Toward Robust Localization and Mapping for Shallow Water ROV Inspections

Nathaniel Goldfarb, Jinkun Wang, Shi Bai and Brendan Englot

***Abstract*— We propose a portfolio of filtering methods for a remotely-operated vehicle (ROV) that relies on a compass, gyro, depth sensor and ultra-short baseline (USBL) positioning system for localization in shallow-water environments. Our goal is to maintain an accurate state estimate that will be suitable as a basis for decision-making in the course of carrying out an autonomous underwater inspection. We employ Kalman filters in conjunction with an outlier filter to produce a reliable estimate in the presence of noisy and spurious data. Our localization result is then used as a basis for 3D occupancy mapping, in which further filtering and inference techniques are applied to remove false returns from the ROV's scanning sonar. Localization results from two field deployments of the ROV are given.**

I. Introduction

The goal of this paper is to present a multi-faceted method for the localization of a remotely operated vehicle (ROV) in an unstructured and disturbance-filled shallow water environment. We are interested in port and harbor infrastructure inspection scenarios in which the general location of the infrastructure is known, but its integrity may have deteriorated and a perfect prior map does not exist (and thus cannot be used to drive a localization process). We consider an ROV that relies on an ultra-short baseline (USBL) positioning system, a compass, gyro, and depth sensor for localization, with a scanning sonar for mapping. Each of these modes of sensing presents advantages and disadvantages in different portions of the environment. The USBL will be reliable at close range to its base station, and when the line-of-sight to the robot is clear. The compass will fail in close proximity to steel structures. We will assume that structures will not be reliably present in the robot's field of view to the extent needed to use sonar scan-matching in the localization process. A key objective is to reject spurious measurements in instances when these sensors are unreliable, and filter additive noise when the sensors are well-behaved.

The proposed localization methodology is implemented using a VideoRay Pro4 ROV equipped with a Tritech Micron scanning sonar and MicronNav USBL positioning system, shown in 1. Unlike prior work that has relied largely on sonar scan-matching to support localization [5], [11], our ROV system must operate in environments that are cluttered in some areas and open in others, without the frequent availability of unique features from surrounding structures. Other work has relied on a USBL system to supply near-truthful positioning during inspection tasks [8], but our system will typically operate in shallow water where a USBL offers lower-precision positioning, and at long ranges from the base station the measurements will be spurious and prohibitively noisy. Unlike prior work that has used the VideoRay system for bridge inspection [6], our system is intended to operate in areas where the infrastructure in the environment may have deteriorated, our prior map is unreliable, and localization independent from a prior map is required.

Fig. 1: Photo of Stevens Institute of Technology's VideoRay Pro4 System, equipped with Micron sonar and MicronNav positioning system.

Our localization methodology is comprised of an outlier fillter, which determines whether a measurement is spurious, and a Kalman filter [3], into which all non-spurious measurements are introduced. A simple linear, single-integrator model of vehicle dynamics is adopted, and separate Kalman filters are implemented for the robot's translational and rotational degrees of freedom, which are measured and propagated at different rates. In this work, the ROV is not carrying out aggressive maneuvers, and is treated instead as a slow-moving sensor node that is propagated from one waypoint to another in the course of mapping and inspecting the environment. The Mahalanobis distance is adopted as our metric for the evaluation of outliers, and localization results from two field experiments are given. In the first, the USBL base station is allowed to sway freely in the water, contributing to larger quantities of spurious positioning data, and substantial noise. In the second, the base station is physically stabilized to limit its range of motion, resulting in fewer outliers and less severe noise. We conclude with the presentation of preliminary results on 3D mapping, which

N. Goldfarb, J. Wang, S. Bai and B. Englot are with the Department of Mechanical Engineering, Stevens Institute of Technology, Castle Point on Hudson, Hoboken NJ 07030 USA, {ngoldfar, jwang92, sbai1, benglot} @stevens.edu.

