2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

The Common State Filter for SLAM

Martin P. Parsley and Simon J. Julier

Abstract— **This paper presents the Common State Filter (CSF), a novel and efficient suboptimal Multiple Hypothesis SLAM (MHSLAM) method for Kalman Filter-based SLAM algorithms. Conventional MHSLAM algorithms require the entire vehicle and map state to be copied for each hypothesis. The CSF, by contrast, maintains a single, common instance of the vast majority of the map and only copies the map portion that varies substantially across different hypotheses. We demonstrate the performance of the algorithm on the Victoria Park data set.**

I. INTRODUCTION

A key requirement for robust and practical simultaneous localisation and mapping (SLAM) is the ability to handle *ambiguity*. Ambiguity can arise in several ways. The most common is the data association problem. This may be addressed by a number of methods including nearest neighbour [1], matching appearance descriptors [2] and landmark configurations [3]–[5], however the association is not guaranteed to be unique. A second source of ambiguity are time delayed measurements with unknown time delays [6]. A third cause lies in multiple model estimation for a platform whose active movement class (of a predefined set) at any instant in time is not known [7]. One means of handling bearing-only SLAM is to use range-parameterised filters, which create a set of discrete range hypotheses [8]–[10].

In principle, all of these types of ambiguity can be represented in a Multiple Hypothesis SLAM (MHSLAM) framework: the set of all feasible candidates are enumerated and, for each candidate, a new filter is created [11]. At subsequent time steps, measurement likelihoods are used to recursively compute the probability that a given hypothesis is correct. Although MHSLAM is an exact solution, it has not been widely used for two key reasons. The first is that the number of hypotheses can grow exponentially over time [11]. This problem can be mitigated through the use of appropriate pruning strategies and hypothesis merging methods [12], [13]. Second, each hypothesis must contain a complete copy of the vehicle and all of the beacons in the map. For large maps, even maintaining just a few hypotheses becomes prohibitively expensive [4], [11], [14].

Particle filters are an alternative to EKF-based MHSLAM, and can inherently handle multiple hypotheses. The most popular such method is FastSLAM [15]. However, particle filters have their own drawbacks compared to the EKF, namely inconsistency [16], and thus an EKF-based solution still has merit. In addition, the number of particles required for performing MHSLAM may still grow exponentially as the filter must sample over robot paths and data associations [15].

Martin Parsley was supported by a UK Research Council EPSRC Doctoral Training Award number EP/P502802/1.

M. Parsley and S. Julier are with the Department of Computer Science, University College London, Gower Street, London, WC1E 6BT, UK
M.Parsley@cs.ucl.ac.uk; S.Julier@cs.ucl.ac.uk

In this paper we consider the problem of reducing the computational and storage costs for EKF-based MHSLAM. Our solution, which we call the Common State Filter (CSF), exploits the intuition that the values of most map states are very similar across all the different hypotheses. The CSF maintains a single, common instance of the majority of the map and each hypothesis only stores the small number of states which vary substantially across the different hypotheses. The algorithm leads to substantial reductions in both computational and storage costs with only a slight reduction in accuracy due to the suboptimality.

The structure of the paper is as follows. Section II provides a brief overview of SLAM and introduces the notation used. In Section III we describe the problem of data association and show how it induces an MHSLAM structure. Section IV introduces and describes the basic principles of the Common State Filter (CSF). A practical demonstration of the CSF on the Victoria Park data set is given in Section V. Finally, conclusions and a summary are given in Section VI.

II. SLAM WITHOUT AMBIGUITIES

The structure of the full covariance SLAM solution is as follows. At time k, the true state $\mathbf{x}(k)$ consists of the vehicle pose $\mathbf{x}_v(k)$ and the set of n static beacons $\mathbf{x}_{1\dots n}$,

$$\mathbf{x}(k) = \left[\begin{array}{cccc} \mathbf{x}_v^T & \mathbf{x}_1^T & \dots & \mathbf{x}_n^T \end{array}\right]_k^T .$$

The mean and covariance of this estimate are

$$\hat{\mathbf{x}}(i|j) = \left[\begin{array}{cccc} \hat{\mathbf{x}}_v^T & \hat{\mathbf{x}}_1^T & \dots & \hat{\mathbf{x}}_n^T \end{array}\right]_{i|j}^T = \left[\begin{array}{cc} \hat{\mathbf{x}}_v^T & \hat{\mathbf{x}}_b^T \end{array}\right]_{i|j}^T , \tag{1}$$

$$\mathbf{P}(i|j) = \left[\begin{array}{cccc} \mathbf{P}_v & \mathbf{P}_{v1}^T & \dots & \mathbf{P}_{vn}^T \\ \mathbf{P}_{v1} & \mathbf{P}_{11} & \dots & \vdots \\ \vdots & \vdots & \ddots & \mathbf{P}_{1n}^T \\ \mathbf{P}_{vn} & \mathbf{P}_{1n} & \dots & \mathbf{P}_{nn} \end{array}\right]_{i|j} = \left[\begin{array}{cc} \mathbf{P}_v & \mathbf{P}_{vb}^T \\ \mathbf{P}_{vb} & \mathbf{P}_b \end{array}\right]_{i|j} , \tag{2}$$

where $(i|j)$ denotes the estimate at time i given observations up to and including time j [17]. $\mathbf{P}_v$ is the vehicle pose covariance, $\mathbf{P}_{nn}$ is the nth beacon covariance, $\mathbf{P}_{vj}$ are the cross correlations between the vehicle and the jth beacon, and $\mathbf{P}_{nj}$ are the cross correlations between the nth and jth beacons.

