Towards Constant-Time SLAM on an Autonomous Underwater Vehicle Using Synthetic Aperture Sonar

Paul M. Newman[1], John J. Leonard[2], Richard J. Rikoski[2]
[1]Oxford University Department of Engineering Science
[2]MIT Dept. of Ocean Engineering *

April 2003

Abstract

This paper applies a new constant-time, consistent and convergent Simultaneous Localization and Mapping (SLAM) algorithm to an autonomous underwater vehicle (AUV). A constant-time SLAM algorithm offers computation independent of workspace size and is one key component in the development of truly autonomous agents. The real-time deployment of such a system would be a landmark achievement for the mobile robotics community. This paper describes progress towards this goal focusing on the sub-sea domain — an area set to benefit massively from the autonomy afforded by SLAM. The primary sensor used in this work is a sixteen element synthetic aperture sonar (SAS) carried on the nose of the AUV "Caribou". Using a novel target detection strategy, data gathered from a 40 minute survey is processed by the new SLAM algorithm and the results compared to both a ground truth and the quadratic time "gold standard" full covariance SLAM algorithm.

1 Introduction and Motivation

The Simultaneous Localization and Mapping (SLAM) problem is of fundamental importance in the quest for autonomous mobile machines. SLAM research seeks to enable a platform, starting with no prior information and using only onboard sensors, to move through its environment and build a consistent map of its surroundings as well as an estimate of its own trajectory.

SLAM is a hard problem [7, 23, 9, 22] and may not always be the best technique to employ to achieve mobile autonomy. For example, in many land based applications navigation and mapping can be aided by in-place infrastructure - GPS, radio beacons or geometric *a priori* maps for example. Navigation and mapping in the sub-sea domain is a different matter however. No wide-coverage underwater GPS equivalent exists (although some small oil and mineral rich areas are well populated with acoustic beacons at surveyed locations [31]). At present only a few percent of the earth's seabed has been explored. What publicly available bathymetry maps [20] do exist of explored areas are coarse and unsuitable for precision navigation (CEP < 20 m).

Underwater SLAM is a key technology in development of Autonomous Underwater Vehicles (AUVs). A SLAM enabled AUV could be deployed from the surface over unknown terrain and descend to execute a mission of unspecified duration. The state-of-the-art in AUV navigation technology uses a fusion of Doppler Velocity Log (DVL) and inertial measurements [17]. Although impressive performance has been reported by such systems (0.1% of distance travelled) they are essentially odometric in nature and prone to the accumulating drift and error growth inherent in all such dead-reckoning approaches. The immediate impact of sub-sea SLAM is likely to be in terms of bounded localization error rather than mapping. A SLAM algorithm on an AUV would still utilize doppler and inertial measurements. However it is the repeated measurement of the relationship between vehicle and distinct physical artifacts on the sea bed that would lead to bounded error in localization - even though the built map of the artifacts used to achieve this may not be of interest *per-se*.

One approach to the sub-sea navigation problem is 'on-the-fly' acoustic feature deployment[25]. In this scenario instead of detecting naturally occurring landmarks low cost or recoverable transponders are deployed in unknown locations which enable range only measurements between vehicle and beacon to be made. An optimization procedure is then employed to deduce both vehicle trajectory and beacon location. In some ways the problem is made easier by using transponders but in others much harder. Firstly the data-association problem is removed. The correspondences between measurements and features, in this case transponders, are known because each transponder replies to an acoustic interrogation at a unique frequency. However the ranges measured are huge in comparison to the movement of the vehicle between sequential measurements. Hence not only are the transponder locations only partially observable (because of the range-only data) but also the problem is inherently ill conditioned over small vehicle path lengths.

The range only SLAM problem has much in common with the structure from motion (SFM) problem [1, 13, 9, 22, 21] which in turn has a strong duality with bearing only and range only SLAM [4, 5]. The motivation for this approach is clear. Submerged, on-the-fly calibration of transponders would enable an AUV to lay and extend its own beacon network to fit adaptive mission and navigation criteria. Alternatively, the transponders could simply be dropped from a surface vessel or air-craft with no regard to calibration. With small, cheap acoustic transponders now available, this approach could find application over barren flat terrain and complex reefs alike. Figure 1 shows the results of a non-linear least squares bundle adjustment [13] solving for both vehicle trajectory and transponder locations.

Although workable, the in-mission deployment of features is only an interim solution that is likely to be unsuitable for

*The authors would look to extend their great thanks to all the members of the GOATS 2002 team, Don Eickstedt, MIT Sea Grant, Mike Bosse, Henrik Schmidt and the MIT Acoustics Group.

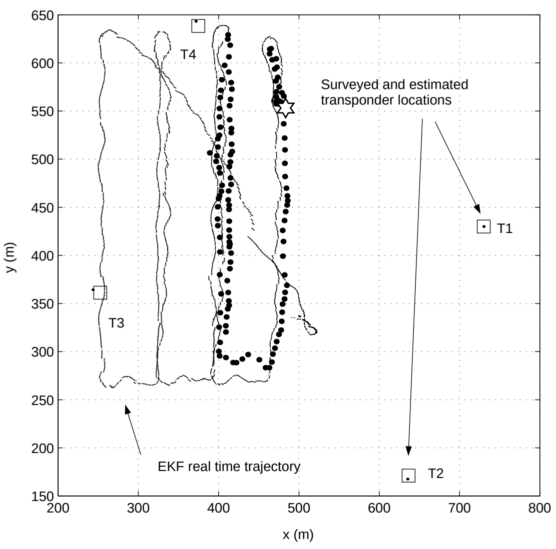


Figure 1: Result of the range-only optimization. The GPS derived positions of the transponders are shown as squares and the estimated locations as nearby black dots. The estimated vehicle trajectory is plotted on top of that derived from the onboard navigation (using surveyed beacon locations). The AUV surfaced at the end of the dive and acquired a GPS fix (shown as a star) which is coincident with both the onboard navigation and range only estimated tracks.

a substantial proportion of AUV applications. It is far more preferable to be self-contained and use naturally occurring landmarks (if available) to aid the navigation process. This paper focuses on this ideal.

Section 2 discusses contemporary work in the SLAM problem especially with regard to issues relating to operating in truly large areas. In the light of this, Section 3 presents a new constant-time SLAM algorithm, dubbed CTS. Section 4 takes the CTS algorithm and applies it to Synthetic Aperture Sonar (SAS) data collected from an AUV. The recorded onboard navigation estimates are used as a ground truth to compare SLAM derived vehicle trajectories. Finally Section 5 discusses areas of future and intended research that lie on the path to developing a truly autonomous, SLAM-enabled AUV.

2 Prior work and background

The seminal work of Smith, Self and Cheeseman [27] proposed an algorithm (commonly known as the “Full Covariance solution”) with complexity $\mathcal{O}(n^2)$ where n is the number of features mapped. The Full Covariance solution is accepted as the “Gold Standard” Algorithm in terms of performance (not applicability) and in the case of linear observation and vehicle models is provably optimal and consistent. The properties of this algorithm and its nomenclature are used by the new approach proposed in this paper and for completeness it is now summarized.

