Australian Government
Department of Defence
Defence Science and
Technology Organisation

Simultaneous Localisation and Map Building using the Probabilistic Multi-Hypothesis Tracker

Samuel Davey

Intelligence, Surveillance and Reconnaissance Division
Information Sciences Laboratory

DSTO–TR–1691

ABSTRACT

This report presents an algorithm for efficiently solving the Simultaneous Localisation and Map Building (SLAM) problem. The SLAM problem requires both the dynamic estimation of the sensor location and the tracking of features of interest in the environment using the sensor measurements. The problem is difficult because the unknown sensor and feature locations are coupled through the sensor measurement. It has been shown that under linear Gaussian conditions, a Kalman Filter solution converges to a solution relative to the unknown starting location. However, this approach does not scale well with the number of features in the scene, and is unfeasible for large maps. The algorithm introduced here is based on the Probabilistic Multi-Hypothesis Tracker (PMHT) and exploits a factorisation of the problem to reduce the computational requirements of the Kalman Filter approach. The new algorithm is demonstrated on a benchmark data set recorded in Victoria Park.

Published by

DSTO Information Sciences Laboratory
PO Box 1500
Edinburgh, South Australia, Australia 5111

Telephone: (08) 8259 5555
Facsimile: (08) 8259 6567

AR No. AR 013-343
March, 2005

Simultaneous Localisation and Map Building using the Probabilistic Multi-Hypothesis Tracker

EXECUTIVE SUMMARY

This report introduces a new algorithm for solving the feature-based Simultaneous Localisation and Map Building problem (SLAM). In feature-based SLAM, one or more platforms move through an unknown environment containing stationary objects (landmarks) that can be observed with onboard sensors. The aim is to simultaneously estimate the position and orientation of the platform, while producing a map of the landmarks. SLAM provides a method for frame-to-frame registration of sensor observations when navigation information is lacking, or not of high enough accuracy.

The standard approach to feature-based SLAM is to stack all of the unknowns into a system state vector, and then use a Kalman Filter to estimate the state values. However, this state vector becomes very large as the number of landmarks increases, and the algorithm resource requirements scale as the square of the number of unknowns. This makes the direct Kalman Filter approach infeasible for large maps. In addition, the Kalman Filter approach assumes that the association of measurements to landmarks is known, and is only compatible with simple data association methods.

The algorithm presented here is based on the Probabilistic Multi-Hypothesis Tracker (PMHT), which is a multi-target tracking approach that provides good scaling properties with the problem size. By generalising the standard PMHT, an algorithm is found that scales linearly with the number of landmarks in the map, while offering performance at least as good as the Kalman Filter approach.

The performance of the PMHT SLAM algorithm is analysed using simulated data and compared with that of the Kalman Filter approach. PMHT incorporates a kind of probabilistic data association, and this is demonstrated to give significant performance improvements when measurement errors are high. The performance of the algorithm is also presented for a benchmark SLAM data set recorded by the University of Sydney. The algorithm output is shown to be comparable with that achieved by state-of-the-art algorithms, even when navigation measurements are denied.

Contents

Figures

Notation

k_{tnr}	assignment index for the rth measurement from sensor n at scan t (the true measurement source)
$\mathbf{K}_{tn} = \{k_{tnr}\}_{r=1}^{\eta_t^n}$	all assignments for sensor n at scan t
$\mathbf{K}_t = \{\mathbf{K}_{tn}\}_{n=1}^{N}$	all assignments at scan t
$\mathbf{K} = \{\mathbf{K}_t\}_{t=1}^{T}$	all assignments
M	number of landmarks (targets) in world
M_{FOV}	number of landmarks in field-of-view
N	number of sensors
$Q(\cdot)$	EM auxiliary function
T	number of scans (batch length)
$\boldsymbol{x}_t^m$	state of landmark m at scan t
$\mathbf{X}^m = \{\mathbf{X}_t^m\}_{t=1}^{T}$	all landmark of target m
$\mathbf{X} = \{\mathbf{X}^m\}_{m=1}^{M}$	all landmark states
$\boldsymbol{y}_t^n$	state of sensor n at scan t
$\mathbf{Y}^n = \{\boldsymbol{y}_t^n\}_{t=1}^{T}$	all states of sensor n
$\mathbf{Y} = \{\mathbf{Y}^n\}_{n=1}^{N}$	all sensor states
$\boldsymbol{z}_{tnr}$	the rth measurement at scan t from sensor n
$\mathbf{Z}_{tn} = \{\boldsymbol{z}_{tnr}\}_{r=1}^{\eta_t^n}$	all measurements at scan t from sensor n
$\mathbf{Z}_t = \{\mathbf{Z}_{tn}\}_{n=1}^{N}$	all measurements at scan t
$\mathbf{Z} = \{\mathbf{Z}_t\}_{t=1}^{T}$	all measurements
$\zeta_t^n\left(\boldsymbol{z}_{tr}\|\boldsymbol{y}_t^n, \boldsymbol{x}_t^{k_{tr}}\right)$	measurement pdf
η_t^n	number of observations at from sensor n at scan t
π_{tn}^m	assignment prior, $P(k_{tnr} = m)$
$\mathbf{\Pi}$	set of all assignment priors
$\phi_t^0\left(\boldsymbol{y}_0^n\right)$	the state prior for sensor n
$\phi_t^n\left(\boldsymbol{y}_t^n\|\boldsymbol{y}_{t-1}^n\right)$	the state evolution pdf for sensor n
$\psi_t^0\left(\boldsymbol{x}_0^m\right)$	the state prior for target m
$\psi_t^m\left(\boldsymbol{x}_t^m\|\boldsymbol{x}_{t-1}^m\right)$	the state evolution pdf for target m
$\hat{\cdot}\,(i)$	estimate calculated on the ith EM iteration

1 Introduction

The standard target tracking problem is to estimate the location (and perhaps other parameters) of scatterers in the sensor field of view using sensor measurements. It is usually assumed that the sensor location and orientation are known with sufficient accuracy [Blackman & Popoli 1999]. A notable exception is the registration problem, when multiple sensors are fused together. In registration, the sensor position and orientation are known to within a small error, and this error is estimated by comparing measurements or tracks on common targets (e.g. [Bar-Shalom & Blair 2000]). Navigation is the reverse of the tracking problem. Here the locations of the features are known (the sensor has a map) and the dynamic sensor position is estimated (e.g. [Bar-Shalom et al. 2001]).

The problem where both the sensor position and the feature (target) locations are to be estimated is referred to as Simultaneous Localisation and Map Building (SLAM) or Concurrent Map Building and Localisation (CML). The inherent difficulty in the SLAM problem is that the sensor data is generally a nonlinear function of the relative difference between the sensor and feature positions. This coupling of the unknowns through the observation process necessitates a joint estimation scheme.

Numerous methods have been used to address the SLAM problem. [Thrun 2002] gives a good overview of mapping and SLAM approaches. This report will be concerned with feature-based SLAM where the environmental map is composed of a set of discrete stationary objects (landmarks). The benchmark method for solving this problem is to use a Kalman Filter with a state vector consisting of the platform location and all of the landmark locations stacked together. When the system is linear and the random elements are Gaussian, this is an optimal approach, and it has been shown that the map derived converges to a relative map with zero uncertainty [Dissanayake et al. 2001]. However, the approach does not scale well with large maps. If there are M landmarks, then the length of the state vector is proportional to M and the covariance matrix is proportional to M^2 (ignoring the contribution of the sensor state). This means that the algorithm's memory requirement grows as M^2 and its computation requirement grows at least as M^2 [1]. For this reason, the stacked state vector approach is infeasible for large maps (large M).

One solution to the complexity of the stacked state vector approach is to update only those landmarks within the locale of the platform for most measurements. The information about other landmarks garnered from these measurements comes only through improved knowledge of the platform, and can be incorporated using global updates at a slower rate. An algorithm based on this principle is presented in [Guivant & Nebot 2001*a*]. This method still has quadratic complexity in the number of landmarks inside the field of view, but it is somewhat insensitive to the total number of landmarks in the world. In contrast, the standard stacked state vector has quadratic complexity in the total number of landmarks. If the number of landmarks currently in view is much smaller than the total number (which is the case for many problems) then this represents a considerable saving.

The stacked state vector methods explicitly assume that the association of measurements to landmarks is known. In practice, this means that nearest neighbour association would be used. Other data association approaches, such as probabilistic data association, or Multi-Hypothesis Tracking could be adapted, but given that the problem generally has a large number of landmarks,

[1]Multiplications involving the covariance matrix will be order M^3 unless the special structure of the matrices involved is exploited.

and the computation cost with even nearest neighbours is prohibitive, it is unlikely that these approaches could be feasibly implemented.

An alternative approach to stacking the state vector is to exploit the Bayesian factorisation of the problem. Conditioned on the sensor location, the landmark estimation problem is independent for the different landmarks. This allows parallel filtering, which is hence linear in the number of landmarks. The difficulty then becomes tracking the sensor position. One algorithm based on this approach is referred to as *FastSLAM* [Montemerlo et al. 2002, Montemerlo et al. 2003, Montemerlo 2003]. FastSLAM uses a particle filter to track the sensor location, and parallel Extended Kalman Filters to track the landmarks. The particle filter is a Monte Carlo based estimation technique [Doucet et al. 2001]. The sensor pdf is represented by a collection of M^p random samples, referred to as particles. For FastSLAM, each particle performs parallel EKFs for each landmark. Since the conditional pdf of the landmarks is tracked, no information about a landmark is gained unless it is observed. FastSLAM uses this fact to reduce the work it needs to do for the unobserved landmarks. FastSLAM claims a complexity of $\log_2 M$ and $M^p M_\alpha$, where M_{FOV} is the number of landmarks currently in the sensor filed of view. If $M_{FOV} \ll M$ and N is not too large, then FastSLAM is more efficient than the Kalman Filter approach. Although the computation requirements scale well, they remain quite high [Guivant & Nebot 2001b].

Besides the computational aspects of FastSLAM, there are other advantages of the approach. The particle filter it uses is not based on linear Gaussian assumptions, and is a preferred method for nonlinear non-Gaussian filtering (e.g. [Ristic et al. 2004]). In applications where the sensor passes close to landmarks, such as with ground vehicles, the nonlinearity may be high, and the Extended Kalman Filter may not be adequate for the problem. FastSLAM also has an in-built data association ability by using the particles to sample the different possible assignment combinations. Thus the sensor filter uses a Monte Carlo approximation for the data association problem as well as the nonlinear filtering.

This report will present a different algorithm, similarly based on the Bayesian factorisation of the SLAM problem. This algorithm uses Probabilistic Multi-Hypothesis Tracking to achieve a complexity only linear in the number of targets, and uses probabilistic data association. The Probabilistic Multi-Hypothesis Tracker (PMHT) of [Streit & Luginbuhl 1995] is an algorithm derived from the application of the Expectation Maximisation (EM) of [Dempster et al. 1977] to the tracking problem. The standard PMHT models the assignment of measurements to targets as hidden variables (nuisance parameters) and estimates target states by taking the expectation over the assignments. Unlike other tracking approaches, PMHT assumes that the assignment of each measurement is an independent realisation of a random process. This assumption allows the PMHT to be optimally realised with linear complexity in the number of targets (i.e. M_{FOV}). PMHT is a data association approach, and the estimation scheme used depends on the problem. For linear Gaussian statistics, the PMHT can be implemented as a bank of parallel Kalman Filters. For nonlinear problems, particle filters have been used.

This report demonstrates how a generalisation of the PMHT can be used to realise a solution for the multi-vehicle SLAM problem. This solution can be implemented as an iterative scheme, alternating between a bank of parallel filters for the landmarks, and a bank of parallel filters for the sensor platforms. For the application considered, there is only a single platform, and the PMHT algorithm is realised with $M_{FOV} + 1$ independent EKFs.

The remainder of the report is set out as follows. Section 2 defines the multi-vehicle SLAM problem, and summarises the stacked state vector and FastSLAM methods for solving it. Section 3 derives a PMHT algorithm for multi-vehicle SLAM. Section 4 compares the PMHT algorithm with the stacked state vector EKF and FastSLAM using simulations. The PMHT algorithm is applied to a benchmark data set recorded in Victoria Park in section 5. Section 6 presents a summary.

2 Problem Definition

Consider the multi-vehicle SLAM problem where there are N sensors observing M landmarks. Not all landmarks are necessarily visible to all sensors at any particular time. At discrete epochs, the sensors observe their environment, but the time between epochs is not necessarily regular, and the sensors may or may not make observations at the same time. The PMHT is derived over a batch of data, but it can be implemented as a sliding window, or a recursive estimator, depending on the requirements of the application. Each of these latter forms is simply obtained from the batch processor, and so a batch will be assumed.

Suppose that there are T observation times (scans). The time of the tth scan will be denoted τ_t and the number of observations from sensor n as η_t^n. η_t^n may be zero. Let $\boldsymbol{z}_{tnr}$ denote the rth observation from sensor n at scan t, and $\mathbf{Z}_{tn}$ be the set of all observations from sensor n at scan t. If sensor n does not collect observations at time τ_t, then $\eta_t^n = 0$ and $\mathbf{Z}_{tn} = \emptyset$, the empty set. $\mathbf{Z}_t$ is the set of all measurements at scan t, and $\mathbf{Z}$ is the grand collection of all of the observed data.