By assumption, the beacons are stationary and no process noise acts upon them. Therefore, the motion model corresponding to the vehicle pose $\mathbf{x}_v$ at time k is

$$\mathbf{x}_v(k) = \mathrm{f}\left[\mathbf{x}_v(k-1), \mathbf{u}(k), k, \mathbf{v}(k)\right], \qquad (3)$$

where $\mathbf{u}(k)$ are the control inputs and $\mathbf{v}(k)$ is the zero-mean control noise with covariance $\mathbf{Q}(k)$. The estimated state $\hat{\mathbf{x}}_v$ propagates according to

$$\hat{\mathbf{x}}_v(k|k-1) = \mathrm{f}\left[\hat{\mathbf{x}}_v(k-1|k-1), \mathbf{u}(k), k, \mathbf{0}\right], \qquad (4)$$

and the covariances according to

$$\mathbf{P}(k|k-1) = \nabla\mathbf{F}(k)\,\mathbf{P}(k-1|k-1)\,\nabla\mathbf{F}(k)^T + \nabla\mathbf{G}(k)\,\mathbf{Q}(k)\,\nabla\mathbf{G}(k)^T + \mathbf{Q}_s, \qquad (5)$$

where $\nabla\mathbf{F}(k)$ is the Jacobian of f, $\nabla\mathbf{G}(k)$ is the control inputs Jacobian which takes the form $\nabla\mathbf{G}(k) = \begin{bmatrix} \nabla\mathbf{G}_v & 0 & \ldots & 0 \end{bmatrix}_k$, the v subscript referring to the vehicle, and $\mathbf{Q}_s$ is optional stabilising noise.

An observation $\mathbf{z}(k)$ can arise from one of three sources - either it is an observation of a beacon already in the map, a new beacon not present, or clutter. When the status of the observation is known implementing SLAM is, in principle, very simple.

If the beacon is already present in the map, the state is updated using the standard Kalman filter update equations. The observation function that links the observation $\mathbf{z}(k)$ of the jth beacon to the state $\mathbf{x}(k)$ is

$$\mathbf{z}_j(k) \quad = \quad \mathbf{h}_j\left[\mathbf{x}(k), \mathbf{w}(k)\right]. \qquad (6)$$

The observation noise $\mathbf{w}(k)$ is zero-mean noise with covariance $\mathbf{R}(k)$. The observation Jacobian $\nabla\mathbf{H}(k)$ for this function is *sparse.*

The update is computed using the standard Kalman filter update equations,

$$\begin{aligned}
\mathbf{K}(k) &= \mathbf{P}(k|k-1)\nabla\mathbf{H}^T(k)\left[\nabla\mathbf{H}\mathbf{P}(k|k-1)\nabla\mathbf{H}^T + \mathbf{R}\right]_k^{-1} && (7)\\
\hat{\mathbf{x}}(k|k) &= \hat{\mathbf{x}}(k|k-1) + \mathbf{K}(k)(\mathbf{z}(k) - \nabla\mathbf{H}(k)\hat{\mathbf{x}}(k|k-1)) && (8)\\
\mathbf{J}(k) &= \mathbf{I} - \mathbf{K}(k)\,\nabla\mathbf{H}(k) &&\\
\mathbf{P}(k|k) &= \mathbf{J}(k)\,\mathbf{P}(k|k-1)\,\mathbf{J}(k)^T + \mathbf{K}(k)\,\mathbf{R}(k)\,\mathbf{K}(k)^T, && (9)
\end{aligned}$$

where $\mathbf{K}(k)$ is the Kalman gain.

New beacons are added to the map following the method in [18]. Clutter observations arise from landmarks which are not static, and should be ignored.

Given that there is no ambiguity the implementation of SLAM is straightforward. However, as explained in the introduction, ambiguities can arise for many reasons. One important cause is data association - if the status of the beacon (known, unknown, clutter) is unknown. In such situations, the basic SLAM algorithm is not capable of modelling the induced uncertainty.

III. Ambiguous Data Association and MHSLAM

In the previous section we described Kalman Filter-based SLAM algorithms when there is no ambiguity in data association. However, ambiguities in data association can arise whenever beacons are not uniquely identifiable. The most well-known cause of this is loop-closing [19], but they can occur whenever the environment has a large number of repetitive beacons. Incorrect association can, at best, cause the creation of additional spurious beacons and, at worst, cause catastrophic filter failure [4]. Basic approaches such as individual compatibility nearest neighbour (ICNN) are computationally simple, but most prone to data association failures [5]. Some methods for data association, such as joint compatibility branch and bound (JCBB) [5], attempt to make the most informed decision using all the observations, however must do so at the time the observations are received. Thus their decisions may still be incorrect.

