have been suggested, and our system employs a vision-based ego-motion estimator to produce a visual odometry model:

$$q(s_t|s_{t-1}, z_t, m) = p(s_t|s_{t-1}, z_t),$$

Note that the visual odometer is independent of odometry inputs, enabling SLAM in the absence of odometry or IMU measurements. We can do this using structure from motion techniques taking advantage of our stereo camera setup.

Let I_t and I_{t-1} represent the pairs of stereo images taken with the moving robot's camera at two consecutive time steps. For each pair of images we detect points of interest, compute SIFT descriptors for them and perform stereo matching resulting in 2 sets of landmarks L_{t-1} and L_t. We compute the camera motion using a non-linear optimization algorithm minimizing the re-projection error of the 3D coordinates of the landmarks. We employ the Levenberg-Marquardt (LM) non-linear optimization algorithm (Hartley and Zisserman, 2000; Arun et al., 1987).

In Elinas et al. (2006), we show how we can efficiently solve for the camera that relates the two coordinate systems and also obtain the covariance of the solution as the inverse of $J^T J$, where J is the Jacobian matrix. The solution and its covariance determine a Gaussian distribution that we can sample from and use as the proposal.

After taking an observation z_t of a landmark (described in the next section), each particle in the current generation of particles is weighted according to the probability of the current observation z_t, conditioned on that particle's map:

$$w_{i,t} = \frac{p(z_t|s_{i,t}, m_i)p(s_{i,t}|s_{i,t-1}, u_t)}{q(s_{i,t}|s_{i,t-1}, u_t, z_t, m_i)}\, w_{i,t-1} \quad (8)$$

$$= p(z_t|s_{i,t}, m_i)\, w_{i,t-1} \quad (9)$$

$$= k \exp(-0.5\Delta z^T \Sigma^{-1} \Delta z)\, w_{i,t-1} \quad (10)$$

where for this derivation we assume the standard proposal, $\Delta z = h(s_{i,t}) - z_t$, $h(\cdot)$ is a generative model of the observation as a function of pose, and Σ is the sum of the measurement covariance and prediction covariance. The derivation from reasoning over all time steps to considering only the current time step is based on the Markov properties of the system and is presented by Montemerlo et al. (2002). Each particle is weighted according to how well the current observation agrees with the map constructed from that particle's trajectory. Because we have noisy data association, we must make sure that the outliers do not bias the weight computation. We describe in Section 5.3 how we deal with this problem.

5.2. *Proposal Distribution Derived from a Mixture Model*

One of the drawbacks of using a local motion model as the proposal distribution is that it does not take into account that the observation might be better explained at a more distant location (an undesirable consequence of filter over-confidence and particle depletion). So there is no guarantee that the proposed hypotheses will be in the correct space and filter divergence might result. In order to eliminate this problem we have also developed a model that samples from a mixture proposal distribution. This approach has been studied before both

in Monte Carlo Localization (Thrun et al., 2000) and vision-based SLAM (Karlsson et al., 2005). We differ in our choice of mixture components and their computation that is driven by our selection of sensor and map representation. Specifically, new hypotheses are proposed from either the motion model or a global pose estimator:

$$q(s_t|s_{t-1}, z_t, m) = \phi p(s_t|s_{t-1}, z_t) + (1-\phi)q_{global}(s_t|m)$$

where ϕ is known as the mixing ratio. In the following, we describe the computation of q_{global}.

Let z_t be the current observation and m_i be the map of the ith particle learned thus far. Given the correspondences in z_t, we can compute the transformation that relates the two coordinate frames using weighted-least squares. The procedure is similar to that described earlier except that in this case instead of minimizing the re-projection error of the 3D points, we minimize their Euclidean distance in 3D. This approach is sensitive to outliers that are present in our data association. To overcome this problem, we employ a RANSAC-based approach (Bolles and Fischler, 1981) where we select subsets of the point correspondences and compute a candidate robot pose for each. We have found that generating as few as 200 candidate poses, $\tilde{s}_{t_{1\ldots200}}$, is sufficient for good localization given our noisy observations. For computational efficiency, we only compute the candidate poses with respect to the map of the most likely particle at time $t-1$.

In order to sample from this distribution, we evaluate, using Eq. (13), the probability of our latest observation given each candidate pose. We then fit a Gaussian distribution to these data points such that:

$$q_{global} = N_{global}(\mu, \Sigma)_{\tilde{s}_{t_{1\ldots200}}} \tag{11}$$

In order to compute the weights for the ith particle correctly, we must evaluate the ratio of the target and proposal distributions,

$$w_{i,t} = \frac{p(z_t|s_{i,t}, m_i)p(s_{i,t}|s_{i,t-1}, u_t)}{(1-\phi)q_{global}(s_{i,t}|z_t, m_i) + \phi p(s_{i,t}|s_{i,t-1}, u_t)} w_{i,t-1} \tag{12}$$

Each of the distributions involved is a Gaussian that we have already described how to compute. One should notice that the weights for the particles are equal to the observation likelihood scaled by the ratio of the probability of the pose under the motion model and the weighted sum of the probability under the motion model and the *global* distribution. That is, those particles that are supported by both models are given weights that are mostly proportional to the observation probability while those that disagree are penalized.

