2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

A Hybrid SLAM Representation for Dynamic Marine Environments

Charles Bibby
Active Vision Lab
Oxford University
Email: cbibby@robots.ox.ac.uk

Ian Reid
Active Vision Lab
Oxford University
Email: ian@robots.ox.ac.uk

Abstract— **We present a hybrid SLAM system for marine environments that combines cubic splines to represent the trajectories of dynamic objects, point features to represent stationary objects and an occupancy grid to represent land masses. This hybrid representation enables SLAM to be applied in environments with moving objects, where solutions using point features alone are computationally prohibitive or where dense objects e.g. landmasses can not be represented correctly using point features. Estimation is achieved using a sliding window framework with reversible data-association and reversible model-selection. Our main contributions are: (i) a hybrid representation of the environment; (ii) occupancy grid fusion is continually refined for the duration of the sliding window; (iii) the trajectories of dynamic objects are represented using cubic splines and (iv) radar scans are re-rendered at a sub-scan resolution to compensate for the egomotion during the scan acquisition period. We show that the continual refinement of the occupancy grid greatly improves the quality of the resultant map, leading to a better estimate of the egomotion and therefore better estimates of the trajectories of dynamic objects. We also demonstrate that the use of cubic splines to represent trajectories has two major advantages: (i) the state space is compressed i.e. many vehicle poses can be represented using a single spline section and (ii) the trajectory becomes continuous and so fusing information from asynchronous sensors running at multiple frequencies becomes trivial.**

The efficacy of our system is demonstrated using real marine radar data, showing that it can successfully estimate the positions/velocities of objects and landmasses observed during a typical voyage on a small boat.

I. INTRODUCTION

Simultaneous Localisation and Mapping (SLAM) has been studied for over two decades and although many consider it a 'solved problem', the task of producing an estimation framework to do SLAM in the 'real world' is non-trivial; it requires significant amounts of domain knowledge, good underlying representations, carefully selected heuristics and non-trivial tuning. The 'real world' scenario we are interested in is the marine environment observed from a small boat using radar. This has applications in GPS denied environments, for example during a military operation. This challenging environment contains landmasses, stationary objects and dynamic objects, and it is desirable to model them all within a single framework. To add to the difficulty, the marine radar sensor is prone to poor angular resolution, reflections, interference and clutter. We tackle these difficulties using sliding window estimation, reversible decision making and hybrid mapping, allowing us to do SLAM in environments with large numbers of moving objects [2][19][18]. The sliding window provides a fixed period of time during which the estimated landmasses, stationary and dynamic objects, and the egomotion trajectory can be refined, as well as allowing reversible data-association and model-selection. This is crucial to the success of the system because it is only possible to make the correct decision about the true origin of a measurement (clutter, stationary or dynamic) when given enough time to observe temporal characteristics.

This work is based upon our previous work [2], which uses a sliding window filter [15] to achieve reversible data-association and model-selection and shows that this leads to more consistent estimation. The first shortfall of the method proposed in [2] is that point features can not represent large objects e.g. landmasses and so a heuristic clustering method is used to represent these large objects as a mixture of point features. The problem with this approach is that the clustering method will not produce the same cluster centres for different scans, leading to poor data-association and therefore a reduction in overall accuracy. We address this problem by using occupancy grids [10][9][5], allowing us to deal with objects of arbitrary size and shape probabilistically by breaking them into small grid cells and then computing the posterior probability that a grid cell is occupied. The second problem with [2] is that a pose is stored and estimated at every time step for all dynamic objects, resulting in the system quickly falling below real-time performance for any significant number of dynamic objects. We tackle this problem by using a cubic spline representation for the trajectories of dynamic objects and the egomotion, which allows them to be automatically compressed based on their kinematics. In other words, if a dynamic object moves in a straight line for five minutes then the system can represent this using the equivalent of only two poses, one at the beginning and one at the end of the trajectory. In practice the kinematics are rarely that simple and so the system finds the appropriate compromise, which typically results in around a 70-80% reduction in the number of states required in the estimation process. The third problem with [2], which is also common to the majority of work in this research area, is that the sensor is treated as a synchronous snapshot of the world, whereas in reality the sensor data is actually aquired whilst the platform is moving. This results in errors in the estimation, which we demonstrate in the results section as shadowing in the occupancy grid (caused by measurements falling in the wrong grid cell). Our spline representation allows us to compensate

for this by re-rendering the sensor data to account for the egomotion undergone during the sensor aquistion period. The sliding window is used to continually refine the egomotion trajectory and the occupancy grid (similar to [16]), using generalised expectation maximisation [11][8]. This improves the estimate of the egomotion, the occupancy grid and the dynamic objects as well as providing the necessary time required to get data-association and model-selection correct.

The notion of hybrid mapping is becoming increasingly popular, with [13] incorporating features within an occupancy grid framework and [12] breaking the occupancy grid into triangular patches and using feature based methods to estimate their positions. Our method of using cubic splines to represent trajectories within a SLAM system is the first of its kind, providing an elegant and compact way of representing trajectories in a continuous manner. The closest related work is [14], which uses Bezier splines to represent stationary objects i.e. walls and corridors. In contrast, our method uses splines to represent the trajectories of dynamic objects and hence the spline parameter represents time. This has three major advantages: (i) the number of parameters required for the spline is less than having a pose at each time step; (ii) the continuous nature of the spline makes it trivial to add measurements to the system at arbitrary times, making it easy to use asynchronous measurements from sensors running at different frequencies and (iii) it is now possible to compute a position/velocity at any point in time along a trajectory and so it is possible to re-render scans at a sub-scan resolution compensating for the egomotion during scan acquisition.