The most general approach to the problem, adopted from multiple target tracking, is to use multiple hypotheses [11], [15]. Such a Multiple Hypothesis SLAM (MHSLAM) propagates a set of $m(k)$ hypotheses. The main advantage over other methods is its generality, and the ability to defer association decisions, while making full use of the information available. Dropping ambiguous observations as an alternative is restrictive, and leads to increased ambiguity further on [14].

In MHSLAM each hypothesis maintains its own mean and covariance estimate and a weight, which is the probability that the state is correct. Therefore, the MHSLAM state can be written as

$$\left\{ \ \{\hat{\mathbf{x}}, \mathbf{P}, w\}^1 \quad \{\hat{\mathbf{x}}, \mathbf{P}, w\}^2 \quad \ldots \quad \{\hat{\mathbf{x}}, \mathbf{P}, w\}^m \ \right\}_{k|k}. \qquad (10)$$

A normalised weighting $w^{1\cdots m}(k)$ is maintained for each filter, and represents the likelihood that that filter is the correct hypothesis. In a SLAM context this has been referred to as a "brute force" method [11], and the Gaussian Sum filter (GSF) when using multiple Gaussians to approximate another distribution [9], [20]. In all these cases, the filter maintains a weighted bank of independent EKFs, each one representing a hypothesis (combination of possible beacon locations).

The steps for the algorithm are as follows:

1) Predict the state forwards in each hypothesis to the current time. This is achieved by executing the mean and covariance prediction equations (4) and (5).
2) Perform data association of the observation with each hypothesis. The likelihood of a measurement being consistent with a given hypothesis is computed. Several approaches could be used, but we use gating [1], with a threshold for association of 10^{-3}.
3) Update and reweight. For a gated measurement, the update equations (7) to (9) are applied and the weight is updated (assuming a Gaussian likelihood)

$$w^h(k|k) = w^h(k-1|k-1)\,\frac{1}{(2\pi)^{n/2}\sqrt{|S^h|}}\,e^{-\frac{1}{2}(\nu^T S^{-1}\nu)^h}, \qquad (11)$$

where ν^h is the innovation, n is the dimension of the innovation vector, and S^h is the innovation covariance of the observed beacons in filter h.
4) Pruning. In this case filters with a normalised weight falling below a threshold $\frac{w_t}{m(k)}$ ($m(k)$ being the number

of hypotheses at time k) are trimmed and the remaining set renormalised.

5) Spawn new hypotheses. A given observation is ambiguous if there is more than one beacon that can correspond to it. This is usually defined to be those beacons whose likelihood is greater than a threshold d_w. It is also possible that the observation may correspond to a new beacon that is not in the state, or be clutter. Of the possible associations, only one is correct. Because incorrectly associating an observation with a beacon in the state can have negative consequences [3], [4], we spawn off a number of parallel instances of the filter, each of which make a separate hypothesis as to which beacon, if any caused that observation. The assumption is that in time the correct hypothesis will show itself to be the most likely, allowing the rest of the (incorrect) hypotheses to be pruned.

For subsequent measurements where there are multiple hypotheses, data association must be performed on each filter in turn. As the number of hypotheses spawned can grow quickly, it is important that the filter is able to detect and prune unlikely hypotheses as quickly as possible, to keep the computational and storage cost feasible. Each filter that survives at the end can be considered to correspond to a chain of measurement association hypotheses which over time have been shown to have a high likelihood of being correct.

If a hypothesis requires the addition of a new beacon, it is added to the state using the same method as in regular SLAM.

When there are multiple hypotheses $1 \dots m$, the representative mean and covariance at time k is computed from [9],

$$\bar{\mathbf{x}}(k|k) = \left\{ \sum_{h=1}^{m} w^h \hat{\mathbf{x}}^h \right\}_{k|k}, \quad (12)$$

$$\bar{\mathbf{P}}(k|k) = \left\{ \sum_{h=1}^{m} w^h \left(\mathbf{P}^h + (\hat{\mathbf{x}}^h - \bar{\mathbf{x}})(\hat{\mathbf{x}}^h - \bar{\mathbf{x}})^T \right) \right\}_{k|k}, \quad (13)$$

where w^h is the normalised weighting of the filter h and the overbars show that this is the state representative of all the hypotheses.

However, MHSLAM has two important disadvantages. First, the number of hypotheses increase exponentially with the number of ambiguous associations. Second, a complete state must be maintained for each hypothesis. Clearly there is a large computational storage and update complexity associated with this. This can be avoided to a certain extent by considering hypotheses sequentially, as in lazy data association [14]. Even if aggressive pruning techniques are used, maintaining even a small number of hypotheses can be prohibitively expensive. If we consider a state comprising n beacons over m hypotheses, the storage requirements of full MHSLAM are $O(mn^2)$, with an update complexity of $O(mn^3)$. One way to make MHSLAM tractable is to reduce the computational and storage costs associated with each hypothesis. Several methods attempt to approximate MHSLAM using a single state [21]. [22] loses information by neglecting correlations between new features, while [23] does not prevent hypothesis mixing, thus may become inconsistent.