Let us assume that there are n features in the environment, and that they are static. The global frame, designated by G, is a unique, immutable coordinate-frame that is defined at the beginning of a mission. The true state at time k is designated by $\mathbf{x}(k) = [\mathbf{x}_v(k)^T\ \mathbf{x}_1(k)^T \ldots\ \mathbf{x}_n(k)^T]^T$, where $\mathbf{x}_v(k)$ represent the location of the vehicle, and $[\mathbf{x}_1(k)^T \ldots\ \mathbf{x}_n(k)^T]^T$ represents the locations of the environmental features. We assume that the vehicle moves from time k to time $k+1$ in response to a known control input, $\mathbf{u}(k)$, that is corrupted by noise. Let U^k designate the set of all control inputs from time 0 through time k, $Z(k)$ designate the set of sensor measurements obtained at time k, and Z^k designate the set of all measurements obtained from time 0 through time k. For each measurement $\mathbf{z}_j(k) \in Z(k)$, there is a corresponding assignment index $\mathbf{a}_j$. The value of $\mathbf{a}_j$ is i if measurement $\mathbf{z}_j(k)$ originates from feature i. Let A^k designate the set of all assignment indices from time 0 through time k. Assuming that the associations are known, the objective is to compute recursively the probability distribution for the location of the robot and the features, with reference to the global reference frame G, given the measurements, control inputs, and assignments:

$$p(\mathbf{x}_v(k), \mathbf{x}_1(k), \ldots, \mathbf{x}_{n_k}(k) | Z^k, A^k, U^{k-1}).$$

For the Linear-Gaussian (LG) SLAM problem, the Kalman filter provides the optimal estimate of this pdf, which is described by its mean $[\hat{\mathbf{x}}_v(k)^T \hat{\mathbf{x}}_1(k)^T \ldots\ \hat{\mathbf{x}}_n(k)^T]^T$ and covariance $\mathbf{P}(k)$. The properties of single-map LG SLAM solution are well-known [6].

Recent related work in SLAM includes submap decomposition methods [19, 10, 14, 32, 28], FastSLAM [23], sparse extended information filters (SEIF’s) [30], scan-matching [11, 29, 12] and topological approaches [16, 2]. To date, all published SLAM algorithms are subject to the scaling problem — the fact that the SLAM task gets harder as more and more features are mapped.

The unbounded growth of computation with map size essentially prohibits large scale sustainable mobile autonomy. Because of the difficulties encountered by SLAM algorithms when applied to larger environments, the “map scaling” problem has been identified as one of the key issues for research in this area. Davison’s [3, 15] “postponement method” and later Guivant and Nebot’s [10] Compressed Filter allow computational resources to be focused on maintaining a representation of the local area while postponing the computation required to manage all other mapped features. However eventually the same $\mathcal{O}(n^2)$ computation must be completed - once again placing a limit on the size of environment in which the algorithm can be deployed. In a similar vein, the Constrained Sub-map Filter [32] and the Geometric Projection Filter [24] seek to delay the full $\mathcal{O}(n^2)$ computation. Other techniques such as decoupled stochastic mapping [19] and SEIFs [30] achieve $\mathcal{O}(1)$ performance, but make approximations that require empirical testing to verify state estimation consistency.

3 A Constant-Time SLAM Algorithm

This section summarizes a new Constant-Time SLAM (CTS) algorithm, described in detail in [18]. Figure 2 illustrates the broad structure of the algorithm. The CTS algorithm is characterized by the following elements and criteria:

1. SLAM processing occurs within local maps.
2. Map management — updating and creating new maps
3. Map Location estimation — determining the best global location estimate for a submap

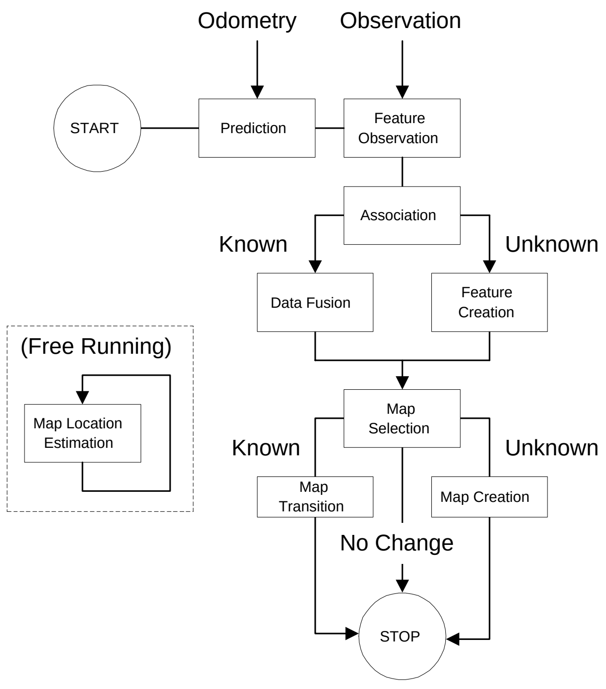


Figure 2: Flow chart for each cycle of the CTS algorithm. The top half of the figure is as expected for a standard SLAM algorithm with odometry and measurement data being used to update location estimates or augment the current map with new features. Following this the CTS algorithm makes a decision based on vehicle location which existing map if any should be made active. The map location "improvement loop" runs asynchronously and seeks to use differing estimates of features shared between maps (each map has an independent estimate of a shared feature) to improve the globally referenced feature uncertainty.

4. Map Transitioning — The relocation of a vehicle into a previously built submap.

5. Computation of global state estimates for all features in a map

Partitioning a robot's total work-space into a patchwork of two or more local active work-spaces has an intuitive appeal and this approach has been widely adopted. However, no previous method satisfies each of the three criteria of provable consistency,spatial convergence and constant-time updates. For example, Julier and Uhlmann [14] provide a consistent, constant-time algorithm for large-scale SLAM, based on split covariance intersection, but this method does not achieve "tight" convergence to the error bounds that would be obtained with a full covariance solution. Methods such as the compressed filter [10], the constrained local submap filter [32], and sequential map joining [28] are provably consistent and convergent, but are $\mathcal{O}(n^2)$, where n is the number of the features in the environment.

3.1 Definitions and Operations

We now define several terms and basic operations that will facilitate description of the new method. A **location vector** is a parameterization of both position and orientation of one coordinate-frame, i with respect to another, j. In $\mathcal{R}^2$ this is represented as a translation by $[x, y]$ followed by a rotation θ. These three parameters are encapsulated in the 3 vector $T_i^j = [x, y, \theta]^T$.

An **entity** is a parameterization of a vehicle or landmark. Each entity is labelled with a unique positive integer — this is referred to as the entity's ID. We can attach a coordinate frame $\mathcal{F}_i$ to any entity i and describe it using a location vector in another coordinate frame. The vector T_i^j should be understood to be a parameterization of a transformation from $\mathcal{F}_i$ to $\mathcal{F}_j$. The uncertainty in this transformation is represented by Σ_i^j.

A **map** is a collection of entities all described with respect to a local coordinate frame. Each map has a unique integer id. Each map has associated with it a **Map Root** entity i and a **Map Location** vector T_m^G. The Map Location vector describes the pose of a map's local coordinate frame in the global frame G. The local coordinate frame of a map is coincident with one of the entities in the local map; this entity is referred to as the Map Root. In other words, the Map Root lies at the origin of the local map, and it is the entity to which all other entities in a map are referenced. If an entity is the Map Root, then by definition its location vector will be $[0, 0, 0]^T$ and its global location vector given by T_m^G — the location of the map in global coordinates.[1] We use the pre-superscript notation $[m, j]$ to denote that all entities in map m are referenced to $\mathcal{F}_j$ where j is the root entity of the map.