We begin in Section II by introducing our notation and showing how to do Hybrid SLAM in Dynamic Environments (HSLAMIDE)[1]; Section III explains how to use cubic splines within the framework; Section IV shows the results of using the system on real radar data and finally Section V concludes and discusses ideas for future work.

II. HYBRID SLAM IN DYNAMIC ENVIRONMENTS

We will now introduce our method for Hybrid SLAM in Dynamic Environments (HSLAMIDE). The most important changes from traditional SLAM are: (i) we use a hybrid representation using occupancy grids, cubic splines and point features; (ii) the map becomes time dependent and (iii) model-selection parameters are introduced. For simplicity we will begin by explaining the system without using cubic splines and then in Section III we will demonstrate how to retrofit splines to the system. Below is a list of the notation we will be using:

- τ: The beginning of the sliding window.
- T: The end of the sliding window.
- $\mathbf{x}_t$: The state vector at time t describing the vehicle's pose (location and orientation $[x, y, \theta]$).
- $\mathbf{r}_t^+$: A complete radar scan obtained at time t.
- $\mathbf{z}_t$: Range-bearing measurements extracted from the scan $\mathbf{r}_t^+$, which satisfy a constraint on permissible object size.
- $\mathbf{r}_t$: The residual radar scan at time t having removed the measurements $\mathbf{z}_t$.
- $\mathbf{m}^k$: State vector describing the location of object k.
- $\mathbf{O}$: The occupancy grid representing landmasses.
- $\mathbf{M}_t = \{\mathbf{m}_t^1, \ldots, \mathbf{m}_t^k\}$: The set of all objects at time t.
- $\mathbf{X}_{\tau:T} = \{\mathbf{x}_\tau, \ldots, \mathbf{x}_T\}$: The set of vehicle poses.
- $\mathbf{Z}_{\tau:T} = \{\mathbf{z}_\tau, \mathbf{r}_\tau, \ldots, \mathbf{z}_T, \mathbf{r}_T\}$: The set of all measurements i.e. radar scans + extracted range-bearing measurements.
- $\mathbf{M}_{\tau:T} = \{\mathbf{M}_\tau, \ldots, \mathbf{M}_T\}$: The map consisting of stationary and dynamic objects.
- $\mathbf{D}_{\tau:T} = \{\mathbf{d}_\tau, \ldots, \mathbf{d}_t\}$: The data-association.
- $\mathbf{V} = \{v^1, \ldots, v^k\}$: The model-selection parameters.

Figure 1 is a Bayesian network that shows our formulation of the HSLAMIDE problem. The joint distribution corresponding to Figure 1 is:

$$P(\mathbf{X}_{\tau:T}, \mathbf{M}_{\tau:T}, \mathbf{O}, \mathbf{D}_{\tau:T}, \mathbf{V}, \mathbf{Z}_{\tau:T}, \mathbf{R}_{\tau:T}) = P(\mathbf{x}_\tau, \mathbf{M}_\tau) P(\mathbf{O}) P(\mathbf{d}_{\tau:T}) P(\mathbf{V}) \times \prod_{t=\tau+1}^{T} \Big\{ P(\mathbf{z}_t|\mathbf{x}_t, \mathbf{M}_t, \mathbf{d}_t) P(\mathbf{x}_t|\mathbf{x}_{t-1}) P(\mathbf{M}_t|\mathbf{M}_{t-1}, \mathbf{V}) P(\mathbf{r}_t|\mathbf{x}_t, \mathbf{O}) \Big\}, \quad (1)$$

where:

- $P(\mathbf{x}_\tau, \mathbf{M}_\tau)$ is the prior on the vehicle state and map at the beginning of the sliding window, which has a mean $\tilde{\mathbf{p}}_\tau$ and covariance $\mathbf{P}_\tau$.
- $P(\mathbf{O})$ is the prior on the occupancy grid and is taken to be 0.4 for each grid cell.
- $P(\mathbf{d}_{\tau:T})$ is the prior on the data-association and is taken to be the uninformative uniform distribution.
- $P(\mathbf{V})$ is the prior on the model-selection parameters and assumes that new objects are dynamic (refer to step 3 of the algorithm for details).
- $P(\mathbf{z}_t|\mathbf{x}_t, \mathbf{M}_t, \mathbf{d}_t)$ is the measurement model for objects in the map i.e. the probability of the measurement $\mathbf{z}_t$ given the vehicle pose $\mathbf{x}_t$, the map $\mathbf{M}$ and the data-association $\mathbf{d}_t$.
- $P(\mathbf{x}_t|\mathbf{x}_{t-1})$ is the motion model i.e. the probability of the new pose $\mathbf{x}_t$ given the last vehicle pose $\mathbf{x}_{t-1}$.
- $P(\mathbf{M}_t|\mathbf{M}_{t-1}, \mathbf{V})$ is the motion model for the map given the current estimate of the model-selection parameters. We use a constant position for stationary objects and three constant velocity models with noise in $\dot{x}$ and $\dot{y}$ for dynamic objects.
- $P(\mathbf{r}_t|\mathbf{x}_t, \mathbf{O})$ is the measurement model for a single (residual) radar scan i.e. the probability of the radar scan $\mathbf{r}_t$ given the vehicle pose $\mathbf{x}_t$ and the occupancy grid $\mathbf{O}$.

Solving (1) with a single optimisation is intractable and so we propose the following steps based on generalised

[1]Note that we use the terminology "Dynamic Environments" to refer to an environment with dynamic objects, not an environment which evolves, say, seasonally.