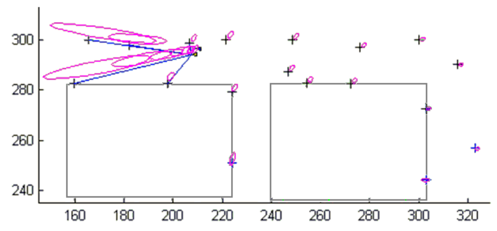
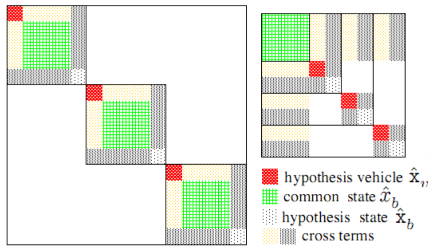


Fig. 1. Map showing a full MHSLAM run.



🟥 hypothesis vehicle $\hat{\mathbf{x}}_v$
🟩 common state $\hat{x}_b$
hypothesis state $\hat{\mathbf{x}}_b$
▨ cross terms

Fig. 2. Full MHSLAM (left) and CSF (right) covariance matrices shown as a single large matrix for 3 hypotheses. The common state beacon covariances are square hatches, the hypothesis state beacons are sparse black dots. The cross terms between the vehicle and map and hypothesis covariances are shown in lighter and darker shading respectively.

We have developed an algorithm, known as the Common State Filter (CSF) to address these issues.

IV. The Common State Filter

The intuition for the CSF comes from Fig. 1. This figure shows an example of a vehicle performing planar SLAM, with the crosses representing beacons, and the covariance ellipses representing 3σ beacon estimates. There are 6 hypotheses, and the ellipses for all the beacons on the map across all the hypotheses have been superimposed (i.e. 6 ellipses are shown per beacon). Close examination shows that the map may be segregated into beacons that are "near" (i.e. being observed or which have been observed recently), and "far". "Near" beacon estimates vary substantially across hypotheses, while "far" beacons have very similar estimates, as the map shows. Full MHSLAM makes no distinction between the two types of states and replicates the entire map across all hypotheses. This is inefficient, as it can contain hundreds of states in a large map. Based on the above intuition, the CSF removes the need to copy distant beacons across all hypotheses.

The benefit of this method is illustrated in Fig. 2 which visually indicates the structure of the covariance matrices required to maintain MHSLAM and CSF covariances. The overall storage requirements of the CSF are less than in full MHSLAM; this becomes more apparent as the number of hypotheses or the size of the state grows.

This state partitioning concept is similar to postponement [24], a method of amortising the computational costs associated with the Kalman filter, by deferring the updates of "far" beacons in the state to a more convenient time. Postponement, as applied to single hypothesis SLAM is fully optimal and is intended to give the platform increased flexibility in the times at which it updates beacons in the state. The CSF by comparison is a suboptimal method that is mainly intended to reduce the storage and computational costs of MHSLAM at a given point in time.

In deriving the CSF let us consider the full MHSLAM structure in (10) above. This will become the set of hypothesis states $\left\{ \begin{bmatrix} \hat{\boldsymbol{x}}_v^T & \hat{\boldsymbol{x}}_b^T \end{bmatrix}^T, \boldsymbol{P}, w \right\}^{1 \dots m}$. We augment this with a common state $\{\hat{\mathsf{x}}_b, \mathsf{P}_b\}$, which remains the same across all hypotheses. The structure of all the hypotheses becomes

$$\hat{\mathbf{x}}^{1\dots m}(k|k) = \left\{ \begin{bmatrix} \begin{bmatrix} \hat{\mathsf{x}}_b \end{bmatrix} \\ \begin{bmatrix} \hat{\boldsymbol{x}}_v \\ \hat{\boldsymbol{x}}_b \end{bmatrix}^1 \end{bmatrix}, \dots, \begin{bmatrix} \begin{bmatrix} \hat{\mathsf{x}}_b \end{bmatrix} \\ \begin{bmatrix} \hat{\boldsymbol{x}}_v \\ \hat{\boldsymbol{x}}_b \end{bmatrix}^m \end{bmatrix} \right\}_{k|k}. \tag{14}$$

The associated covariance structure of the common state and each hypothesis state, along with the cross terms between them is

$$\mathbf{P}^{1\dots m}(k|k) = \\ \left\{ \begin{bmatrix} [\mathsf{P}_b] & \begin{bmatrix} \boldsymbol{p}_{\mathrm{bv}}^{\mathrm{T}} & \boldsymbol{p}_{\mathrm{bb}}^{\mathrm{T}} \end{bmatrix}^1 \\ \begin{bmatrix} \boldsymbol{p}_{\mathrm{bv}} \\ \boldsymbol{p}_{\mathrm{bb}} \end{bmatrix}^1 & \begin{bmatrix} \boldsymbol{P}_{\mathrm{v}} & \boldsymbol{P}_{\mathrm{vb}}^{\mathrm{T}} \\ \boldsymbol{P}_{\mathrm{vb}} & \boldsymbol{P}_{\mathrm{b}} \end{bmatrix}^1 \end{bmatrix}, \dots, \begin{bmatrix} [\mathsf{P}_b] & \begin{bmatrix} \boldsymbol{p}_{\mathrm{bv}}^{\mathrm{T}} & \boldsymbol{p}_{\mathrm{bb}}^{\mathrm{T}} \end{bmatrix}^m \\ \begin{bmatrix} \boldsymbol{p}_{\mathrm{bv}} \\ \boldsymbol{p}_{\mathrm{bb}} \end{bmatrix}^m & \begin{bmatrix} \boldsymbol{P}_{\mathrm{v}} & \boldsymbol{P}_{\mathrm{vb}}^{\mathrm{T}} \\ \boldsymbol{P}_{\mathrm{vb}} & \boldsymbol{P}_{\mathrm{b}} \end{bmatrix}^m \end{bmatrix} \right\}_{k|k}. \tag{15}$$

The covariance matrix of each hypothesis state is shown in uppercase, and the cross terms with the common state are shown in lowercase. Beacons that have not had a recent observation will eventually be moved into the common state, as described in Section IV-C. The joint common and hypothesis state across all hypotheses is shown in (14), with corresponding covariance structure (15).

