Feature Tracking For Underwater Navigation using Sonar

John Folkesson, John Leonard, Jacques Leederkerken, and Rob Williams

Abstract—Tracking sonar features in real time on an underwater robot is a challenging task. One reason is the low observability of the sonar in some directions. For example, using a blazed array sonar one observes range and the angle to the array axis with fair precision. The angle around the axis is poorly constrained. This situation is problematic for tracking features in world frame Cartesian coordinates as the error surfaces will not be ellipsoids. Thus Gaussian tracking of the features will not work properly. The situation is similar to the problem of tracking features in camera images. There the unconstrained direction is depth and its errors are highly non-Gaussian. We propose a solution to the sonar problem that is analogous to the successful inverse depth feature parameterization for vision tracking, introduced by [1]. We parameterize the features by the robot pose where it was first seen and the range/bearing from that pose. Thus the 3D features have 9 parameters that specify their world coordinates. We use a nonlinear transformation on the poorly observed bearing angle to give a more accurate Gaussian approximation to the uncertainty. These features are tracked in a SLAM framework until there is enough information to initialize world frame Cartesian coordinates for them. The more compact representation can then be used for a global SLAM or localization purposes. We present results for a system running real time underwater SLAM/localization. These results show that the parameterization leads to greater consistency in the feature location estimates.

I. INTRODUCTION

Perception and mapping with underwater robots is a particularly challenging problem domain. While a few applications of vision have been used, as in [2], sonar sensing is generally more applicable to marine environments, and examples of sonar mapping can be seen in [3], [4], [5]. Typical AUV systems aid feature tracking and initialization with motion estimates provided by a Doppler Velocity Log (DVL). We demonstrate a sonar feature tracking solution in the absence of DVL, which is a relatively expensive sensor.

There has been much work done on the problem of Simultaneous Localization and Mapping, *SLAM*. Many different algorithms are now available to choose from, ([6], [7], [8] are a cross-section). The problems with scaling to large environments is dealt with in [9]. The success of any SLAM implementation is critically dependent on components other than the SLAM algorithm itself, such as, careful motion modeling for dead-reckoning estimates, robust methods for global matching of map sections and the local tracking of features. Feature tracking or the matching of measurements taken from different robot poses is of central importance. This paper focuses on this aspect of the SLAM problem.

Dept. of Mech. Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, johnfolk@mit.edu, jleonard@mit.edu, jckerken@mit.edu

This work was supported by Office of Naval Research.

Feature tracking can be considered as the fine scale SLAM problem. Here we care less about computational complexity and more about accuracy and correctness of our inferences. The large scale SLAM problem will never be more accurate than the fine scale one. It is on the fine scale that the difficult decisions of separating spurious measurements from true ones and matching these to previous measurements are made. Mistakes here can severely distort the map and pose estimates.

There are two complementary approaches to track features in sensor data. One is to track them purely in the measurement space. This space may be for example a range bearing space for sonar, a camera image or a planar cross-section from a SICK scan. This approach uses some characteristic motion or appearance in the measurements that indicates a match to track the features. Examples include [10] for underwater sonar. Or using SIFT feature descriptors [11]. Also in this category are scan matching methods [12]. Presumably independence from motion estimates leads to some improvement in robustness. It also can allow one to do SLAM with no motion estimate at all as in [13].

The other approach is to use some motion estimate combined with a world model of the environment that gave rise to the measurement. The motion estimate can be simply a uniform distribution of possible motion [14]. Or a more conventional odometry model. These methods can run into trouble if the measurements do not provide full observations of the world model [15], [16]. It is then that the model needs to be carefully crafted to avoid mistakes such as overconfident inferences.

II. OVERALL APPROACH TO FEATURE TRACKING USING SONAR

In order to carry out missions autonomously underwater, the robot needs to remain localized in a GPS deprived environment. Sonar is generally very useful for sensing of the local environment underwater. Sonar gives precise range information to objects. By arranging the sonar transducers in an array and measuring the phase of the returned signals at each element, one can obtain bearing information as well. The precision of the bearing depends on the array configuration. For linear arrays the bearing angle from the linear axis will be much more precise than the angle around the axis.

