HybridSLAM: Combining FastSLAM and EKF-SLAM for reliable mapping

Alex Brooks[1] and Tim Bailey[1]

Australian Centre for Field Robotics, University of Sydney
{a.brooks,t.bailey}@acfr.usyd.edu.au

Abstract: This paper presents HybridSLAM: an approach to SLAM which combines the strengths and avoids the weaknesses of two popular mapping strategies: FastSLAM and EKF-SLAM. FastSLAM is used as a front-end, producing local maps which are periodically fused into an EKF-SLAM back-end. The use of FastSLAM locally avoids linearisation of the vehicle model and provides a high level of robustness to clutter and ambiguous data association. The use of EKF-SLAM globally allows uncertainty to be remembered over long vehicle trajectories, avoiding FastSLAM's tendency to become over-confident. Extensive trials in randomly-generated simulated environments show that HybridSLAM significantly out-performs either pure approach. The advantages of HybridSLAM are most pronounced in cluttered environments where either pure approach encounters serious difficulty. In addition, the HybridSLAM algorithm is experimentally validated in a real urban environment.

1 Introduction

The Simultaneous Localisation and Mapping (SLAM) problem has attracted immense attention in the mobile robotics literature. The problem involves building a map while computing a vehicle's trajectory through that map, based on noisy measurements of the environment and the vehicle's odometry.

This paper specifically addresses the problem of feature-based SLAM, where the environment is modelled as a discrete set of features, each described by a number of continuous state variables. The standard solution is to take a Bayesian approach, explicitly modelling the joint probability distribution over possible vehicle trajectories and maps. There are currently two popular approaches to modelling this distribution. The first is to linearise and represent the joint probability with a single high-dimensional Gaussian. This is the approach taken by EKF-SLAM [8] and its variants [2]. The second is to use a Rao-Blackwellised particle filter, representing the vehicle's trajectory using a set of particles and conditioning the map on the vehicle's trajectory. Examples of this approach include FastSLAM [16] and others (*e.g.* [9]).

Both approaches have their strengths and weaknesses, and it is not difficult to produce examples where one or the other will fail. In particular, the EKF is prone to failure where significant vehicle uncertainty induces linearisation errors [3], or where significant clutter induces ambiguity in data association. The latter issue is problematic because the standard EKF formulation requires that hard data association decisions be made, by selecting the most likely hypothesis. Once a bad association is made it cannot be un-done and can cause the filter to fail catastrophically [2].

The probability of making a potentially-catastrophic incorrect decision can be reduced by using batch association methods. Nearest-neighbour association considers each observation in isolation, under the assumption that landmarks are independent. Joint compatability (*e.g.* JCBB [11] improves robustness by simultaneously considering all observations made from a single pose, but cannot consider relations between observations made from different poses. Sliding-window methods simultaneously consider all observations made from poses in a small window of recent history, but can be computationally intractible without the assumption of landmark independence for multiple observations from a single pose [12][6].

In contrast to the EKF, FastSLAM does not suffer from linearisation problems (because it does not linearise the vehicle pose), and is much more robust in situations of association ambiguity. For data association, each particle is allowed to make independent decisions, hence FastSLAM can maintain a probability distribution over all possible associations [16]. As more observations arrive, particles which made poor association decisions in the past tend to be removed in the resampling process, hence the majority of particles tend to converge to the correct set of associations. For the purposes of data association, FastSLAM automatically allows information to be integrated between observations at a *single* time step (as JCBB does), and between *multiple* time steps (as sliding-window methods do). The former occurs due to landmarks being conditionally-independent given the vehicle path, the latter due to each particle's memory of past associations. Furthermore, FastSLAM is simple to implement relative to complicated batch association algorithms.

The disadvantage of FastSLAM is its inability to maintain particle diversity over long periods of time [4]. The fundamental problem is that the particle filter is really operating in a very high-dimensional space: the space of vehicle *trajectories* (not momentary poses). The number of particles needed is therefore exponential in the length of the trajectory. When a smaller number is used, the filter underestimates the total uncertainty, and eventually becomes inconsistent. While this may still produce good maps, problems are encountered when the full uncertainty is required, for example when large loops need to be closed.

For remembering long-term uncertainty, the EKF is far superior. By using a continuous (Gaussian) representation, uncertainty does not degrade purely as a function of trajectory length. Linearisation errors are still a concern [3], but these are far less severe than FastSLAM's particle diversity problems.