Finally, our empirical results show that using a constant mixing ratio tends to generate noisy robot trajectories. This is the result of the bad and ambiguous data associations in our observations. Also, the observations are often dominated by landmarks that were seen most recently biasing the *global* distribution towards the most recently added landmarks. This potentially prohibits the closure of large loops. To correct this, we only generate and sample from q_{global} as a function of the ratio of old landmarks to total landmarks observed at time t. If this ratio is larger than 30% then we set $\phi = 0.5$ and sample evenly from both mixture components, otherwise we set $\phi = 0$ and only sample from $p(s_t|s_{t-1}, z_t)$. An old landmark in our case is defined as one that has not been observed for longer than 2 minutes. Using this approach, we rely on the standard proposal in the short term and the mixture proposal for closing loops.

5.3. *Weight Normalization and Adaptive Resampling*

Special consideration must be taken when computing the particle weight, particularly where large numbers of feature observations, with significant potential for outlier correspondences, are present. We consider the weight computation for a standard proposal distribution here, with similar implications for the other proposal distributions. Weights are computed by accumulating the log likelihood of observations over time, and employing a normalization technique described below to prevent catastrophic numerical outcomes.

Following from Eq. (12),

$$\log w_{i,t} = \log p(z_t|s_{i,t}, m_i) + \log w_{i,t-1} \tag{13}$$

$$= -0.5 \min(T_l, \Delta z^T \Sigma^{-1} \Delta z) + \log w_{i,t-1} \tag{14}$$

where the maximum observation deviance T_l is selected so as to prevent outlier observations from significantly affecting the observation likelihood. However, given the potentially large numbers of observations, even with a reasonable setting for T_l, the magnitude of the log-likelihood can be such that raising it to the exponential to evaluate the ith particle weight results in machine-precision zero—we employ a normalization procedure described in Sim et al. (2005a) that preserves accuracy while preventing catastrophic numerical results. We note the following simplification. Let $H_i = \log p(z_t|s_{i,t}, m_i)$. Without loss of generality, assume that sample index $i = 0$ corresponds to the particle that minimizes $|H_i|$. Then for all particles:

$$H_i = (H_0 + H_i'). \tag{15}$$

where $H_i' = H_i - H_0$. Substituting into Eq. (13) and subsequently normalizing the weights yields:

$$w_i = \frac{\exp(H_0 + H_i')}{\sum_{j=1}^{N} \exp(H_0 + H_i')} \tag{16}$$

$$= \frac{\exp(H_0)\exp(H_i')}{\exp(H_0)\sum_{j=1}^{N} \exp(H_i')} \tag{17}$$

$$= \frac{\exp(H_i')}{\sum_{j=1}^{N} \exp(H_i')} \tag{18}$$

Note that for sample $\{s_{0,t}, m_0\}$, $H_i' = 0$, so by subtracting H_0 we guarantee that at least one sample weight has a numerator of 1 in normalizing Eq. (8), and the denominator is at least 1.0. This approach effectively eliminates the probability mass associated with outliers that is common to all particles. It is also important to note that using this approach assures that all particles have comparable weights—every particle has the same number of input observations, and outliers are represented in the model on a per-particle basis. Hence, a particle with more outlier matches will have a lower weight than a particle with better data association.

Once particle weights are calculated, the filter determines whether it is appropriate to resample. Resampling too often can result in sample starvation in the filter, resulting in poor modeling of the posterior distribution over states. We use the technique suggested in Stachniss et al. (2005) to evaluate the effective sample size N_{eff}:

$$N_{eff} = \frac{1}{\sum_i w_i^2}$$

The general heuristic is to resample when N_{eff} falls less than $N/2$, where N is the number of samples in the filter. Resampling involves sampling probabilistically with replacement to produce the next generation of particles. When any particle is not chosen for advancement it is pruned, and all nodes in the landmark tree to which it refers have their reference counts decremented, and are deleted if their reference counts reach zero. After resampling, the particle weights are reset to $1/N$.

6. Implementation

Our system architecture is described in Fig. 5. We use a multi-threaded design, enabling user and robot responsiveness while the filter is computing observation updates. Our current robot interface connects to RWI B-14 and Activmedia Powerbot bases. We typically run the system on a desktop system which communicates with the robot and its camera over a high-speed wireless connection.

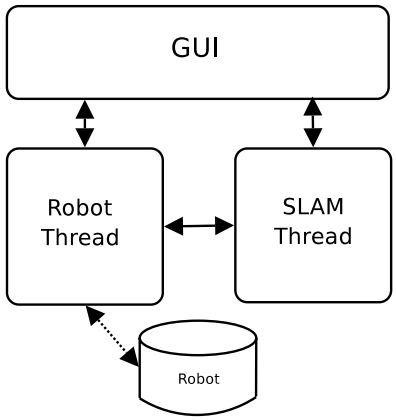

Figure 5. System architecture overview.