This is a DRAFT. As such it may not be cited in other works. The citable Proceedings of the Conference will be published in IEEE Xplore shortly after the conclusion of the conference.

978-0-933957-43-5 ©2015 MTS

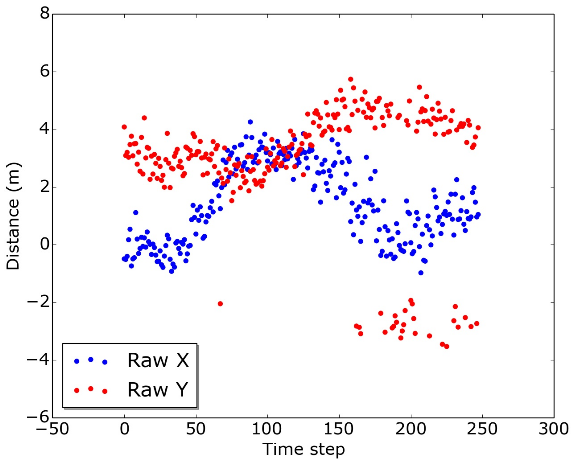


Fig. 2: Raw x and y positioning data, plotted against time. Each time step represents a sample from the robot's positioning system, which operated at 2Hz. Outliers can be seen at bottom right.

leverages our localization results and includes the use of additional filtering methods for range sensors that have met with much success in ground robotics applications [10].

II. Outlier Filtering of ROV Positioning Data

The ROV's raw position and orientation data would not yield a meaningful map of the structures observed by the sonar without additional filtering. Inaccuracies are caused by noise and spurious outliers in the pose data, and false returns from the sonar. A representative example of raw positioning data in the horizontal plane is plotted against time in Figure 2. The data was gathered in the Hudson River, in water approximately 3 meters deep, while the robot was within a 10 meter distance of the USBL dunking transducer "base station". This data was combined with the output of the robot's depth sensor to produce a 3D position estimate, rather than rely on the z-measurement supplied by the USBL.

For the outlier filter, the Mahalanobis distance was adopted as our metric of choice to find and reject spurious data. This widely-used metric [1] is a measure of how far a point falls from the the mean of multivariate data. It is defined by the following formula:

$$MD_i = ((\mathbf{R}[i] - \bar{\mathbf{R}}))^T (cov(\mathbf{x_{list}}, \mathbf{y_{list}}))(\mathbf{R}[i] - \bar{\mathbf{R}}))^{\frac{1}{2}} \tag{1}$$

where $\mathbf{R}$ is a matrix whose rows are populated with the positioning system's measurements in x and y collected at each measurment step i, $\bar{\mathbf{R}}$ contains the mean values of the x and y measurements, and $cov(\mathbf{x_{list}}, \mathbf{y_{list}})$ is the statistical covariance matrix for the x and y degrees of freedom.

Our experiments were conducted by moving the ROV from waypoint to waypoint to collect measurements, with the vehicle spending some time loitering at each waypoint. With this procedure in mind, we adopted a specialized use of the Mahalanobis distance: instead of taking the mean of all data collected in the survey, we used a rolling mean. This was intended to capture the positioning data at the most recent waypoint by looking only at data within a window of finite size. Our average of the data within this window was then updated on every run, popping the oldest data point, pushing the newest, and taking a new average. This insured that the Mahalanobis distance, mean, and standard deviation were based on the most recent data and were not effected by old data from earlier waypoints in the survey.

One iteration of this procedure is outlined in Algorithm 1. Several different window sizes were examined to find the optimal size. The intuition employed when choosing a window size was that a window size too large would be adversely affected by old waypoints, and consequently it would incorrectly reject outliers. A window size too small would not reject outliers because it would not have enough history to correctly determine the vehicle's present location. A window size of 10 was chosen to keep track of a sufficient number of previous positioning data. The criteria for outlier rejection is given in Algorithm 2. The performance of the outlier filter over USBL positioning data, with different window sizes, is given in Figure 3. It is evident that in this example, a window size of 5 is inaccurate for much of the robot's trajectory, and window sizes of 10 and 15 offer improved performance, although both include some of the spurious data visible in the bottom right of each plot.