A. *CSF prediction*

The CSF prediction step applies (4) and (5) to each hypothesis in turn. In the covariance update step (5), the structure of the control inputs Jacobian corresponding to hypothesis h at time k, $\nabla\mathbf{G}^h(k)$ is $\begin{bmatrix} \mathbf{0} & \nabla\mathbf{G}_v^h & \mathbf{0} \end{bmatrix}_k^T$. $\nabla\mathbf{F}^h(k)$ has the form

$$\nabla\mathbf{F}^h(k) = \operatorname{diag}\left(\left[\mathbf{1}, \nabla\mathbf{F}_v^h(k), \mathbf{1} \right]^h \right)$$

with respect to (14), where diag$(\cdot)$ is the diagonal matrix, and $\nabla\mathbf{F}_v^h(k)$ is the vehicle Jacobian for the hypothesis h.

B. *CSF update*

When there is a single hypothesis, the single map is updated using (7) to (9). When there are multiple hypotheses, we cannot use the standard Kalman update equations to update $\hat{\mathbf{x}}$ and $\mathbf{P}$ as the cross correlations would update the common state, corrupting it in the case of incorrect hypotheses. We use the Schmidt-Kalman [25] [26] form of the update equations to update $\hat{\boldsymbol{x}}^h$ (and its corresponding $\boldsymbol{P}^{\mathrm{h}}$ and $\boldsymbol{p}^{\mathrm{h}}$) without updating the common state $\hat{\mathsf{x}}$ or its covariance P.

Common state beacons which have been associated with an observation (in any hypothesis) are transferred into each

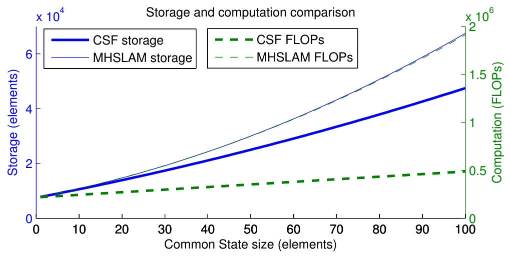


Fig. 3. Covariance storage and computational requirements for the CSF and full MHSLAM as the state size increases. This assumes each hypothesis state has 50 elements, and that there are 5 hypotheses.

hypothesis state and are updated using their respective hypothesis vehicles. Moving beacons from the common state into each hypothesis state is simple; for a beacon j, reorder every state in (14) and the rows and columns of every covariance matrix in (15), so that P_j and its cross terms are placed at the end of the hypothesis state in $\boldsymbol{P}_{\mathrm{b}}$.

As all beacons with observations will be found in the hypothesis state, any non-zero terms in the observation Jacobian will be confined to entries in the hypothesis states, so we need only consider this part of the Jacobian, $\nabla\mathbf{H}^h$. The gain $\mathbf{K}^h(k)$, which corresponds to the h hypothesis state is computed from (7). The hypothesis state is updated by (8) and the covariance by

$$\mathbf{J} = \mathbf{K}^h(k)\,\nabla\mathbf{H}^h(k), \\ \begin{bmatrix} \mathsf{P} & (\boldsymbol{p}^{\mathrm{h}})^T \\ \boldsymbol{p}^{\mathrm{h}} & \boldsymbol{P}^{\mathrm{h}} \end{bmatrix}_{k|k} = \begin{bmatrix} \mathsf{P} & \mathbf{J}^T (\boldsymbol{p}^{\mathrm{h}})^T \\ \mathbf{J}\,\boldsymbol{p}^{\mathrm{h}} & \mathbf{J}\,\boldsymbol{P}^{\mathrm{h}}\,\mathbf{J}^T \end{bmatrix}_{k|k-1} + \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{K}^h\,\mathbf{R}\,(\mathbf{K}^h)^T \end{bmatrix}_k. \tag{16}$$

Because the common state P is not updated, there is a computational improvement over the standard Kalman update.

Following the update stage, the hypotheses weights are updated and trimmed based on (11), in the same way as in full MHSLAM.

New beacons are appended to their respective hypothesis states in the same way as in regular MHSLAM, following the method in [18].

The CSF reduces the full MHSLAM storage and update complexity to $O(n(m+n))$ for storage and $O(n(m+n^2))$ for the update. Fig. 3 compares the storage requirements of full MHSLAM and the CSF for a typical scenario, showing how even for a small number of hypotheses the storage savings of the CSF over full MHSLAM become readily apparent. This efficiency comes at a penalty; there is an information loss associated with this simplification, however we have found this to be small in practice.