Let the state of sensor n at scan t be denoted by $\boldsymbol{y}_t^n$. The set of states for all scans is denoted as $\mathbf{Y}^n$, and the set of all sensor states as $\mathbf{Y}$. Similarly, let the state of landmark m at scan t be denoted by $\boldsymbol{x}_t^m$. The set of landmark states for all scans is denoted as $\mathbf{X}^m$, and the set of all landmark states as $\mathbf{X}$.

Let k_{tnr} denote the true source of measurement $\boldsymbol{z}_{tnr}$. The index k_{tnr} gives the landmark that gave rise to measurement $\boldsymbol{z}_{tnr}$, thus $k_{tnr} \in 1 \ldots M$. k_{tnr} is referred to as the assignment index and is unknown (or else data association is unnecessary). The sets $\mathbf{K}_{tn}$, $\mathbf{K}_t$, and $\mathbf{K}$ are defined as above.

Assume that the prior distribution of the state of each sensor is known, and is given by $\phi_t^0\left(\boldsymbol{y}_0^n\right)$ for sensor n. Similarly, the known prior distribution of each landmark is given by $\psi_t^0\left(\boldsymbol{x}_0^m\right)$ for landmark m. If these are unknown, then it is assumed that an estimation scheme exists based on measurements (a la track initiation).

The sensor dynamics are described by the evolution pdf $\phi_t^n\left(\boldsymbol{y}_t^n | \boldsymbol{y}_{t-1}^n\right)$, and the landmark dynamics by $\psi_t^m\left(\boldsymbol{x}_t^m | \boldsymbol{x}_{t-1}^m\right)$. Both of these are assumed known.

Finally, the observation process is described by a known measurement pdf that is denoted as $\zeta_t^n\left(\boldsymbol{z}_{tnr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^m, k_{tnr} = m\right)$. This pdf may be different for each sensor.

2.1 Stacked State Vector Solution

The standard Kalman Filter solution is based on stacking the state vectors of the various unknown objects. This approach was first introduced in [Smith et al. 1990].

The problem described above contains M unknown landmark state vectors and N unknown sensor positions. Let the system state be defined as

$$\Xi_t = \left[\boldsymbol{y}_t^{1\mathrm{T}}, \ldots \boldsymbol{y}_t^{N\mathrm{T}}, \boldsymbol{x}_t^{1\mathrm{T}}, \ldots \boldsymbol{x}_t^{M\mathrm{T}}\right]^{\mathrm{T}}. \tag{1}$$

That is, Ξ_t is the stacked vector of all the states at scan t.

Assume that the assignments are known, and that the random processes are linear and Gaussian. That is,

$$\begin{aligned}
\phi_0^n\left(\boldsymbol{y}_0^n\right) &\sim N\left(\bar{\boldsymbol{y}}_0^n, \mathsf{P}_0^{y,n}\right), & (2)\\
\phi_t^n\left(\boldsymbol{y}_t^n|\boldsymbol{y}_{t-1}^n\right) &\sim N\left(\mathsf{F}_t^{y,n}\boldsymbol{y}_{t-1}^n, \mathsf{Q}_t^{y,n}\right), & (3)\\
\psi_0^m\left(\boldsymbol{x}_0^m\right) &\sim N\left(\bar{\boldsymbol{x}}_0^m, \mathsf{P}_0^{x,m}\right), & (4)\\
\psi_t^m\left(\boldsymbol{x}_t^m|\boldsymbol{x}_{t-1}^m\right) &\sim N\left(\mathsf{F}_t^{x,m}\boldsymbol{x}_{t-1}^m, \mathsf{Q}_t^{x,m}\right), & (5)\\
\zeta_t^n\left(\boldsymbol{z}_{tnr}|\boldsymbol{y}_t^n, \boldsymbol{x}_t^m, k_{tnr}=m\right) &\sim N\left(\mathsf{H}_t^{y,n}\boldsymbol{y}_t^n + \mathsf{H}_t^{x,n}\boldsymbol{x}_t^m, \mathsf{R}_t^n\right), & (6)
\end{aligned}$$

where $N(\mu, \Sigma)$ is a normal density function with mean μ and covariance matrix Σ.

The evolution pdf for the system state is then also linear and Gaussian, given by

$$p\left(\Xi_t|\Xi_{t-1}\right) \sim N\left(\mathcal{F}_t\Xi_{t-1}, \mathcal{Q}_t\right), \tag{7}$$

where $\mathcal{F}_t$ and $\mathcal{Q}_t$ are block diagonal matrices, e.g.

$$\mathcal{F}_t \equiv \begin{bmatrix}
\mathsf{F}_t^{y,1} & 0 & & \cdots & & 0\\
0 & \ddots & \ddots & & & \\
 & \ddots & \mathsf{F}_t^{y,N} & 0 & & \vdots\\
\vdots & & 0 & \mathsf{F}_t^{x,1} & \ddots & \\
 & & & \ddots & \ddots & 0\\
0 & & \cdots & & 0 & \mathsf{F}_t^{x,M}
\end{bmatrix}.$$

Similarly the system prior is given by

$$p\left(\Xi_0\right) \sim N\left(\bar{\Xi}_0, \mathcal{P}_0\right), \tag{8}$$

where $\bar{\Xi}_0$ is the stacked vector of the prior means, and $\mathcal{P}_0$ is the block diagonal prior covariance matrix.

The mean of the measurement pdf for measurement $\boldsymbol{z}_{tnr}$ can be written as

$$\mathsf{H}_t^{y,n}\boldsymbol{y}_t^n + \mathsf{H}_t^{x,n}\boldsymbol{x}_t^m = \mathsf{H}_{tnr}\Xi_t, \tag{9}$$

where H_{tnr} is a sparse matrix with $\mathsf{H}_t^{y,n}$ and $\mathsf{H}_t^{x,n}$ positioned appropriately

$$\mathsf{H}_{tnr} = \left[\ldots 0 \ldots \mathsf{H}_t^{y,n} \ldots 0 \ldots \mathsf{H}_t^{x,n} \ldots 0 \ldots\right].$$

A stacked measurement matrix can now be constructed

$$\mathcal{H}_t = \begin{bmatrix}
\mathsf{H}_{t11}\\
\vdots\\
\mathsf{H}_{t1\eta_t^1}\\
\mathsf{H}_{t21}\\
\vdots\\
\mathsf{H}_{tN\eta_t^N}
\end{bmatrix}, \tag{10}$$

and a stacked measurement vector

$$\mathcal{Z}_t = \left[z_{t11}{}^{\mathrm{T}}, \ldots z_{t1\eta_t^1}{}^{\mathrm{T}}, z_{t21}{}^{\mathrm{T}}, \ldots z_{tN\eta_t^N}{}^{\mathrm{T}} \right]^{\mathrm{T}}, \tag{11}$$

so that the probability of all of the measurements in scan t is

$$p\left(\mathbf{Z}_t | \mathbf{Y}_t, \mathbf{X}_t, \mathbf{K}_t\right) = p\left(\mathcal{Z}_t | \Xi_t, \mathbf{K}_t\right) \sim N\left(\mathcal{H}_t \Xi_t, \mathcal{R}_t\right), \tag{12}$$

where $\mathcal{R}_t$ is a block diagonal matrix made up of the covariances corresponding to each individual measurement.

It should now be clear that a Kalman Filter can be implemented to estimate Ξ_t from $\mathcal{Z}_t$. Under the linear Gaussian conditions, this will be an optimal estimator.

The primary difficulty with this approach is that the solution does not scale well with the problem size. The length of the system state vector, Ξ, grows linearly with the number of landmarks, M. This means that the covariance matrix used by the Kalman Filter will grow as M^2. The sparseness of the matrices $\mathcal{F}_t$ and$\mathcal{Q}_t$ can be exploited to reduce the required computation overhead, but the system covariance matrix will not be sparse because the measurement process couples the states. For problems with a large number of landmarks, this approach becomes infeasible.

2.1.1 Data Association

The above analysis assumed that the assignment of measurements to landmarks was known, i.e. k_{tnr} is known. This will usually not be true. To relax this assumption, an association process such as global nearest neighbours can be used. The algorithm requires hard assignments, so it may perform poorly in ambiguous situations.

2.1.2 Nonlinearity

The obvious way to extend the Kalman Filter approach for nonlinear problems is to use an Extended Kalman Filter. This replaces the true state evolution and measurement functions with linear approximations based on a truncated Taylor series expansion. The matrices $\mathcal{F}_t$ and $\mathcal{H}_t$ are then formed using Jacobians. The EKF is known to perform poorly for highly nonlinear systems, and may diverge.

Alternatively, the stacked state vector problem could be tackled with a nonlinear filter, such as a particle filter. However, particle filters are not well suited to problems with a high dimensional state, because one needs to use a large number of particles to populate the state space.

2.1.3 Large Maps

When the number of landmarks is large, the Kalman Filter approach is infeasible. One solution is to only update a subset of the landmarks in the locale of the sensor [Guivant & Nebot 2001*a*]. When the sensor moves to a new locale, the measurements not included for these landmarks can be lumped together and the map corrected.

2.2 FastSLAM

The FastSLAM[2] approach of [Montemerlo 2003] is based on the exact Bayesian factorisation of the problem,

$$P(\mathbf{X},\mathbf{Y}|\mathbf{Z}) = P(\mathbf{Y}|\mathbf{Z}) \prod_{m=1}^{M} P(\mathbf{X}^m|\mathbf{Y},\mathbf{Z}). \tag{13}$$

The landmark terms on the right hand side can each be solved with independent EKFs, but the sensor term is more difficult. FastSLAM uses a particle filter approach to approximate this. To the author's knowledge, FastSLAM has only been applied to single platform problems.

There are two published versions of the FastSLAM algorithm, referred to as *FastSLAM1.0* [Montemerlo et al. 2002] and *FastSLAM2.0* [Montemerlo et al. 2003] respectively. These algorithms are substantially the same, except that the FastSLAM2.0 uses a different proposal distribution. Under FastSLAM2.0, each particle does more work, but the algorithm can get by with fewer particles. In the limit, as the number of particles becomes large, the two are equivalent. [Montemerlo 2003] gives a comparison of the two variants, and demonstrates under which conditions one is preferred over the other.

2.2.1 Data Association

FastSLAM naturally incorporates probabilistic data association. It does this by sampling the possible association combinations with the particles. So, the collection of particles spans the set of assignment hypotheses, and it is hoped that the particles that survive are those carrying the correct hypotheses.

2.2.2 Nonlinearity

The heart of FastSLAM is a particle filter for estimating the platform state. The particle filter is a natural choice for nonlinear problems, and it should be anticipated that FastSLAM would offer an advantage in this type of problem.

2.2.3 Large Maps

The landmark estimates are conditional estimates given the platform state. Since each particle has (at least potentially) a different state value, each particle must carry out its own landmark estimation. However, since the landmarks are not dynamic, there is no prediction, and the conditional density only changes when a landmark is observed, so the algorithm needs to do $M^p \times M_{FOV}$ landmark updates, where M^p is the number of particles and M_{FOV} is the number of landmarks in the field of view. This means that there are $M - M_{FOV}$ landmarks that are not updated for each particle.

After the update, each particle has a weight, and the weighted combination of the particles is the pdf estimate. It turns out that particles will become degenerate and have zero weight unless

[2]This section will have to be evolutionary as my understanding of this algorithm improves

resampling is done. In resampling, the new set of particles is drawn from the updated set using the weights as a pmf for the selection. This means that the filter has to do M^p particle copies during resampling. Results presented in [Montemerlo 2003] demonstrate that the particle copy process actually limits the performance of the algorithm since each particle contains M state estimates. It turns out that many of the landmark estimates that are out of the current field of view will be the same because the resampling causes the higher weight particles to be selected more frequently. This means that some of the resampling copy process is creating duplicate copies of the same data which is wasteful of memory and computation. By using pointers to trees of landmark estimates, FastSLAM has an efficient method for handling these out of sight landmarks. [Montemerlo 2003] argues that the complexity of this scheme is logarithmic in M since the binary tree used to store landmark estimates has depth $\log_2 M$. So, assuming that $M_{FOV} \ll M$, then the computation time of the algorithm is limited by $M^p \log_2 M$.

Arriving at a complexity only logarithmic in the number of landmarks seems remarkable (less than linear!) but this conclusion is a little deceptive. The algorithm must do M^p particle updates each of which consists of a platform prediction (requiring only a random number draw), M_{FOV} local landmark updates, a weight calculation, and M resamplings, each of which is heuristically $\log_2 M$ in complexity. In comparison the stacked state vector using only local landmarks needs to assign measurements and perform a M_{FOV} stacked state EKF. The particle filter weight calculation requires $M^p * M_{FOV}$ pdf evaluations, each of which is expensive. Even considering only the landmark updates, the particle filter uses $M^p * M_{FOV}$ EKFs whereas the stacked EKF uses one of size M_{FOV}. Although the single stacked EKF is costly, FastSLAM does numerous small filters. Overall, the stacked state vector on the local area is independent of M except on global updates, and FastSLAM is not. So, it appears that the low complexity claims of FastSLAM are overstated. However, FastSLAM does have advantages for data association and nonlinear problems.