Algorithm 1 Rolling Mahalanobis distance

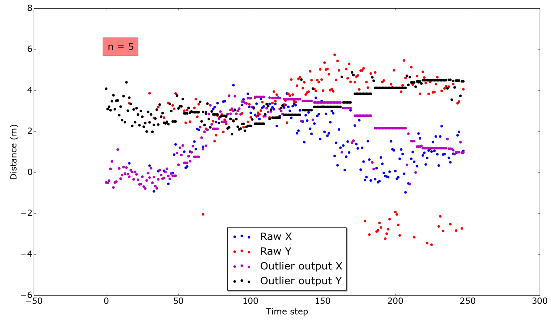
$[x_{new}, y_{new}] \leftarrow$ *getNewestData()*;
$\mathbf{R}.deleteRow(0)$;
$\mathbf{R} \leftarrow \mathbf{R} \cup [x_{new}, y_{new}]$;
$cov(x_{list}, y_{list}) \leftarrow computeCovariance(\mathbf{R})$;
$\bar{\mathbf{R}} \leftarrow computeMean(\mathbf{R})$;
$MD_i = computeMD(\mathbf{R}, cov(x_{list}, y_{list}), \bar{\mathbf{R}})$;
$MDList \leftarrow MDList \cup MD_i$;
$MDList.delete(0)$;

Algorithm 2 findOutlier($MD_i, MDList, w$)

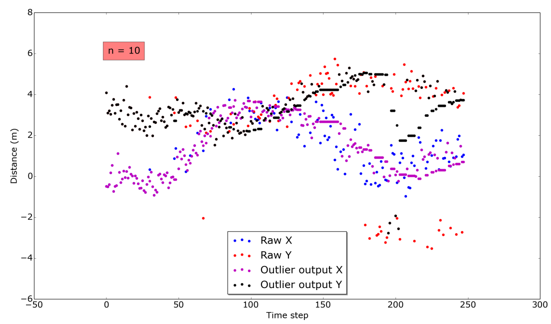
$isOutlier \leftarrow 0$;
if $MD_i >$ mean($MDList$) +w*std($MDList$) **then**
  $isOutlier \leftarrow 1$;
return $isOutlier$

III. Kalman Filtering

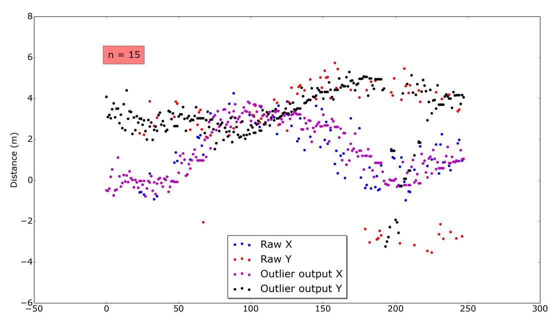
While the outlier filter removed most of the spurious data, the remaining data were still observed to be noise-corrupted and in need of additional filtering. To produce a more accurate ROV pose estimate, Kalman filtering was implemented. A single integrator linear model of vehicle dynamics was employed, which assumed constant velocity. Under the slow and conservative movement of the vehicle during our simulated inspections, this approach proved to be effective, despite its simplicity. Separate filters were applied to the translation and orientation of the robot, which were



(a) Window size of 5.



(b) Window size of 10.



(c) Window size of 15.