We define **root-shifting** to be the operation, $\mathcal{S}$, that changes the root of a map from i to j. After this operation all location vectors in a map will be referenced to $\mathcal{F}_j$ rather than $\mathcal{F}_i$, $i, j \in m$.

Finally, the global location of a feature j in a local map m which is referenced to feature i, ${}^{[m,i]}T_j$ is computed by simple composition of the local location vector and the already globally referenced Map Location vector:

$$ {}^{[m,\cdot]}T_j^G = T_m^G \oplus {}^{[m,i]}T_j \qquad (1)$$

For ease of reference, Appendix A summarizes the notation of transformations and operations used in this paper.

3.2 Managing local maps

At any one time, there is a single active map. For each map m, we compute a partial solution, $p(\mathbf{x}_m(k)|Z_m^k, A_m^k, U_m^{k-1})$, where $\mathbf{x}_m(k)$ designates the local map state, and Z_m^k, A_m^k, and U_m^{k-1} represent subsets of the measurements, associations, and control inputs, respectively. Each measurement is used in only a single map. (This is vital for ensuring consistency of the global Map Location estimation process.) Each map m contains an estimated mean $[\hat{\mathbf{x}}_v^m(k)^T \hat{\mathbf{x}}_1^m(k)^T \dots \ \hat{\mathbf{x}}_n^m(k)^T]^T$ and covariance $\mathbf{P}^m(k)$ corresponding to selected vehicle locations (for time steps when map m is the active map), and only a subset of the features. These estimates are the same as the location vectors ${}^{[m,i]}T_j$ and associated uncertainty ${}^{[m,i]}\Sigma_j$ for the features in the local map. Note that no constraint is placed on the manner in which local SLAM is undertaken. For example, it need not be a Kalman Filter based — all that is required is a probablistic state estimation of vehicle and landmark parameterization.

[1] In the SLAM literature, the term "base reference" [28] is a synonym for our term Map Root. Note that in the general case with orientation, a single point feature will be insufficient to define a reference frame. In 2-D, two points will be required and in 3-D three points will be required.

For each map we define a map center to be the vehicle location at the time of the creation of a map. About this center is defined a region of radius r. This defines a bound on vehicle location and not feature locations — any feature that is observed from a position inside the map region will be added to the local map. The estimated location of the vehicle is used to deduce which map(s) the vehicle is in, and when to make transitions. When the vehicle travels more than $r + h$ from the center, the vehicle is considered to have left the current map. The parameter h is a hysteresis term, to prevent excessive map switching.

We assume that the density of discernible features in a local area is bounded. This provides a bound on the number of features that can belong to a map.

When a vehicle leaves a map, we must determine which map (if any) the vehicle has transitioned to. A list of possible candidates is drawn up from a look up table indexed by quantized vehicle locations. If more than one candidate exists, we choose the map with the lowest ID, i.e., the oldest map. If no candidates are found, then a new map is created at the current location (a distance $r + h$ from the center of the previous map). All of these operations can be performed in constant time.

3.3 Estimation with multiple maps

This section summarizes how CTS uses multiple *local* maps to reduce the uncertainty in *globally* referenced feature estimates. The corner-stone of the whole approach is the use of multiple estimates of shared features (from different maps) to provide alternative expressions for the global uncertainty in the map location vector. If, by re-referencing (root shifting) an entire map m to a shared feature, s, and then using an alternative global estimate of that new "root-feature"'s location, ${}^{[p,\cdot]}T_s^G$,using some other map p, the global uncertainty of other features in the map would be decreased then CTS performs the root shift and *replaces* T_p^G with ${}^{[p,\cdot]}T_s^G$ and Σ_p^G with ${}^{[p,\cdot]}\Sigma_s^G$. This procedure is called "Map Location Estimation".

For constant-time operation, Map Location estimation is performed only when the vehicle transitions from one map to another. Alternatively, the procedure can be performed periodically (or off-line at the end of the mission) to all maps. Multiple iterations result in global convergence to a near-optimal solution, but the computation complexity is no longer $\mathcal{O}(1)$. This estimation procedure is made explicit in the boxed algorithm in Figure 3 .

Figure 4 is a graphical representation of the map location estimation procedure. It shows the improvement of one map using a expression for the global uncertainty of a feature shared with another map.

In the steps listed in Figure 3 the criteria for new root selection is based on finding minimizing the minimum feature uncertainty the map selected for improvement. This is not the only sensible choice however. Depending on the degree of internal correlation between features in a map, the root shifting operation may increase the globally referenced uncertainty of some features while decreasing others. This motivates and alternative criteria — minimization of the median decrease in feature uncertainty. Out experiments show that this is an equally valid approach. Indeed, the choice of criteria is not crucial, the root shifting operation hastens the global convergence of the estimates by utilizing "better" estimates of features in other maps. However, once all maps are referenced to shared features, the convergence properties of the intra-map SLAM means that global convergence will occur, albeit at a slower rate, without further root shifting.

1. Select a map p, to improve which is currently referenced to the root entity i.
2. Create a set, $\mathcal{N}$, containing the ID's of all nearby maps including p — the map to be improved.
3. For each $q \in \mathcal{N}, \quad q \neq p$ create a set $\mathcal{C}_{p,q}$ of ID's of features that are present in both maps.
4. For the frame $\mathcal{F}_k$ attached to feature $k \in \mathcal{C}_{p,q}$, calculate its globally referenced location ${}^{[q,\cdot]}T_k^G$ and uncertainty ${}^{[q,\cdot]}\Sigma_k^G$ using the location estimate of feature k within map q and its current location estimate.
5. Pick the map $q^\star$ and entity ID $k^\star$ such that:
$$[q^\star, k^\star] = \arg\min\{|\ {}^{[q,\cdot]}\Sigma_k^G\ |\}$$
6. If $q^\star = p$ and $k^\star = i$ then stop. The map p cannot be improved
7. Root Shift map p to $k^\star$ from i — reference all entities in map p to a coordinate frame attached to entity $k^\star$.
8. Replace T_p^G and Σ_p^G with ${}^{[q^\star,\cdot]}T_{k^\star}^G$ and ${}^{[q^\star,\cdot]}\Sigma_{k^\star}^G$ respectively.

Figure 3: The CTS map location estimation algorithm

Given an independent, consistent estimate of the location of a submap p (with root i) with respect to the global frame, G, we can produce a consistent global estimate of the location of any feature j in map p by composition of map and feature locations:

$$ {}^{[p,\cdot]}T_j^G = T_p^G \oplus {}^{[p,i]}T_j \qquad (2)$$

This estimate is consistent because T_p^G and ${}^{[p,i]}T_j$ are independent. ${}^{[p,i]}T_j$ is "internal" to the map and T_p^G is "external" to the map.

The existence of a "shared" feature, s, between two maps p and q allows the location estimate T_p^G, of map p, (with root feature s), to be replaced with T_{p+}^G where

$$T_{p+}^G = T_q^G \oplus {}^{[q,w]}T_s \qquad (3)$$

and w, the root of map q is any feature id in map q. Equation 3 should be interpreted as finding an alternative expression for the global location of a shared feature s using quantities associated with map q instead of p. As map p has a root at the shared feature s this expression is by definition an alternative expression for the map location. The minimization step of the algorithm is concerned with finding the best choice of shared feature s.