The BlueView blazed array sonars contain pairs of linear arrays arranged in V shapes. Each pair detects in a range of angles from the centerline of the V, see figure 1. The output from the pair is provided as an image and each linear array can be extracted from the image data. The resolution is

poor in the direction perpendicular to the plane of the V. We model the sensor as two separate sensors each with its own offset transformation from the robot frame. In this work we have only looked at data from the horizontal mounted pair. With the sonar mounted looking forward, the data from this pair gives information on objects in the path of the robot. Figure 2 shows the robot with the forward mounted sonar. The effective range for this application of the BlueView 900

Fig. 1. This shows how one pair of linear sonar arrays are arranged to give a wedge shaped field of view, *FOV*, forward. There are 2 sensor coordinate frames defined in this figure, right and left. They differ in the *z* axis direction. The angle θ around the *z*-axis is poorly resolved.

kHz sonar is 4-40 meters while the 450 kHz model is 8-60 meters. In close, the level of noise increases making separating the detections difficult. One can detect some fixed number of maxima in each ping return. The result is a large number of detections most of which are not useful for localization. In order to filter out all the signals that are not from stationary solid objects, we model the objects that gave rise to them and see how well the model predicts the subsequent detections. True features will be seen again at the predicted range/bearings. False ones will not. This filtering

Fig. 2. The Nekton Ranger robot equipped with a nose mounted BlueView Proviever 450 kHz blazed array sonar appears on the left. The REMUS AUV is shown to the right.

is dependent on being able to make inferences based on the model. It is therefore essential to accurately capture the uncertainty of the accumulated information on the objects. The nature of this information is accurate range and some bearing from a smoothly moving robot. An estimate of the robot's 6 degree of freedom motion is also available from dead reckoning.

We began to examine the problem using an Extended Kalman Filter (EKF) and Cartesian coordinates for the features. A separate EKF was started for each new tracked object. When the object was predicted correctly a number of times it was initialized in another EKF that built a global map. This approach has several drawbacks. First the measurements during the initial tracking phase are not used to correct the robot pose of the main EKF. This creates a pressure to make quick decisions on feature initialization. Second the EKF is a Gaussian filter. The linearized error ellipsoids of the features in this representation can not represent the true probability distribution of the measurements.

In order to improve the tracking of the features these two drawbacks were addressed. The first one by replacing the EKF with an estimator that works with the full state of all robot poses. This estimator can then incorporate information even after a delay. Thus all the measurements are added to the estimate at the time of initialization. The rush to initialize is eliminated as is the loss of information.

The second drawback is addressed by the novel feature representation that can exactly model the uncertainty of the first measurement and well approximate it for measurements taken from roughly the same range and bearing. This representation is analogous to the feature representation proposed in [1] for a similar vision SLAM problem. The idea proposed there was to represent the vision features as the pose they were first seen from and parameters that gave the relative position of the feature to the pose. The inverse depth was the choice that gave the most Gaussian like distribution for the features observed with a camera.

The sonar used here has uncertainties that are relatively independent in sensor frame polar coordinates, see figure 1, where the *z* axis is along the sonar's array axis. Polar coordinates are ϕ , the angle to the *z*-axis, θ , the angle of the projection to the *xy* plane with the *x*-axis and *r*, the range. The errors are symmetric around their means in this space. The θ error distribution is still not close to Gaussian.

The statements above apply if the origin of the polar coordinates is the sensor frame. Using the world frame origin would defeat the purpose. Thus we need to include in the representation of the feature the 6 coordinates that specify a sensor frame. We use the frame it was first seen from. These 6 parameters are already being estimated by the tracker as part of the motion estimate. Therefore the extra 6 parameters are not an extra cost for the local estimator. On the other hand, to represent the features in a global map containing perhaps hundreds of features having 9 parameters each would be a significant cost over a 3 parameter representation. After the feature has been seen from enough vantage points its uncertainty distribution will converge to something more Gaussian in the world frame. This convergence can be checked by thresholding the information in the three Cartesian directions. This threshold has negligible effect on the results as good features eventually are seen enough times and no information is ever lost. So the idea is to maintain one local estimator with all the robot poses and then summarize the local estimate at a low frequency into what we call a

composite measurement. This summarize compact data has the features transformed to a Cartesian coordinate system in a common world frame. The composite measurement is then passed to the global navigation module or other user of the information.