3 Probabilistic Multi-Hypothesis Tracker for SLAM

The fundamental philosophy behind the PMHT approach to SLAM presented here is an iterative application of partial Expectation Maximisation (EM). This will be explained by first briefly explaining EM for the simpler tracking problem, where $\mathbf{Y}$ is known (i.e. the standard PMHT).

The task is to estimate the target states, $\mathbf{X}$, given the observed data $\mathbf{Z}$. However, the assignment of the measurements, $\mathbf{K}$, is unknown. In an EM context, the assignments are treated as *missing data* or *nuisance parameters*; it is unimportant what $\mathbf{K}$ are, except that it appears one must solve for the joint posterior $P(\mathbf{X},\mathbf{K}|\mathbf{Z})$ to obtain the optimal $\mathbf{X}$. EM theory applied to this problem defines an auxiliary function

$$Q(X|\hat{\mathbf{X}}) = \sum_{\mathbf{K}} P(\mathbf{K}|\hat{\mathbf{X}},\mathbf{Z}) \log P(\mathbf{X},\mathbf{K},\mathbf{Z}) \tag{14}$$

that is the expectation of the joint likelihood over the assignments. The probability of the assignments is evaluated at some guess of the states, $\hat{\mathbf{X}}$. One finds the state $\mathbf{X}$ that maximises $Q(X|\hat{\mathbf{X}})$ and this becomes an improved guess of the states. EM theory states that the iterative application of this process converges to the same $\mathbf{X}$ that optimises $P(\mathbf{X},\mathbf{K}|\mathbf{Z})$, subject to local maxima. This is useful if $Q(X|\hat{\mathbf{X}})$ is easy to maximise, which it is for the tracking case.

In the case of SLAM, the problem has an additional unknown, $\mathbf{Y}$, the sensor states. So the direct application of PMHT would require the joint estimation of $\mathbf{X}$ and $\mathbf{Y}$, as with the Kalman Filter method. However, if partial EM is used instead, the problem is much simpler. Partial EM is where one seeks not to estimate all of the states, but to estimate some of them, and keep the others fixed. If $\mathbf{Y}$ is kept fixed, it is equivalent to assuming that the current estimated sensor trajectory is the true trajectory, and then doing the standard tracking problem. One iteration of the EM for this would be

- Calculate the probability of the assignments given the assumed sensor trajectory, $\hat{\mathbf{Y}}$, and the current landmark state estimates $\hat{\mathbf{X}}_{i-1}$. This defines the auxiliary function.
- Determine a new landmark state estimate $\hat{\mathbf{X}}_i$ by optimising the auxiliary function. This can be done in parallel for each landmark - the auxiliary function contains independent terms.

A similar EM process can be obtained by only updating the sensor states. Here the landmark positions are assumed known. For this case, the sensor states can be independently estimated by a bank of filters. By combining the two partial updates, an iterative scheme is achieved which will converge, subject to local maxima. The algorithm is thus of the form

- Choose an initial guess of the landmark positions, $\hat{\mathbf{X}}_0$, and the sensor trajectories, $\hat{\mathbf{Y}}_0$.
- Calculate the assignment probabilities to determine the auxiliary function.
- Refine the sensor trajectory estimates to find $\hat{\mathbf{Y}}_i$ by maximising the auxiliary function with landmarks fixed at $\mathbf{X} = \hat{\mathbf{X}}_{i-1}$.
- Calculate the assignment probabilities to determine the auxiliary function.
- Refine the landmark estimates to find $\hat{\mathbf{X}}_i$ by maximising the auxiliary function with sensors fixed at $\mathbf{Y} = \hat{\mathbf{Y}}_{i-1}$.
- Repeat 2 . . . 5 until convergence.

3.1 The Algorithm

Assume that all state sequences are independent of each other, and all measurements are conditionally independent, given the relevant states. Assume that all measurements from the same sensor are identically distributed given the relevant states. Assume that the assignments are IID random variables from a prior distribution $\pi_t^m = P(k_{tr} = m)$.

Due to the independence assumptions, the complete data likelihood is given by:

$$P(\mathbf{X}, \mathbf{Y}, \mathbf{K}, \mathbf{Z}) = P(\mathbf{X})P(\mathbf{Y})P(\mathbf{K})P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K}) \tag{15}$$

where,

$$P(\mathbf{Y}) = \prod_{n=1}^{N} \phi_t^0 (\boldsymbol{y}_0^n) \prod_{t=1}^{T} \phi_t^n \left(\boldsymbol{y}_t^n | \boldsymbol{y}_{t-1}^n\right) \tag{16}$$

$$P(\mathbf{X}) = \prod_{m=1}^{M} \psi_t^0 (\boldsymbol{x}_0^m) \prod_{t=1}^{T} \psi_t^m \left(\boldsymbol{x}_t^m | \boldsymbol{x}_{t-1}^m\right) \tag{17}$$

$$P(\mathbf{K}) = \prod_{t=1}^{T} \prod_{r=1}^{n_t} \pi_t^{k_{tr}} \tag{18}$$

$$P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K}) = \prod_{t=1}^{T} \prod_{r=1}^{n_t} \zeta_t^n \left(\boldsymbol{z}_{tr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^{k_{tr}}\right) \tag{19}$$

So,

$$\log P(\mathbf{X}, \mathbf{Y}, \mathbf{K}, \mathbf{Z}) = \log P(\mathbf{X}) + \log P(\mathbf{Y}) + \log P(\mathbf{K}) + \log P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K}) \tag{20}$$

where each of the terms to the right above is itself a sum of terms due to the factorised nature of the problem.

The assignments are missing data, so the EM auxiliary function requires $P(\mathbf{K}|\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ which can be found using Bayes' Rule and some simple manipulations:

$$\begin{aligned}
P(\mathbf{K}|\mathbf{X}, \mathbf{Y}, \mathbf{Z}) &= \frac{P(\mathbf{X}, \mathbf{Y}, \mathbf{K}, \mathbf{Z})}{P(\mathbf{X}, \mathbf{Y}, \mathbf{Z})} \\
&= \frac{P(\mathbf{X}, \mathbf{Y}, \mathbf{K}, \mathbf{Z})}{\sum_{\mathbf{K}} P(\mathbf{X}, \mathbf{Y}, \mathbf{K}, \mathbf{Z})} \\
&= \frac{P(\mathbf{X})P(\mathbf{Y})P(\mathbf{K})P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K})}{P(\mathbf{X})P(\mathbf{Y}) \sum_{\mathbf{K}} P(\mathbf{K})P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K})} \\
&= \frac{P(\mathbf{K})P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K})}{\sum_{\mathbf{K}} P(\mathbf{K})P(\mathbf{Z}|\mathbf{X}, \mathbf{Y}, \mathbf{K})} \\
&= \frac{\prod_{t=1}^{T} \prod_{r=1}^{n_t} \pi_t^{k_{tr}} \zeta_t^n \left(\boldsymbol{z}_{tr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^{k_{tr}}\right)}{\sum_{\mathbf{K}} \prod_{t=1}^{T} \prod_{r=1}^{n_t} \pi_t^{k_{tr}} \zeta_t^n \left(\boldsymbol{z}_{tr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^{k_{tr}}\right)}
\end{aligned}$$

$$= \prod_{t=1}^{T} \prod_{r=1}^{n_t} \frac{\pi_t^{k_{tr}} \zeta_t^n \left(\boldsymbol{z}_{tr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^{k_{tr}} \right)}{\sum_{m=1}^{M} \pi_t^m \zeta_t^n \left(\boldsymbol{z}_{tr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^m \right)} \tag{21}$$

$$\equiv \prod_{t=1}^{T} \prod_{r=1}^{n_t} w_{k_{tr}tr} \tag{22}$$

So, the conditional probability of the assignments is given by the product of individual per measurement terms. In common PMHT parlance, the w_{mtr} is referred to as the weight for measurement $\boldsymbol{z}_{tr}$ and target model m. Notice that this weight is simply a likelihood ratio evaluated using the current state estimates.

Combining all this leads to the auxiliary function

$$\begin{aligned} Q(\mathbf{X}, \mathbf{Y} | \mathbf{X}^i, \mathbf{Y}^i) &= \sum_{\mathbf{K}} P(\mathbf{K} | \mathbf{X}, \mathbf{Y}, \mathbf{Z}) \log P(\mathbf{X}, \mathbf{Y}, \mathbf{K}, \mathbf{Z}) \\ &= \log P(\mathbf{X}) + \log P(\mathbf{Y}) + \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{r=1}^{n_t} w_{mtr} \log \pi_t^m \\ &\quad + \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{r=1}^{n_t} w_{mtr} \log \zeta_t^n \left(\boldsymbol{z}_{tr} | \boldsymbol{y}_t^n, \boldsymbol{x}_t^m \right) \end{aligned} \tag{23}$$

Notice that the measurement term is composed of pairs of states, each with one sensor and one target, coupled together by the measurement function. Thus (at worst) all the targets and all the sensors are coupled together through this term. However, none of the targets are coupled with each other, and none of the sensors are coupled with each other. This is pretty much the standard PMHT result which allows the normal algorithm to be achieved with linear complexity in the number of targets. So, if $\mathbf{X}$ were known, $\mathbf{Y}$ could be estimated with independent filters for each sensor. Similarly, if $\mathbf{Y}$ were known, $\mathbf{X}$ could be estimated with independent filters for each target. These are the standard navigation and tracking (mapping) functions respectively. This means the problem is amenable to partial EM.

3.2 Statement of Algorithm

The PMHT can be implemented either as a batch algorithm, or recursively. Here the batch form is presented. The recursive algorithm is trivially recovered by choosing a unit length batch.

1. Initialise the target and sensor states, $\hat{\mathbf{X}}(0)$ and $\hat{\mathbf{Y}}(0)$.
2. Calculate the assignment weights, w_{mtr}
3. Refine the sensor estimates, $\hat{\mathbf{Y}}(i)$, assuming the target states are fixed (ie $\mathbf{X} = \hat{\mathbf{X}}(i-1)$). This is independent for the sensors; N parallel filters/smoothers.
4. Recalculate the assignment weights, w_{mtr}

5. Refine the target estimates, $\hat{\mathbf{X}}(i)$, assuming the target states are fixed (ie $\mathbf{Y} = \hat{\mathbf{Y}}(i)$). This is independent for the targets; M parallel filters/smoothers.

6. Return to step 2 until convergence.

In theory, convergence should be checked by evaluating the auxiliary function and monitoring its growth (each step should increase it by an asymptotically smaller amount). However, in practice calculating the auxiliary function is more expensive than the rest of the algorithm, and most of the time it converges in only two or three iterations. So, a practical scheme is likely to rather run the EM process for a fixed number of iterations (perhaps three).

Under linear Gaussian statistics, the state estimates can all be determined with Kalman Filters/Smoothers. The sensor filters can be implemented using synthetic measurements which are effectively a mean pseudo-measurement, combining the various observations, with an associated synthetic covariance.

3.3 Data Association

PMHT is inherently a data association algorithm. The algorithm assumes that the data association is unknown and uses EM to treat it as nuisance parameters. The resulting algorithm is somewhat like a probabilistic data association approach.

3.4 Nonlinearity

PMHT is not restricted to a particular estimation algorithm. In the general statement of the algorithm, it simply requires that the implementation use the optimal estimator for the problem. For nonlinear problems, an EKF could be used, or a particle filter could be implemented. Both forms have been used by various authors, and the choice depends on what estimator is appropriate for the problem.

So far, the example problem presented in the next section has assumed that the state estimation can be adequately solved using Extended Kalman Filters/Smoothers. Further work is required to obtain an analytic result demonstrating that this is indeed the linear approximate solution - or to determine what is, if something else.

3.5 Large Maps

The PMHT approach is well suited to large maps. Due to the conditional factorisation, there is no need to update the estimates of landmarks outside of the field of view. The complexity in the number of landmarks inside the field of view is linear, so the total cost of the algorithm comparable to FastSLAM with a single particle, and much less than that of the stacked state vector Kalman Filter.

4 Comparison of PMHT SLAM with Other Approaches

The previous section derived a PMHT based algorithm for solving the SLAM problem. As stated earlier, this is far from the first solution to SLAM, so it is important to compare it with existing approaches to the problem. Three metrics will be considered for this comparison: platform position estimation accuracy, computation time requirement, and data storage requirement. The first measures the quality of the algorithm output, and the others measure the cost of using the algorithm. The performance of the PMHT will be compared with the stacked state vector EKF approach and FastSLAM. In the EKF case, implementation is relatively simple, and quantitative results are readily obtained. However, implementation of FastSLAM is more difficult. To avoid erroneous results due to poor implementation, inferences about performance compared with FastSLAM will be made using the results comparing FastSLAM performance and EKF performance given in [Montemerlo 2003].

4.1 Estimation Accuracy

To measure estimation accuracy, it is necessary to compare the algorithm output with truth. For the real experiment in section 5, the only truth available is a set of GPS reports, that fluctuate by several metres between reports. This fluctuation is comparable, if not greater than the SLAM output error, so this data does not provide a useful set for quantitative error analysis. In order to get reliable error statistics, simulated data was be used.