Fig. 3: Results of the outlier filter over USBL positioning data, with three different window sizes. Filtered data is plotted for every time instant that a measurement was received, and a zero-order hold is applied over time instants where the underlying positioning data has been rejected.

sampled at different rates (positioning at 2Hz, and orientation at 20Hz). Positioning data was supplied by the USBL and depth sensor, and orientation data was supplied by internally blended measurments of roll, pitch and yaw from the robot's gyro and 3D tilt-compensated compass. In the models employed, external control inputs were neglected, as the ROV was driven slowly under manual control, and additive process noise was assumed to influence each degree of freedom. The z-coordinate supplied by the robot's depth sensor was assumed perfect, and not included in the filter, due to its precision in comparison to the other sensors. The process noise covariance Q and the observation noise covariance R were assumed to be identity matrices multiplied by scalar coefficients, denoted by q and r respectively, allowing for simple tuning of each filter, which had comparable levels of noise among all degrees of freedom in position and in orientation. The models assumed in each Kalman filter are given below:

Translation Kalman filter model

$$\begin{bmatrix} \dot{x}_{i+1} \\ \dot{y}_{i+1} \\ x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ dt & 0 & 1 & 0 \\ 0 & dt & 0 & 1 \end{bmatrix} * \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ x_i \\ y_i \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ w_x & 0 \\ 0 & w_y \end{bmatrix}$$

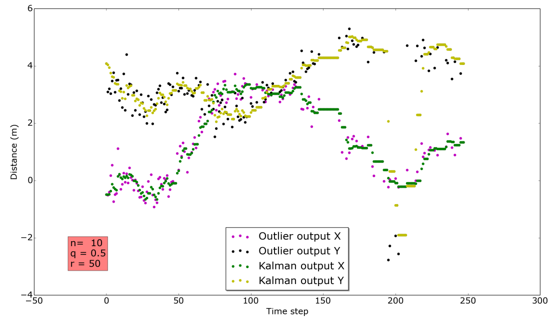
Orientation Kalman filter model

$$\begin{bmatrix} \dot{\phi}_{i+1} \\ \dot{\theta}_{i+1} \\ \dot{\gamma}_{i+1} \\ \phi_{i+1} \\ \theta_{i+1} \\ \gamma_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ dt & 0 & 0 & 1 & 0 & 0 \\ 0 & dt & 0 & 0 & 1 & 0 \\ 0 & 0 & dt & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \dot{\phi}_i \\ \dot{\theta}_i \\ \dot{\gamma}_i \\ \phi_i \\ \theta_i \\ \gamma_i \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ w_\phi & 0 & 0 \\ 0 & w_\theta & 0 \\ 0 & 0 & w_\gamma \end{bmatrix}$$

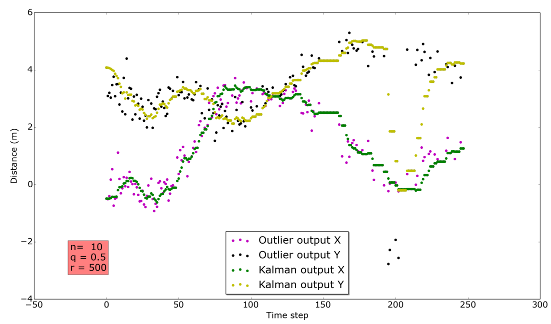
Figure 4 gives the results of the Kalman filter applied to the positioning data of Figure 3. Reducing the influence of the spurious data at the bottom right of each plot was challenging, and larger multipliers of the R matrix were more successful at doing so. Figure 5 gives results of the Kalman filter applied to the ROV's corresponding angular degrees of freedom, in which substantial noise in the roll and pitch measurements was successfully mitigated.