The state of our tracker is then a chain of recent poses at the times of feature measurements and the estimated ranges and bearings from some of the poses to features that were recently observed. The measurements are initially used to create infant tracked objects. These infants are updated to their maximum likelihood positions given the tracker state. They do not affect that state. When and if enough information is collected on an infant¹ it is graduated to a feature, its state added to the tracker state, and all its measurements added to the state estimator. No information is lost. We summarize our state evolution over one iteration:

- 1) Use the dead-reckoning estimate to add a pose state.
- 2) Match the measurements to the tracked objects.
- 3) If unmatched create a new infant tracked object.
- 4) Update the state of all infants at the current state.
- 5) Conditionally graduate infants to features.
- 6) Update the state.

For this paper we will focus on the tracking of the features to form composite measurements. The interesting problems of data fusion for dead-reckoning, use of GPS, matching features globally and making the global estimate are all considered beyond the scope of this paper. Let it suffice to say that we had solutions to these available to test our feature tracking ideas on a robot operating in real time underwater.

III. THE TRACKING FILTER

The estimator for tracking the features is an incremental implementation of a square root filter [17]. This filter has a number of advantages over other Gaussian estimators. First and foremost is that by working with the square root of the information matrix we also square root the condition number of the matrix. This has the effect of eliminating the numerical difficulties that invariably cause problems for SLAM using an EKF or information filter. This is the strongest argument for using the square root formulation.

Since we maintain only a short chain of recent poses and features in the filter, the matrices involved in the calculation tend to remain sparse. We take advantage of this sparseness to re-linearize the system when we graduate infants to features and when we form composite measurements from the older part of the state. Thus we maintain all the measurements in a graphical data structure and can calculate the likelihood of any state exactly.

The state is initially just the filter's base robot pose. Upon receiving a new sonar ping a dead-reckoning estimate of the robot pose is added to the state. This dead-reckoning estimate is in the form of a measurement between the current pose and the previous pose as well as some measurements of the absolute 'Earth' frame such as compass, depth, pitch and roll. We add the i^{th} dead-reckoning measurement as a matrix. The

measurement matrix, A_i is the square root of the information matrix H_i , which is the inverse of the covariance matrix for the measurement C_i .

$$A_i^T A_i = H_i = C_i^{-1}. \quad (1)$$

The goal of the filter is minimization of the minus log likelihood or energy [16]. The energy gets an additional term from the i^{th} measurement that looks like:

$$\Delta E_i = (A_i \Delta \mathbf{x} - \mathbf{b}_i)^T (A_i \Delta \mathbf{x} - \mathbf{b}_i) \quad (2)$$

The $\Delta \mathbf{x}$ is the change in state from the current estimated state (where A_i and $\mathbf{b}_i$ are evaluated). This A_i matrix is computed from the Jacobians of the incremental transformation between the two poses and the covariance of the incremental coordinates. There are additional rows then added to A_i for each earth frame measurement. In our case A_i has 6 rows and 12 columns for the part that links the two poses and 4 additional rows for the earth measurements of rotation and depth. These last four rows have only 4 non-zero columns. The $\mathbf{b}_i$ is the measurement mean scaled appropriately see [17].

The solution is found by minimizing the total energy E :

$$E = \sum_i |(A_i \Delta \mathbf{x} - \mathbf{b}_i)|^2. \quad (3)$$

This equation is best solved iteratively after adding each new A_i term. The full equation can be viewed as solving one matrix equation where the total A is formed by stacking all the A_i from each measurement with zeros added for the extra columns. This total A matrix is then rectangular with rows for every measurement and columns for every state. It is sparse. We never form this matrix but instead form a matrix R that is the upper triangle matrix obtained by doing the decomposition, $QR = A$ with Q an orthonormal matrix. As R is zero for all rows below its number of columns we discard those rows after each iteration. The right hand side is also saved $\tilde{\mathbf{b}} = Q^T \mathbf{b}$. Again only the rows corresponding to R are retained. The other rows of $Q^T \mathbf{b}$ give the energy of the minimum energy state but do not give the state.

The new measurement A_i is appended to R by adding rows to the bottom. The same is done to the right hand side. Then the QR decomposition is performed. After that the added rows of R are all zero and can be deleted. The new state $\mathbf{x}$ can be found by back substitution. After solving for $\Delta \mathbf{x}$ we move the state and then can zero out all of $\tilde{\mathbf{b}}$. The linear approximation is that we do not re-evaluate R at the new state. R would be the same if the system were truly linear² and $\tilde{\mathbf{b}}$ would come out as zero. We are now ready for the next measurement.