7. Experimental Results

For the purposes of our experiments, we used an RWI B14 robot with a BumbleBee stereo head from Point Grey Research. The robot was driven by a human operator through a laboratory environment consisting of two rooms of total size approximately 19 m by 16.5 m, and the robot collected 8500 stereo images along a trajectory of approximately 120 m. The entire sequence and odometry log were saved for testing under the various operating modes of our system. Using the same procedure, we also collected a second data set with 3500 images, as the robot traversed a corridor environment. All of the filter tests were conducted on a Pentium Xeon 3.2 GHz PC with 4 GB of RAM. We begin by reporting on the accuracy of the different models and then proceed to discuss our study of the scalability of the RBPF with respect to the number of particles and map size.

During typical run-time behaviour of the filter, each 320×240 image yields approximately 400 SIFT keys, of which roughly 75 yield good stereo matches, and subsequently typically 12–25 are successfully matched against the SIFT keys in the map. This drastic reduction in the number of keys helps prevent outliers (since we impose tight thresholds for matching at each step), boosting the confidence and performance of the filter.

7.1. Accuracy

The main goal of any SLAM architecture is robust, accurate mapping. However, without an accurate map in the first instance, it is difficult to reliably evaluate the performance of a SLAM algorithm. Our approach to map evaluation was to record the filter's estimate of the robot's position as it visited a set of pre-determined waypoints. Map accuracy was then defined in terms of the accuracy of the robot's trajectory, as evaluated by

Figure 6. Sample frames from our test sequences. The top row shows example frames from the data used for constructing the map in Fig. 8 and the bottom rows shows example frames from the data used for constructing the map in Fig. 10.

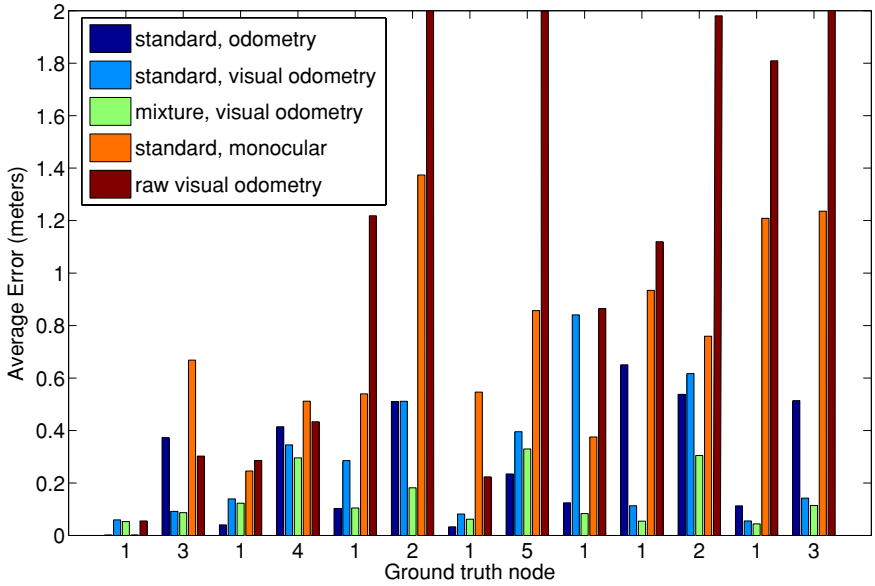

Figure 7. The localization error for the filter using just the standard proposal, mixture proposal and the raw visual odometry estimates. Note that the robot starts at the left-most node in the plot and finishes at the right-most node. The legend is described in Table 1.

its ability to localize at the waypoints. We defined five waypoints located in the corners and center of the main room that the robot explored and measured their positions. Figure 9 illustrates the set of waypoints and their positions superimposed on a learned occupancy grid map of the same area. The waypoints were visited in the sequence 1, 3, 1, 4, 1, 2, 1, 5, 1, 1, 2, 1, 3 where the robot traversed a large loop through an adjacent room in the 1,1 phase of the sequence. In addition, we run an experiment using data from a corridor environment closing a large loop in the order of 100 meters. However, we do not have ground truth information for these data. Figure 6 shows a few examples of the frames in the two data sets.

For the following set of experiments, we ran the filter with 500 samples, and varied the operating modality of the filter. Figure 7 shows the error at each ground truth node for all the different models. As expected the filter outperformed the raw odometers in all cases. In addition, using the visual odometry performed better than using mechanical odometry. The filter that implements a mixture proposal performed better overall. Figure 8 shows an example of the 3D landmark map and the occupancy grid learned for the best particle at the end of the trajectory. The map shown was generated using the mixture proposal with visual odometry and variable mixing ratio. The map consists of about 30,000 landmarks. Table 1 summarizes