This paper presents a hybrid mapping strategy, which combines the strengths of both approaches. We advocate the use of FastSLAM as a front-end to an EKF-SLAM back-end. The FastSLAM front-end is used to build local maps, and is allowed to run for long enough to disambiguate associations. Before the trajectory length becomes so long that particle diversity becomes problematic, a single Gaussian is computed from the FastSLAM posterior. This Gaussian local map is then fused into the global map. At the point of fusion, a hard decision must be made about the associations between local-map features and global-map features. However, a large local map can provide sufficient constraints to make the probability of a bad decision extremely low. The result is a SLAM algorithm which is robust to linearisation errors and data association ambiguities on a local scale, and can also close large loops on a global scale.

The remainder of the paper is structured as follows. Section 2 gives a brief review of FastSLAM and shows how a single Gaussian can be produced from a FastSLAM posterior, under the assumption of known associations. This assumption is removed in Section 3, which addresses the problem of producing a Gaussian posterior given unknown associations. Section 4 describes the algorithm for fusing local sub-maps produced by the front-end into the global map. Section 5 describes a set of experiments and discusses the results, and Section 6 concludes.

2 Conversion of Factored FastSLAM Distribution to a Single Gaussian

The aim of a SLAM algorithm is to compute the posterior

$$p(\mathbf{v}_{0:t}, \mathbf{M} | \mathbf{Z}_{0:t}, \mathbf{U}_{0:t}, \mathbf{v}_0) \tag{1}$$

where $\mathbf{v}_{0:t} = \mathbf{v}_0, \ldots, \mathbf{v}_t$ denotes the path of the vehicle, the map $\mathbf{M} = \theta_1, \ldots, \theta_N$ denotes the positions of a set of landmarks, $\mathbf{Z}_{0:t}$ represents the set of all observations, $\mathbf{U}_{0:t}$ denotes the set of control inputs, and $\mathbf{v}_0$ denotes the initial vehicle state. The subscript $_{0:t}$ indicates a set of variables for all time-steps up to and including t, while the subscript $_t$ is used to indicate the variable at time-step t. Many SLAM algorithms (including the one described in this paper) marginalise out past poses, estimating only

$$p(\mathbf{v}_t, \mathbf{M} | \mathbf{Z}_{0:t}, \mathbf{U}_{0:t}, \mathbf{v}_0) \tag{2}$$

Fusion into a global EKF-SLAM back-end requires the local mapping algorithm to produce this distribution in the form of a single multi-dimensional Gaussian. This section shows how a distribution of this form can be extracted from FastSLAM, under the restrictive assumption that the environment contains N uniquely-identifiable landmarks, all of which have been observed at least once. This assumption will be removed in Section 3.

2.1 The FastSLAM Posterior as a Gaussian Mixture Model (GMM)

The FastSLAM algorithm factors distribution 1 as follows [16]:

$$p(\mathbf{v}_{0:t}, \mathbf{M}|\mathbf{Z}_{0:t}, \mathbf{U}_{0:t}, \mathbf{v}_0) = p(\mathbf{v}_{0:t}|\mathbf{Z}_{0:t}, \mathbf{U}_{0:t}, \mathbf{v}_0) \prod_n p(\theta_n|\mathbf{v}_{0:t}, \mathbf{Z}_{0:t}, \mathbf{U}_{0:t}) \quad (3)$$

where the first factor represents the vehicle's path and subsequent factors represent landmark positions given the vehicle's path. This factored distribution is represented as a set of P samples, with the p^{th} particle S_t^p consisting of a weight w_t^p, a vehicle pose $\mathbf{v}_t^p$ (past poses are marginalised out), and N Gaussian landmark estimates described by their mean $\mu_{n,t}^p$ and covariance $\Sigma_{n,t}^p$:

$$S_t^p =< w_t^p, \mathbf{v}_t^p, \underbrace{\mu_{1,t}^p, \Sigma_{1,t}^p}_{\text{landmark } \theta_1}, \ldots, \underbrace{\mu_{N,t}^p, \Sigma_{N,t}^p}_{\text{landmark } \theta_N} > \quad (4)$$

While it is convenient to represent individual landmark distributions as isolated low-dimensional Gaussians, each particle can equivalently be represented using

$$S_t^p =< w_t^p, \mathbf{x}_t^p, \mathbf{P}_t^p > \quad (5)$$

where $\mathbf{x}_t^p = [\mathbf{v}_t^p, \mu_{1,t}^p, \ldots, \mu_{N,t}^p]$ denotes the concatenation of the vehicle states with all landmark states, and $\mathbf{P}_t^p$ denotes a block-diagonal covariance matrix constructed from the vehicle covariance and the covariances of each landmark:

$$\mathbf{P}_t^p = \begin{bmatrix} \mathbf{P}_{vv,t}^p & & & \\ & \Sigma_{1,t}^p & & \\ & & \ddots & \\ & & & \Sigma_{N,t}^p \end{bmatrix}$$

In this case the vehicle covariance $\mathbf{P}_{vv,t}^p$ is zero because each particle has no uncertainty associated with the vehicle states.