One can see that the QR decomposition is the critical matrix operation. The second major advantage of the square root filter is that Q is discarded. It enters the energy equation paired with its transpose and is thus not needed. Maintaining Q orthonormal as the dimensions become large is the main

¹There is a threshold on the trace of the information matrix of the infant.

²By linear we mean precisely that the A_i do not depend on the state.

weakness of QR decomposition. Thus not needing Q eliminates a second numeric problem as the size of our system gets large.

The QR decomposition itself is constant time if the states involved in the new measurement happen to be the last columns of R . If that is not the case we can swap rows and columns of R to make it true. This we do one row and column step at a time. We then do QR to restore the upper triangular form of R . Rearranging like this takes time L where L is the distance in rows that we have to move. If the R matrix has become dense then the complexity becomes L^2 . Thus we hope to generate little fill in R and have to move rows only short distances.

In practice, for this application the active states all are kept near the bottom and the states near the top are never needed. As the robot moves new pose and feature states are added at the bottom of R .

The computational complexity is tied to the sparseness of the matrix. This is in turn dependent on the number of loops in the path as revisiting causes fill in the matrix during execution of the algorithm. In this tracking filter we only work with a short stub of the full robot trajectory. So we have no loops. This means that the complexity for computing the current pose and current local features is constant time as is the cost of computing the covariance for those few states. The complexity to compute the covariance of the entire state would be N^2 if sparseness is maintained and N^3 worst case, where N is the dimension of our state. We never really need to compute, for instance, the covariance of one of the inner robot pose states.

Measurements from pose states to feature states are handled in exactly the same way as pose to pose measurements.

To compute the covariance of a set of n states we again move the n states to the bottom and solve:

$$R\tilde{s} = \tilde{I} \quad (4)$$

Here $\tilde{I}$ is a $N \times n$ matrix with the $n \times n$ identity matrix on the bottom. And $\tilde{s}$ is the solution that contains the square root of the covariance we seek. As

$$C = (R^T R)^{-1} = R^{-1}(R^T)^{-1} \quad (5)$$

and $\tilde{s}$ is the n right most columns of an upper triangular R^{-1} , we need only back substitute to the bottom n rows of $\tilde{s}$, call them a $n \times n$ matrix s . The sub-matrix of C that we want is just ss^T . This is one way that Mahanolobis testing can be done. We have a faster equivalent way to check matches.

We can simply move the states of the feature measurement to the bottom. Then copy these rows and columns of R to form a smaller R matrix with only the relevant parts. We append the A_i and b_i to this and do the QR on it. The sum of squares for the elements of the right hand side vector, $\tilde{b} = Q^T b$, for appended rows is the Mahanolobis distance for the match. They represent the increase in the energy that would result from making the match.

A new sonar feature detection is matched to existing tracked features using a threshold on this energy increase. For infant objects the matching criteria will also be the increase

in energy but only the infant's coordinates are allowed to move when computing the minimum energy state of its measurements.

IV. FEATURE REPRESENTATION

The blazed array sonar we are using to detect strongly reflecting point features has a relatively good resolution in range, about 3 cm and in the angle from the axis of the transducer array, about 1 degree. The angle around the array axis, θ , is very poorly resolved with an resolution of about 20 degrees. The intensity of a given target's signal is roughly Gaussian. The feature detector will generate feature measurements for the brightest spots above some noise threshold. This results in a probability of detection for a target that is essentially flat, 20 degrees wide, and centered on 0. If this is then projected into Cartesian xyz coordinates the result is an arc of equally likely target locations.

If one uses xyz as the feature representation in the tracking filter, this arc will be approximated by an ellipsoid. Obviously this is a poor model. A significant improvement was obtained by parameterizing the feature with a range and two bearing angles from the sensor pose it was first seen from. These three parameters along with the robot pose from the filter state and fixed sensor offset give the feature location in 3 space. This proved to be rather successful. There was still some model inaccuracy due to the flattened distribution being fit by a bell shaped Gaussian. This is addressed now.

Fig. 3. Comparison of the shape of the two distributions used to model the probability of a detection given that an object was at a given theta.