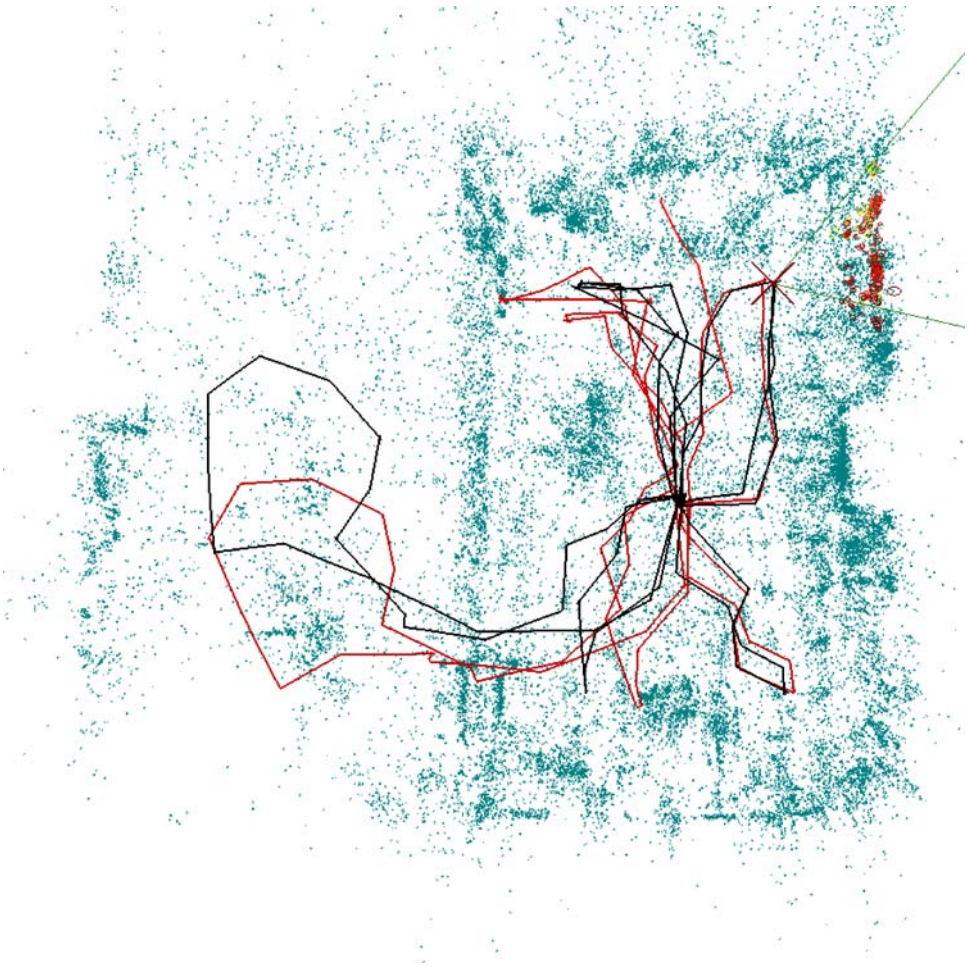

Figure 8. Example of the map learned with a mixture proposal and visual odometry for two adjacent rooms size 19 × 16.5 meters. The figure depicts a top-down view of the map of 3D landmarks used for localization. Shown in black is the filter trajectory and in red the raw visual odometry trajectory. The robot's final position is marked with a red 'X', and the set of landmarks observed in the final few frames is highlighted with red and yellow ellipses.

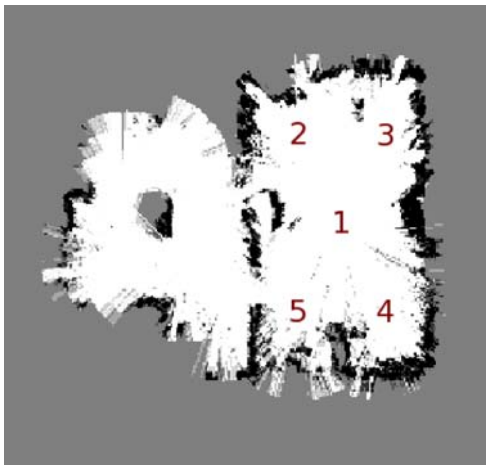

Figure 9. The occupancy grid constructed using the maximum likelihood trajectory for camera location in Fig. 8. In the grid, white pixels represent empty space while black pixels represent occupied space. Gray pixels denote unobserved areas. The grid's resolution is 15 × 15 cm.

the configuration of the filter for each run, along with the mean error in the trajectory estimate over all ground truth nodes. Figure 10 shows another example of a learned 3D map for a corridor environment and Fig. 11 shows its corresponding occupancy grid. It should be noted that since the filter does not perform backwards correction, the corridors are not perfectly aligned. Further post-processing improvements can be realized by using the filter-based solution to initialize a full bundle-adjustment.

It is important to note that even the worst performing approach (SLAM with a monocular camera) outperformed the robot's dead reckoning. The erroneous monocular estimates are largely due to the robot's failure to re-localize when it closed the large loop through the second room. We hope that advanced mixture proposal techniques, such an extension to the mixture proposal that we have presented for stereo vision, will lead to improved performance with the monocular modality. We also note that the accuracy of the visual odometry approach facilitates the use of non-robotic platforms, such as hand-held

Table 1. Summary of operating modes and experimental results for evaluating map accuracy.