C. *Merging*

Once all except one hypothesis have been trimmed, the hypothesis state beacons are moved into the common state. If we consider the remaining hypothesis state c, then just prior to merging, the state and covariance appear as follows;

$$\hat{\mathbf{x}}(k|k) = \begin{bmatrix} \hat{\mathsf{x}}_b^T & \begin{bmatrix} \hat{\boldsymbol{x}}_v^T & \hat{\boldsymbol{x}}_b^T \end{bmatrix}^c \end{bmatrix}_{k|k}^T,$$

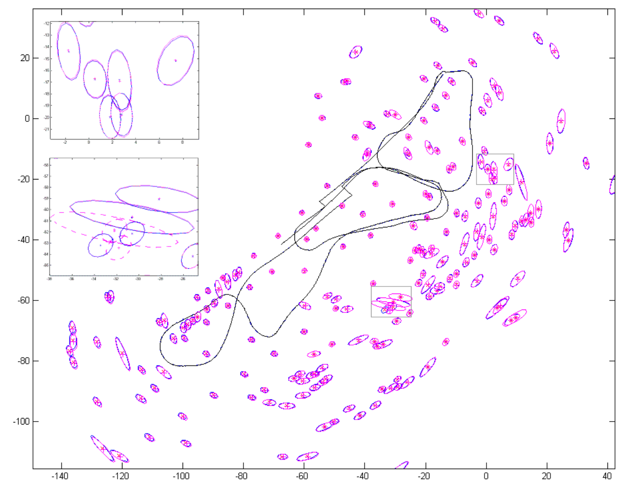


Fig. 4. Overview of the map and vehicle trajectories. The maps and trajectories created by full MHSLAM and the CSF have been overlaid, showing how they are almost identical. Note the vehicle trajectory being corrected at the end of each blackout. Close-ups show the selected regions in detail. Bottom: different beacon positions due to different data association decisions being made at this location. Top: the vast majority of the map shows almost identical agreement between the filters.

$$\mathbf{P}(k|k) = \begin{bmatrix} [\mathsf{P}_b] & \begin{bmatrix} \boldsymbol{p}_{\mathrm{bv}}^{\mathrm{T}} & \boldsymbol{p}_{\mathrm{bb}}^{\mathrm{T}} \end{bmatrix}^c \\ \begin{bmatrix} \boldsymbol{p}_{\mathrm{bv}} \\ \boldsymbol{p}_{\mathrm{bb}} \end{bmatrix}^c & \begin{bmatrix} \boldsymbol{P}_{\mathrm{v}} & \boldsymbol{P}_{\mathrm{vb}}^{\mathrm{T}} \\ \boldsymbol{P}_{\mathrm{vb}} & \boldsymbol{P}_{\mathrm{b}} \end{bmatrix}^c \end{bmatrix}_{k|k} .$$

Moving the vehicle to the top, the merged state is

$$\hat{\mathbf{x}}^{'}(k|k) = \begin{bmatrix} \hat{\boldsymbol{x}}_v^T & \hat{\mathsf{x}}_b^T & \hat{\boldsymbol{x}}_b^T \end{bmatrix}_{k|k}^T ,$$

and the covariance

$$\mathbf{P}^{'}(k|k) = \begin{bmatrix} \boldsymbol{P}_{\mathrm{v}} & \boldsymbol{p}_{\mathrm{bv}}^{\mathrm{T}} & \boldsymbol{P}_{\mathrm{vb}}^{\mathrm{T}} \\ \boldsymbol{p}_{\mathrm{bv}} & \mathsf{P}_b & \boldsymbol{p}_{\mathrm{bb}}^{\mathrm{T}} \\ \boldsymbol{P}_{\mathrm{vb}} & \boldsymbol{p}_{\mathrm{bb}} & \boldsymbol{P}_{\mathrm{b}} \end{bmatrix}_{k|k} .$$

V. Victoria Park Experiment

We compared the performance of the Common State and full MHSLAM filters on part of the popular Victoria Park data set [27], shown in Fig. 4. As we did not make any provisions for large maps (e.g. submapping), we ran each test for a total of 10 500 time steps (230s), this being sufficient to demonstrate the method. Because there are very few native data association problems in the data set [28], we created them by allowing the vehicle to run for 5000 time steps to build up the map, then discarding the sensor data for selected time periods, thus allowing motion error and data association ambiguity to build up. We compared the results with those obtained from full MHSLAM using all the available observations. The GPS data was always discarded, as its accuracy is questionable in this region of the map.

The blackouts occured at 5100, 6000, 7800 and 8900 time steps (with durations 700, 1500, 600 and 600 time steps respectively). These spacings were chosen as they caused the nearest neighbour approach to fail at the end of each blackout.

The vehicle was a pickup truck, described in [27], with motion and sensor specifications from personal correspondence with J. Guivant. The motion standard deviations

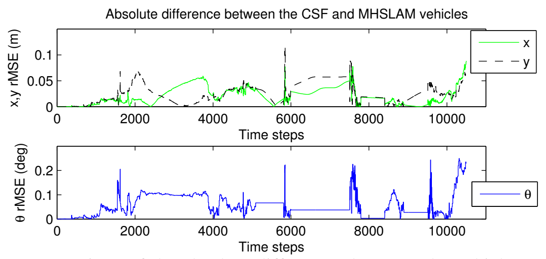


Fig. 5. Comparison of the absolute differences between the vehicle poses of full MHSLAM and the CSF. (Top) the x and y positions and (bottom) the orientation. During the blackouts the vehicle orientation difference remained constant, as the orientations were solely determined by the control inputs during this time.