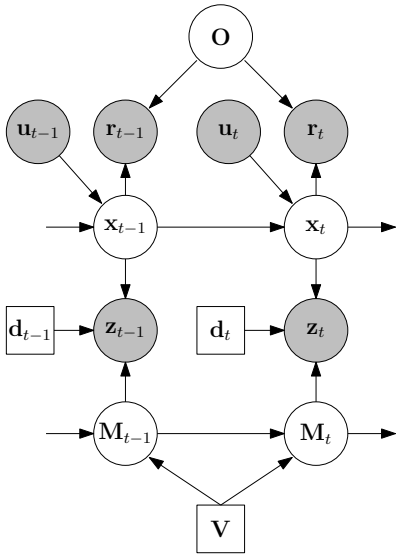

Fig. 1. A Bayesian network representing HSLAMIDE (Note:- Square boxes indicate discrete variables).

expectation maximisation [11] to solve the optimisation in real-time (see [2] for details):

1) $\mathbf{D}^{'} = \arg\max_{\mathbf{D}} P(\mathbf{X}, \mathbf{M}, \mathbf{O}, \mathbf{D}, \mathbf{V}, \mathbf{Z}, \mathbf{R})$
2) $\{\mathbf{X}^{'}, \mathbf{M}^{'}\} = \arg\max_{\{\mathbf{X},\mathbf{M}\}} P(\mathbf{X}, \mathbf{M}, \mathbf{O}, \mathbf{D}^{'}, \mathbf{V}, \mathbf{Z}, \mathbf{R})$
3) $\mathbf{V}^{'} = \arg\max_{\mathbf{V}} P(\mathbf{X}^{'}, \mathbf{M}^{'}, \mathbf{O}, \mathbf{D}^{'}, \mathbf{V}, \mathbf{Z}, \mathbf{R})$
4) $\mathbf{O}^{'} = \arg\max_{\mathbf{O}} P(\mathbf{X}^{'}, \mathbf{M}^{'}, \mathbf{O}, \mathbf{D}^{'}, \mathbf{V}^{'}, \mathbf{Z}, \mathbf{R})$

Note: The subscript $_{\tau:T}$ has been dropped from all terms to save space and the $'$ notation indicates the new estimate. We will now explain in detail each of the steps.

Step 1: performs the data-association using a probabilistic data-association filter [1] with an initial Mahalanobis gate of 4 and a uniform distribution modeling the outlier process. This method allows uncertainty in data-association to be modeled and adds robustness to outliers.

Step 2: is a least-squares optimisation for the vehicle trajectory and the map. Taking the logarithm of (1) and using the notation $\|\mathbf{x}\|^2_{\mathbf{P}} = \mathbf{x}^T\mathbf{P}^{-1}\mathbf{x}$ we can write:

$$\{\mathbf{X}^{'}, \mathbf{M}^{'}\} = \underset{\{\mathbf{X},\mathbf{M}\}}{\arg\min}\Bigg\{ \|\tilde{\mathbf{p}}_\tau - \mathbf{p}_\tau\|^2_{\mathbf{P}_\tau} + \sum_{t=\tau}^{T}\Big\{ \|f_x(\mathbf{x}_{t-1}) - \mathbf{x}_t\|^2_{\mathbf{Q}^0} + \|g(\mathbf{r}_t, \mathbf{O}) - \mathbf{x}_t\|^2_{\mathbf{B}} + \sum_{k=1}^{K}\Big(\big\|f_m(\mathbf{m}^k_{t-1}, v_k) - \mathbf{m}^k_t\big\|^2_{\mathbf{Q}^k_{v_k}} + \Big\|h(\mathbf{x}_t, \mathbf{m}^{d^k_t}_t) - \mathbf{z}_t\Big\|^2_{\mathbf{R}}\Big)\Big\}\Bigg\}, \quad (2)$$

where $\|\tilde{\mathbf{p}}_\tau - \mathbf{p}_\tau\|^2_{\mathbf{P}_\tau}$ is the Gaussian prior on the vehicle's state and the map at the beginning of the sliding window. $\|f_x(\mathbf{x}_{t-1}) - \mathbf{x}_t\|^2_{\mathbf{Q}^0}$ is the motion model of the vehicle taken to be constant velocity with covariance $\mathbf{Q}^0$. $\|g(\mathbf{r}_t, \mathbf{O}) - \mathbf{x}_t\|^2_{\mathbf{B}}$ is the registration between a single (residual) radar scan $\mathbf{r}_t$ and the occupancy grid $\mathbf{O}$ given an estimate of the vehicle's pose (location and orientation $[x, y, \theta]$) with covariance $\mathbf{B}$. This registration is computed using the level-set based registration technique described in [3]. $\big\|f_m(\mathbf{m}^k_{t-1}) - \mathbf{m}^k_t\big\|^2_{\mathbf{Q}^k_{v_k}}$ is the motion model of the object, which can be one of four motion models: stationary with zero uncertainty or constant velocity with three different levels of covariance $\mathbf{Q}^k_{v_k}$. Finally, $\Big\|h(\mathbf{x}_t, \mathbf{m}^{d^k_t}_t) - \mathbf{z}_t\Big\|^2_{\mathbf{R}}$ is a range-bearing measurement model with covariance $\mathbf{R}$ for the measurement $\mathbf{z}_t$.

Step 3: computes the model-selection parameters $\mathbf{V}^{'}$ and is computed using the following discrete Bayesian update:

$$P(v_k|\mathbf{m}^k_t) = \frac{P(\mathbf{m}^k_t|v_k)P(v_k|\mathbf{m}^k_{t-1})}{P(\mathbf{m}^k_t)}. \quad (3)$$

$P(v_k|\mathbf{m}^k_{t-1})$ is initially set to $[0.1, 0.3, 0.3, 0.3]$, where the first element corresponds to the stationary model and then the next three consecutive elements correspond to increasing amounts of motion model noise. The term $P(\mathbf{m}^k_t|v_k)$ is the likelihood for a given model and is a Gaussian on velocity with standard deviations of $[1, 5, 10, 20]$ knots for the four models respectively. We then take the Maximum a Posteriori (MAP) estimate for use in the motion model. It is worth noting that once the stationary model is selected there is no way to go back to dynamic, since the stationary motion model is a hard constant position and hence any subsequent velocities are zero (in practice this is dealt with by adding a new dynamic landmark).