Run title	Mechanical odometry	Visual odometry	Stereo/Monocular	Mean pose error (m)
Dead Reckoning	no	yes	stereo	1.6
Standard (odometry)	yes	no	stereo	0.28
Standard (visual odometry)	no	yes	stereo	0.28
Mixture (visual odometry)	no	yes	stereo	0.14
Monocular	yes	no	monocular	0.71

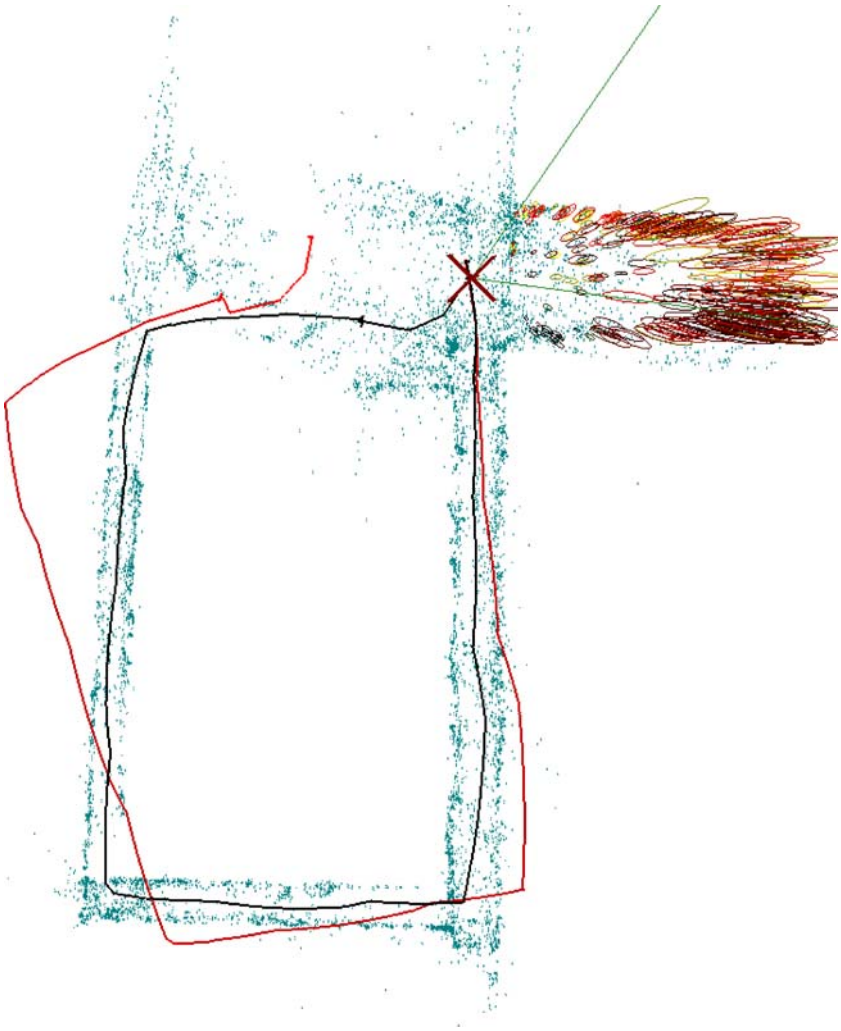

Figure 10. Example of the map learned with a mixture proposal and visual odometry for a corridor environment. The figure depicts a top-down view of the map of 3D landmarks used for localization. Shown in black is the filter trajectory and in red the raw visual odometry trajectory.

cameras, and other devices where odometry may not be available.

7.2. RBPF Run-Time Behavior

In this section we examine the empirical behavior of our RBPF implementation. For this section we consider only the standard proposal distribution and measure run-time and memory performance relative to variations in the sample set size, N. For these experiments, we employed a data set consisting of 8500 images along an exploratory trajectory similar to that presented above, and in the same laboratory environment. Figure 12 presents the map constructed by our system using 100 samples, with an architectural drawing overlaid to illustrate the features of the environment.

Figure 13 illustrates the run-time performance of the filter as we varied the number of samples in the filter, from

Figure 11. The occupancy grid constructed using the camera location for the maximum likelihood trajectory in Fig. 10. In the grid, white pixels represent empty space while black pixels represent occupied space. Gray pixels denote unobserved areas. The grid's resolution is 15×15 cm.

values of 1 up to 16000, roughly doubling the number of samples at each increment. The performance is measured in terms of the mean processing time per frame, in milliseconds, over the run of 8500 frames. In all cases the number of landmarks represented *per map* at the end of each of these runs was 12,316. Note that the plot is in log-log format.

It is worthwhile noting that the filter can run at 1 Hz at approximately 3000 samples, enabling real-time performance for large sample sets. It is also interesting to note that the slope of the log-log plot is less than 1 up to approximately 2000 samples, but begins to demonstrate superlinear behavior beyond this value. Note that the worst case performance of the RBPF should be $O(NK)$. The main reasons for the superlinear performance appears to be an increased cache and page fault frequency as N grows, as well as non-constant time performance of the standard C++ memory allocators. We have mitigated this latter issue somewhat by implementing custom allocators that improve performance for sample allocation.

We note at this point that for reliable performance the filter typically requires at least 400 samples (which can run at about 10 Hz), and will likely require more samples for very large environments.