The simulated scenario contained a single platform that observed landmarks using a radial sensor (range and bearing). The accuracy of this sensor was varied by scaling a base measurement variance according to

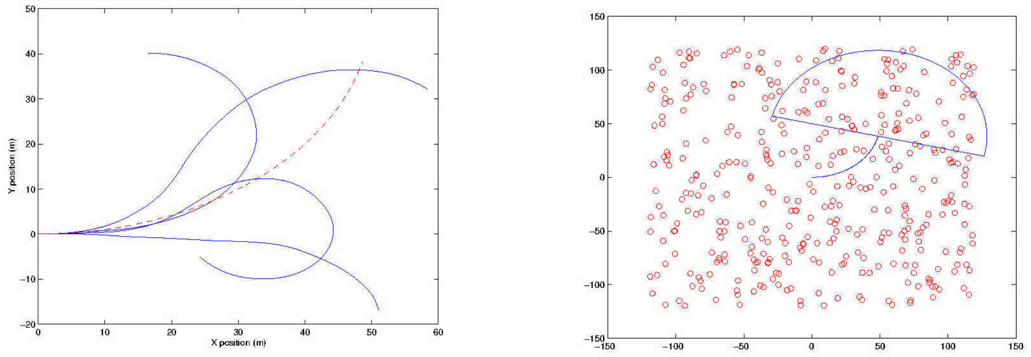
$$\mathsf{R}(\kappa) = \kappa^2 \begin{bmatrix} 0.1^2 & 0 \\ 0 & 0.05^2 \end{bmatrix}. \tag{24}$$

When κ was large, then data association became important.

The platform state was the same as was used for the Victoria Park experiment described later. Namely,

$$y_t \equiv \left\{ \begin{array}{c} \text{X position} \\ \text{Y position} \\ \text{heading} \\ \text{speed} \\ \text{steering angle} \end{array} \right\}_t , \tag{25}$$

with white Gaussian process noise on the speed and steering angle to emulate manoeuvres. The platform is initialised at the origin with zero heading, a speed of $\pi/2$ and a steering angle of $\tan^{-1}(L/50)$, where L is the axle length of the vehicle from the Victoria Park experiment, i.e. 2.83 metres. This corresponds to a 50 metre radius turn at one revolution per 200 seconds. The state evolution equations for this model are given in (36) ... (40). Two process noise examples were considered. The low process noise case had no process noise on the speed and noise with a standard deviation of 0.0001 radians on the steering angle. This is effectively no process noise. The high process noise case had standard deviations of 0.04 metres per second and 0.02 radians respectively.



(a) Example platform trajectories (b) Example low Q trial

Figure 1: Simulation examples

A fixed number of landmarks was randomly placed inside the world volume. To avoid extreme non-linearity, any landmark that was within three metres of the vehicle path was removed. The scenario runs for 200 sensor updates at the same rate as the laser sensor in the Victoria Park experiment, approximately 4.7 Hertz.

Figure 1(a) shows a few realisations of the high process noise example. The trajectory with zero process noise is shown as a dashed line. This demonstrates that the process noise is indeed high, but the true trajectories are still smooth due to the integration effect of the state model. Figure 1(b) shows an example trial with low process noise. The sensor field-of-view is shown as a semi-circle. Due to the random landmark placement, some landmarks are quite close together.

Monte Carlo trials were run for varying sensor accuracy (κ) and the two process noise values. Estimation performance was measured using two quantities

Divergent track rate is the percentage of trials on which the worst estimation error is more than 3 metres.

RMS estimation error is averaged over time and trials for non-divergent trials.

Figure 2 shows the RMS estimation error for the two algorithms as a function of the measurement accuracy. As expected, the error is higher for the high process noise case. For the high process noise and high measurement error case, both the PMHT and the EKF give similar error performance, but for all other cases, the PMHT has lower error.

Table 1 gives the percentage of divergent platform estimates. The PMHT is clearly far superior to the EKF in this area. In fact, the PMHT only ever diverges for the most difficult scenario where process and measurement noise are both high.

From these results, it is clear that the PMHT algorithm gives much better estimation performance than the EKF. This is primarily because of data association. The EKF here uses a nearest neighbour based data association method. The EKF has relatively poor performance for the lowest measurement noise case ($\kappa = 1$). This is because the low measurement noise results in a high Kalman gain, and when incorrect assignments are made, the spurious measurement correction causes the platform estimate to diverge.

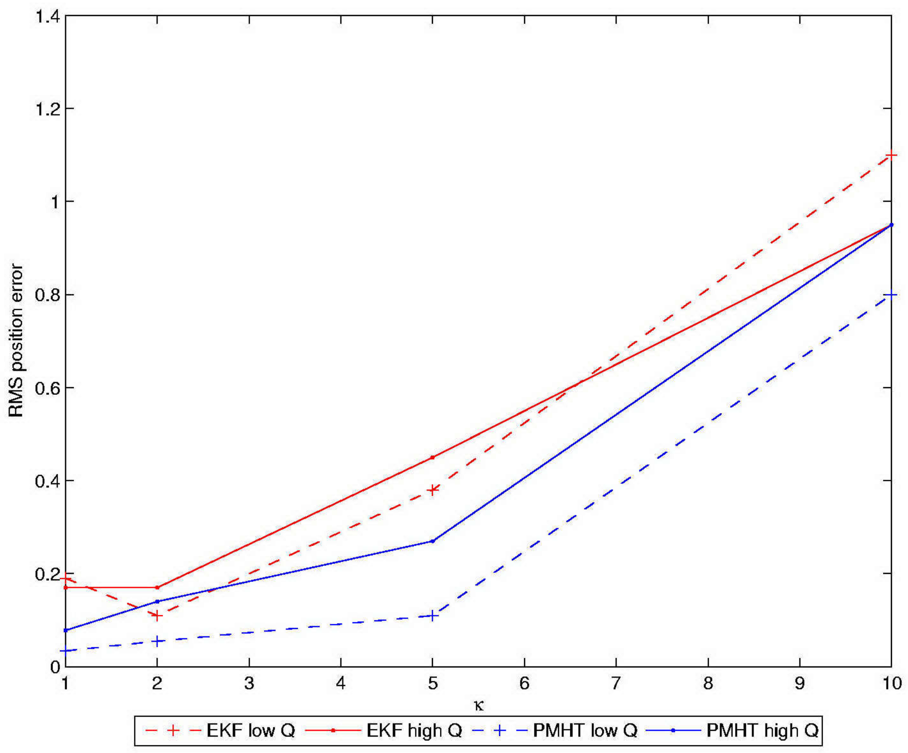


Figure 2: RMS position estimation error

	EKF	PMHT	EKF	PMHT
κ	low Q	low Q	high Q	high Q
1	34	0	74	0
2	3	0	24	0
5	14	0	38	0
10	71	0	83	30

Table 1: Percentage of divergent tracks

No attempt was made to implement FastSLAM, so there are no simulation results. However, it is assumed that it would have been able to achieve similar RMS and divergence performance as the PMHT, if enough particles were used. Since FastSLAM uses a sampling approach to data association, the number of particles would need to be high enough to at least enumerate all probable association hypotheses, which in the more challenging parameter combinations was very high. The cost of using many particles is explored in the following sections.

4.2 Computation Overheads

The computation requirements of the algorithm were also tested using simulated data. The scenario was similar to that used above, except that the process noise was set to zero for both speed and steering angle. This meant that the true platform trajectory was always a circular orbit. A longer duration was used: one thousand sensor measurements were simulated, which corresponded to slightly more than one circuit of the orbit. In the accuracy experiment, the landmarks were randomly placed, but here they were positioned in a regular grid to make it easier to control the landmark density. The sensor measurement noise was made low ($\kappa = 1$) to simplify data association. Since the process noise was zero, the Kalman gain was also always zero, with the result that the state estimate was simply predicted from the starting point. This was an unrealistic scenario, but it doesn't change the computation resource required for the problem.

The cpu time elapsed for a single scan update was recorded for every scan and over a number of Monte Carlo trials. This time was measured as a function of the number of landmarks in the sensor field-of-view, and the total number of landmarks in the map. The number of landmarks in the field-of-view changes from scan to scan depending on the sensor position and orientation, and was varied by running trials with different landmark densities. For a particular landmark density, the number of landmarks in the map was varied by simply using a proportionally larger map. When the map size was very large, most of the landmarks were never observed. However, the SLAM algorithm is initialised with an estimate for each landmark, so it is functionally equivalent to a scenario where the sensor has observed every landmark at some past time.

The computation requirements of each of the PMHT SLAM steps are now discussed to produce an expected complexity for the algorithm.

Initialisation Before measurements are processed, all states are initialised. This is linearly dependent on the total map size, but very cheap, and not expected to impact on overall computation cost. For each scan, the sensor state is initialised by predicting ahead from the previous time. This is independent of the map size, and also cheap.

Gate landmarks The simplest realisation of this process is to measure the range and angle to each landmark, using current estimates, and test whether these are within the sensor field of view. This is linear in the number of landmarks.

Calculate weights The most costly part of the weight calculation is evaluating the measurement density terms. This cost is reduced by gating the measurements. However, the number of gated measurements is dependent on the landmark density. This will be only slightly more than linear in the number of landmarks in the field of view, provided that the number of gated measurements is small.

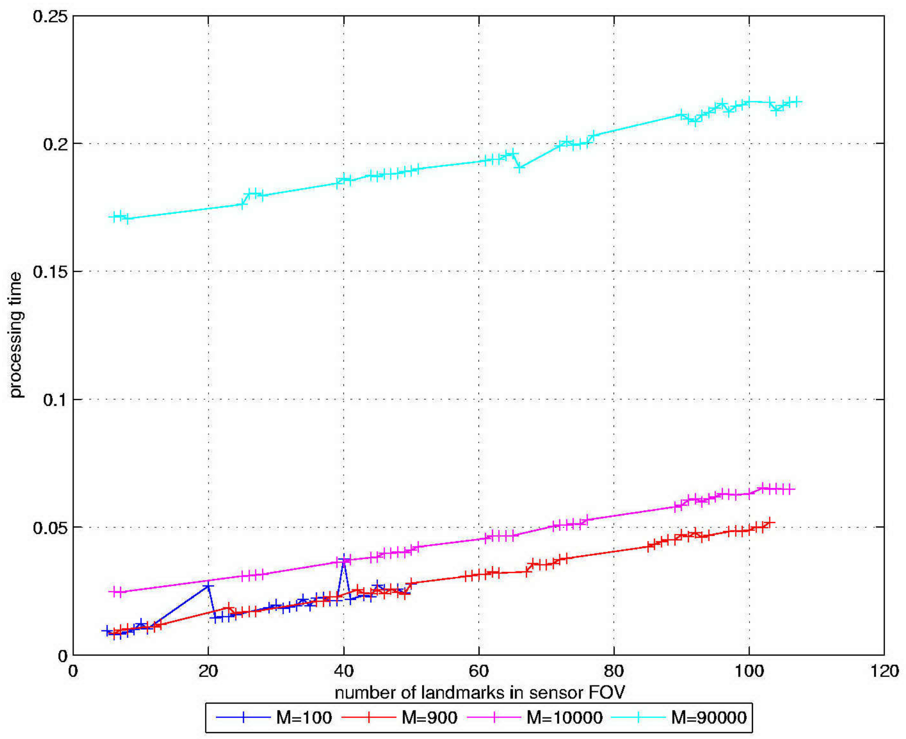


Figure 3: Computation time as a function of map size

Update platform state The most costly part of the platform update is incorporating the sensor measurements. By using a composite measurement approach, this is linear in the number of landmarks.

Update landmark state Each landmark EKF is independent of the number of landmarks, except through the influence this has on the number of gated measurements. Again, provided this is small, the landmark update is linear in the number of landmarks in the field of view.

Most of the above steps are independent of the number of total number of landmarks, and only linear in the number in the field of view. The exception is the initial field-of-view gating, which is linear in the total map size. This step would be a requirement for any SLAM approach; to remove it makes the rest of the algorithm dependent on the whole map size, not simply the number in the field-of-view.

Figure 3 shows how the processing time for a single time update varies as a function of the number of landmarks in the field of view. Each curve shows a particular total map size, M. The cost is clearly linear in the number of landmarks in the field of view. There is also an offset between the curves for larger total map sizes. This offset is linearly dependent on the map size, and is due to the field-of-view gating. Figure 4 shows the computation time excluding that used for the gating. It is independent of the total map size. The method used here for gating is a simple direct implementation, and methods exist to improve this. One example might be to group the landmarks into tiles and to gate the tiles instead. As stated earlier, this process is necessary for any algorithm, and so optimising it is not explored here.