In order obtain better estimates of the vertical positions of the features, we transformed θ to a parameter that would be better modeled by a Gaussian,

$$u = k(\alpha\theta + \theta^3), \alpha \ll 1 \quad (6)$$

Where k is a coefficient chosen to give the right width to the distribution. The α (used $\alpha = 10^{-10}$), is just to give some slope near the origin. The rational being that if the u distribution was taken as Gaussian then the θ distribution would become much flatter in the center and steeper in the tails, see Figure 3. This is in the same spirit as using inverse depth as a parameter for vision point features. This gave us an improvement in the consistency of our estimated depths for the tracked features compared to the estimated standard deviation. One issue was the small value of the slope in the u/θ plane at the origin. This caused various Jacobians to become very large. Thus a modest covariance in u could be projected to an overly large covariance in z . There

is no problem in the tracking filter itself but rather when forming the covariance for the composite measurements. For those we used a Jacobian that corresponded to the slope of the transformation over the finite distances in u of ± 1 standard deviation. This solved that problem.

The other issue is that experiments were carried out with the robot moving at constant depth and zero pitch. Thus the information on the feature z coordinate is very weak and bi-modal. The probability of the feature being above or below the robot is about the same due to the symmetry. Nevertheless, we can conclude that there is an increased movement of the features in the z direction and a more realistic covariance estimate. The linear theta model has a strong maxima at theta equal to zero. This causes the z estimate to be over confident. The more realistic covariance should be elongated in the z direction. Most times the features moved down to the true positions. There were times that they moved up to the other mode usually due to too few detections of the feature. Figure 4 shows a result from one experiment. We see how the estimates are more consistent using the nonlinear parameterization of the features.

Figure 5 shows how an individual feature estimate moved over time. This helps one get a feel for how a series of measurements all with mean at the same depth and large depth uncertainty can converge to solutions below that depth. The weakly observable dimension is thus made convergent by careful modeling a series of measurements.

Fig. 4. In (a) we show a vertical cross-section of the map produced by tracking with a linear θ feature parameter. The robot path is shown as the solid red line. The point features are shown by the $1\sigma^2$ covariance ellipsoids of their estimated positions. All features are estimated to be at the depth of the robot, as the model gave a strongly peaked maximum at that angle. The true depth of the features is 5 m, shown as the dotted grid line. The results using Eq. 6 are shown in (b). The estimates have all moved closer to the correct depth. One can also see how the error ellipsoids are more elongated in the depth direction which is more like the true uncertainty.

V. COMPOSITE MEASUREMENTS

The purpose of the tracking filter is to provide summarized composite measurements of the robot motion and detected features to a navigation module. The navigation module can then use the measurements to do SLAM, localization to a prior map, and other navigation tasks. A composite measurement provides an estimate of two pose states and a number of feature states and includes an accurately calculated covariance between these states. This allows a global SLAM filter to update its state from the previous

Fig. 5. Here we focus on a single feature and show the progress from first detection to initialization and then the first few adjustments of its position. We are using Eq. 6 as the theta parameterization. Each plot shows the z vs y coordinates. The dots are the sonar detections at the mean positions (i.e., right in the middle of the model's distribution). The small circles are the estimated locations of the infant object after adding each measurement. The large circle is the position of the newest estimate. The earlier positions are shown in each frame for comparison. We can see that the initial estimated location agrees with the sonar measurement. Then after two measurements a location far below the nominal measurement modes is the best fit to the model. As more measurements are added the location begins to settle to a moderate depth below the measurement modes. After the seventh measurement the feature is initialized. The estimated feature locations from the previous frames are shown by smaller x's. The large x is the feature position for that frame.

pose to the new pose while adding all the features. The features can then be merged where possible and the previous state marginalized out. It essentially combines the prediction and update over a section of the path. All correlations are accounted for and are in fact more accurate than they would be if the measurements were added one at a time to the global filter. This is because we re-linearize the entire set taken together.

The important point is to avoid breaking off a map section that contains infant objects that are still collecting data. If an infant is not matched to for 10 seconds it is deleted. If a feature is seen more than 10 times it is normally initialized. So after at most 100 seconds an infant has either been initialize or deleted. In practice there are usually no infants attached to poses more than 15 seconds old.

The pose states older than the oldest pose with an infant can be broken off into a composite measurement. We try to collect a fair amount of states before forming a composite so that the states have had time to be fully observed and the covariance can constrain all the dimensions properly. We generally try to collect about 30 nodes or more into a composite. The procedure is:

- 1) Cut off the younger states and their measurements from the filter and start a new filter with them. Some features that have measurements on both sides of the cut-off

- will need to be copied and the correspondence noted.
- 2) Call the oldest and youngest poses that are left P_1 and P_2 respectively. Move P_1 , P_2 and all the features with their reference frame poses to the bottom rows of R .
- 3) Compute the covariance of those states.
- 4) Convert from the 9 dimensional feature representation to a 3 dimensional xyz world frame parameterization. The Jacobians of this coordinate change are used to compute a new covariance.
- 5) The mean is taken as the current values of all the new state variables. These are P_1 , P_2 and the xyz for each feature.