Using this representation, the FastSLAM distribution over vehicle and map states is a Gaussian Mixture Model (GMM): each particle is a Gaussian component with weight, mean, and covariance given by Equation 5.

2.2 Conversion to a Single Gaussian

The parameters of a single Gaussian with mean $\mathbf{x}_t$ and covariance $\mathbf{P}_t$ can be computed from the GMM using a process known as moment matching [5]:

$$\mathbf{x}_t = \sum_p w_t^p \mathbf{x}_t^p \quad (6)$$

$$\mathbf{P}_t = \sum_p w_t^p \left[\mathbf{P}_t^p + (\mathbf{x}_t^p - \mathbf{x}_t)(\mathbf{x}_t^p - \mathbf{x}_t)^T\right] \tag{7}$$

In Equation 7, the first term in the square brackets is the covariance of the particle's individual map (due to sensor noise), while the second term is due to the variation between particles' maps (due to vehicle noise).

3 Conversion using Unknown Associations

This section extends to the more realistic case where landmarks are not uniquely identifiable. Since FastSLAM particles are free to make data association decisions independently, both the number of landmarks and their ordering may differ between particles. However, since the particles are to be merged into a single high-dimensional Gaussian distribution (for fusion into the global map), it is necessary to track the correspondences between landmarks in different particles' maps. The aim is to produce a single common set of features from the particles' individual sets of features.

While the true set of landmarks is unknown, suppose each landmark in the environment is assigned a unique index. Each particle can be augmented with a set of correspondence variables of the form $\gamma_{i,t}^p$, indicating the correspondences between landmarks in the maps of individual particles and landmarks in the environment: $\gamma_{i,t}^p = n$ indicates that the i^{th} landmark in the p^{th} particle's map corresponds to the n^{th} landmark in the environment. Each particle is therefore of the form:

$$S_t^p = < w_t^p, \mathbf{v}_t^p, \underbrace{\mu_{1,t}^p, \Sigma_{1,t}^p, \gamma_{1,t}^p}_{\text{landmark } \theta_1^p} \ldots, \underbrace{\mu_{N^p,t}^p, \Sigma_{N^p,t}^p, \gamma_{N^p,t}^p}_{\text{landmark } \theta_{N^p}^p} > \tag{8}$$

Sections 3.1 and 3.2 describe procedures for (a) tracking these correspondence variables within a particle's map, and (b) generating a single Gaussian given a set of particles with non-identical correspondence variables.

3.1 Tracking Correspondences

The correspondences between features are tracked using a simple voting scheme. Consider a single observation. Each particle can vote to either

1. ignore the observation as spurious,
2. associate the observation with a particular existing map feature, or
3. create a new map feature and associate the observation with this.

Each particle votes in proportion to its weight, and the winner takes all: a single correspondence variable is computed based on the majority vote. All particles must use this correspondence variable for features modified or created by this observation, over-writing any previous correspondence variable.

This scheme forces the particles to form a consensus about the common set of features, but allows that consensus to be overturned at a later time if future information alters the particle weights. In principal it is possible for particles to disagree wildly about the number of features, but in practice a large majority tends to form quickly, out-voting a small set of dissenting particles.

3.2 Producing a Gaussian Given Correspondences

The correspondence variable γ provides a mapping from features in each particle's individual map to the common set of features $\mathbf{M}$. Let the function δ denote the reverse mapping: $\delta_t(n,p) = i$ indicates that the n^{th} feature in the common set is represented by the i^{th} feature in the p^{th} particle's map.

Given these variables, each particle can be represented using a weight, mean, and covariance as in Equation 5 with the mean

$$\mathbf{x}_t^p = [\mathbf{v}_t^p, \mu^p_{\delta_t(1,p),t}, \ldots, \mu^p_{\delta_t(N,p),t}] \tag{9}$$

and the block-diagonal covariance matrix

$$\mathbf{P}_t^p = \begin{bmatrix} \mathbf{P}^p_{vv,t} & & & \\ & \Sigma^p_{\delta_t(1,p),t} & & \\ & & \ddots & \\ & & & \Sigma^p_{\delta_t(N,p),t} \end{bmatrix}$$

In other words, each particle's individual map can be represented by a single multi-dimensional mean and covariance, produced by arranging the individual map feature states in the order of the common features about which they are a hypothesis. A Gaussian distribution can then be produced using Equations 6 and 7.