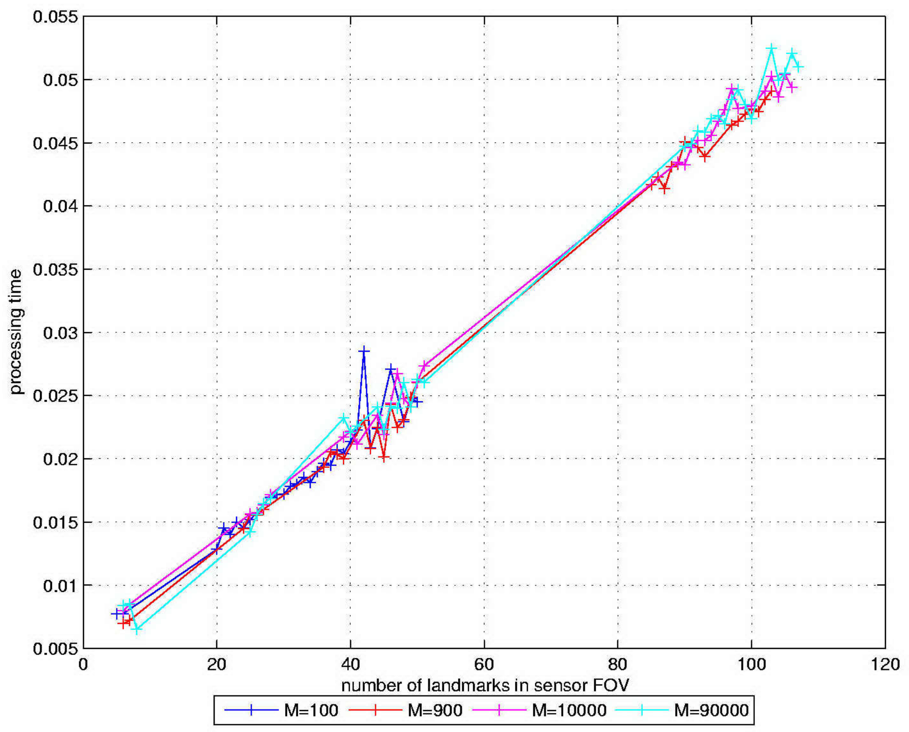


Figure 4: Computation time excluding FOV gating

4.2.1 Comparison

The stacked EKF algorithm will scale quadratically with the total map size. Using local region updates as in [Guivant & Nebot 2001*a*], this is reduced to quadratic in the number of landmarks in the field-of-view with lower rate full cost updates. In either case, the PMHT algorithm is clearly quicker.

The full algorithm for FastSLAM is laid out in [Montemerlo 2003]. Simplistically, it comprises the following steps:

1. For each particle
   (a) Sample a new pose
   (b) Calculate the likelihood of each measurement-landmark pair
   (c) Choose a hard data association
   (d) Update features according to data association
   (e) Calculate particle weight
2. Re-sample particles

The computation cost is linearly dependent on the number of particles, much as the PMHT cost is linearly dependent on the number of EM iterations used. In [Montemerlo 2003], the computation cost is measured for a fixed landmark density and a varying map size. It shows that the complexity of FastSLAM is logarithmic in the map size, and the cause of this cost is the maintenance of landmark estimates in the re-sampling phase of the particle filter. Thus, the dominant cost is step 2 from above. Notice that step 1(b) involves a pdf evaluation for every landmark and measurement for every particle, which is $M \times M_{FOV} \times M_p$ if M_{FOV} landmarks are detected in the field-of-view and there are M_p particles. This linear complexity in M is not apparent in the results of [Montemerlo 2003], which reflects that the resampling is a more costly exercise: a quadratic will appear linear if the linear coefficient is relatively large. The dominant cost in the PMHT algorithm for large maps was the field-of-view gating. This step is undertaken to reduce the cost of likelihood calculations and is cheaper than 1(b). Since step 1(b) is not significant in the overall FastSLAM cost, it implies that the whole PMHT algorithm cost is insignificant compared with running FastSLAM.

4.3 Storage Overheads

The data storage requirements for the PMHT algorithm are easy to determine. Each landmark in the filter has an independent estimator. Assuming that the state vector for landmarks has length d_M and an EKF is used, then each landmark requires $d_M + d_M^2$ floating points. Similarly, each platform requires $d_N + d_N^2$. This is very much smaller than the stacked EKF, which would require $(Nd_N + Md_M) + (Nd_N + Md_M)^2$. For the example in section 5, there are nominally 150 landmarks, leading to a difference of two orders of magnitude. For a two dimensional map with $M \gg N$, the stacked state vector uses approximately M times more memory.

The memory usage of FastSLAM is presented in section 3.8 of [Montemerlo 2003]. Here FastSLAM is demonstrated to use approximately 3 MB of memory for a map with 50,000 landmarks. For a map this size, the PMHT SLAM algorithm would require approximately 2.3 MB, around the same as FastSLAM.

5 Victoria Park Data

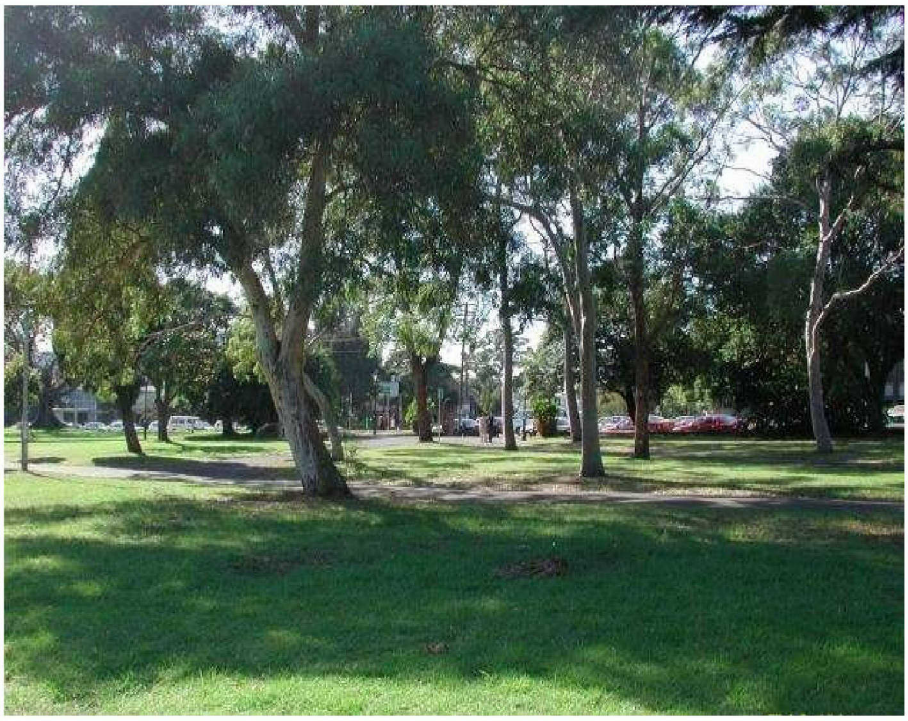
The Victoria Park data set is a benchmark SLAM data set recorded by the University of Sydney. In the experiment, a utility was fitted with various sensors and driven around Victoria Park (at the University of Sydney), which contains sparse trees. A laser range finder was used to observe the trees, and inertial measurements of the vehicle's speed and steering direction were collected. GPS data was collected to provide ground truth. Due to occlusion, the GPS signal was not available over the whole experiment, so there are patches where no truth data is available. Partial occlusion also caused the GPS constellation to vary with platform position. Constellation changes cause jumps in the GPS position estimate of up to five metres. This makes the GPS reports useful for gauging gross platform localisation performance, but not for quantifying estimation error. The data set contains over seven thousand laser measurements, collected over approximately half an hour, during which time the vehicle moved over four kilometres. The inertial measurements were collected at a much higher rate than the laser measurements (around 8.5 times faster).

Figure 5 shows the Victoria Park environment. In the experiment, the trees were the intended landmarks. However, other objects were present, and were detected at various times. Figure 5(a) is an image taken from the platform's perspective. It demonstrates how trees close to the sensor may occlude others farther away. It also shows some of the non-tree objects in the environment, such as poles, cars and pedestrians. Figure 5(b) shows an aerial image of the park. It highlights the varying density of the trees in the park, and also environment outside the park. In particular, there is a road to the South East of the park that has numerous moving vehicles. These vehicles were detected, but did not satisfy the stationary target assumption. The two images highlight one of the major difficulties in the data set: track maintenance under occlusion, where some targets are stationary but temporarily hidden, but others are not stationary. Non-stationary targets will degrade the SLAM performance, since it assumes all visible objects are stationary landmarks. However, discrimination between the various object types visible to the sensor is non-trivial.

Figure 6 contrasts the trajectory obtained purely by integrating the inertial measurements with the GPS reports. This highlights the inadequacy of the inertial measurements as an assumed truth, and motivates the need for applying a SLAM algorithm to this data. The main deficiency in the inertial data is that it tends to over-estimate turns. Numerous loops can be seen in figure 6 that are not found in the GPS reports. The individual GPS reports are not linked with a line because of their intermittent nature; putting a straight line through the gaps would not properly represent the vehicle trajectory.

5.1 Vehicle Geometry

The geometry of the sensor platform is illustrated in figure 7. The laser range finder is located on the front left side of the car, and the speed measurement is made at the rear axle. Relevant dimensions are indicated in the diagram. The physical location of the sensors on board the platform is of importance because many of the landmarks are observed from very close range, at times less than 2 metres. In the figure, the distance between the car's axles is labelled L, the width of the car is $2H$, and the dimensions a and b define the position of the laser sensor relative to the front axle and the car's centreline respectively.


(a) Platform's perspective

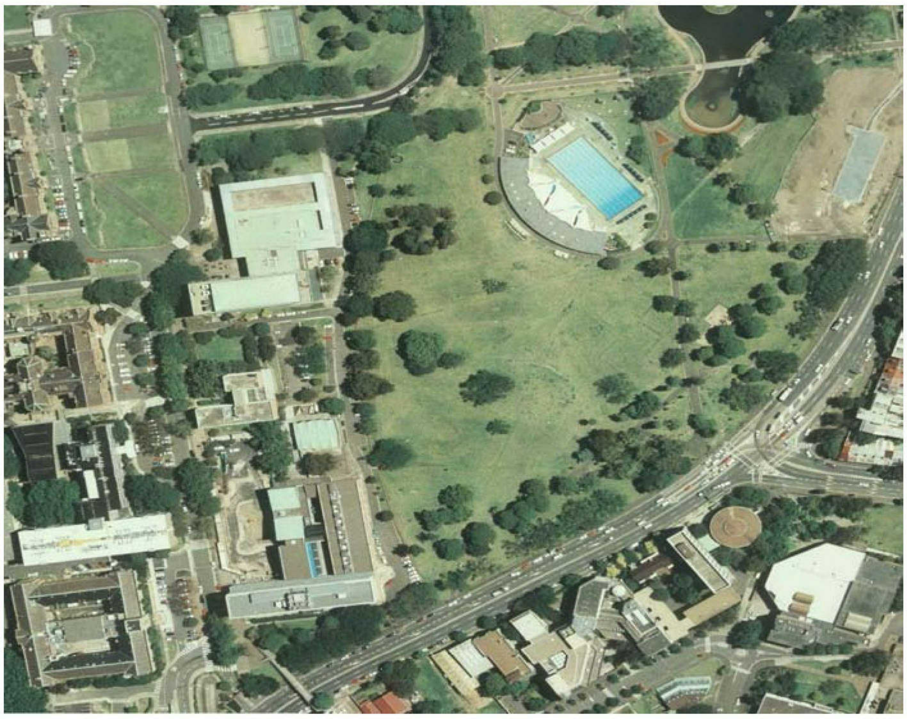
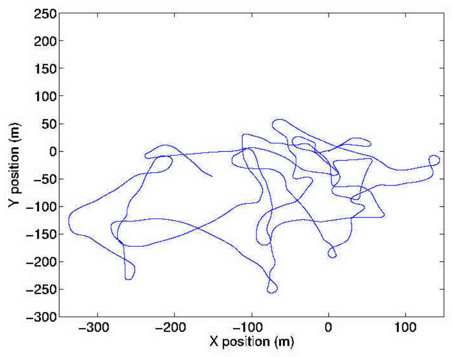
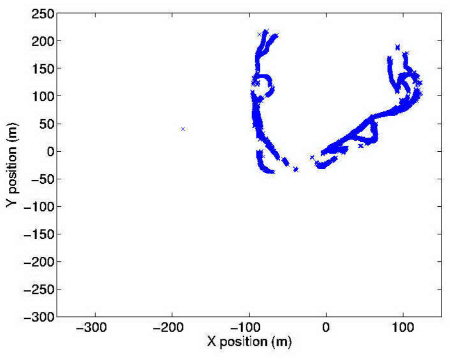