IV. Positioning System Stabilization

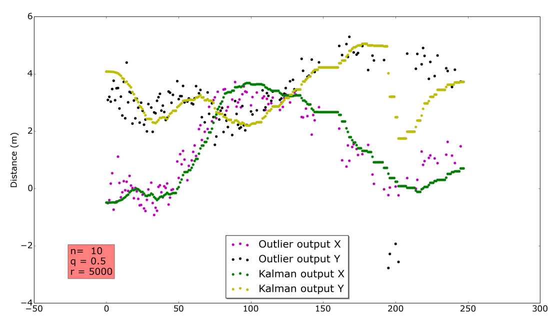
A second test was conducted in which the ROV was used to inspect the T/V Kings Pointer at the U.S. Merchant Marine Academy in Kings Point, NY. The trajectory of the ROV as it was manually driven along the side of the hull is given in Figure 6. Rather than swaying freely in the water as was the case for the results of Figures 2-4, the USBL positioning system's dunking transducer was rigidly secured to obtain more accurate positioning data. A telescoping PVC system was built to solve this problem, which consisted of two concentric pipes that allowed the depth of the transducer to be varied. A bracket was made to secure the dunking



(a) $R = r_{init} I$

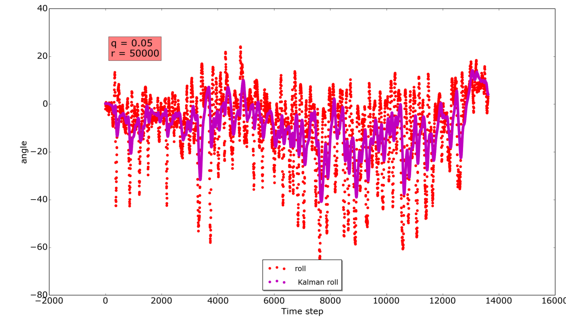


(b) $R = 10 * r_{init} I$

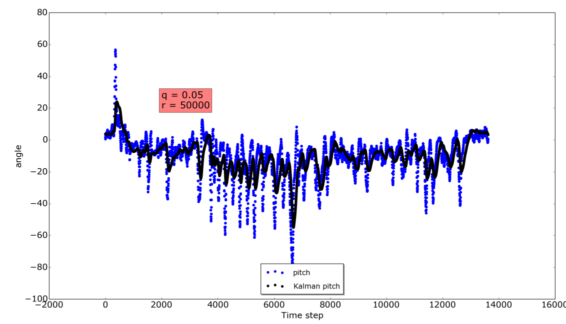


(c) $R = 100 * r_{init} I$

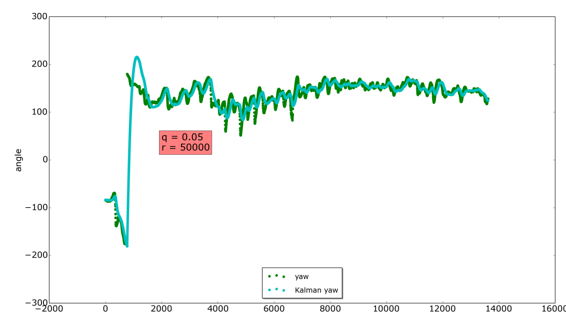
Fig. 4: Results of a linear Kalman filter applied to the USBL positioning data, under a fixed choice of process noise covariance Q, with three different selections of the sensor noise covariance R, varying across two orders of magnitude.



(a) Roll angle data and estimate.



(b) Pitch angle data and estimate.



(c) Yaw angle data and estimate.

Fig. 5: Results of a Kalman filter applied to the ROV's angular degrees of freedom, for the pose history that corresponds with the position history shown in Figures 2-4.

transducer to the end of the pipe, while its cable was fed through the pipes. The system extended more than a meter into the water and was then secured to the side of the ship with a brace made from 80-20 aluminum bars, tied with ratchet straps. This system is depicted in Figure 7. Outcomes from the outlier filter and Kalman filter applied to the resulting positioning data are given in Figure 8. Although the positioning data is less noisy in general, the outlier filter successfully eliminates most of the spurious measurements visible at top of Figure 8a.