Step 4: re-renders the occupancy grid based on the updated vehicle trajectory $\mathbf{X}^{'}$ and the radar scans $\{\mathbf{r}_\tau, \ldots, \mathbf{r}_T\}$. This step is implemented using the Graphics Processing Unit (GPU) and makes the standard assumption of independence between grid cells [17]. The posterior for each grid cell is computed using floating point textures and fragment shaders on the GPU. The benefit of this is that we can re-render several minutes worth of radar data within a few milliseconds on standard hardware. Our underlying cubic spline representation (see Section III) allows us to re-render at a sub-scan resolution i.e. we can compensate for the egomotion of the vehicle during the time taken to acquire a radar scan.

New Objects: these are added as part of the data-association step. A new object must have a Mahalanobis distance of at least 16 from every existing object – the reasoning behind this is that we want to be absolutely sure that the measurement generating a new object was not generated from an object in the system. Once a new object is detected it is then added using the predicted object location (given the vehicle pose and the measurement) and the corresponding uncertainty.

Object Deletion: we have three criteria for deleting objects: (i) if no measurements are associated to it within the sliding window; (ii) if the measurement density is less than 30% during the first 10 seconds of an object's life and (iii) if the object velocity exceeds 40 knots.

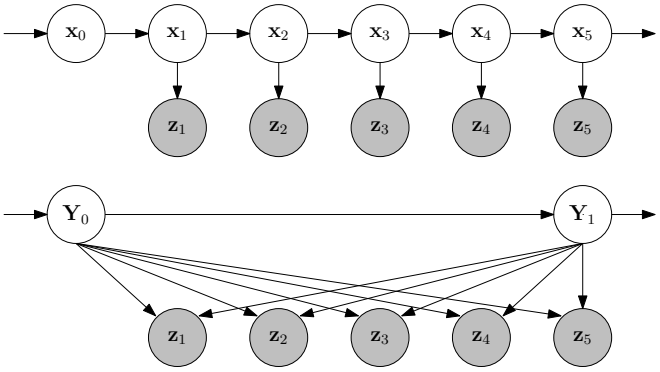

Fig. 2. Bayesian Networks: (top) a traditional formulation and (bottom) our method using splines.

Object Merging: because we use a probabilistic data-association filter it is possible that two initially separate objects can converge onto the same trajectory by sharing measurements. We deal with this problem by measuring the sum-of-squared differences between overlapping trajectories and if it is small enough we merge the two objects.

III. CUBIC SPLINES AS A CONTINUOUS TRAJECTORY REPRESENTATION

The key concept is to represent the vehicle's trajectory $[\mathbf{x}_0, \ldots, \mathbf{x}_t]$ as a set of cubic spline sections, which requires a much smaller parameter set compared with the requirement for a full representation. For instance, if a 2D vehicle is moving with a constant rate of change of acceleration for 2 seconds and its fastest sensor runs at 100Hz (e.g. an inertial sensor), then those two hundred poses (200 poses $\times$ 3 parameters/pose = 600 parameters) can now be represented by 12 spline parameters i.e. 2% of the original size. This huge compression then allows us to solve much larger sliding window lengths. The second major benefit is that the trajectory now has a continuous representation rather than discrete time steps, which makes it easy to deal with asynchronous measurements/constraints within the system. Figure 2 illustrates the difference between a traditional approach and our method using splines. The top Bayesian network represents a traditional approach where for each incoming measurement $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_5]$ (e.g. a radar, sonar or laser scan) there is a corresponding pose in the state vector $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_5]$. In contrast, our method uses a cubic spline section described by the two knots $[\mathbf{Y}_0, \mathbf{Y}_1]$ to represent the poses. This has two consequences: (i) each measurement constraint in the original Bayesian network is now projected into two constraints, one for each knot and (ii) the motion model constraints are now represented by a single constraint between the knots. We will now explain how the top Bayesian network can be solved using linear algebra and then elaborate on how to introduce cubic splines, highlighting the required modifications.

A. The Traditional Solution

The joint distribution for the top Bayesian network is:

$$P(\mathbf{X}, \mathbf{Z}) = P(\mathbf{x}_0) \prod_{t=1}^{5} P(\mathbf{z}_t|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{x}_{t-1}), \qquad (4)$$

where $P(\mathbf{x}_0)$ is the prior, $P(\mathbf{z}_t|\mathbf{x}_t)$ is a Gaussian measurement model and $P(\mathbf{x}_t|\mathbf{x}_{t-1})$ is a Gaussian motion model. Let us now take the logarithm of (4) to obtain a non-linear least-squares problem:

$$\mathbf{X} = \arg\min_{\mathbf{X}} \Big\{ \|\tilde{\mathbf{x}}_0 - \mathbf{x}_0\|^2_{\mathbf{P}} + \sum_{t=1}^{T} \big(\|f(\mathbf{x}_{t-1}) - \mathbf{x}_t\|^2_{\mathbf{Q}} + \|h(\mathbf{x}_t) - \mathbf{z}_t\|^2_{\mathbf{R}} \big) \Big\}. \qquad (5)$$

This minimisation can be solved by linearising the non-linear terms and re-writing as a matrix equation:

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \Big\{ \|\boldsymbol{\delta}\mathbf{x}_0 - \{\mathbf{x}_0 - \tilde{\mathbf{x}}_0\}\|^2_{\mathbf{P}} + \sum_{t=1}^{T} \big(\|\{\mathbf{F}_{t-1}\boldsymbol{\delta}\mathbf{x}_{t-1} + \boldsymbol{\delta}\mathbf{x}_t\} - \{\mathbf{x}_t - f(\mathbf{x}_{t-1}, \mathbf{u}_t)\}\|^2_{\mathbf{Q}} + \|\{\mathbf{H}_t\boldsymbol{\delta}\mathbf{x}_t\} - \{\mathbf{z}_t - h(\mathbf{x}_t)\}\|^2_{\mathbf{R}} \big) \Big\}, \qquad (6)$$

where $\mathbf{F}_{t-1}$ is the Jacobian of $f(.)$ w.r.t. $\mathbf{x}_{t-1}$ and $\mathbf{H}_t$ is the Jacobian $h(.)$ w.r.t. $\mathbf{x}_t$. Equation (6) can be factorised and written as a system of linear equations ($\mathbf{A}\mathbf{x} = \mathbf{b}$), for a detailed look at this process refer to [7].

B. Using Cubic Splines

Each cubic spline is constructed with N piecewise third-order polynomials passing through control points (knots) $[\mathbf{Y}_0, \ldots, \mathbf{Y}_N]$. These knots are parameterised by their position $\mathbf{y}_n = \{x_n, y_n, \theta_n\}$ and velocity $\dot{\mathbf{y}}_n = \{\dot{x}_n, \dot{y}_n, \dot{\theta}_n\}$. If p is a parameter in the range $[0 \ldots 1]$ and n references one of the N spline sections then a single spline section is defined as:

$$\mathbf{ss}_n(p) = \mathbf{a}_n + \mathbf{b}_n p + \mathbf{c}_n p^2 + \mathbf{d}_n p^3$$

where

$$\begin{aligned} \mathbf{a}_n &= \mathbf{y}_n \\ \mathbf{b}_n &= \dot{\mathbf{y}}_n \\ \mathbf{c}_n &= 3(\mathbf{y}_{n+1} - \mathbf{y}_n) - 2\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n+1} \\ \mathbf{d}_n &= 2(\mathbf{y}_n - \mathbf{y}_{n+1}) + \dot{\mathbf{y}}_n + \dot{\mathbf{y}}_{n+1}. \end{aligned}$$

Let us now consider the vehicle's trajectory as a spline – given the spline parameters $\boldsymbol{\Phi} = \{\mathbf{Y}_0, \ldots, \mathbf{Y}_N\}$ and a time t we can write a function that returns the vehicle's state:

$$\mathbf{s}(t,\mathbf{\Phi}) = \begin{bmatrix} x_n(p) \\ y_n(p) \\ \theta_n(p) \end{bmatrix} = \begin{bmatrix} a_n^x + b_n^x p + c_n^x p^2 + d_n^x p^3 \\ a_n^y + b_n^y p + c_n^y p^2 + d_n^y p^3 \\ a_n^\theta + b_n^\theta p + c_n^\theta p^2 + d_n^\theta p^3 \end{bmatrix}$$
$$n = \left\lfloor \frac{t}{\lambda} \right\rfloor$$
$$p = \frac{t}{\lambda} - n, \tag{7}$$

where λ is the number of time steps per spline section, which for simplicity is assumed constant, and the function $\lfloor . \rfloor$ computes the greatest integer less than the argument. In practice we allow λ to change and so solving for n and p turns into a binary search. We also require the jacobian of $\mathbf{s}(t,\mathbf{\Phi})$ w.r.t. $\mathbf{\Phi}$:

$$\mathbf{S}_t = \frac{\partial s(t,\mathbf{\Phi})}{\partial \mathbf{\Phi}} = \begin{bmatrix} k & 0 & 0 & l & 0 & 0 & m & 0 & 0 & n & 0 & 0 \\ 0 & k & 0 & 0 & l & 0 & 0 & m & 0 & 0 & n & 0 \\ 0 & 0 & k & 0 & 0 & l & 0 & 0 & m & 0 & 0 & n \end{bmatrix}$$

where

$$k = 1 - 3p^2 + 2p^3$$
$$l = p - 2p^2 + p^3$$
$$m = 3p^2 - 2p^3$$
$$n = -p^2 + p^3$$

this jacobian $\mathbf{S}_t$ only depends upon the scalar value p and is therefore suitable for a look up table implementation.

To modify (5) and (6) to a spline representation, we make the direct substitution $\mathbf{X} = \mathbf{\Phi}, \mathbf{x}_t = \mathbf{s}(t,\mathbf{\Phi})$ and include $\mathbf{S}_t$ to project any jacobians w.r.t $\mathbf{x}_t$ onto the spline parameter set $\mathbf{\Phi}$, here are the modified versions:

$$\mathbf{\Phi} = \arg\min_{\mathbf{\Phi}} \Big\{ \|\tilde{\mathbf{x}}_0 - \mathbf{s}(0,\mathbf{\Phi})\|^2_{\mathbf{P}} + \sum_{t=1}^{T} \big(\|f(\mathbf{s}(t-1,\mathbf{\Phi})) - \mathbf{s}(t,\mathbf{\Phi})\|^2_{\mathbf{Q}} + \|h(\mathbf{s}(t,\mathbf{\Phi})) - \mathbf{z}_t\|^2_{\mathbf{R}} \big) \Big\}. \tag{8}$$

$$\hat{\mathbf{\Phi}} = \arg\min_{\mathbf{\Phi}} \Big\{ \|\mathbf{S}_0 \delta\mathbf{\Phi} - \{\mathbf{s}(0,\mathbf{\Phi}) - \tilde{\mathbf{x}}_0\}\|^2_{\mathbf{P}} + \sum_{t=1}^{T} \big(\|\{\mathbf{H}_t \mathbf{S}_t \delta\mathbf{\Phi}\} - \{\mathbf{z}_t - h(\mathbf{s}(t,\mathbf{\Phi}))\}\|^2_{\mathbf{R}} + \|\{\mathbf{F}_{t-1}\mathbf{S}_{t-1}\delta\mathbf{\Phi} - \mathbf{S}_t\delta\mathbf{\Phi}\} - \{\mathbf{s}(t,\mathbf{\Phi}) - f(\mathbf{s}(t-1,\mathbf{\Phi}))\}\|^2_{\mathbf{Q}} \big) \Big\}. \tag{9}$$