(b) Aerial perspective

Figure 5: The Victoria Park Environment



(a) Integrated inertial data



(b) GPS reports

Figure 6: Reliability of Inertial Data

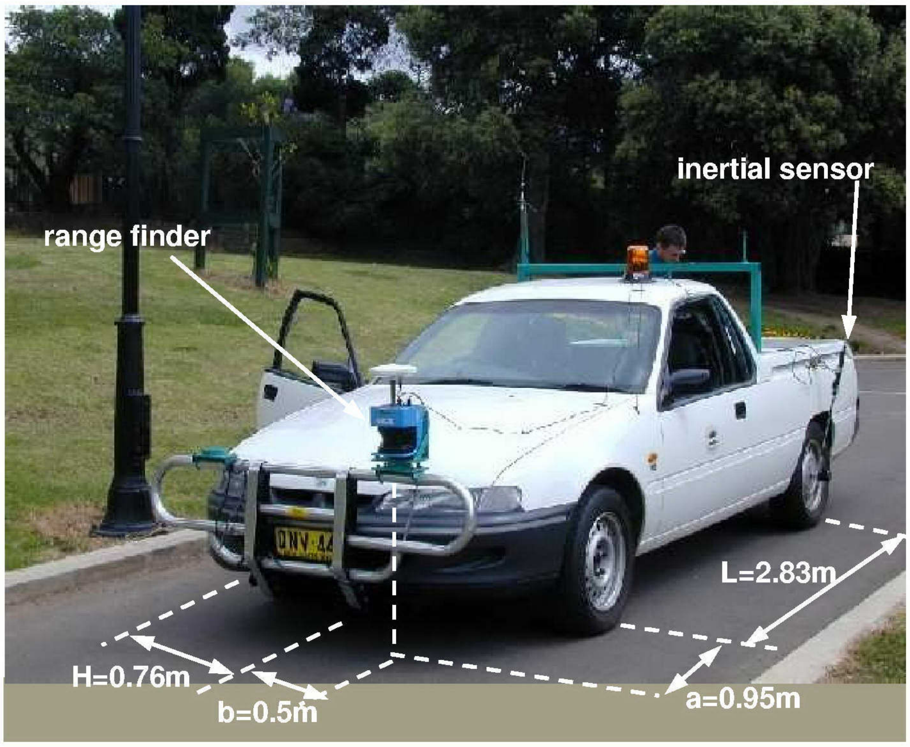


Figure 7: Vehicle Geometry

5.2 Measurements

There are two sensors available, a laser range finder and an inertial sensor. The inertial sensor measures the vehicle speed and steer direction. The steer direction is the angle between the front wheels and the direction vector of the vehicle. These are reported in metres per second and radians respectively. Each measurement has an associated time stamp measured in milliseconds, and the sensor collects a measurement every twenty five milliseconds (40 Hertz).

The laser range finder provides a scan of 361 angles in a 180 degree arc (i.e. 0.5 degree resolution) in front of the vehicle. For each angle, the range to the nearest reflector is reported. If no reflector is found within 80 metres, a null value is given. This data is also accompanied by a time stamp in milliseconds and the sensor scans at a revisit time of slightly faster than 214 milliseconds (about 4.7 Hertz). Professor Eduardo Nebot provided a detector with the data set. This detector takes the raw laser data and produces reports, each giving the location and size of a reflector. The processed laser data thus consists of a time stamp, and a set of observations each of which gives a reflector's range in metres, relative bearing in radians, and estimated size in metres. The detector is able to estimate the reflector size because the sensor resolution is high, and the trees are visible in several azimuth bins. So the processed laser data can be treated as measurements from a polar sensor that also measures a non-kinematic attribute. For the filter implemented here, it has been chosen to ignore the size measurement, although it could be exploited for data association.

5.3 Models

The scenario takes place in a two dimensional world (ignoring terrain topography). The landmarks are modelled as stationary objects, so their state vector is a simple two dimensional position vector

$$\boldsymbol{x}_t^m \equiv \left\{ \begin{array}{c} \text{X position} \\ \text{Y position} \end{array} \right\}_t^m . \tag{26}$$

There is only one sensor platform, and its motion is approximated using a constant speed, constant steering angle model

$$\boldsymbol{y}_t \equiv \left\{ \begin{array}{c} \text{X position} \\ \text{Y position} \\ \text{heading} \\ \text{speed} \\ \text{steering angle} \end{array} \right\}_t , \tag{27}$$

with white Gaussian process noise on the speed and steering angle to account for manoeuvres. This model allows for measurement noise in the inertial data by not treating them as simple control inputs, but rather observations of control inputs. Without perturbations (manoeuvres) the model reverts to a circular orbit with a fixed speed.

It is assumed that there is no slip in the wheels of the car, which is an acceptable approximation for moderate dynamics, and gives the following relations for the state evolution.

$$\dot{\boldsymbol{y}}[1] = \boldsymbol{y}[4] \cos(\boldsymbol{y}[3]) \tag{28}$$

$$\dot{\boldsymbol{y}}[2] = \boldsymbol{y}[4] \sin(\boldsymbol{y}[3]) \tag{29}$$

$$\dot{\boldsymbol{y}}[3] = \frac{\boldsymbol{y}[4]}{L} \tan\left(\boldsymbol{y}[5]\right) \tag{30}$$

Time recursions for the state elements are found by integrating the instantaneous heading over the revisit interval

$$\begin{aligned}
\boldsymbol{y}_{t+1}[1] &= \boldsymbol{y}_t[1] + \frac{L}{\tan(\boldsymbol{y}_t[5])}\left\{\sin\left(\boldsymbol{y}_{t+1}[3]\right) - \sin\left(\boldsymbol{y}_t[3]\right)\right\}, &(31)\\
\boldsymbol{y}_{t+1}[2] &= \boldsymbol{y}_t[2] + \frac{L}{\tan(\boldsymbol{y}_t[5])}\left\{\cos\left(\boldsymbol{y}_t[3]\right) - \cos\left(\boldsymbol{y}_{t+1}[3]\right)\right\}, &(32)\\
\boldsymbol{y}_{t+1}[3] &= \boldsymbol{y}_t[3] + \frac{\boldsymbol{y}_t[4]\tan(\boldsymbol{y}_t[5])}{L}\left(\tau_{t+1} - \tau_t\right), &(33)\\
\boldsymbol{y}_{t+1}[4] &= \boldsymbol{y}_t[4] + \omega_t[1], &(34)\\
\boldsymbol{y}_{t+1}[5] &= \boldsymbol{y}_t[5] + \omega_t[2]. &(35)
\end{aligned}$$

When the turn rate is zero (i.e. straight line motion), these simplify in the limit to

$$\begin{aligned}
\boldsymbol{y}_{t+1}[1] &= \boldsymbol{y}_t[1] + \boldsymbol{y}_t[4]\cos\left(\boldsymbol{y}_t[3]\right)\left(\tau_{t+1} - \tau_t\right), &(36)\\
\boldsymbol{y}_{t+1}[2] &= \boldsymbol{y}_t[2] + \boldsymbol{y}_t[4]\sin\left(\boldsymbol{y}_t[3]\right)\left(\tau_{t+1} - \tau_t\right), &(37)\\
\boldsymbol{y}_{t+1}[3] &= \boldsymbol{y}_t[3] + \boldsymbol{y}_t[5]\left(\tau_{t+1} - \tau_t\right), &(38)\\
\boldsymbol{y}_{t+1}[4] &= \boldsymbol{y}_t[4] + \omega_t[1], &(39)\\
\boldsymbol{y}_{t+1}[5] &= \boldsymbol{y}_t[5] + \omega_t[2]. &(40)
\end{aligned}$$

The above time recursions are used to predict ahead the state of the platform at scan $t+1$ given the state at scan t. The straight line equations are used when the estimated turn rate is low to avoid numerical problems.

The above state models the position and movement of the centre of the rear axle of the car. The sensor is located at a position $\boldsymbol{y}_t^l$ that is geometrically related to the platform state by the relation

$$\begin{aligned}
\boldsymbol{y}_t^l[1] &= \boldsymbol{y}_t[1] + \sqrt{(L+a)^2 + b^2}\cos\left(\boldsymbol{y}_t[3] + \beta\right), &(41)\\
\boldsymbol{y}_t^l[2] &= \boldsymbol{y}_t[2] + \sqrt{(L+a)^2 + b^2}\sin\left(\boldsymbol{y}_t[3] + \beta\right), &(42)\\
\beta &= \arctan\left(\frac{b}{L+a}\right), &(43)
\end{aligned}$$

where the dimensions L, a, and b are defined as shown in figure 7. Using the values for this scenario, as shown in the figure, this gives

$$\begin{aligned}
\boldsymbol{y}_t^l[1] &\approx \boldsymbol{y}_t[1] + 3.813\ \cos\left(\boldsymbol{y}_t[3] + 0.1315\right),\\
\boldsymbol{y}_t^l[2] &\approx \boldsymbol{y}_t[2] + 3.813\ \sin\left(\boldsymbol{y}_t[3] + 0.1315\right).
\end{aligned}$$

The laser measurements give the relative range and bearing to the reflector from the sensor. Thus the measurement functions are

$$\begin{aligned}
\boldsymbol{z}_{t1r}[1] &= \sqrt{\left(\boldsymbol{x}_t^m[1] - \boldsymbol{y}_t^l[1]\right)^2 + \left(\boldsymbol{x}_t^m[1] - \boldsymbol{y}_t^l[1]\right)^2} + \nu_{t1r}[1], &(44)\\
\boldsymbol{z}_{t1r}[2] &= \arctan\left(\frac{\boldsymbol{x}_t^m[2] - \boldsymbol{y}_t^l[2]}{\boldsymbol{x}_t^m[1] - \boldsymbol{y}_t^l[1]}\right) - \boldsymbol{y}_t[3] + \nu_{t1r}[2], &(45)
\end{aligned}$$

where ν_{t1r} is the measurement noise.

The inertial measurements are purely a function of the platform states. Assuming that there is no slipping in the front wheels, these are related to the states by

$$z_{t2r}[1] = \boldsymbol{y}_t[4] + \nu_{t2r}[1], \tag{46}$$
$$z_{t2r}[2] = \boldsymbol{y}_t[5] + \nu_{t2r}[2]. \tag{47}$$

It is obvious that both the state evolution and the measurement equations are nonlinear. Under Generalised EM, this does not necessarily require the PMHT to use a nonlinear filter to estimate the states. Generalised EM states that choosing a new state that increases the auxiliary function, rather than optimises it, will lead to the same solution on convergence. So, one might look to find a linear estimator for the states that is guaranteed to increase the auxiliary function, and simply perform more iterations. Alternatively, a nonlinear estimator could be used, such as a particle filter. For this problem, the benchmark is a stacked state vector EKF solution, so it will be assumed that the EKF approximation to the nonlinear problem is sufficient. Namely, the PMHT will be implemented using a bank of EKFs for the landmarks, and an EKF for the platform.

5.3.1 EKF Implementation

The EKFs used for the landmarks are standard polar measurement, cartesian constant state filters. So the state evolution matrix, F^x, and the process noise matrix, G^x, are identity. The process noise covariance is very small, since the landmarks are non-dynamic, $\mathsf{Q}^x = 10^{-16}\mathsf{I}$ (where I is the identity matrix). The measurement matrix is given by the Jacobian of the measurement function

$$\mathsf{H}_t^{xm} = \begin{bmatrix} \Delta X_t^m / r_t^m & \Delta Y_t^m / r_t^m \\ -\Delta Y_t^m / (r_t^m)^2 & \Delta X_t^m / (r_t^m)^2 \end{bmatrix}, \tag{48}$$

where

$$\begin{aligned} \Delta X_t^m &= \boldsymbol{x}_t^m[1] - \boldsymbol{y}_t[1], \\ \Delta Y_t^m &= \boldsymbol{x}_t^m[2] - \boldsymbol{y}_t[2], \\ r_t^m &= \sqrt{(\boldsymbol{x}_t^m[1] - \boldsymbol{y}_t[1])^2 + (\boldsymbol{x}_t^m[2] - \boldsymbol{y}_t[2])^2}. \end{aligned}$$

The sensor platform estimator has been implemented as a stacked measurement Extended Kalman Smoother. Since the dynamics are nonlinear, the evolution matrix is also a Jacobian

$$\mathsf{F}_t^y = \begin{bmatrix} 1 & 0 & f13 & f14 & f15 \\ 0 & 1 & f23 & f24 & f25 \\ 0 & 0 & 1 & f34 & f35 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{49}$$

where

$$f13 = -\frac{L}{\sin^2(\boldsymbol{y}_t[5])}\Big\{ \cos(\boldsymbol{y}_t[3]) - \cos\big(\boldsymbol{y}_{t-1}[3]\big) \Big\},$$

$$
\begin{aligned}
f14 &= (\tau_t - \tau_{t-1})\cos(\boldsymbol{y}_t[3]),\\
f15 &= \frac{L}{\sin^2(\boldsymbol{y}_t[5])}\Big\{\sin(\boldsymbol{y}_t[3]) - \sin(\boldsymbol{y}_{t-1}[3])\Big\} + \frac{(\tau_t - \tau_{t-1})\cos(\boldsymbol{y}_t[3])\,\boldsymbol{y}_t[4]}{\sin(\boldsymbol{y}_t[5])\cos(\boldsymbol{y}_t[5])},\\
f23 &= -\frac{L}{\sin^2(\boldsymbol{y}_t[5])}\Big\{\sin(\boldsymbol{y}_t[3]) - \sin(\boldsymbol{y}_{t-1}[3])\Big\},\\
f24 &= (\tau_t - \tau_{t-1})\sin(\boldsymbol{y}_t[3]),\\
f25 &= \frac{L}{\sin^2(\boldsymbol{y}_t[5])}\Big\{\cos(\boldsymbol{y}_t[3]) - \cos(\boldsymbol{y}_{t-1}[3])\Big\} + \frac{(\tau_t - \tau_{t-1})\sin(\boldsymbol{y}_t[3])\,\boldsymbol{y}_t[4]}{\sin(\boldsymbol{y}_t[5])\cos(\boldsymbol{y}_t[5])},\\
f34 &= \frac{(\tau_t - \tau_{t-1})\,L}{\tan(\boldsymbol{y}_t[5])},\\
f35 &= \frac{(\tau_t - \tau_{t-1})\,\boldsymbol{y}_t[4]}{L\cos^2(\boldsymbol{y}_t[5])}.
\end{aligned}
$$

The process noise matrix is given by

$$
\mathsf{G}_t^y = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \tau_t - \tau_{t-1} & 0 \\ 0 & \tau_t - \tau_{t-1} \end{bmatrix}, \tag{50}
$$

i.e. the process noise is white noise acceleration in speed and steering angle. The process noise covariance is the diagonal matrix, $\mathsf{Q}_t^y = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix}$.