In the limit each map becomes internally fully correlated and the "min" operation will have no further effect so that for any feature j

$$T_p^G \oplus {}^{[p,\cdot]}T_j = T_q^G \oplus {}^{[q,\cdot]}T_j \qquad (4)$$

where $\cdot$ is any choice of root. In other words no root-shifting and replace operation can be found that improves the global

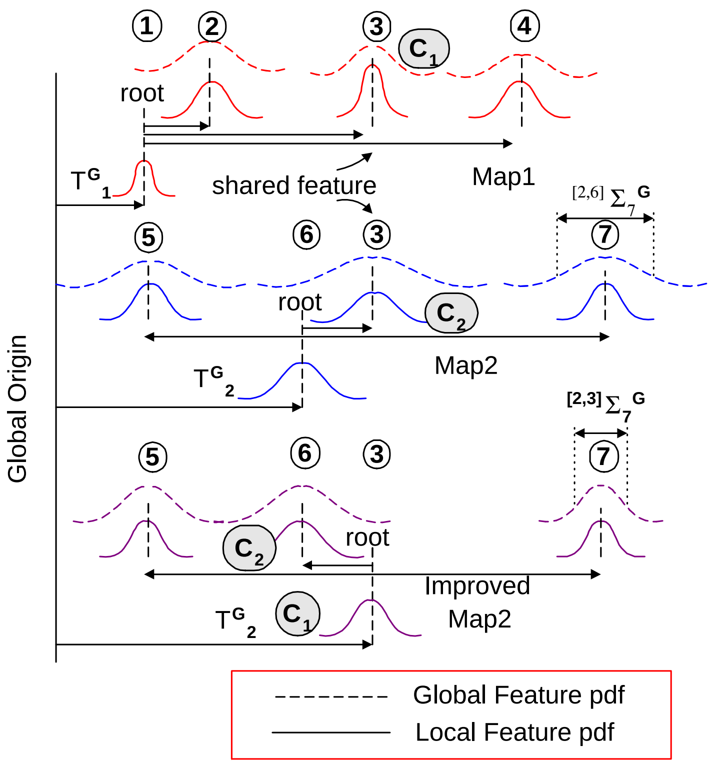


Figure 4: A 1D depiction of the CTS map location estimation procedure. Here maps 1 and 2 share feature 3. Initially map 1 is referenced to (has as root) feature 1 and map 2 to feature 6. The dotted pdf's represent the global uncertainties in the features derived from composition of local estimates with the map location estimates T_1^G,T_2^G. The CTS algorithm deduces that map 2 can be improved by the use of the global estimate of the shared feature derived from map 1. Map 2 is root shifted to feature 3 (the shared feature) and its location estimate replaced by ${}^{[1,\cdot]}T_3^G$ (labelled as C_1). The resulting improvement in map 2 is illustrated by the resulting reduction in ${}^{[2,\cdot]}\Sigma_7^G$ — the global uncertainty in feature 7.

uncertainty of feature j. For the linear case

$$\Sigma_p^G + {}^{[p,\cdot]}\Sigma_j = \Sigma_q^G + {}^{[q,\cdot]}\Sigma_j \tag{5}$$

The maps are rooted on features and so as $k \to \infty$ then

$${}^{[p,\cdot]}\Sigma_j = {}^{[q,\cdot]}\Sigma_j = 0 \tag{6}$$

and so

$$\Sigma_p^G = \Sigma_q^G \tag{7}$$

which is true for all choices of p and q. Therefore the globally referenced feature location uncertainty ${}^{[\cdot,\cdot]}\Sigma_j^G$ is the same independent of choice of map $(\cdot)$. The value of this limiting value is clearly given by the smallest possible uncertainty in Map Location which is the uncertainty of the first feature initialized in the first map [6].

To show that the global location estimates produced by Equation 2 are consistent, we rely on the following three properties: (1) local map state estimates ${}^{[p,i]}T_j$ (obtained from the local SLAM solution $[\hat{\mathbf{x}}_v^m(k)^T \hat{\mathbf{x}}_1^m(k)^T \ldots \hat{\mathbf{x}}_n^m(k)^T]^T$) are consistent, (2) global state estimates for Map Locations T_m^G are consistent, and (3) the composition of these two pieces of information — local state estimates within a map and global information concerning the location of the map — is consistent.

The consistency of local maps follows directly from the properties of the Kalman filter which in the linear gaussian case is the optimal Bayesian estimator. Clearly choosing to use a possibly inconsistent estimator such as the EKF in a nonlinear scenario will invalidate these claims. However the LG case allows statements to be made regarding the *underlying* properties of the CTS algorithm. In a non-linear implementation, the consistency of the LG case can be matched to an arbitrary degree by using Monte-Carlo estimators in each sub-map. Regardless of local estimation techniques, the CTS algorithm preserves its constant-time property because it only ever operates on a bounded set of features (the bound coming from our assumption regarding a bounded spatial density of observable/mappable features).

Local maps have three differences from the full solution (a) their base reference (root) is defined by one of the features in the map, (b) relocation is periodically performed to re-initialize the local map when the vehicle transitions back into it, and (c) the base reference of the local map is periodically shifted from one feature in the local map to another (Root Shifting). None of these three differences result in a loss of consistency for the local SLAM solution.

The global Map Location estimate T_m^G for a given map, m, is consistent because it is created via the composition of transformations derived from other local maps, and each local map is independent of other local maps. The composition of transformations from different local maps is a consistent operation (for the linear case).

Finally, the composition of the local map state estimates performed in Equation 2 is a consistent operation, because T_m^G and ${}^{[m,i]}T_j$ are independent of one another.

While the location estimates for different maps are correlated with one another, and this correlation is not computed by the algorithm, the method is none-the-less consistent because this correlation is never needed. We never fuse Map Location estimates estimates, but rather, perform wholesale replacement. The algorithm keeps track of the best estimate for the global location of the root entity of a given map. A guiding principle of this algorithm is that estimated quantities that are "external" to a map never effect an internal quantity.

When the vehicle transitions between maps the marginal distribution of the vehicle is thrown away and the vehicle is relocated from scratch into the next map. This loss of this information does not affect absolute convergence but only the rate of convergence. Because relocation is a consistent operation [15], each partial solution retains all the properties of a Kalman filter SLAM solution [6], and hence is provably consistent and convergent.

In CTS each local map is referenced to a feature. Therefore in the limit the uncertainty for each feature in a local map converges to zero [6] — the relationship between features within a map becomes perfectly known. This implies that, in the limit, in any local map utilizing only a subset of Z^k, the relationship between features becomes perfectly known as $k \to \infty$. In other words, for local maps in which the base reference (root) is a feature in the local map, the covariance of any feature tends to zero.

With this argument in hand we see by induction that in the limit, the lower bound in global uncertainty achieved in submap 1 is "inherited" by all other submaps. In addition, it is the global uncertainty of the first feature mapped within submap 1 that limits the lower bound of all other features within it . This then is entirely analogous to the full covariance solution's limiting precision being driven by the first observed feature.

There are two differences between what occurs in submap 1 in comparison to a full covariance solution that couples estimates for all features in a single map: (a) submap 1 has fewer features in it, and (b) not all of the observations of features that are contained in submap 1 are processed in the submap 1 solution. We believe that consideration (a), the fact that submap 1 has fewer features in it, is what is sacrificed in this approach. Even if some measurements are ignored in submap 1 (vs. the full solution), in the limit as $k \to \infty$, both maps will converge to a the same well-defined lower bound. With enough additional time the submap 1 solution can "catch up" to the full solution. However, the fact that the full solution has more features enables in it cannot be compensated for, and hence the full solution achieves a slightly tighter bound. This in effect is the "cost" of computing multiple partial solutions and subsequently combining them, rather than computing one full solution. Our simulations [18] have shown that this result is extremely small. The next section presents results using the CTS algorithm with real data in a truly challenging environment.