The base frame is the youngest pose from the previous composite measurement. We haven't any absolute measurement of x and y . The composite measurement is therefore only relative to the base in x and y . The navigation filter will have its own estimate of x and y for the base frame from matching the features globally, GPS, or other information. We need to give a measurement which is relative to that estimate.

We add the base frame state to our composite measurement using the dead-reckoning measurement between it and the pose P_1 . we then marginalize out the pose P_1 and delete the rows for the x and y of the base pose. We subtract the base pose x and y from all x and y in the mean. The navigation filter can then add in its own estimate for the base x and y . No rows with information are deleted.

The entire filter can be re-linearized around the current state while forming the composite measurement. This calculation is order N where N is dimension of the filter. The linear complexity is a result of the short and simple paths that the robot follows for these estimates. For long paths with loops the complexity would be cubic. This ability to re-linearize the sub system is a major advantage. Re-calculating all the A_i also avoids having to do any messy and time consuming rearranging of the R matrix. We just rebuild the two sections around the current state estimate. The calculation is thus also linear in the number of measurements.

VI. EXPERIMENTS

The experiments were carried out on a Nekton Ranger AUV. The Ranger is a small AUV about 1.3 m long and 20 cm in diameter. It has a single propeller aft that is steerable around two axes. It carried a payload consisting of the BlueView ProViewer P900 sonar and a CPU for running our navigation program. The Ranger moves at 80 cm/sec and the ping rate was about 1 Hz.

The features used were 12 radar reflectors deployed about 1 m above the river bed in a pattern of two rows of five and a center row of two reflectors. A prior map of the feature locations was extracted by hand from side-scan sonar data collected by a REMUS AUV, see top of Figure 6. This was meant to be an accurate mock up of the actual application of navigation in a field of floating underwater mines. The REMUS maps had a estimated error of about 5 m due to errors in localizing the vehicle and in extracting the features from the data. The GPS errors from both vehicles also

contributes to the differences in the two maps. The Ranger consistently found the features offset from the locations on the prior map. The task was to lock the robot to the map and not to find the true locations of the features in the GPS frame.

In Figure 6 (b) we show one result of matching the tracked features, forming composite measurements, and then updating a global EKF that was initialized with the *a priori* map and its uncertainty. The robot saw all the features except the top feature in the center row which nevertheless had its covariance ellipsoid shrink as a result of its *a priori* error being highly correlated to the other features. This example uses the cubic parameterization of the features. When the same data is processed using the linear theta distribution the result was poorer in several respects. First the center row feature near the bottom was not tracked sufficiently for initialization and thus never used. Second the feature estimates of depth all became over confident and less accurate than those of the non-linear model. Third, on the final dive some features were miss-matched causing an error in the pose estimate.

The linear model did work rather well. As expected, its estimates tend to move around less in depth. It failed to track some borderline features, that is, features that we see on the edge of the field of view for a short time.

On the other hand we see that the feature tracker using the non-linear theta parameterization works better in some tangible ways. The most consistent difference between the linear and non-linear models are in the estimation of feature depth and its covariance. We have validation of the overall concept of doing high dimensional SLAM on small snippets of the robot trajectory, forming composite measurements and updating a global filter at low frequency.

This result is one sample of a series of experiments over a one week period in November 2006. The results were that the tracking and matching performed satisfactorily on all runs without changing any parameters.

We mention that we also ran the tracker as an extended information filter and found that the matrices did occasionally become poorly conditioned which caused errors in the composite measurements. The problem could be solved by taking extra care but the square root filter had no such problems and appeared to be more accurate. In other words the square root filter is more robust.

VII. CONCLUSION

We have demonstrated a real time implementation of feature tracking using a blazed array forward looking sonar on an underwater robot. The tracking of features was improved by accurately modeling the features with a novel feature parameterization. The tracking filter provides accurate detailed information on the robot's local environment that can be used by other on board modules such as SLAM, localization or homing type control on a target. This was demonstrated to be sufficiently accurate for the real tasks assigned to the robot in the application.