This approach requires that two corner cases be addressed. Firstly, a particle's individual map may contain multiple features referring to the same common feature. In this case, δ simply selects one at random (another alternative would be to use the fusion of the features). Secondly, a common feature may have no corresponding features in a particular particle's map. In this case, the particle is ignored when computing the mean and covariance of that common feature.

In addition to computing the mean and covariance of the landmarks' positions, independent state estimates (such as the probability of existence of each landmark) can be computed using weighted sums analogous to Equation 6. Maintaining an estimate of the probability of existence of each feature means that features which are not generally agreed upon will have a low probability of existence. These features can be ignored when fusing the local map into the global map.

4 Sub-Map Fusion

This paper uses a sub-mapping strategy similar to CLSF [17] and others [14]. Figure 1 illustrates the approach: the filter maintains both a Gaussian global map and a Gaussian local map. The (uncertain) global coordinate frame of the local map is stored in the global map. The filter periodically fuses the local map into the global map to produce a single map in the global coordinate frame. After fusion, a new local map is started with the vehicle beginning at the local origin with no uncertainty and no landmarks.

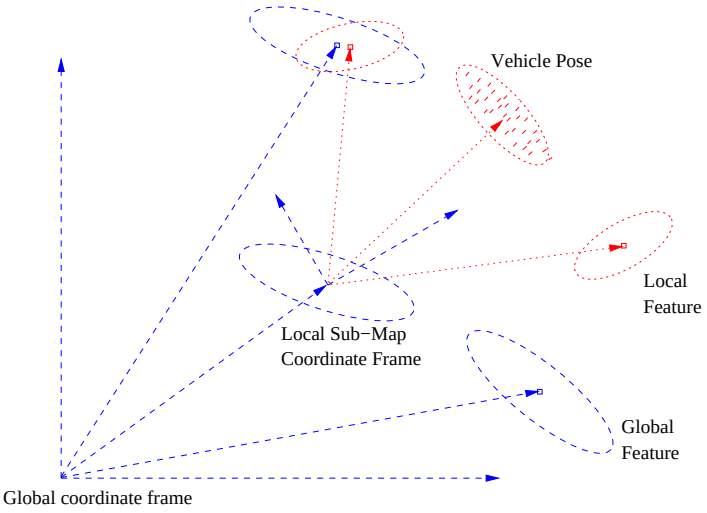

Fig. 1. An illustration of the sub-mapping strategy. In this case, the local map is converted to a Gaussian from a FastSLAM representation, in which the local vehicle pose is represented using particles.

Map fusion requires two steps:

1. local features are initialised in the global map, and
2. features are associated and fused.

Note that the vehicle pose in the local map is considered to be just another feature, and hence does not require any special treatment in the procedure that follows.

Let $\mathbf{x}^-$, $\mathbf{x}^+$, $\mathbf{P}^-$, and $\mathbf{P}^+$ refer to the global map mean and covariance immediately before and after fusion, and $\mathbf{x}_L$ and $\mathbf{P}_L$ refer to the local map mean and covariance. Local features (including the vehicle pose) are initialised in the global map by augmenting the state and covariances as follows:

$$\mathbf{x}^+ = \begin{bmatrix} \mathbf{x}^- \\ g(\mathbf{v}^-, \mathbf{x}_L) \end{bmatrix}$$

$$\mathbf{P}^{+} = \begin{bmatrix} \mathbf{P}^{-}_{vv} & \mathbf{P}^{-}_{vm} & \mathbf{P}^{-T}_{vv}\nabla_v g^T \\ \mathbf{P}^{-T}_{vm} & \mathbf{P}^{-}_{mm} & \mathbf{P}^{-T}_{vm}\nabla_v g^T \\ \nabla_v g\mathbf{P}^{-}_{vv} & \nabla_v g\mathbf{P}^{-}_{vm} & \nabla_v g\mathbf{P}^{-}_{vv}\nabla_v g^T + \nabla_z g\mathbf{P}_L\nabla_z g^T \end{bmatrix}$$

where $g(\mathbf{v}^-, \mathbf{x}_L)$ transforms the local map into the global coordinate frame, using $\mathbf{v}^-$: the vehicle's global pose at the time of the previous fusion (now the origin of the local sub-map).

New features are then associated with old features, using joint compatibility [11]. Two features θ_1 and θ_2 are fused using a zero-noise observation $\mathbf{z}_f$ of the difference between features:

$$\mathbf{z}_f(\theta_1, \theta_2) = \theta_1 - \theta_2 \tag{10}$$

and setting $\mathbf{z}_f = \mathbf{0}$. This leaves two identical features, one of which is removed. Finally, the old global vehicle pose estimate is replaced by the new estimate, and a new local map is started.