V. 3D Mapping

A Tritech Micron DST sonar was used to map the structures in the surrounding environment, with the data, registered to the robot's position estimates, gathered in the form of point clouds. A representative sonar scan of the hull of the T/V Kings Pointer is given in Figure 9. Radial lines in the figure represent acoustic interference between the sonar and the MicronNav positioning system's vehicle-mounted modem. Extracting true returns from the hull was attempted by selecting returns within a specific interval of

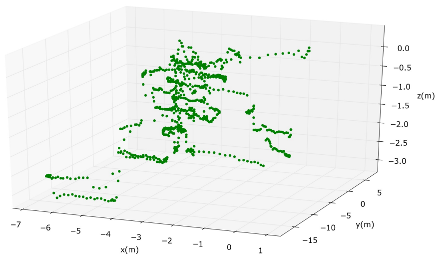


Fig. 6: The 3D trajectory of the ROV as it was driven manually along the hull of the T/V Kings Pointer, with the aid of the USBL stabilization apparatus shown in Figure 7.

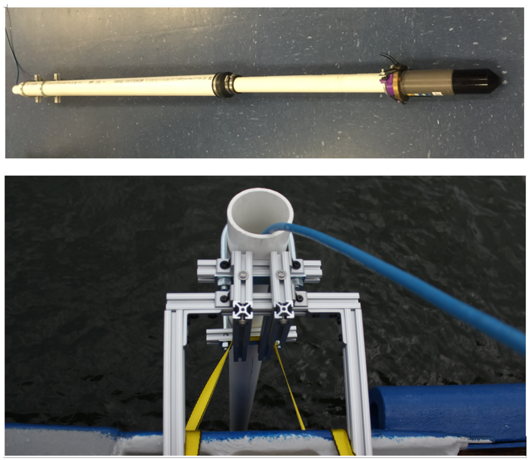
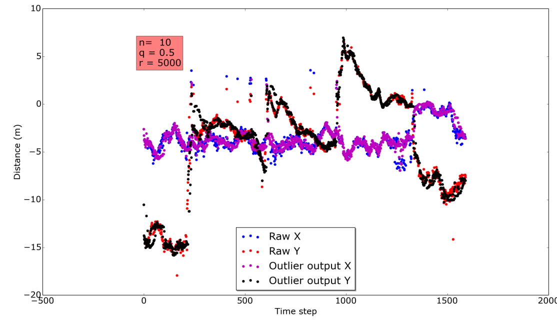
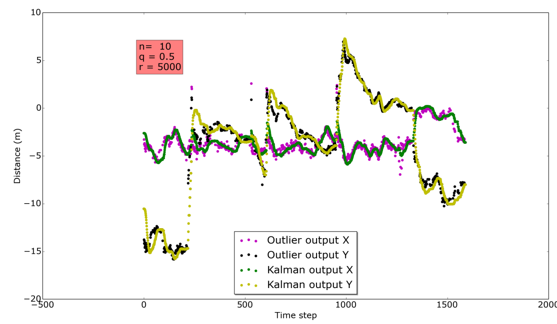


Fig. 7: Device used to stabilize the motion of the USBL positioning system.



(a) Outlier filter output.



(b) Kalman filter output.

Fig. 8: Results of the outlier filter and Kalman filter applied to the x and y positioning measurements that are shown in Figure 6.

intensities and over a specific window of ranges, with the aim of eliminating acoustic interference and other erroneous returns.

To build a real-time map, the robot operating system (ROS) [7], the point cloud library (PCL), [9] and the OctoMap occupancy mapping server [4] were used. Real-time data gathering in Linux was additionally supported by the use of the VideoRay ROS packages developed at the Georgia Institute of Technology [2]. Octomap was used to build and maintain a 3D map and publish it to ROS's RViz visualization tool, where it was displayed. The resolution of the map's grid cells was set to $0.5m$ because the data was sparse and a coarse resolution was needed to identify structures. The data passed through the intensity and range filters was then sent to a widely-used range beam filtering algorithm [10]. This algorithm determined the probability that a point was a true hit. Any point determined to have a high probability of being a hit was collected, and the surviving data was processed by PCL's voxel filter. This final step achieves sub-sampling by converting the data from a point cloud to a coarser set of voxels, and back to a point cloud with points at the centroid of each voxel. The data was then set to the OctoMap server to be rendered in 3D.