Fig. 6. Figure (a) shows 3 separate estimates. One, the *a priori* hand map made from the REMUS AUV data, the $\times$'s. One of the 12 features was not seen in the REMUS data. Two, the estimated positions of these same 11 points as produced by our SLAM algorithm, shown as 1-sigma ellipsoids (black dots). These are a subset of the estimates shown in (b). There is no effort to match the two maps in (a). Three, the path shown is the GPS corrected path not using the map at all. The jumps in the path are GPS fixes. Figure (b) shows path and map after correlating in a separate global EKF using the output of the tracking filter. The robot matched to the *a priori* map on all six dives. The new estimated positions of the prior map points are distinguished by the $\times$'s. Here the jumps are both from GPS fixes on the surface and matches to the *a priori* map while submerged. Notice how the very large GPS adjustment in one loop at the top of map (a) is hardly noticeable in (b) due to the better navigation estimate while under water

The estimator used was both computationally efficient and numerically stable. It could and was periodically completely re-linearized around new estimated states consisting of up to one minute of robot trajectories.

We showed how the feature representation and estimator could be used to effectively deal with highly non-Gaussian measurements. The features did converge to depths closer to the actual depths in a majority of the cases. The covariances of the feature locations were more realistic as well.

When combined with an EKF SLAM algorithm initialized with an uncertain prior map, we showed that we could correlate the robot pose to the prior map and navigate accurately relative to that map. This was demonstrated repeatedly on data collected over 5 days.

REFERENCES

- [1] J. Montiel, J. Civera, and A. J. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Proc. of the Robotics Science and Systems Conference (RSS06)*, vol. 1, 2006.
- [2] S. Williams and I. Mahon, "Simultaneous localization and mapping on the Great Barrier Reef," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA04)*, vol. 1, 2004, pp. 1771–1776.
- [3] D. Ribas, P. Ridao, J. Neira, and J. Tardos, "Slam using an imaging sonar for partially structured underwater environments," in *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS06)*. IEEE, 2006.
- [4] J. J. Leonard, R. Carpenter, and H. J. S. Feder, "Stochastic mapping using forward look sonar," *Robotica*, vol. 19, pp. 467–480, 2001.
- [5] I. Tena, S. de Raucourt, Y. Petillot, and D. Lane, "Concurrent mapping and localization using sidescan sonar," *IEEE Journal of Ocean Engineering*, vol. 29, no. 2, pp. 442–456, Apr. 2004.
- [6] M. G. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte, and M. Corba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, June 2001.
- [7] A. Eliazar and R. Parr, "Dp-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks," in *Proc. of the Joint Conference on Artificial Intelligence (IJCAI-03)*, 2003.
- [8] G. Grisetti, C. Stachniss, and W. Burgard, "Improved grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA05)*, vol. 1, 2005, pp. 2432–2438.
- [9] U. Frese and L. Schroder, "Closing a million-landmarks loop," in *Proc. of the IEEE/RJS Conference on Intelligent Robots and Automation (IROS 2006)*, 2006.
- [10] R. Rikoski and J. J. Leonard, "Trajectory sonar perception," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA03)*, 2003, pp. 963–971.
- [11] S. Se, D. G. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *International Journal of Robotics Research*, vol. 21, no. 8, pp. 735–58, 2002.
- [12] F. Lu and E. Milios, "Globally consistent range scan alignment for environmental mapping," *Autonomous Robots*, vol. 4, pp. 333–349, 1997.
- [13] A. Davison, "Real-time simultaneous localization and mapping with a single camera," in *Proc. of the IEEE International Conference on Computer Vision*, 2003.
- [14] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using FastSLAM," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA03)*, vol. 1, 2003, pp. 1985–1991.
- [15] J. Sola, A. Monin, M. Devy, and T. Lemaire, "Undelayed initialization in bearing only slam," in *Proc. of the IEEE/RJS International Conference on Intelligent Robots and Systems*, 2005, pp. 2499–2504.
- [16] J. Folkesson, P. Jensfelt, and H. I. Christensen, "Graphical SLAM using vision and the measurement subspace," in *Proc. of the IEEE/RJS International Conference on Intelligent Robots and Systems*, 2005.
- [17] F. Dellaert, "Square root sam: Simultaneous location and mapping via square root information smoothing," in *Robotics: Science and Systems*, 2005.