4 Experimental Results — Application to an AUV

This section presents CTS SLAM results using data from the 2002 "GOATS"[8] experiment near the coast of Italy. The experiment was one component of an ongoing program of research into the to the search, detection and identification of subsea mines (the mine counter measure (MCM) problem) by autonomous underwater vehicles. The vehicle used was an Odyssey III type vehicle shown in figure 5. The large "prong" structure protruding from the front of the AUV is the receiver array of MIT synthetic aperture sonar (SAS). The SAS is a high power transmit/receive acoustic device and was the primary payload of the MIT component of the GOATS 2002 experiment.

The SAS is an extraordinary payload. It offers remarkable detection capabilities but at the same time substantially changes the vehicle's hydro-dynamics and controllability. The "wet end" of the SAS system consists of two rows of eight transducers and an acoustic source (a sub bottom profiler) used to ensonify the sea bed. The two rows of hydrophones allow after, beam forming, good azimuthal angular resolution but only course resolution in elevation.

4.1 Obtaining a "Ground Truth"

A network of four small long baseline (LBL) acoustic transponders were deployed in the work area and their surface drop positions surveyed with a high precision differential GPS system. In the shallow operating area the actual seabed location of the transponders is likely to be a within a few (¡5) meters of the drop coordinates. The Odyssey III class vehicle "Caribou" shown in figure 5 was fitted with an acoustic transceiver, which transmits to the main vehicle computer the time elapsing between the transmission of an interrogation pulse and detection of a reply pulse from each of the four beacons. The measurements from this "absolute" sensor and the north seeking gyro and DVL fitted to the vehicle

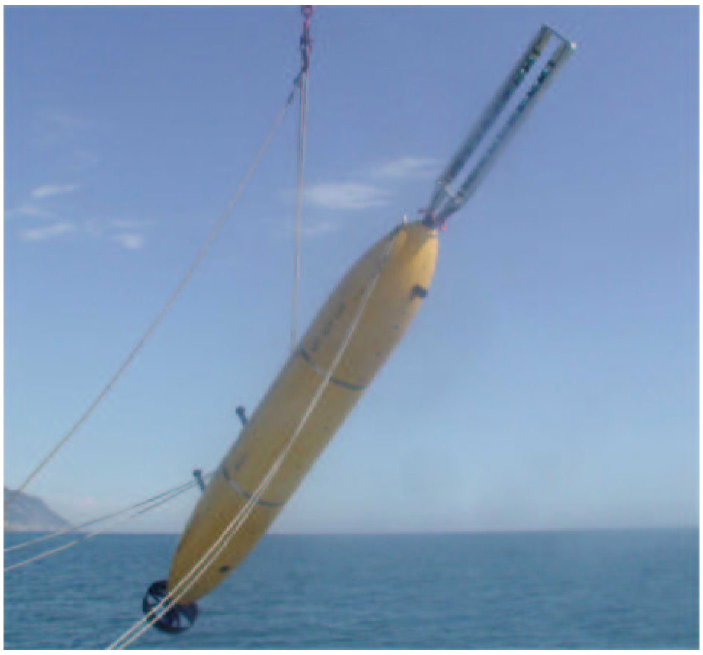


Figure 5: Caribou the MIT Department of Ocean Engineering's AUV being launched in the GOATS 2002 Experiment near Italy. The forked "prong" structure at the front of the vehicle is a 16 element Synthetic Aperture Sonar (SAS).

we used by a ten-state navigator running on the vehicle during the mission to execute the "Zamboni" pattern illustrated in figure 6. Offline the estimates are smoothed by application of Rauch Tung and Striebel forwards/backwards filter to derive estimates of vehicle location at time k conditioned on all measurements in the mission : $\hat{\mathbf{x}}(k|Z^N)\forall k \leq N$ where Z^N is the complete set of all measurements. This sequence of estimates is used as the ground truth for the following results.

4.2 SAS processing using Trajectory Sonar Perception

Trajectory Sonar Perception [26] is a method of tracking locally curved objects using sonar. Over small regimes, it is assumed that surfaces can be described by a radius of curvature ρ and a center of curvature $[x_c\ y_c\ z_c]^T$. A point has zero radius of curvature, a plane has a radius of curvature of $\pm\infty$. Assuming normal reflections from surfaces, the measurement prediction equations for range r, azimuth θ, and elevation ϕ are

$$r = \sqrt{(x_c - x_r)^2 + (y_c - y_r)^2 + (z_c - z_r)^2} - \rho \tag{8}$$

$$\theta = \arctan(\frac{y_c}{z_c}) \tag{9}$$

$$\phi = \arccos(\frac{x_c}{\sqrt{x_c^2 + y_c^2 + z_c^2}}). \tag{10}$$

Since the state of the object is not known, these equations cannot be used directly for tracking. Instead, using prior observations of targets, subsequent measurements are predicted via a Taylor series expansion of substantial derivatives.

$$\begin{bmatrix} r_1 \\ \theta_1 \\ \phi_1 \end{bmatrix} = \begin{bmatrix} r_0 + \triangle t \frac{Dr}{Dt} + h.o.t. \\ \theta_0 + h.o.t. \\ \phi_0 + h.o.t. \end{bmatrix}. \tag{11}$$

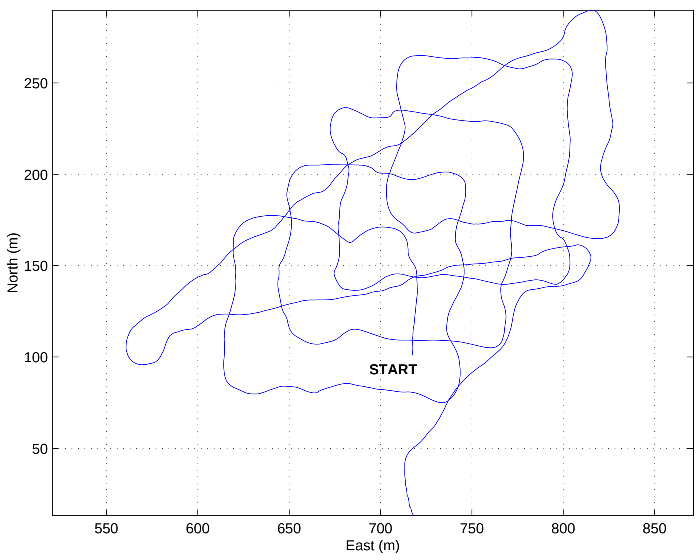


Figure 6: The ground truth trajectory of the vehicle derived by conditioning on all LBL and odometric mission measurements.