For laser measurements, the platform estimator uses a stacked measurement matrix, similar to (10), but where the elements are of the form of (48).

For inertial measurements, the measurement matrix is given by

$$
\mathsf{H}_t^y = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{51}
$$

Both sensors are assumed to have diagonal measurement covariances (e.g. the range measurement is not correlated with the angle measurement). These are given by

$$
\mathsf{R}_{\text{laser}} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.001 \end{bmatrix}, \tag{52}
$$

$$
\mathsf{R}_{\text{inertial}} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}. \tag{53}
$$

The values of measurement covariance and process covariance were chosen by trial and error to give good performance.

5.3.2 Implementation Issues

There are a few other issues that had to be addressed to get good performance on the Victoria Park data. There are two main issues to deal with: occlusion and loop closing.

As shown earlier, some of the landmarks in the sensor field-of-view may be completely or partially occluded by other landmarks at closer range. This causes valid tracks to be truncated if standard target existence type models are used for track maintenance. In addition, the laser sensor appears to be somewhat unreliable even when there are no occlusions, especially at longer range. These features make it difficult to properly deal with landmark existence. The approach taken was to insert new tracks, and if they survived for longer than a threshold duration, then they were protected and never truncated. A simplistic effort to account for occlusion was incorporated, but this was conservative: only tracks that were certain to be occluded were discounted, and used primarily to limit the number of tracks used in data association.

Loop closing becomes an issue because the sensor measurement variance is relatively low. After the platform completes a loop, there will be some residual alignment error that prevents data association of the old landmarks. The error is actually in the platform state estimate, but the algorithm requires that the measurement variance be inflated to allow for correct association. An ad hoc scaling of the measurement variance was incorporated at times when orientation error is likely to be high: during turns, and when the platform closes large loops.

5.3.3 Performance without inertial input

The PMHT algorithm was also run without the inertial input data. This did not require re-tuning of the process or measurement noise matrices, and was simply achieved by not providing one of the input sources.

5.3.4 Alignment

The GPS reports are given in latitude and longitude in metres relative to an unknown reference point. In order to compare them with the SLAM estimate, the two need to be registered and time aligned. The initial GPS report defines the starting position for the vehicle, but the initial heading needs to be inferred. The GPS reports are not synchronous with the platform sensors, so the state estimates need to be re-sampled at the GPS times. This is done using the evolution model and predicting the state ahead from the closest earlier time point. Once the trajectories are time aligned, the initial heading can be determined using a least squares fit. This is given by

$$\tan\theta = \frac{\sum_t \left(A_t^x B_t^y - A_t^y B_t^x\right)}{\sum_t \left(A_t^x B_t^x + A_t^y B_t^y\right)}, \tag{54}$$

where the GPS report at time t is (A_t^x, A_t^y), and the filter output is (B_t^x, B_t^y). This gives an initial heading of approximately 35 degrees.

5.4 Results

The platform trajectory estimate produced by the PMHT algorithm is shown in figure 8. The GPS reports are shown as crosses, the PMHT SLAM estimate with all inputs as a solid line, and

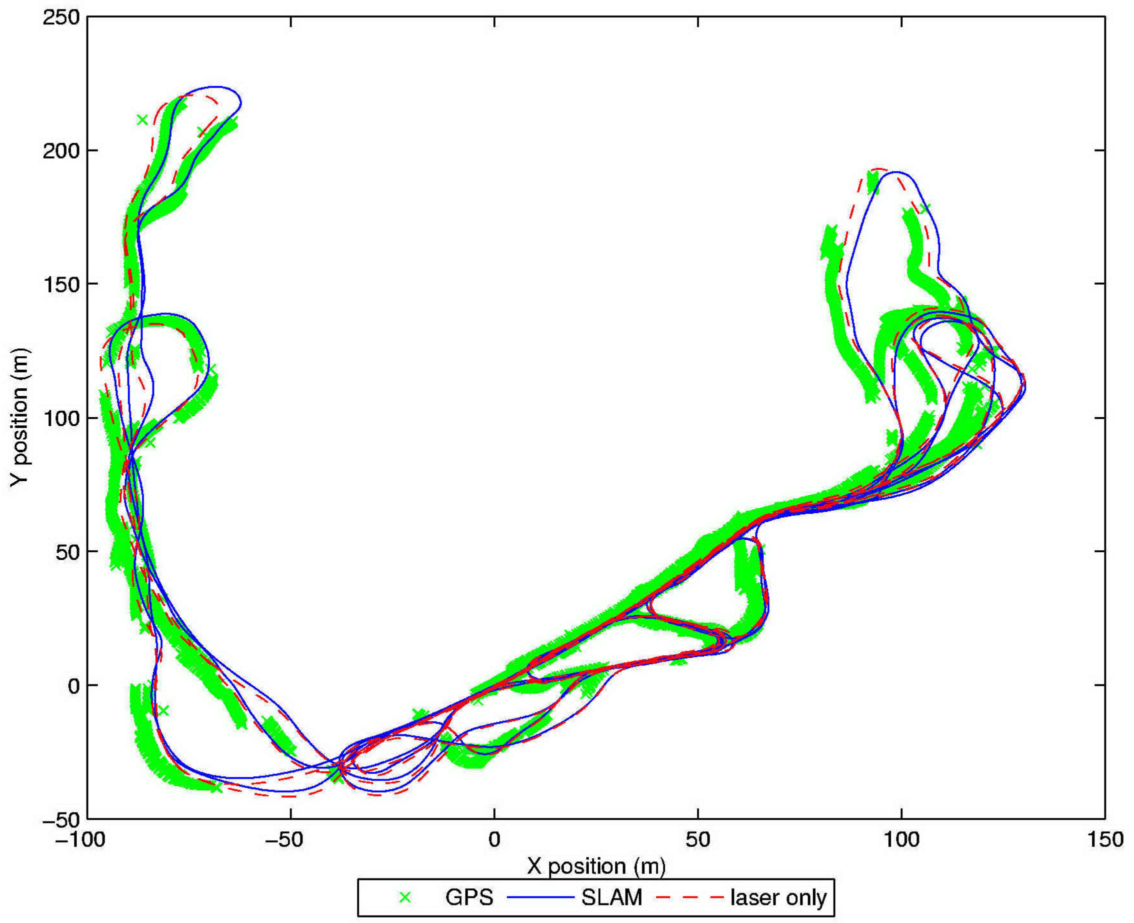


Figure 8: PMHT estimated trajectory

the PMHT SLAM estimate with only laser measurements as a dashed line. Overall, the estimates are relatively close to the GPS reports. Removing the inertial measurements did not significantly degrade the performance.

The main discrepancy between the GPS and PMHT estimates is in the North-Eastern part of the map. This part of the trajectory consists of several loops connected to the main area via a single path. Just before the path is a turn, and it appears that the PMHT algorithm made a small error in estimating the magnitude of this turn. This angular error translates to a relatively large position error at the farther ends of the path.

Figure 9 compares the PMHT SLAM output with that obtained from a stacked state vector EKF using the same dynamic models. The EKF was more difficult to tune and required somewhat different process noise values to provide a reasonable output. The two estimates are very close for the bulk of the trajectory, but deviate in the North-East and Western parts of the map. As described above, both of these sections are individual loops with connecting sections along turns. Small angular discrepancies during these turns lead to large position deviations. The difference between these two outputs is due to data association.

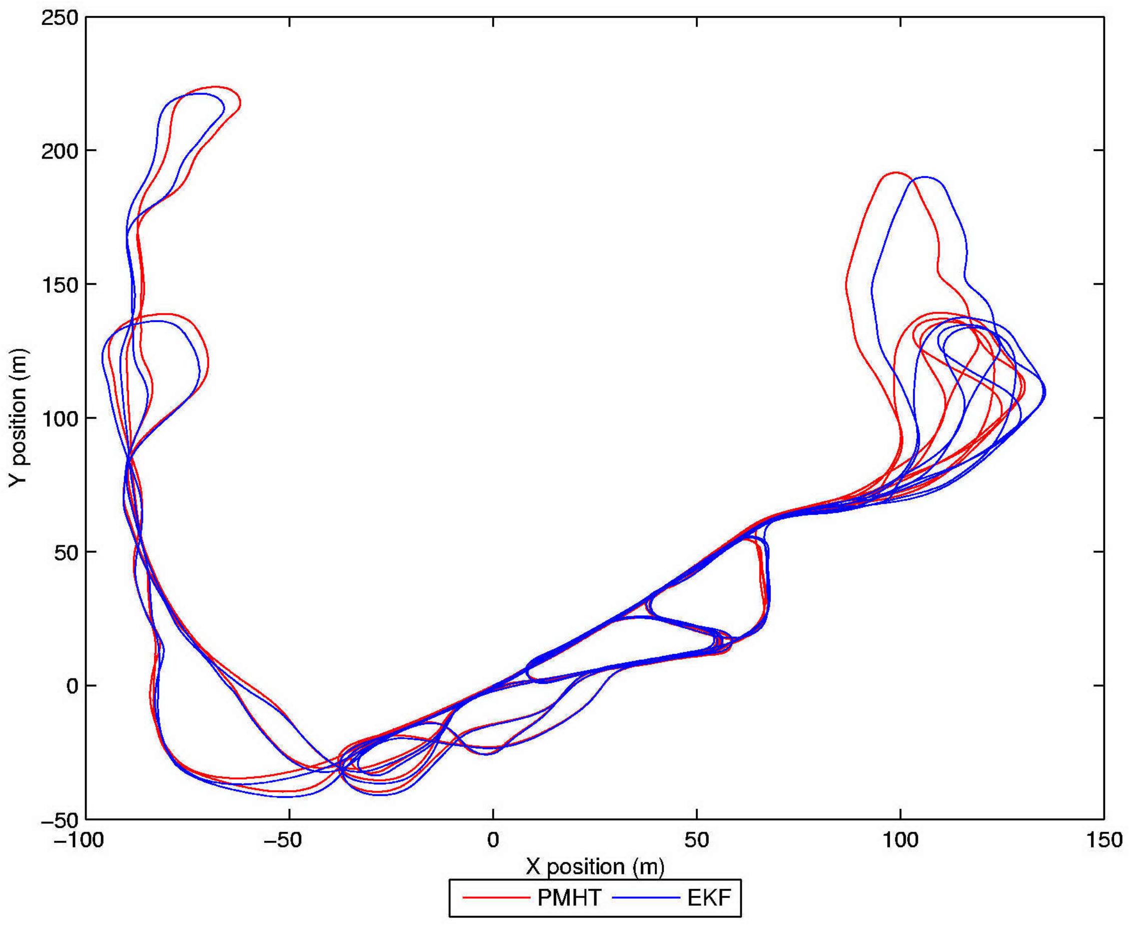


Figure 9: PMHT estimate compared with EKF estimate

5.4.1 Comparison with other algorithms

At this point in time, other authors have not made numerical output on this data available. Montemerlo provided an RMS error figure between the FastSLAM output and the GPS reports of "approximately 4m". The PMHT SLAM output has an RMS error of 4.2m. As described earlier, the GPS reports suffer from relatively large jumps as occlusion of the GPS satellites varies, and do not provide a useful truth source. It is likely that the SLAM output is more accurate than the GPS, and the figure of 4m is mainly due to errors in the GPS.

Independent of the accuracy of the GPS, RMS error is a poor measure of performance in this example. As discussed above, small angular errors can lead to large position deviations for long legs of the path. Thus the RMS position error is not indicative of the algorithm performance.

Along with the data set, Neito provided an image of the estimated trajectory using a stacked state vector EKF, overlaid on a map of Victoria Park. This image has been used in annual reports of the Australian Centre for Field Robotics, and represents the output of the algorithm in [Guivant & Nebot 2001*a*]. The PMHT output was manually registered onto the image using the tree position estimates near the starting point of the vehicle. Once again, small angular deviations lead to large position discrepancies that make the estimated tree locations at the extremes of the map poor choices for registration. Figure 10 shows the PMHT output overlaid on top of the EKF picture. The image provided by Neito shows the EKF output as a solid blue line, and the estimated tree locations as circles with dots at the centre. The GPS reports are overlaid on the image based on the PMHT registration and are marked as crosses. The PMHT output is marked as a yellow line.