C. Consequences Of The Spline Representation

By parameterising a set of poses with a cubic spline section we are setting a hard motion model (fixed rate of change of acceleration $\dddot{\mathbf{x}}$) for each spline section i.e. between the knots. All vehicles in the 'real-world' have inertia and in consequence many usually exhibit relatively smooth trajectories, at least on some scale. This fact is used in typical Kalman Filter style solutions to give a prior on the likely motion a vehicle might undergo. Our cubic spline representation assumes a fixed rate of change of acceleration along the spline section and is therefore able to deal with constant position, constant velocity, constant acceleration and constant rate of change of acceleration motion models. Therefore, our cubic spline method is able to not only capture the dynamics of our marine vessel, but also the large majority of vehicles that are typically modeled in the robotics community. Naturally there are some exceptions (either always, or more typically, for a short time period) where this dynamic model is not appropriate. In these cases knots can be placed at every time instance, which is identical to the standard solution, though of course we lose the compression available in the more usual case.

Let us now consider two cases: (i) what happens along a spline section and (ii) what happens at the knots. The hard motion model is enforced along the spline section implicitly because of the cubic spline representation. At the knots, we fix $\mathbf{x}$ and $\dot{\mathbf{x}}$ to be equal (C1 continuity), but allow $\ddot{\mathbf{x}}$ and $\dddot{\mathbf{x}}$ to change to anything i.e. a uniform probability distribution over $\ddot{\mathbf{x}}$ and $\dddot{\mathbf{x}}$.

Note also, that the splines are a particular case of a Gaussian Process [4]. Here we have a particular choice of basis set, given by the spline basis. Gaussian distributions over the knots $\mathbf{Y}$ in the Bayes' net (Figure 2) lead to a distribution over the spline functions. We can, in particular, find the covariance associated with *any* point on the spline via the jacobians $\mathbf{S}$ as $\mathrm{cov}(\mathbf{s}(t,\mathbf{\Phi})) = \mathbf{S}_t \,\mathrm{cov}(\mathbf{\Phi}) \mathbf{S}_t^\top$.

D. Knot Placement

Initially the trajectory is over represented with a knot for every pose (the traditional solution). The system then removes knots selectively where $\ddot{\mathbf{x}}$ and $\dddot{\mathbf{x}}$ are sufficiently similar. Thus, knots only remain where they are required to accurately represent the trajectory; the representation automatically finds the correct number of knots based on the current environmental conditions e.g. more in a rough sea than in a calm sea. We explore the effect of knot placement in more detail in the subsequent results section.

While this procedure is chosen for expedience, a more rigorous procedure might use, say, Minimum Description Length to determine the correct number and placement of knots, as in [6]. Further, at present we do not consider the case of adding knots back in. We have not had to do this in any of our experiments, and we expect such occurrences to be very rare. Nevertheless, it might be necessary if, for example, the system changes crucial data-associations within the sliding window (which of course, it can) meaning a trajectory has been incorrectly represented by a spline. One way to achieve this would be to recompute the splines whenever data associations change, or to monitor the spline's innovation sequence for biases, which would be indicative of an inappropriate model. Note that this is possible within the sliding window since no decisions have been committed to the filter via marginalisation.

IV. RESULTS

We present results using radar data obtained from a 50 minute voyage on a small boat using a Garmin GMR18. The principle of radar is to send out pulses of microwave radiation and to measure the time-of-flight for reflections to be received back, enabling accurate measurements of both the bearing and range of objects. Figure 3 shows a sample of raw radar data from our experiments. We use a sliding window length of 90 seconds to achieve the temporal fusion of the radar data, Figure 4 and Figure 5 show examples of the typical output of the system. The occupancy grid is drawn in the background and overlaid with the stationary and dynamic objects (each object in the system is given a unique identifier). The dynamic objects are represented by cubic splines, which are drawn in black and the covariance (which can be computed for any time along the spline) is drawn at regular intervals in a light grey.

Figure 8 shows the results obtained if the occupancy grid is only fused once at the head of the sliding window as it would be in a filtering style solution to the problem. There are three clear inadequacies visible in the figure: (i) the occupancy grid exhibits incorrect shadowing effects (caused by radar data falling into the wrong grid cell); (ii) the range-bearing measurements extracted from the radar sweeps have large amounts of noise and (iii) the system suffers from false-negatives and false-positives. All of these problems stem from a poor estimate of the egomotion. In contrast, Figure 9 shows the results of our proposed method using a 90 second sliding window to continually re-estimate the occupancy grid, egomotion, dynamic objects, model-selection and data-association. The results are that: (i) the occupancy grid is clear and sharp giving a good representation of the surrounding environment; (ii) the range-bearing measurements have much less noise on them making it easier to obtain better estimates of the trajectories of dynamic objects and (iii) because of the reduced noise in the range-bearing measurements the model-selection is better at correctly classifying the objects.