The substantial derivative is the rate of change of a moving observer's measurements of a field. The geometric quantities we wish to measure can be modelled as a field, and the robot is clearly a moving observer, so this is appropriate. In vector, notation, the substantial derivative is

$$\frac{Dr}{Dt} = \frac{\partial r}{\partial t} + \dot{\mathbf{x}} \cdot \nabla r. \tag{12}$$

The abbreviated Taylor series we use contains first order terms for range, and zeroth order terms for the angles. In general, the n^th order range terms are equivalent to the $(n-1)^{th}$ order angular terms. The extent of the range Taylor series was determined using two non-dimensional numbers. If the first, denoted $\#_1$, exceeds $\frac{1}{4}$, first order terms are necessary. Likewise, if the second, denoted $\#_2$, exceeds $\frac{1}{4}$, second order terms are needed. These numbers determine whether the first and second order terms contribute more than a quarter wavelength to the total change in range — we are assuming that a quarter wavelength is the smallest measurable range change.

$$\#_1 = \frac{V}{\lambda f_s} \tag{13}$$

$$\#_2 = \frac{V^2}{2 r_{min} \lambda f_s^2} \tag{14}$$

For the experiment, a representative robot velocity,V was $1\frac{m}{s}$, a representative wavelength,λ, was $.1m$, and the sampling frequency, f_s was $3Hz$. This made $\#_1 = \frac{10}{3}$, which exceeds $\frac{1}{4}$, necessitating the first order range term. In fact, the maximum velocity at which first order terms can be neglected is $\frac{\lambda f_s}{4} = .075\frac{m}{s}$. Heave and sway velocities between $.1\frac{m}{s}$ and $.25\frac{m}{s}$ were routinely observed, necessitating the modelling of all three velocities.

Although ranges that were less than the ten meter water depth were discarded, using an unrealistic minimum range value $r_{min} = 5m$, the second non-dimensional number is calculated as $\#_2 = .1111$. Because an unnecessarily small r_{min} was chosen, this number is artificially high, yet it is still less than $\frac{1}{4}$, demonstrating that only first order terms are needed. The range substantial derivative is

$$\frac{Dr}{Dt} = -u\cos(\theta)\sin(\phi) - v\sin(\theta)\sin(\phi) - w\cos(\phi) \tag{15}$$

where u, v, and w are the robot's surge, sway, and heave velocities respectively. The x axis assumed to be along track, the y axis is to port, and the z axis points up from the body.

A very simple tracking algorithm was used. Range, azimuth, and elevation were found by beamforming and thresholding the signals from a 16 element tuning fork array. Measurement trajectories were initialized from individual measurements. Using the initial measurement, a prediction was made of subsequent measurements. If, within 5 time steps, a second measurement gated with the prediction, it was added to the trajectory, and became the measurement that was used for predictions. If a trajectory was not updated for 5 time steps, it was terminated.

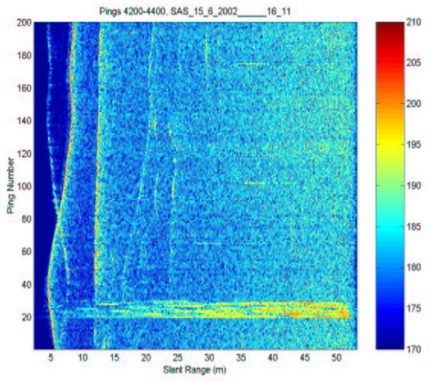
Running in a lag behind the trajectory augmenter was a second filter that checked for outliers and occasions when the early tracker mistakenly switched targets. The last seven measurements were analyzed. The first three and the last three were used to estimate the measurement trajectory (ie a reference range and the range substantial derivative). If, using a Mahalanobis test, the two pieces were determined to be different, the trajectory was split. If the two pieces could not be said to be different, but the fourth measurement could not be gated with either, it was discarded as an outlier. This was the entire algorithm.

4.3 Application of CTS

The experimental results presented here were generated post-mission using manual data association. The SAS measurement trajectories (the range components of which are shown in Figure 7) were projected into cartesian space from the smoothed vehicle trajectory shown in Figure 6. The angular variance of the SAS measurement set made automatic data association and measurement clustering (such as k-means) unreliable and so the clustering was performed manually. This labelled data set was then piped into the CTS algorithm for processing. The relevance of this human intervention to the results presented here should be clearly stated and understood. Clearly for true autonomy (ie deployment) manual data association is out of the question. However, the aim of this section of the paper is to 1) describe an ongoing programme of work to achieve constant-time subsea SLAM, 2) compare CTS performance/consistency viz a viz the full covariance solution and 3) to highlight outstanding issues — of which automatic data association is one. Importantly there is nothing about the CTS algorithm that makes data association harder than when employing the full covariance approach.

The DVL (doppler velocity log) data was used as an dead reckoning input to the filter. The onboard compass, inclinometers and depth sensors were assumed to produce unbiased measurements of orientation and depth and allowed the estimated vehicle state to be reduced to x and y coordinates alone following roll pitch and yaw compensation. The errors on SAS measurements (in cartesian coordinates relative to the vehicle) were modelled as a gaussian process with 5 m standard deviation in x and y coordinates.

Figure 8 shows the estimated trajectory of the vehicle using full covariance and CTS methods. Figure 9 is a zoomed



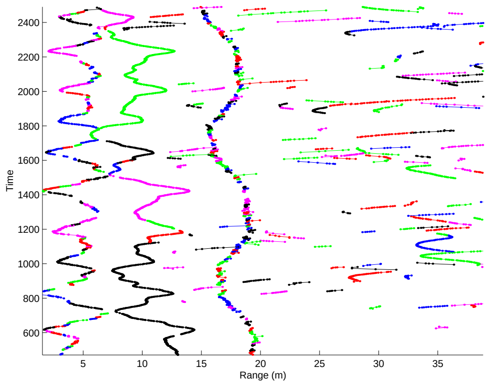


Figure 7: SAS range measurement trajectories. The SAS processing algorithm employed here detects "measurement trajectories" from typical raw amplitude data (top) from the 16 hydrophones. The fundamental idea is that continuous, smooth (but unknown) motion of the vehicle will cause returns from reflective artifacts to form characteristic arcs in the raw data. Beam forming across the hydrophone array allows the estimation of elevation and azimuthal angles to these artifacts contacts. The first two strong returns on the left of the lower figure are direct reflections from sea bed and surface. The second return is a multiple refection between these boundaries and is so a measurement of the water column height — the undulations corresponding to the rise and fall of the ocean floor over the mission. The broad stripe across the raw data plot (top) is interference from an acoustic modem.

view of the globally referenced estimates of two mapped features. If required, a single estimate of feature location conditioned on all measurements can be created by employing the map joining technique presented in [28].

Figure 10 is perhaps the most important figure. It uses the ground truth derived from the long base line measurements to compare the performance, in terms of estimated vehicle location, of the CTS and full covariance solutions. The CTS algorithm is always consistent — its covariance never drops below that of the optimal full covariance result. The solid red lines indicate the theoretically lowest possible vehicle covariance bound for either algorithm. The CTS algorithm was developed after the GOATS 2002 mission and so the vehicle trajectory is not ideal to illustrate the convergence properties of the algorithm [18] — a repetition of the pattern would be preferable. Nevertheless the bounding of vehicle uncertainty is clearly visible and unlike the full covariance solution the constant-time nature of CTS means it can be achieved independently the extent of the mapped area.