The EKF algorithm does a better job of estimating the trajectory in the North-East corner, which was identified earlier as a problem with the PMHT. However, the EKF does a poor job of estimating the trajectory along the Western part of the map, compared with the GPS reports. There are a few differences between the algorithms: the assumed platform model and assumed noise variances, and data association.

Overall, it appears that the PMHT SLAM output is roughly equivalent to the FastSLAM output, and superior to that of the EKF.

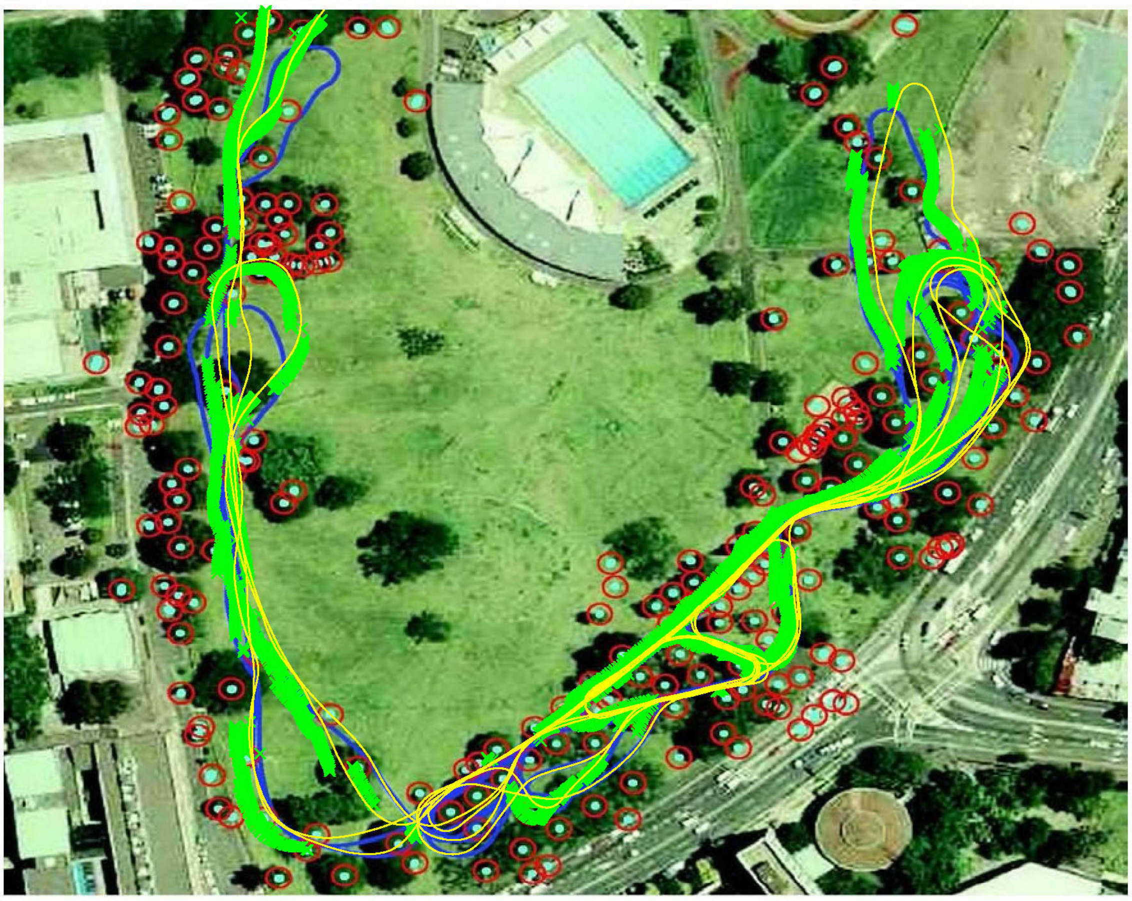


Figure 10: PMHT output overlaid on EKF output

6 Summary

This report has introduced a new approach for solving the Simultaneous Localisation and Map Building problem (SLAM). The algorithm described is based on the Probabilistic Multi-Hypothesis Tracker, and has the advantage of linear complexity in the number of landmarks in the sensor field-of-view.

The computation expense of PMHT SLAM was demonstrated to be significantly lower than current SLAM algorithms, while its estimation performance is superior to that of the standard EKF approach.

The algorithm was demonstrated on benchmark data from Victoria Park, and showed comparable performance with the published results from other algorithms. The PMHT performance on this data might be improved by using a better non-linear estimator, such as a particle filter, or an unscented Kalman filter.

At this stage, no quantitative comparison has been performed with FastSLAM, the current state of the art algorithm. This was deliberately avoided to prevent perceived implementation faults. In the future it would be ideal to pursue a collaborative comparison of the two algorithms.

References

Bar-Shalom, Y. & Blair, W. D. (2000) *Multitarget-Multisensor Tracking: Applications and Advances III*, Artech House, Norwood, Massachusetss, USA.

Bar-Shalom, Y., Li, X. R. & Kirubarajan, T. (2001) *Estimation with Applications to Tracking and Navigation*, Wiley-Interscience.

Blackman, S. S. & Popoli, R. (1999) *Design and Analysis of Modern Tracking Systems*, Artec House, Norwood, Massachusetss, USA.

Dempster, A. P., Laird, N. M. & Rubin, D. B. (1977) Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistics Society* **140**, 1–38.

Dissanayake, M. W. M. G., Newman, P., Clark, S., Durrant-Whyte, H. & Csorba, M. (2001) A solution to the simultaneous localization and map building (SLAM) problem, *IEEE transactions on Aerospace and Electronic Systems* **17**(3), 229–241.

Doucet, A., De Freitas, N. & Gordon, N. (2001) *Sequential Monte Carlo Methods in Practice*, Springer Verlag.

Guivant, J. E. & Nebot, E. M. (2001*a*) Optimization of the simultaneous localization and map-building algorithm for real-time implementation, *IEEE Transactions on Robotics and Automation* **17**(3), 242–257.

Guivant, J. & Nebot, E. (2001*b*) *Simultaneous Localization and Map Building: Test case for Outdoor Applications*, Technical report, Australian Centre for Field Robotics.

Montemerlo, M., Thrun, S., Koller, D. & Wegbreit, B. (2002) FastSLAM: A factored solution to the simultaneous localization and mapping problem, *in Proceedings of the AAAI National Conference on Artificial Intelligence*, AAAI, Edmonton, Canada.

Montemerlo, M., Thrun, S., Koller, D. & Wegbreit, B. (2003) FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, *in Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Montemerlo, M. (2003) *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem With Unknown Data Association*, PhD thesis, Carnegie Mellon University.

Ristic, B., Arulampalam, S. & Gordon, N. (2004) *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House.

Smith, R., Self, M. & Cheeseman, P. (1990) *Autonomous Robot Vehicles*, Cox, I. J. & Wilfong, G. T. (ed) Springer-Verlag, chapter Estimating Uncertain Spatial Relationships in Robotics, pp. 167–193.

Streit, R. L. & Luginbuhl, T. E. (1995) *Probabilistic multi-hypothesis tracking*, Technical report 10428, NUWC, Newport, Rhode Island, USA.

Thrun, S. (2002) *Exploring Artificial Intelligence in the New Millenium*, Lakemeyer, G. & Nebel, B. (ed) Morgan Kaufmann, chapter 1 Robotic Mapping: A Survey.

DISTRIBUTION LIST

Simultaneous Localisation and Map Building using the Probabilistic Multi-Hypothesis Tracker

Samuel Davey

	Number of Copies
DEFENCE ORGANISATION	
Task Sponsor	
Director General Aerospace Development	1
S&T Program	
Chief Defence Scientist FAS Science Policy AS Science Corporate Management Director General Science Policy Development	1
Counsellor, Defence Science, London	Doc Data Sheet
Counsellor, Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	Doc Data Sheet and Dist List
Scientific Adviser, Army	Doc Data Sheet and Dist List
Air Force Scientific Adviser	Doc Data Sheet and Exec Summ
Scientific Adviser to the DMO M&A	Doc Data Sheet and Dist List
Scientific Adviser to the DMO ELL	Doc Data Sheet and Dist List
Platform Sciences Laboratory	
Director, PSL	Doc Data Sheet and Exec Summ
Information Sciences Laboratory	
Chief, ISRD	Doc Data Sheet and Dist List
Research Leader, WASB	Doc Data Sheet and Dist List
Head, TSF	1
S.J. Davey	1
DSTO Library and Archives	
Library, Fishermans Bend	Doc Data Sheet
Library, Edinburgh	1 and Doc Data Sheet

Library, Sydney	Doc Data Sheet
Library, Stirling	Doc Data Sheet
Library, Canberra	Doc Data Sheet
Defence Archives	1
Capability Development Group	
Director General Maritime Development	Doc Data Sheet
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet
Director Trials	Doc Data Sheet
Chief Information Officer Group	
Deputy Chief Information Officer	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet
Director General Information Services	Doc Data Sheet
Strategy Group	
Director General Military Strategy	Doc Data Sheet
Director General Preparedness	Doc Data Sheet
Assistant Secretary Strategic Policy	Doc Data Sheet
Assistant Secretary Governance and Counter-Proliferation	Doc Data Sheet
Navy	
SO (SCIENCE), COMAUSNAVSURFGRP, NSW	Doc Data Sheet and Dist List
Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet
Deputy Director (Operations) Maritime Operational Analysis Centre, Building 89/90, Garden Island, Sydney Deputy Director (Analysis) Maritime Operational Analysis Centre, Building 89/90, Garden Island, Sydney	Doc Data Sheet and Dist List
Army	
ABCA National Standardisation Officer, Land Warfare Development Sector, Puckapunyal	Doc Data Sheet (pdf format)

SO (Science), Deployable Joint Force Headquarters (DJFHQ)(L), Enoggera QLD	Doc Data Sheet
SO (Science), Land Headquarters (LHQ), Victoria Barracks, NSW	Doc Data Sheet and Exec Summ
Air Force	
SO (Science), Headquarters Air Combat Group, RAAF Base, Williamtown	Doc Data Sheet and Exec Summ
Joint Operations Command	
Director General Joint Operations	Doc Data Sheet
Chief of Staff Headquarters Joint Operation Command	Doc Data Sheet
Commandant ADF Warfare Centre	Doc Data Sheet
Director General Strategic Logistics	Doc Data Sheet
Intelligence and Security Group	
DGSTA, Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1 (pdf format)
Assistant Secretary Capability Provisioning	Doc Data Sheet
Assistant Secretary Capability and Systems	Doc Data Sheet
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation	Doc Data Sheet
Defence Materiel Organisation	
Deputy CEO, DMO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Chief Joint Logistics Command	Doc Data Sheet
Defence Libraries	
Library Manager, DLS-Canberra	Doc Data Sheet

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy Library	1
Head of Aerospace and Mechanical Engineering, ADFA	1
Deakin University Library, Serials Section (M List), Geelong, Vic	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1

INTERNATIONAL DEFENCE INFORMATION CENTRES

US - Defense Technical Information Center	2
UK - Dstl Knowledge Services	2
Canada - Defence Research Directorate R&D Knowledge and Information Management (DRDKIM)	1 (pdf format)
NZ - Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES

DSTO Edinburgh Library	5

Total number of copies:	31

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION **DOCUMENT CONTROL DATA**	1. CAVEAT/PRIVACY MARKING

2. TITLE Simultaneous Localisation and Map Building using the Probabilistic Multi-Hypothesis Tracker	3. SECURITY CLASSIFICATION Document (U) Title (U) Abstract (U)
4. AUTHORS Samuel Davey	5. CORPORATE AUTHOR Information Sciences Laboratory PO Box 1500 Edinburgh, South Australia, Australia 5111

6a. DSTO NUMBER DSTO–TR–1691	6b. AR NUMBER AR 013-343	6c. TYPE OF REPORT Technical Report	7. DOCUMENT DATE March, 2005

8. FILE NUMBER 2005/1022386	9. TASK NUMBER JTW 03/148	10. SPONSOR DGAD	11. No OF PAGES 33	12. No OF REFS 15

13. URL OF ELECTRONIC VERSION http://www.dsto.defence.gov.au/corporate/reports/DSTO–TR–1691.pdf	14. RELEASE AUTHORITY Chief, Intelligence, Surveillance and Reconnaissance Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved For Public Release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SOUTH AUSTRALIA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DEFTEST DESCRIPTORS

Algorithms, Kalman filtering, Sensor mapping, Tracking (position)

19. ABSTRACT

This report presents an algorithm for efficiently solving the Simultaneous Localisation and Map Building (SLAM) problem. The SLAM problem requires both the dynamic estimation of the sensor location and the tracking of features of interest in the environment using the sensor measurements. The problem is difficult because the unknown sensor and feature locations are coupled through the sensor measurement. It has been shown that under linear Gaussian conditions, a Kalman Filter solution converges to a solution relative to the unknown starting location. However, this approach does not scale well with the number of features in the scene, and is unfeasible for large maps. The algorithm introduced here is based on the Probabilistic Multi-Hypothesis Tracker (PMHT) and exploits a factorisation of the problem to reduce the computational requirements of the Kalman Filter approach. The new algorithm is demonstrated on a benchmark data set recorded in Victoria Park.