Figure 6 shows how a sampled spline trajectory from the voyage degrades as an increasing number of knots are removed. The figure is generated by placing a knot at every time step (equivalent to the traditional representation), then progressively removing one knot at a time based on which will have the smallest effect (see Section III-D) and then computing the RMS error between the traditional trajectory and the one using the spline. The figure shows that 80-90% of the knots can be removed before a sharp degradation in the quality of the trajectory (we have observed similar results in terms of the quality of the occupancy grid against knot reduction). It is worth noting at this point that errors of 1m are small given the 10m vessel and the the radar sensor which has a range of 2.8km and a resolution of approximately 9m.

Figure 7 shows the distribution of the compression ratio (percentage of knots removed w.r.t the full representation) for the spline trajectories, this distribution is generated using samples taken from the 50 minute voyage at 2 second intervals. The results are that on average the spline trajectories are achieving a compression ratio of 70-90%, which matches well with the sampled trajectory in Figure 6. This gives an indication that the automatic knot removal procedure is leaving knots at the correct locations and shows that a significant compression is achieved resulting in a smaller system to solve and hence reducing the required computation. The entire system currently runs at 10Hz on a high-end quad-core PC.

For results with ground truth (a marine chart) please refer to the accompanying video material.

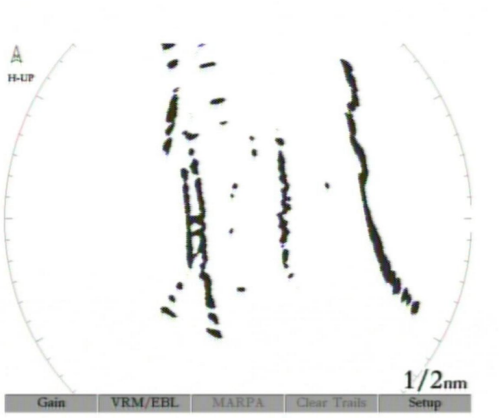

Fig. 3. The raw radar data extracted from the GMR18. Each 360^o scan completes in 1.5 seconds.

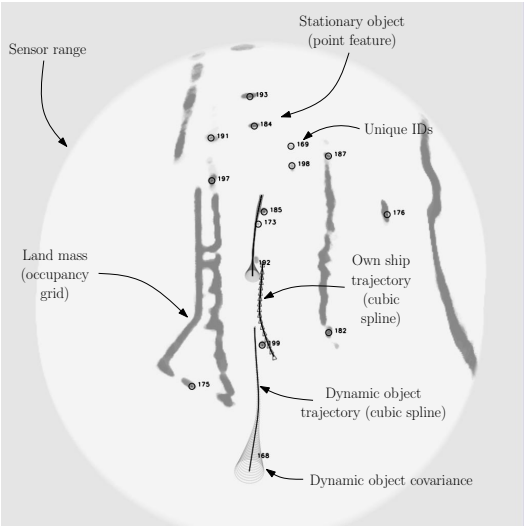

Fig. 4. The system in action – the occupancy grid represents land masses, point features are used for stationary objects and cubic splines are used to represent the trajectories of dynamic objects.

V. CONCLUSIONS AND FUTURE WORK

We have demonstrated how hybrid mapping and sliding window estimation can be used to make a SLAM system that works in complicated dynamic environments using noisy sensors. Our hybrid mapping gives a rich representation

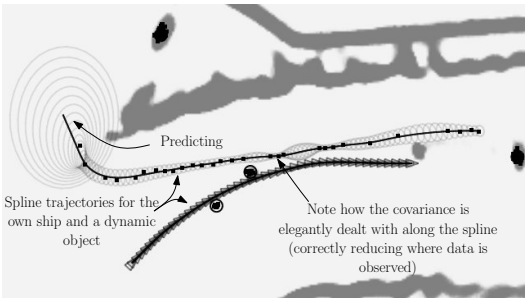

Fig. 5. Example of spline representation in the system.

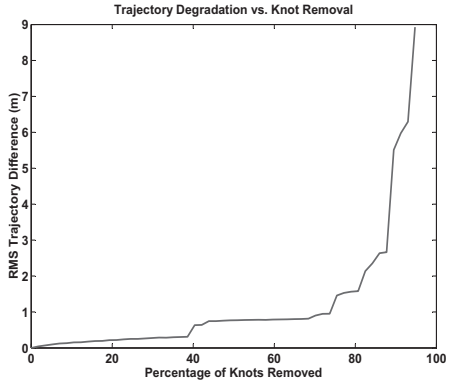

Fig. 6. Quantitative analysis of how trajectories degrade as they are represented using fewer and fewer knots.

of the underlying environment, with an occupancy grid to represent land masses, point features to represent smaller stationary objects and cubic splines to represent the trajectories of dynamic objects. We have also shown how cubic splines can easily be retrofitted to any estimation framework containing dynamic objects and how this achieves state compression, and makes it easy to deal with asynchronous measurements from sensors running at different frequencies. These innovations enable us to perform SLAM in dynamic environments that were too difficult for our previous method [2].

Such a system could be useful to the operator of a marine vessel when dealing with radar data because: (i) intermittent measurements are fused making them clearly visible as stationary objects; (ii) clutter is rejected; (iii) dynamic objects are automatically tracked with the appropriate motion model and (iv) measurements are fused within an occupancy grid to give a clearer picture of the surrounding environment. We are currently looking at how to enrich the information in this system with visual data. The consequences of the spline representation go beyond the present work and it would be very interesting, for example, to investigate splines as a means to perform visual SLAM with a rolling shutter camera.

We gratefully acknowledge the sponsorship of Servowatch Systems Ltd.