Figure 11 illustrates the evolution of globally referenced

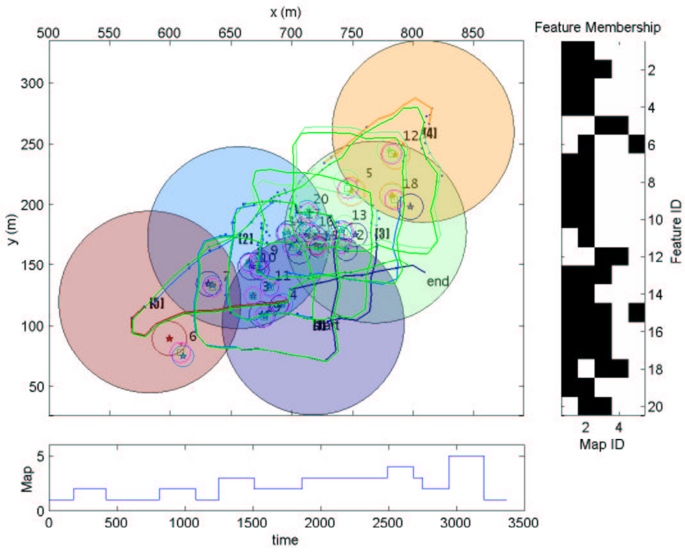


Figure 8: The result of applying the CTS algorithm to the SAS derived data. The mission begins at (700,100) and the AUV executes the "Zamboni" mission of Figure 6. The circles define the five maps as a function of vehicle location. The green trajectory is that derived using the full covariance solution whereas the multi-colored track is the CTS estimated vehicle trajectory. The magenta circles are full covariance error bounds.

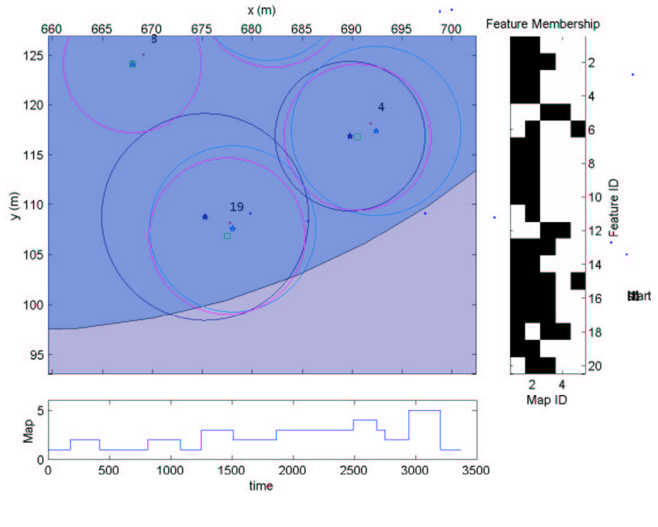


Figure 9: A x-y plane close up of features 19 and 4 as estimated by CTS maps 1 and 2 and the full covariance solution. The green squares are the full covariance estimates with magenta 1σ bounds. The dark and light blue stars are the estimates from CTS maps 1 and 2 respectively with associated error bounds. The red crosses are the human selected true locations of the reflecting artifacts. Note that the full covariance solutions has slightly "tighter" error bounds.

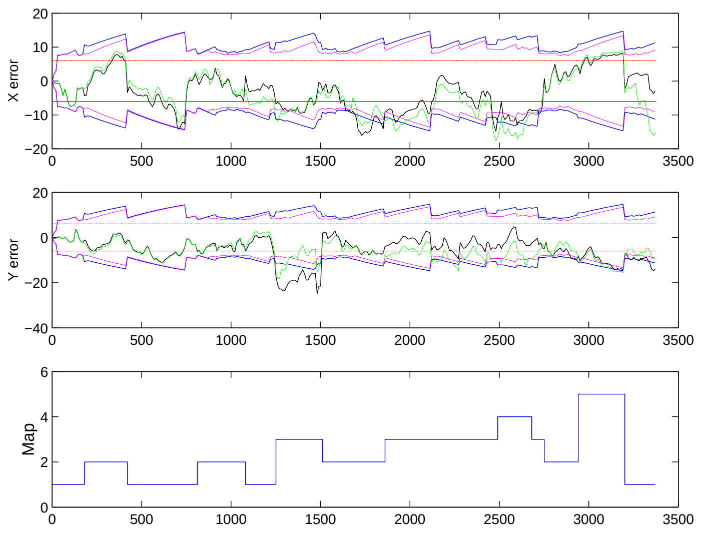


Figure 10: A ground truth comparison vehicle x-coordinate estimates between CTS and the full covariance solution. The error between ground truth and the full solution is plotted in green and for the CTS algorithm and in black. The 1 sigma bound covariances are magenta and blue for the full covariance and CTS solutions respectively. Note how when initially both solutions are in map 1 (the full solution is always in map 1) the two estimates are identical. However when returning to map 1 the CTS solution is as expected slightly less certain (If repeated loops were made this difference would decrease towards zero). Crucially the c CTS solution is conservative, it never produces and estimate with lower uncertainty than the gold standard but will converge to it.

feature estimates and how they compare to those of the full covariance solution as discussed in section 3.3. Although the feature uncertainties do decrease in CTS they do not decrease at the same rate as those in the full solution. Like the full solution, the CTS algorithm is consistent, convergent but what it gains in being constant time it gives up in terms of rate of convergence. If there is no "free lunch" then perhaps this can be considered the "price of lunch".

5 Conclusions and Future Work

We have introduced a new constant-time, consistent and convergent SLAM algorithm and successfully applied it to data collected from an AUV carrying a synthetic aperture sonar (SAS). The SAS data was processed using a novel continuum based technique which used platform motion to detect reflecting artifacts on the sea bed. The performance of the CTS algorithm was commensurate with that of the "gold standard" full covariance solution yet operated in constant time. Several key issues remain to be addressed, however. On a theoretical front, we seek an analytical, closed-form expression for the differing rates of convergence between the CTS and full covariance solutions in the presence of non zero process noise. Secondly, it is probable that a graph-theoretical perspective would be productive in determining the best way in which to root shift the multiple maps of the CTS algorithm. Although convergence occurs regardless, there will be one particular choice of submap origins that produces, with regard to a suitable metric, the best overall global estimate of features. We hypothesize that this is achievable in linear time and need not resort to the quadratic-time fusion of submaps. On a practical front, the upcoming GOATS 2004, experiment will provide another opportunity to gather more SAS and AUV data with an aim to moving closer to a realtime deployment of SLAM on a AUV. In the meantime further refinement will be made to SAS target detection and automatic data association and the CTS algorithm applied to large land based data sets such using non-linear local map estimation.

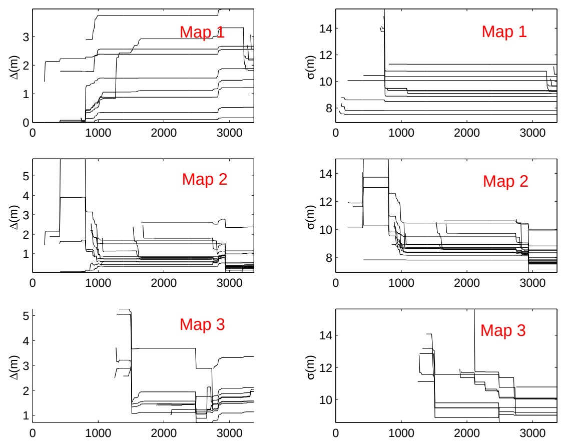


Figure 11: Comparison of feature location estimates between full and CTS solutions. The left hand plots show the difference in the globally referenced x-direction 1 σ contour on feature estimates between the full covariance and CTS solutions. Note how in map 1 the differences increase when in CTS processing the vehicle moves into new maps. This is because *every* observation in the "gold standard algorithm" improves estimates of *every* feature — the cause its quadratic scaling in computation — whereas this is not the case for CTS. The right hand plots show the globally referenced feature x-direction 1 σ contours of the CTS algorithm alone — which, as is expected of any SLAM algorithm, decrease.