5 Experiments

5.1 Filters

Experiments compared the following filters in simulation:

1. FastSLAM v2.0 [16] (FASTSLAM),
2. The sub-mapping algorithm described in Section 4 with EKF front- and back-ends (EKF), and
3. The same sub-mapping algorithm with a FastSLAM front-end and an EKF back-end (HYBRID).

Experiments focussed on the filters' ability to both manage clutter and close large loops. This was done by having the filters close a loop of approximately 400m in a simulated environment, with a sensor subject to false alarms and non-detections. The sensor's reliability was specified by two parameters:

- α: the probability of detecting a true feature, and
- β: the probability of a visible true feature triggering a false alarm with uniformly-distributed range and bearing.

In order to be able to ignore spurious features, all three filters maintain an estimate of the probability of existence of each feature. Let $p(\iota_{n,t})$ denote the estimated probability of existence of the n^{th} feature at time t. This estimate can be updated as in a traditional target-detection framework [13], using

$$p(\iota_{n,t}) \leftarrow p(\iota_{n,t})\frac{\alpha}{p(\iota_{n,t})\alpha + (1 - p(\iota_{n,t}))\beta} \tag{11}$$

where the denominator gives the total probability of making an observation, either through false-alarm or true feature detection.

The two sub-mapping algorithms only fused features whose probability of existence exceeded 0.95, while FASTSLAM ran a periodic cull of old unlikely features. Sub-maps were fused at 10-second intervals. FastSLAM implementations used 500 particles. Note that while FastSLAM is often implemented with fewer particles (*e.g.* 100 [10]), a larger number can be tolerated in our approach because HYBRID's local FastSLAM map is always very small. Both FASTSLAM and the HYBRID front-end used Data Association Sampling (DAS) [16] for computing correspondences.

5.2 Environment

The experimental environments consisted of 50 randomly-generated simulated two-dimensional worlds of size 120m×120m, each containing 35 uniformly-distributed point features, as shown in Figure 2. The vehicle moved at 1m/s and up to 100°/s, observing features using a 360° range-bearing sensor with a maximum range of 25m. The vehicle followed a path near the extremities of the environment, such that one significant loop closure had to be executed. A test run was considered to have failed if the true vehicle pose remained more than 10 standard deviations from the filter's estimate for more than 20 iterations.

SLAM filters were run at the observation frequency of 10Hz. Vehicle linear and rotational velocity was noisy, perturbed with standard deviations equal to $0.25\dot{x}$ and $0.35\dot{\theta}$ respectively, where $\dot{x}$ and $\dot{\theta}$ are commanded linear and rotational velocities. Observation noise standard deviations were set to $0.05m$ and 1° in range and bearing respectively. Identical worlds, odometry, and observations were used for all filters.

5.3 Results

The results are summarised in Table 1. They clearly show that the HYBRID filter out-performs both FASTSLAM and EKF filters. FASTSLAM's tendency to become overconfident meant that it was unable to succeed in a single run (see the following sections for more detail).

Comparing the two sub-mapping approaches, the benefits of using HYBRID versus EKF are apparent with no clutter but most pronounced in high levels of clutter. While the level of clutter in the extreme case is particularly high, it is also unrealistically easy to filter because it is truly random. Clutter in the real world is not independent and identically distributed, but tends to be correlated, clustered around particular (possibly moving) objects.

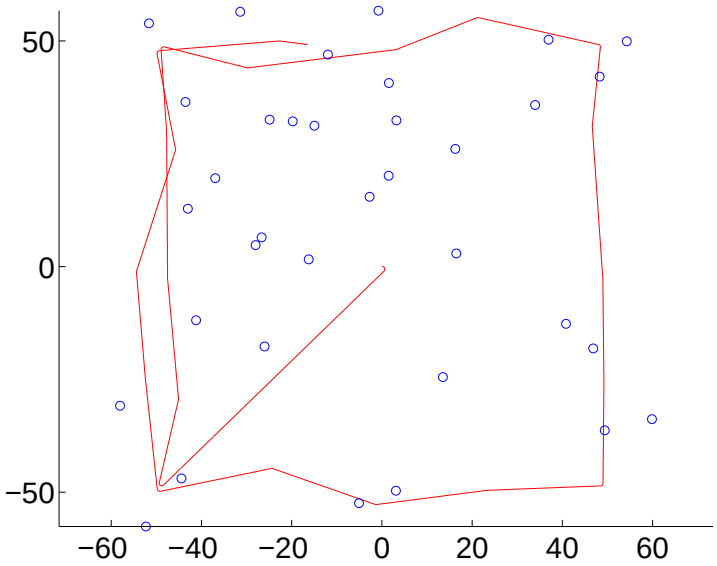

Fig. 2. An example scenario, showing the true vehicle path and landmark positions. The precise path and landmark locations are randomly generated.