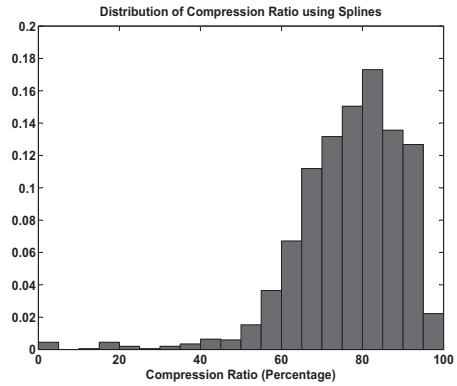

Fig. 7. Quantitative analysis of the amount of trajectory compression achieved during the experiment.

REFERENCES

[1] Y. Bar-Shalom and T. Fortmann. *Tracking and data association*. Academic Press, 1988.

[2] C. Bibby and I. Reid. Simultaneous localisation and mapping in dynamic environments (SLAMIDE) with reversible data association. In *Proc. of Robotics Science and Systems*, 2007.

[3] C. Bibby and I. Reid. Robust real-time visual tracking using pixel-wise posteriors. In *Proc. 10th European Conf. on Computer Vision, Marseille, France*, 2008.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[5] W. Burgard, D. Fox, H. Jans, C. Matenar, and S. Thrun. Sonar-based mapping of large-scale mobile robot environments using em. In *Proc. 16th Int'l Conf. on Machine Learning*, pages 67–76, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[6] T. Cham and R. Cipolla. Automated b-spline curve representation incorporating mdl and error-minimizing control point insertion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:49–53, 1999.

[7] F. Dellaert and M. Kaess. Square root SAM. *Int'l Journal of Robotics Research*, 2006.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 1977.

[9] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Trans. Robotics and Automation*, 3:249–265, 1987.

[10] H. Moravec. Sensor fusion in certainty grids for mobile robots. *Artificial Intelligence Magazine*, 9:61–74, 1988.

[11] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*. Kluwer, 1998.

[12] J. Nieto, J. Guivant, and E. Nebot. The hybrid metric maps (hymms): A novel map representation for denseslam. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 391–396, 2004.

[13] A. K. Pandey, K. M. Krishna, and M. Nath. Feature based occupancy grid maps for sonar based safe-mapping. In *Proc. Int'l Joint Conf. on Artificial Intelligence*, 2007.

[14] L. Pedraza, G. Dissanayake, J. Valls Miro, D. Rodriguez-Losada, and F. Matia. Bs-slam: Shaping the world. In *Proc. of Robotics Science and Systems*, Atlanta, GA, USA, 2007.

[15] G. Sibley, G. S. Sukhatme, and L. Matthies. Constant time sliding window filter SLAM as a basis for metric visual perception. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2007.

[16] S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. In *Machine Learning*, pages 29–53, 1998.

[17] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[18] C. C. Wang, C. Thorpe, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, Taipei, Taiwan, 2003.

[19] D. Wolf and G. S. Sukhatme. Online simultaneous localization and mapping in dynamic environments. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, New Orleans, Louisiana, 2004.

Fig. 8. Qualitative evaluation: Re-rendering turned off.

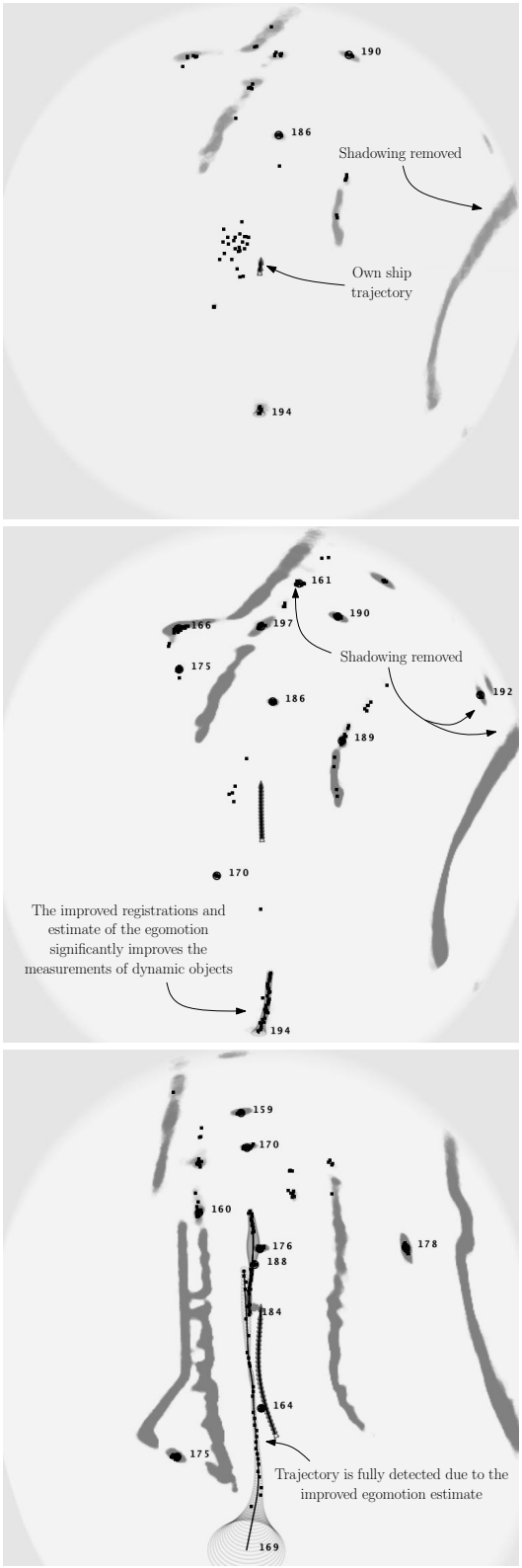

Fig. 9. Qualitative evaluation: Re-rendering turned on.