The range beam "true hit" procedure is adapted from [10] and given in Algorithm 3. The algorithm takes as input a sonar beam z_t and estimates the probability that each bin along the beam is a range return from an observed structure. The likelihood value is computed by weighing both the range and the intensity of each point. The algorithm relies on 6 intrinsic parameters which must be tuned to obtain an accurate map. These included the four weighting parameters; z_{hit}, z_{short}, z_{rand}, z_{max}, and the parameters σ^2 and λ, which are needed to calculate the probabilities of hits and shorts respectively. The intrinsic parameter σ^2 represents the variance of true hits, and λ is the time constant of exponential decay describing spurious short returns. Parameters were obtained by hand-tuning over representative data and observing the effects it had on the map. The procedure for computing the instrinsic parameters is given in Algorithm 4.

The procedure assumes four different types of measurements; hit, short, max, and random. The hit probability is the probability that a point represents the true location of a measured object. The short probability measures the likelihood that a point falls short of the true object's location. The max probability describes points that lie at the maximum range measured by the beam. Finally, the random probability

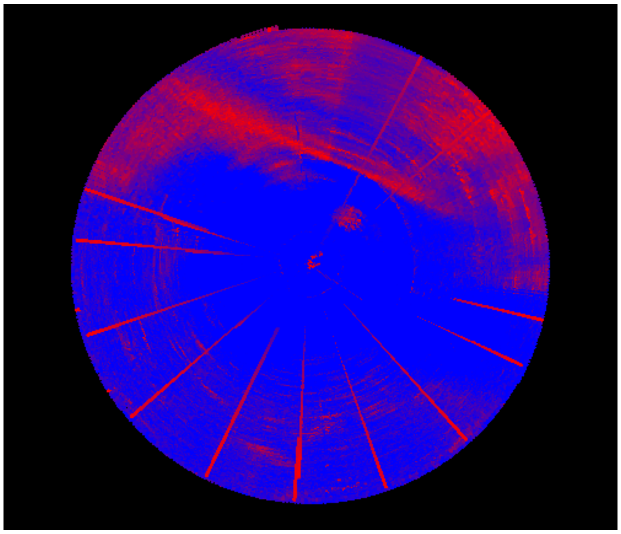


Fig. 9: A representative sonar scan collected during the inspection of the T/V Kings Pointer. Returns from the hull are visible at the top of the image, and radial lines caused by acoustic interference between the sonar and positioning system are visible in the lower half of the image.

Algorithm 3 Range Beam "True Hit" Likelihood

for all bins $i \in \{1, ..N\}$ along range beam t **do**
  $p = z_{hit} p\big(z_t^i | x_t, m\big)_{hit}, + z_{short} p\big(z_t^i | x_t, m\big)_{short}, + z_{max} p\big(z_t^i | x_t, m\big)_{max}, + z_{rand} p\big(z_t^i | x_t, m\big)_{rand}$
  if $p > 0.9$ **then**
    $cloud \leftarrow cloud \cup z_t^i$

describes the likelihood that a point is returning a false reading as a result of noise. For proper tuning, the filter also needs to know the true location of the object, which in the best case scenario would be validated by another sensor. However, with no other perception sensors present, this point was assumed to be the "center of mass" of the intensities returned along a given sonar beam. This true measurement will be denoted as z^*. Each associated probability is then multiplied by its respective weights, which were computed using Algorithm 3, and then the probabilites of each event are summed together to determine the probability that a point represents a true return from an object. The methods to compute each probability are shown in Equations 2 through 5. Any point with more then a 90% probability of being a "true hit" was added to the occupancy map. A resulting occupancy map comprised of a majority of true returns from the hull of the T/V Kings Pointer is given in Figure 10.