	α, β [p(truePositive),p(falsePositive)]			
Filter	1.0, 0.00	0.7, 0.02	0.4, 0.04	0.1, 0.06
FASTSLAM	0%	0%	0%	0%
EKF	79%	76%	72%	30%
HYBRID	87%	89%	83%	63%

Table 1. Success rate for each algorithm, evaluated over 100 trials in each case, for different values of α and β. Failure is defined as the vehicle mean being more than 10 standard deviations from the true vehicle pose for more than 20 iterations.

5.4 Discussion

Failures occurred for a number of reasons. The FASTSLAM failures occurred due to over-confidence, resulting from an inability to remember uncertainty over long trajectories [4].

The two sub-mapping algorithms have two common long-term failure modes, specific to the EKF back-end. Firstly, when sufficient vehicle heading uncertainty accumulates, linearisation errors can cause the filter to become over-confident [3]. Secondly, data association errors were sometimes made when closing the loop. This latter problem could be solved in a variety of ways, for example by running the local mapper for longer when a loop closure is imminent. Both problems were exacerbated as the reliability of the sensor

was decreased, since the total amount of sensor information available to the filter was reduced, increasing vehicle uncertainty.

The differences between the two sub-mapping filters occurred due to differences in short-term mapping in the front-end. Failures in the EKF filter occurred due to spurious observations being associated with landmarks, particularly during transient periods of high vehicle uncertainty (*i.e.* during turns). This is especially problematic when the true feature is not observed simultaneously, and therefore JCBB is unable to resolve the ambiguity.

EKF failures also occurred due to local linearisation errors. Local linearisation errors are kept to a minimum because vehicle uncertainty in the local frame of reference is generally small, since local maps begin with no uncertainty. However local-map uncertainty can be significant, for example if the vehicle begins a new sub-map immediately prior to making a sharp turn amid sparse landmarks.

The HYBRID filter is much more robust to both these local sources of error. The probability of incorporating spurious observations is reduced by the front-end's ability to integrate data association information over time. Vehicle linearisation errors are eliminated because each particle always has zero vehicle uncertainty.

5.5 Linearisation

The effects of linearisation were analysed in more detail in two additional experiments in the same simulated environment, both in the absence of clutter ($\alpha = 1.0$, $\beta = 0.0$). In the first experiment, the vehicle is provided with an accurate compass, eliminating vehicle linearisation errors. The observation model is still linearised, however this introduces only very small errors relative to vehicle non-linearities. Since the Kalman Filter is the optimal estimator in the linear case, the EKF solution is close to optimal in this case, and therefore provides a benchmark against which to compare alternative algorithms.

In the absence of vehicle non-linearities and clutter, one would expect HYBRID and EKF to produce very similar results. The major remaining difference between the two is statistical noise introduced by the HYBRID's FastSLAM front-end, which we minimise by using 5000 particles.

Figure 3 compares the vehicle uncertainty for all three filters, measured using the trace of the vehicle pose covariance matrix. The figure shows that HYBRID can correctly track the uncertainty while FASTSLAM quickly becomes over-confident, even with 5000 particles.

In a second experiment, vehicle uncertainty was introduced by removing the compass. Since the vehicle is executing many uncertain turns (including one sharp turn near the beginning of the trajectory), non-linearity due to vehicle heading uncertainty is problematic. Figure 4 shows the vehicle uncertainty over the entire run.

In this case, the uncertainties of both sub-mapping solutions are certainly too small, due to vehicle linearisation issues. However, since the HYBRID so-

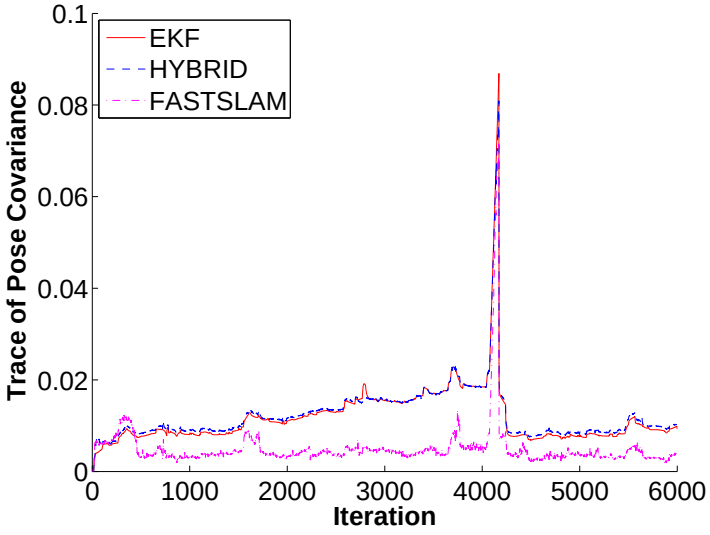

Fig. 3. Vehicle uncertainty over time in a scenario with no vehicle non-linearities. Uncertainty is measured using the trace of the covariance matrix. In order to produce a meaningful comparison with FASTSLAM, the sub-mapping algorithms' estimates are based on both the local and global maps at every iteration, even though the maps are only fused on every 100^{th} iteration.