A Transformation Notation and Operations

The notation is this paper is summarized as:

$$^{[\text{mapid,rootid}]}\text{Variable}^{\text{w.r.t. id}}_{\text{variable id}}$$

Using this notation we can write the transformation from frame $\mathcal{F}_j$ to $\mathcal{F}_k$ in map m with root entity i as $^{[m,i]}T^j_k$. We simplify notation by dropping the right superscript when describing a transformation with respect to the map root:

$$^{[m,i]}T^i_j \rightarrow {}^{[m,i]}T_j$$

The term $^{[m,i]}T_j$ is the pose of an entity j in the local frame of the m^{th} map (which has its Map Root as entity i).

We can manipulate location vectors using the binary transformation operator $\oplus$ and the unary operator $\ominus$ where

$$T^i_j = \ominus T^j_i$$
$$T^i_k = T^i_j \oplus T^j_k$$

Taking two entities j and k from map m, we can express the transformation from j to k as

$$\begin{aligned} {}^{[m,i]}T_k^j &= {}^{[m,i]}T_i^j \oplus {}^{[m,i]}T_k^i \\ &= \ominus{}^{[m,i]}T_j^i \oplus {}^{[m,i]}T_k^i \\ &= \ominus{}^{[m,i]}T_j \oplus {}^{[m,i]}T_k \qquad (16) \end{aligned}$$

where the last step uses the simplification in notation described above.

The root shifting operation is is simply an extension of Equation 16 to act on all entities in the map:

$$\begin{aligned} {}^{[m,j]}T_{1:n} &= \mathcal{S}_{i\to j}({}^{[m,i]}T_{1:n}) \qquad (17) \\ &= \begin{bmatrix} \ominus{}^{[m,i]}T_j \oplus {}^{[m,i]}T_1 \\ \vdots \\ \ominus{}^{[m,i]}T_j \oplus {}^{[m,i]}T_n \end{bmatrix} \qquad (18) \end{aligned}$$

References

[1] P. A. Beardsley, A. Zisserman, and D. W. Murray. Navigation using affine structure from motion. volume 2, pages 85–96, 1994.

[2] H. Choset and K. Nagatani. Topological simultaneous localization and mapping (slam): toward exact localization without explicit localization. *IEEE Transactions on Robotic and Automation*, 17(2):125–137, April 2001.

[3] A. J. Davison. *Mobile Robot Navigation Using Active Vision.* PhD thesis, University of Oxford, 1998.

[4] Matthew Deans and Martial Hebert. Experimental comparison of techniques for localization and mapping using a bearings only sensor. In *Proc. of the ISER '00 Seventh International Symposium on Experimental Robotics*, December 2000.

[5] Matthew Deans and Martial Hebert. Invariant filtering for simultaneous localization and mapping. In *IEEE International Conference on Robotics and Automation*, pages 1042–7, April 2000.

[6] M. W. M. G. Dissanayake, P. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotic and Automation*, 17(3):229–241, June 2001.

[7] H. F. Durrant-Whyte, M. W. M. G. Dissanayake, and P. Gibbons. Towards simultaneous localization and map building (SLAM) in large unstructured environments. In D Koditschek and J. Hollerbach, editors, *Robotics Research: The Ninth International Symposium*, pages pages 162–168, Snowbird, Utah, 2000. Springer Verlag.

[8] J. Edwards, H. Schmidt, and K. LePage. Bistatic synthetic aperture target detection and imaging with an auv. *IEEE J. Ocean Engineering*, 26(4):690–699, 2001.

[9] O. Faugeras, F. Lustman, and G. Toscani. Motion and structure from motion from point and line matches. In *International Conference on Computer Vision*, pages 25–34, June 1987.

[10] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotic and Automation*, 17(3):242–257, June 2001.

[11] J-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *International Symposium on Computational Intelligence in Robotics and Automation*, 1999.

[12] D. Hahnel, D. Schulz, and W. Burgard. Map building with mobile robots in populated environments. In *Proc. IEEE Int. Workshop on Intelligent Robots and Systems*, 2002.

[13] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521623049, 2001.

[14] S. J. Julier and J. K. Uhlmann. Building a Million Beacon Map. In *Sensor Fusion.* SPIE, 2001.

[15] Joss Knight. *Towards Fully Autonomous Mobile Robot Navigation.* PhD thesis, Robotics Research Group, Oxford University Department of Engineering Science, February 2003.

[16] B. Kuipers and P. Beeson. Bootstrap learning for place recognition. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002. AAAI.

[17] M. B. Larsen. High performance doppler-inertial navigation — experimental results. In *IEEE Oceans*, 2000.

[18] J. Leonard and P. Newman. Consistent convergent and constant time SLAM using multiple maps. In *Int. Joint Conf. Artificial Intelligence*, number To Appear, Acapulco, Mexico, August 2003.

[19] J. J. Leonard and H. J. S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In D Koditschek and J. Hollerbach, editors, *Robotics Research: The Ninth International Symposium*, pages 169–176, Snowbird, Utah, 2000. Springer Verlag.

[20] L. Lucido, B. Popescu, J. Opderbecke, V. Rigaud, R. Deriche, Z. Zhang, P. Costa, and P. Larzabal. Segmentation of bathymetric profiles and terrain matching for underwater vehicle navigation. *International Journal of Systems Science*, 29(10):1157–1176, October 1998.

[21] P. F. McLauchlan and D. W. Murray. A unifying framework for structure and motion recovery from image sequences. In *Proc. Int. Conference on Computer Vision*, pages 314–320, Boston, Massachusetts, 1995.

[22] Philip F. McLauchlan. A batch/recursive algorithm for 3d scene reconstruction. In *Int. Conf. Computer Vision and Pattern Recognition*, volume 2, pages 738–743, Hilton Head, SC, USA, 2000.

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

[24] P. Newman and H. F. Durrant-Whyte. An efficient solution to the slam problem using geometric projections. In *Sensor Fusion and Decentralized Control in Robotic Systems IV*, Boston,USA, November 2001.

[25] P. Newman and J. Leonard. Pure range-only subsea slam. In *Proc. IEEE Int. Conf. Robotics and Automation*, number To Appear, Tawain, May 2003.

[26] Richard J. Rikoski, John J. Leonard. Trajectory sonar perception. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2003. Accepted to the 2003 IEEE International Conference on Robotics and Automation.

[27] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. Cox and G. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, 1990.

[28] J.D. Tardós, J. Neira, P.M. Newman, and J.J. Leonard. Robust mapping and localization in indoor environments using sonar data. *Int. J. Robotics Research*, 21(4):311–330, April 2002.

[29] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *Int. J. Robotics Research*, 20(5):335–363, May 2001.

[30] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and Ng. A.Y. Simultaneous mapping and localization with sparse extended information filters. In J.-D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, editors, *Proceedings of the Fifth International Workshop on Algorithmic Foundations of Robotics*, Nice, France, 2002. Forthcoming.

[31] L. L. Whitcomb, D. R. Yoerger, and H. Singh. Combined Doppler/LBL based navigation of underwater vehicles. In *Proc. Int. Symp. on Unmanned Untethered Submersible Technology*, 1999.

[32] S. B. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 406–411, May 2002.