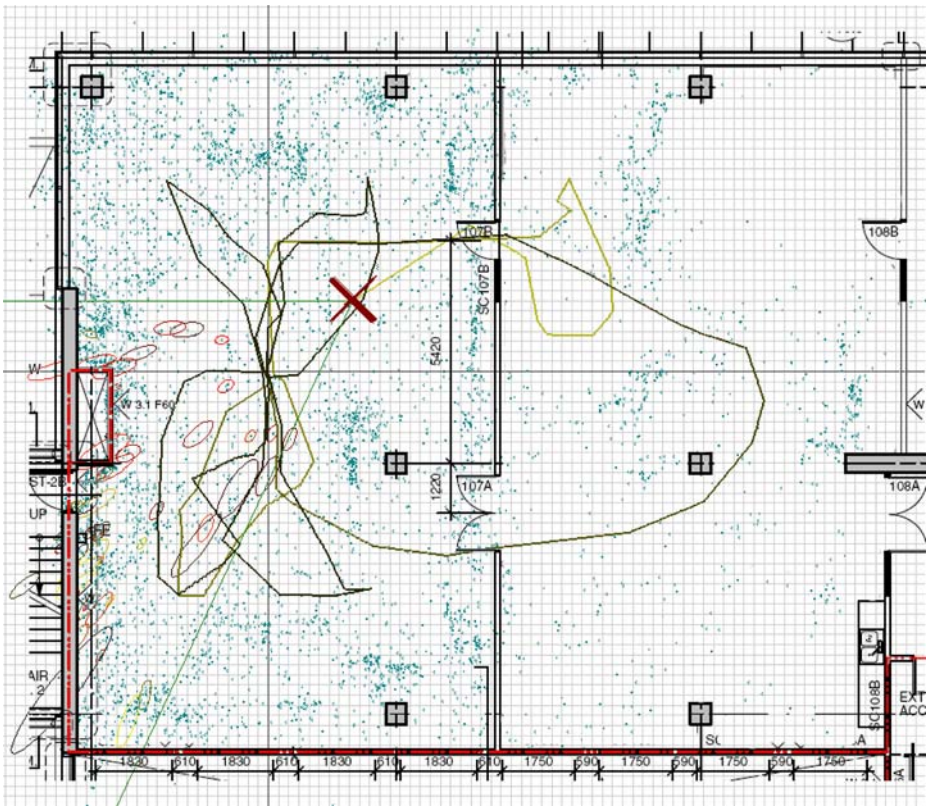

Figure 12. Map constructed using standard proposal distribution, with overlaid architectural drawing. Note that apparently spurious landmarks correspond to furniture or points on the ceiling or floor. The grid marks 25 cm intervals. The trajectory of the filter is plotted, with the set of samples marked by red 'x's (in this case, nearly all are coincident). Note that the robot successfully closes a large loop (going out one door and returning through another), although the precise location of one door is off by about 0.5 m, and also successfully locates the top door on three occasions.

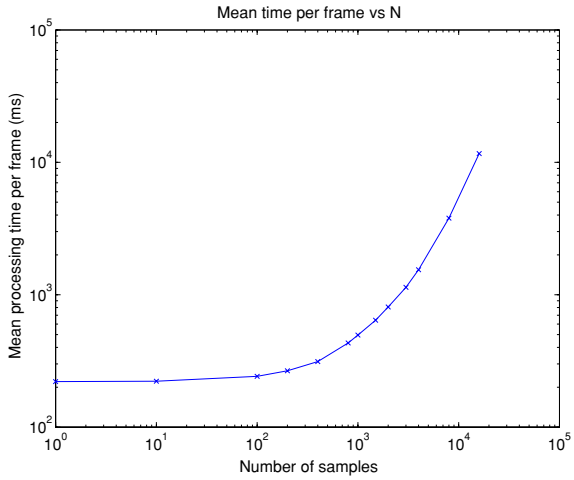

Figure 13. Mean processing time per frame versus number of samples.

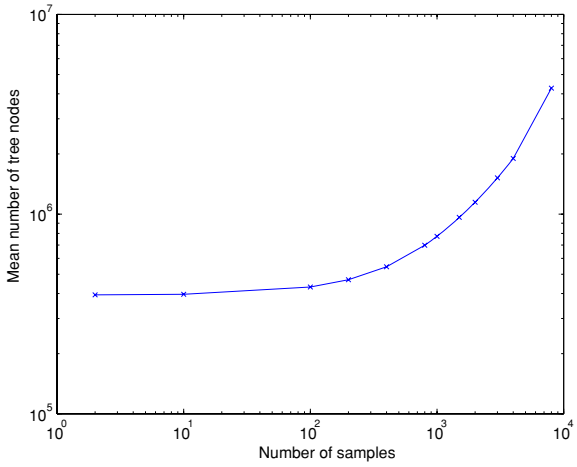

Figure 14. Mean nodes in the FastSLAM tree versus number of samples.

Figure 14 illustrates the memory consumption of the filter for varying N, represented in terms of the total number of nodes in the FastSLAM tree, averaged over all time instances. As N increases, the number of nodes required for map maintenance increases, and while the mean number of nodes is much smaller than the theoretical maximum of $O(NK)$, the slope of the plot at 8000 samples is 1.17, suggesting that the consumption for larger numbers of samples may not scale well.