lution avoids vehicle linearisation in local sub-maps, the larger uncertainties which it reports are closer to the optimal solution. HYBRID is clearly an improvement on EKF in this respect. The swift deterioration of the FASTSLAM estimate is clear in this scenario.

5.6 Live Experiments

The HYBRID filter was used to generate a map in a real urban environment, using a Segway RMP equipped with a SICK laser, as shown in Figure 5. The environment was approximately 135m×90m in size, and was mapped over a 3.75km path. The final map consisted of 452 landmarks, including foreground points, corners, doorways, and 46 retro-reflective strips. Data association was complicated by the presence of difficult-to-model objects such as bushes and trees, people occasionally walking through the area, and the tilting nature of the platform which caused it to make frequent observations of the ground when accelerating. Observations of the ground result in spurious corner features from the intersection of the ground with walls. The final map is shown in Figure 6, with an occupancy-grid overlaid for clarity.

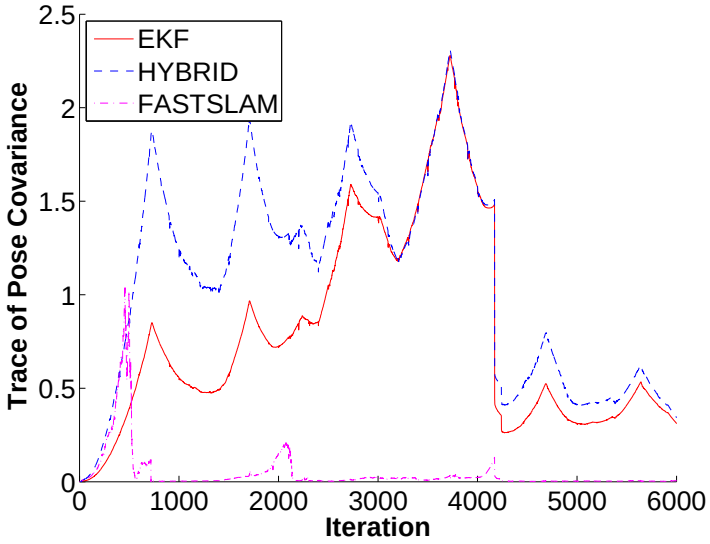

Fig. 4. Vehicle uncertainty over time in a more realistic scenario, with vehicle non-linearities. Sub-map fusion causes small drops in uncertainty every 100 iterations. The large decrease in uncertainty near iteration 4400 is due to the loop closure.

6 Conclusion

This paper presented a hybrid approach to SLAM which combines the strengths of two popular existing approaches: FastSLAM and EKF-SLAM. Using EKF-SLAM as a global mapping strategy allows uncertainty to be remembered over long vehicle trajectories. Using FastSLAM as a local mapping strategy minimises observation-rate linearisation errors and provides a significant level of robustness to clutter and uncertain associations. In experiments involving a large number of randomly-generated trials, the hybrid mapping approach was shown to be clearly superior to either pure approach. The approach was further validated in a live experiment.

One way to view HybridSLAM is as a mixture between pure FastSLAM and pure EKF-SLAM, with the time between map fusions as the mixing parameter. Consider the two extremes: if map fusion occurs on every iteration, HybridSLAM is largely equivalent to EKF-SLAM (with no sub-mapping). If map fusion never occurs, HybridSLAM is identical to FastSLAM.

We see the following as the main areas in which the algorithm described in this paper could be improved:

Fig. 5. The Segway RMP vehicle and the environment

Local-to-Global Interface

The interface between the FastSLAM front-end and the EKF back-end is important: fusing in a moment of transient ambiguity can result in an incorrect decision being irreversibly fused into the back-end. This paper performed experiments with a very simple criterion for map fusion (local-map age equal to 10 seconds), however better strategies are certainly possible and likely to increase robustness. For example, it may be prudent to delay map fusion when the vehicle distribution is multi-modal or particularly non-Gaussian, of when there is significant disagreement about landmark identities. Conversely, it may be of benefit to fuse sooner if particle diversity begins to drop.