(covariance $\mathbf{Q}(k)$) were 0.1m for the velocity and 3^o for the orientation, with additonal stabilising noise $\mathbf{Q}_s = \mathrm{diag}([\ 0.4 \quad 0.4 \quad 0.003\ ]).10^{-3}$ according to (5), where $\mathrm{diag}(\cdot)$ is the diagonal matrix. The observation range and angular standard deviations were 0.15m and 1.5^o respectively.

For comparing the agreement between the filters in Fig. 5, we compared the absolute difference between the representative CSF and full MHSLAM vehicles,$|\ \bar{\mathbf{x}}_v(k) - \bar{\boldsymbol{x}}_v(k)\ |$ over time, $\bar{\mathbf{x}}_v(k)$ and $\bar{\boldsymbol{x}}_v(k)$ being the weighted average poses from (12).

Fig. 5 shows that the CSF and full MHSLAM vehicle positions remain within 12cm of each other; often far less. In the final maps produced the beacon positions were all within 50cm (mean 10cm), and the covariances were very similar, as shown in Fig. 4. The main difference between the maps was the presence of a few (less than 5 in 216) spurious beacons in each filter that were not present in the other. These appear to be caused by an observation being incorrectly classified as a new beacon, though the effect on map quality is minimal; the maps shown in Fig. 4 are typical. These results show that the CSF and full MHSLAM give similar accuracy. It should be noted that in the absence of data from an ideal filter, we cannot infer the accuracy of full MHSLAM over the CSF due to the propagation of linearisation errors.

Fig. 6 shows that where there is more than one hypothesis, the storage requirements of the CSF are consistently lower than those of full MHSLAM. This is most clearly shown at the end of the blackouts (at 5800, 7500, 8400 and 9500 time steps).

Fig. 7 shows the theoretical update cost of full MHSLAM (9) and the CSF (16). In the initial stages there was mainly a single hypothesis, so both filters were performing a regular Kalman update. Following each of the 4 blackouts, when there were multiple hypotheses (as shown in Fig. 6), the computational cost of full MHSLAM far outweighed that of the CSF, whose net update cost actually decreased due to the partitioned update. The sum total FLOPs at the end of the run was 4.42 GFLOPs for full MHSLAM, and 1.55 GFLOPs for the CSF; almost a three-fold decrease. For comparison, an EKF using ICNN (with no blackouts) requires 2.21 GFLOPs (summed over the non-blackout parts of the run). These figures correspond well to the Matlab Profiler times, which for the total updates summed over each run were 140s for full MHSLAM and 40s for the CSF, on a 3 GHz dual-core

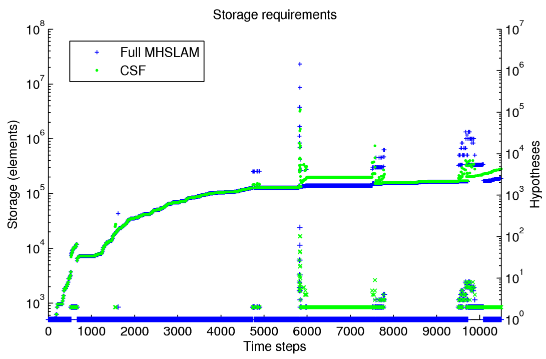


Fig. 6. (Top) Log plot of the number of elements required to store the covariance matrices for all active hypotheses (not exploiting symmetry) for full MHSLAM and the CSF. (Bottom) The number of hypotheses present. The spikes correspond to times at which there were ambiguous beacon observations requiring the creation of multiple hypotheses.

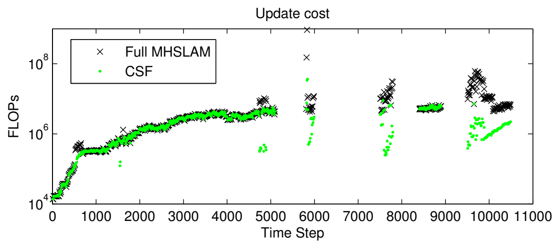


Fig. 7. Computational cost for full MHSLAM (9) and CSF (16) updates over the run (FLOPs). The empty spaces correspond to the sensor blackouts, when no updates were made. The summed FLOPs over the run were 4.42 and 1.55 GFLOPs for full MHSLAM and the CSF respectively.

machine using unoptimised code.

VI. Conclusion

We have presented the Common State filter, a computationally efficient method of performing MHT using the EKF. The performance is comparable to full MHSLAM, at a significant saving in computational storage and update complexity. The test using real data has shown that the information loss has a very small impact on the quality of the map produced. Considering the storage savings available, the CSF is a viable alternative to full MHSLAM in applications which maintain large states.

In further work we will consider the role of a common state vehicle, and the use of an information measure to move key beacons between the common and hypothesis states. We will investigate the performance using simulations in additon to real data. The aim is to both increase the robustness of the filter, whilst ensuring that the storage and computational improvements do not compromise the quality of the results produced.