The main concern for many SLAM implementations is the performance as the number of landmarks increases. Figure 15 shows the average processing time per frame over 400 frame intervals for a selection of sample set sizes N. The rate of growth in the number of landmarks is approximately constant over the entire run (at most 5 new landmarks per frame). This plot demonstrates that while there is a modest increase in the cost of maintaining the map over time, the filter performance does not degrade significantly.

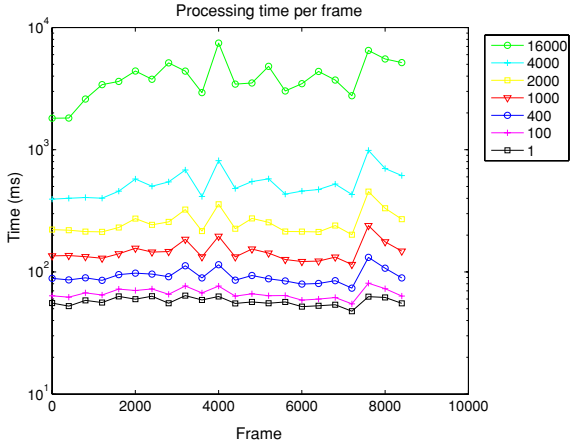

Figure 15. Processing time versus time instance for varying values of sample set size N.

From these results we can conclude that while the FastSLAM landmark tree scales well for large numbers of landmarks (for the environments considered), it is not clear that the filter will scale well for very large numbers of samples. This may explain in part the difficulty other researchers have had in implementing vision-based RBPFs with more than a small number of samples.

8. Conclusion

This paper has presented the design and analysis of an application architecture for conducting experiments in vision-based SLAM using Rao-Blackwellised particle filters. The architecture can successfully manage large maps and large sample sets in real-time. Among the more notable contributions are the development of a motion model that is independent of odometric measurements, and a mixture proposal distribution that facilitates large loop closure. The experimental results presented illustrate the capability of the system for operating in real-time with significant numbers of samples and over large environments. We also demonstrate successful mapping for a variety of modalities, including vision-based ego-motion estimation and monocular sensing. This work represents a significant step beyond the state of the art in terms of both its flexibility and the scale and density of the environments we are successfully mapping with vision.

Our system retains a few limitations that are characteristic of many vision-based SLAM solutions. Monocular SLAM, particularly without odometry, remains challenging due to the inherent scale ambiguity and approximations in landmark initialization. Our approach also requires a well-calibrated camera. We have observed the importance of the camera's exploration strategy (as controlled by a human operator) in constructing accurate maps. Furthermore, in recent work Bailey et al.

demonstrate that RBPF-based SLAM approaches are overconfident over long trajectories (Bailey et al., 2006). In these aspects, our future work will involve examining how intelligent exploration can improve map quality, and how we can further increase the scale of the environments that can be successfully mapped.

Notes

1. We consider real-time operation to be an update frequency greater than 1 Hz.
2. In the monocular case, the complete list of SIFT keys is returned along with their pixel positions—no depth computation is performed.