Another option is to re-use information from slightly before and after the point of map fusion in both temporally-adjacent local maps, for the purposes of data association only. Since uncertainty about data association is not tracked in the back-end, re-use of information in this way will not lead to over-confidence. Future work will address these possibilities.

Scalability and Non-Linearities

For simplicity, this presentation showed a monolithic EKF back-end, which is known to suffer from non-linearities and high computation in large environ-

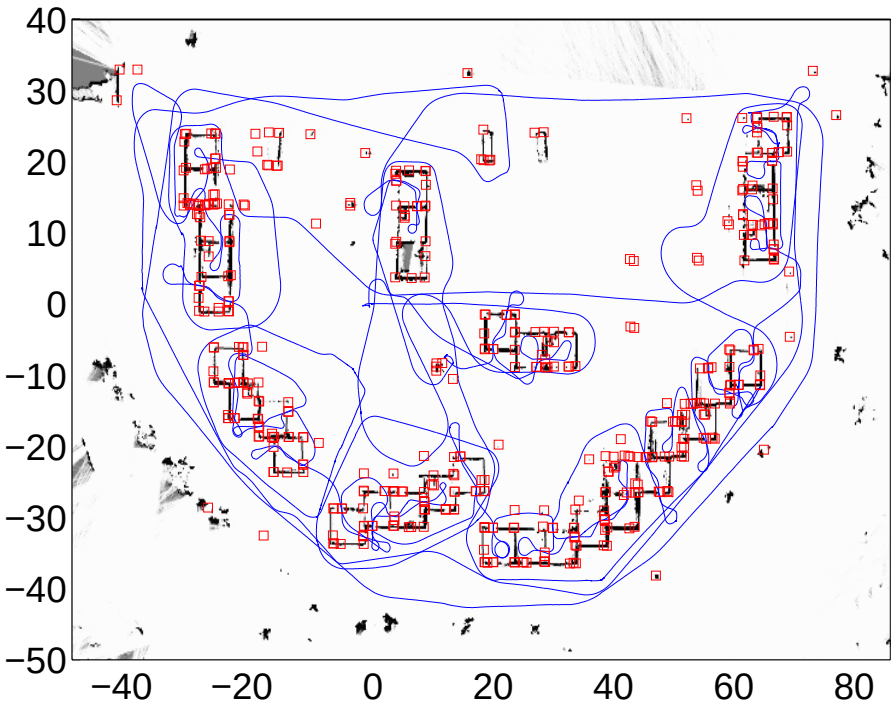

Fig. 6. A map of an urban environment produced by the HYBRID filter. Landmarks are shown as red squares, while the smoothed vehicle path is shown as a blue line. The occupancy grid map is overlaid for clarity.

ments. In future work, the back end could be replaced by more efficient Gaussian representations such as sparse information form [15] or submap methods such as ATLAS [7] of NCFM [1]. The submap forms also address long-term non-linearity.

References

1. T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, University of Sydney, 2002.
2. T. Bailey and H. Durrant-Whyte. Simultaneous localisation and mapping (SLAM): Part II - state of the art. *Robotics and Automation Magazine*, 13(3):108–117, 2006.
3. T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot. Consistency of the EKF-SLAM algorithm. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 3562–3568, 2006.
4. T. Bailey, J. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 424–429, 2006.

5. Y. Bar-Shalom, X. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. John Wiley and Sons, 2001.
6. C. Bibby and I. Reid. Simultaneous localisation and mapping in dynamic environments (SLAMIDE) with reversible data association. In *Proc. Robotics: Science and Systems*, 2007.
7. M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 1899–1906, 2003.
8. H. Durrant-Whyte and T. Bailey. Simultaneous localisation and mapping (SLAM): Part I - the essential algorithms. *Robotics and Automation Magazine*, 13(2):99–110, 2006.
9. A. Eliazar and R. Parr. DP-SLAM 2.0. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 2, pages 1314–1320, 2004.
10. M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 1985–1991, 2003.
11. J. Neira and J. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, 2001.
12. L. Perera, W. Wijesoma, and M. Adams. Data association in dynamic environments using a sliding window of temporal measurement frames. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 753–758, 2005.
13. L. Stone, C. Barlow, and T. Corwin. *Bayesian Multiple Target Tracking*. Artech House, 1999.
14. J. Tardos, P. Newman, and J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Intl. Journal of Robotics Research*, 21(4):311–330, 2002.
15. S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 23(7-8):693–716, 2004.
16. S. Thrun, M. Montemerlo, D. Koller, B. Wegbreit, J. Nieto, and E. Nebot. FastSLAM: An efficient solution to the simultaneous localization and mapping problem with unknown data association. *Journal of Machine Learning Research*, 2004.
17. S. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 406–411, 2002.