References

[1] S. B. Williams, H. Durrant-Whyte, and G. Dissanayake, "Constrained Initialization of the Simultaneous Localization and Mapping Algorithm," *The International J. Robotics Research*, vol. 22, no. 7-8, pp. 541–564, 2003.

[2] A. Davison, "Real-time simultaneous localisation and mapping with a single camera," in *Computer Vision, 2003. Proc. Ninth IEEE International Conference on*, 2003, pp. 1403–1410 vol.2.

[3] U. Frese, "A discussion of simultaneous localization and mapping," *Auton. Robots*, vol. 20, no. 1, pp. 25–42, 2006.

[4] T. Bailey and H. Durrant-Whyte, "Simultaneous localisation and mapping (slam) part 2: State of the art," *Robotics and Automation Magazine*, 2006.

[5] J. Neira and J. Tardos, "Data association in stochastic mapping using the joint compatibility test," *Robotics and Automation, IEEE Trans.*, vol. 17, no. 6, pp. 890–897, 2001.

[6] S. Julier and J. Uhlmann, "Fusion of time delayed measurements with uncertain time delays," in *American Control Conference, 2005. Proc. the 2005*, J. Uhlmann, Ed., 2005, pp. 4028–4033 vol. 6.

[7] S. Ching and E. Davison, "Control of plants which change using switching controllers," in *American Control Conference, 2005. Proc. the 2005*, E. Davison, Ed., 2005, pp. 1181–1185 vol. 2.

[8] T. Lemaire, S. Lacroix, and J. Sola, "A practical 3d bearing-only slam algorithm," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 2449–2454.

[9] N. Kwok, G. Dissanayake, and Q. Ha, "Bearing-only slam using a sprt based gaussian sum filter," in *Robotics and Automation, 2005. Proc. the IEEE International Conference on*, 2005, pp. 1109–1114.

[10] N. Peach, "Bearings-only tracking using a set of range-parameterised extended kalman filters," *IEE Proc. - Control Theory and Applications*, vol. 142, no. 1, pp. 73–80, 1995.

[11] C. Smith, "Integrating mapping and navigation," Ph.D. dissertation, Massachusetts Institute of Technology, 1998.

[12] S. Julier, J. Uhlmann, and D. Nicholson, "A method for dealing with assignment ambiguity," in *American Control Conference, 2004. Proc. the 2004*, vol. 5, June-2 July 2004, pp. 4102–4107 vol.5.

[13] J. Williams and P. Maybeck, "Cost-function-based gaussian mixture reduction for target tracking," in *Information Fusion, 2003. Proc. the Sixth International Conference of*, vol. 2, 2003, pp. 1047–1054.

[14] D. Hähnel, W. Burgard, B. Wegbreit, and S. Thrun, "Towards lazy data association in SLAM," in *Proc. the 11th International Symposium of Robotics Research (ISRR'03)*. Sienna, Italy: Springer, 2003.

[15] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fastslam: A factored solution to the simultaneous localization and mapping problem," 2002.

[16] T. Bailey, J. Nieto, and E. Nebot, "Consistency of the fastslam algorithm," in *Robotics and Automation, 2006. ICRA 2006. Proc. 2006 IEEE International Conference on*, 2006, pp. 424–429.

[17] X.-R. Li and Y. Bar-Shalom, "Multiple-model estimation with variable structure," *Automatic Control, IEEE Trans.*, vol. 41, no. 4, pp. 478–493, Apr 1996.

[18] S. Julier and J. Uhlmann, "A counter example to the theory of simultaneous localization and map building," in *Robotics and Automation, 2001. Proc. 2001 ICRA. IEEE International Conference on*, vol. 4, 2001, pp. 4238–4243 vol.4.

[19] M. Bosse, P. Newman, J. Leonard, and S. Teller, "An atlas framework for scalable mapping," 2002.

[20] D. Alspach and H. Sorenson, "Nonlinear bayesian estimation using gaussian sum approximations," *Automatic Control, IEEE Trans.*, vol. 17, no. 4, pp. 439–448, 1972.

[21] M. Bryson and S. Sukkarieh, "Building a robust implementation of bearing-only inertial slam for a uav," *J. Field Robotics*, vol. 24, no. 1-2, pp. 113–143, 2007.

[22] N. Kwok and G. Dissanayake, "An efficient multiple hypothesis filter for bearing-only slam," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proc. 2004 IEEE/RSJ International Conference on*, vol. 1, 2004, pp. 736–741 vol.1.

[23] J. Sola, A. Monin, M. Devy, and T. Lemaire, "Undelayed initialization in bearing only slam," *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pp. 2499–2504, Aug. 2005.

[24] J. Knight, A. Davison, and I. Reid, "Towards constant time slam using postponement," 2001.

[25] S. F. Schmidt, "Application of state space methods to navigation problems," *Advanced Control Systems*, vol. 3, pp. 293–340, 1966.

[26] P. Y. C. Hwang and R. G. Brown, *Introduction to Random Signals and Applied Kalman Filtering*. J. Wiley, 1992.

[27] J. Guivant and E. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *Robotics and Automation, IEEE Trans.*, vol. 17, no. 3, pp. 242–257, 2001.

[28] 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," 2004.