References

1. Bailey, T., Nieto, J., and Nebot, E. 2006. Consistency of the FastSLAM algorithm. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, IEEE Press: Orlando, FL, pp. 424–429.
2. Barfoot, T.D. 2005. Online visual motion estimation using FastSLAM with SIFT features. In *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*. IEEE/RSJ, IEEE Press: Edmonton, AB, pp. 3076–3082.
3. Beis, J.S. and Lowe, D.G. 1997. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, IEEE Press: Peurto Rico, pp. 1000–1006.
4. Bolles, R.C. and Fischler, M.A. 1981. A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *Proceedings of the Seventh Int. Joint Conf. on Artificial Intelligence*. Vancouver, BC, Canada, pp. 637–643.
5. Davison, A. 2003. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the IEEE Int. Conf. on Computer Vision*. Nice, France, pp. 1403–1410.
6. Dissanayake, G., Newman, P., Clark, S., Durrant-Whyte, H., and Csorba, M. 2001. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241.
7. Doucet, A., de Freitas, N., and Gordon, N. 2001. *Sequential Monte Carlo in Practice*. Springer-Verlag.
8. Doucet, A., de Freitas, N., Murphy, K., and Russell, S. 2000. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Uncertainty in Artificial Intelligence*, pp. 176–183.
9. Eliazar, A.I. and Parr, R. 2004. DP-slam 2.0. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*. IEEE Press: New Orleans, LA, vol. 2, pp. 1314–1320.
10. Elinas, P., Sim, R., and Little, J.J. 2006. σSLAM: Stereo vision SLAM using the Rao-Blackwellised Particle Filter and a novel mixture proposal distribution. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation*. Orlando, FL, USA, pp. 1564–1570.
11. Elinas, P. and Little, J.J. 2005. σMCL: Monte-Carlo localization for mobile robots with stereo vision. In *Proceedings of Robotics: Science and Systems*. Cambridge, MA, USA, pp. 373–380.
12. Eustice, R., Singh, H., and Leonard, J. 2005. Exactly sparse delayed-state filters. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. Barcelona, Spain, pp. 2428–2435.
13. Eustice, R., Singh, H., Leonard, J., Walter, M., and Ballard, R. 2005. Visually navigating the RMS Titanic with SLAM information filters. In *Proceedings of Robotics Science and Systems*. Cambridge, MA.
14. Faugeras, O.D. 1993. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press.
15. Folkesson, J., Jensfelt, P., and Christensen, H.I. 2005. Graphical SLAM using vision and the measurement subspace. In *Int. Conf. on Intelligent Robotics and Systems (IROS)*. IEEE/JRS, Edmonton, Canada, pp. 325–330.
16. Hähnel, D., Fox, D., Burgard, W., and Thrun, S. 2003. A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proc. of the Conference on Intelligent Robots and Systems (IROS)*, pp. 206–211.
17. Hartley, R., and Zisserman, A. 2000. *Multiple View Geometry in Computer Vision*. Cambridge University Pr.: Cambridge, UK.
18. Karlsson, N., Di Bernardo, E., Ostrowski, J., Goncalves, L., Pirjanian, P., and Munich, M.E. 2005. The vSLAM algorithm for robust localization and mapping. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*. Barcelona, Spain, pp. 24–29.
19. Kwok, N.M. and Dissanayake, G. 2004. An efficient multiple hypothesis filter for bearing-only SLAM. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems*. IEEE Press: Sendai, Japan, pp. 736–741.
20. Leonard, J.J., and Durrant-Whyte, H.F. 1991. Simultaneous map building and localization for an autonomous mobile robot. In *Proceedings of the IEEE Int. Workshop on Intelligent Robots and Systems*. Osaka, Japan, pp. 1442–1447.
21. Lowe, D.G. 1999. Object recognition from local scale-invariant features. In *Proceedings of the Int. Conf. on Computer Vision*. IEEE Press: Corfu, Greece, pp. 1150–1157.
22. Lowe, D.G. 1991. Fitting parameterized three-dimensional models to images. *IEEE Trans. Pattern Analysis Mach. Intell. (PAMI)*, 13(5):441–450.
23. Montemerlo, M. and Thrun, S. 2003. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proceedings of the 2003 International Conference on Robotics and Automation*, pp. 1985–1991.
24. Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. 2002. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conf. on Artificial Intelligence*. AAAI, Edmonton, pp. 593–598.
25. Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. 2003. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the Eighteenth Int. Joint Conf. on Artificial Intelligence (IJCAI-03)*. Morgan Kaufmann Publishers: San Francisco, CA, pp. 1151–1156.
26. Murphy, K. 1999. Bayesian map learning in dynamic environments. In *1999 Neural Information Processing Systems (NIPS)*, pp. 1015–1021.
27. Paskin, M.A. 2003. Thin junction tree filters for simultaneous localization and mapping. In *Proceedings of the Eighteenth Int. Joint Conf. on Artificial Intelligence (IJCAI-03)*. Morgan Kaufmann Publishers: San Francisco, CA, pp. 1157–1164.
28. Se, S., Lowe, D.G., and Little, J.J. 2002. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *Int. J. Robotics Research*, 21(8):735–758.
29. Sim, R., Elinas, P., Griffin, M., and Little, J.J. 2005. Vision-based SLAM using the Rao-Blackwellised particle filter. In *Proceedings of Reasoning with Uncertainty in Robotics*. Edinburgh, Scotland, pp. 9–16.
30. Sim, R., Griffin, M., Shyr, A., and Little, J.J. 2005. Scalable real-time vision-based SLAM for planetary rovers. In *IEEE IROS Workshop on Robot Vision for Space Applications*. IEEE, IEEE Press: Edmonton, AB, pp. 16–21.
31. Smith, R., Self, M., and Cheeseman, P. 1990. Estimating uncertain spatial relationships in robotics. In *Autonomous Robot Vehicles*, I.J. Cox and G.T. Wilfong (Eds.). Springer-Verlag, pp. 167–193.

32. Arun, K.S., Huang, T.S., and Blostein, S.D. 1987. Least-squares fitting of two 3-d point sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(5):698–700.
33. Solà, J., Monin, A., Devy, M., and Lemaire, T. 2005. Undelayed initialization in bearing only SLAM. In *Proc. Int. Conf on Intelligent Robots and Systems (IROS)*. IEEE/RSJ, IEEE Press: Edmonton, AB, pp. 2751–2756.
34. Stachniss, C., Grisetti, G., and Burgard, W. 2005. Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*. Barcelona, Spain, pp. 667–672.
35. Thrun, S., Liu, Y., Koller, D., Ng, A.Y., Ghahramani, Z., and Durrant-Whyte, H. 2004. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 23(7–8):693–716.
36. Thrun, S., Fox, D., and Burgard, W. 2000. Monte Carlo localization with mixture proposal distribution. In *Proceedings of the 2000 National Conference of the American Association for Artificial Intelligence (AAAI)*, pp. 859–865.
37. Zhang, Z., and Shan, Y. 2001. *Incremental motion estimation through local bundle adjustment.* Microsoft Research Technical Report MSR-TR-01-5.