$$p(z_t^k | x_t, m)_{hit} = \begin{cases} \eta \mathcal{N}(z_t^k, z_t^*, \sigma^2), & \text{if } 0 < z_t^k < z_{max}. \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

$$\eta = (\int 0 z_{max} \mathcal{N}(z_t^k, z_t^*, \sigma^2) dz_k^*)^{-1}$$

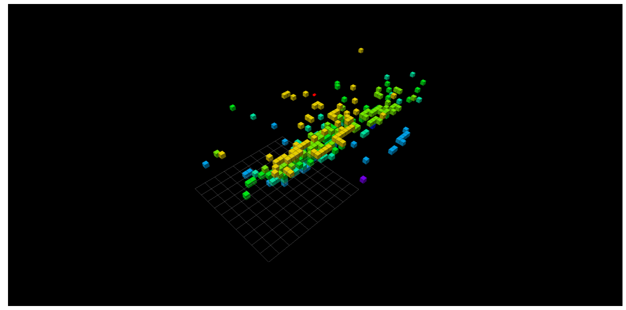


Fig. 10: A 3D occupancy map primarily comprised of true returns from the hull of the T/V Kings Pointer, collected during the execution of the trajectory illustrated in Figure 6. Colors indicate the relative height of the map's contents. The single red cell indicates the location of the ROV at the termination of the inspection.

Algorithm 4 Calculate Intrinsic Parameters

$totalZ = \sum_i z_i$
for all z_i in Z **do**
  $\eta = \Big[\ \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{hit}, + \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{short} + \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{max} + \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{rand}\ \Big]^{-1}$

  $e_{i,hit} = \eta \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{hit}$
  $e_{i,short} = \eta \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{short}$
  $e_{i,max} = \eta \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{max}$
  $e_{i,rand} = \eta \mathrm{p}(\ z_i \mid x_i\ ,\ m\ )_{rand}$

  $z_{hit} = |totalZ|^{-1} \sum_i e_{i,hit}$
  $z_{short} = |totalZ|^{-1} \sum_i e_{i,short}$
  $z_{max} = |totalZ|^{-1} \sum_i e_{i,max}$
  $z_{rand} = |totalZ|^{-1} \sum_i e_{rand}$

  $\sigma^2 = \frac{\sum_i e_{i,hit} \left(z_i - z_i^*\right)}{\sum_i e_{i,hit}}$

  $\lambda = \frac{\sum_i e_{i,short}}{\sum_i e_{i,short} z_i}$

$$p(z_t^k | x_t, m)_{short} = \begin{cases} \frac{\lambda e^{-\lambda z_t^k}}{1 - e^{-\lambda z_t^k}}, & \text{if } 0 < z_t^k < z_t^*. \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

$$p(z_t^k | x_t, m)_{max} = \begin{cases} 1, & \text{if } z_t^k = z_{max}. \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

$$p(z_t^k | x_t, m)_{rand} = \begin{cases} \frac{1}{z_{max}}, & \text{if } 0 < z_t^k < z_{max}. \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

VI. Conclusion

The methods presented in this paper have aided in the development of reliable 3D localization and mapping for an ROV in a shallow water environment. The two step approach of using an outlier filter and Kalman filter work to achieve improved localization. Without these tools, the map created from the raw sensor data is unusable for any practical purpose. However, this data alone is still is not enough to build an accurate map; the sonar data is initially too noisy to automatically identify objects in the environment. A multi-step filtering process was applied to the sonar data to remove false hits and reflections. The probabilistic method used in this paper assumes uncertainty in the data, which aids in the removal of noise and extracts meaningful returns from structures in the environment. In order to achieve a usable map in an underwater unstructured environment, it is necessary to filter both the pose information and the sonar data, and further improvements will be required to allow autonomous decision-making over this data.

ACKNOWLEDGEMENTS

We thank Prof. Carolyn Hunter and the crew of the T/V Kings Pointer at the U.S. Merchant Marine Academy in Kings Point, NY for granting us access to the ship. We also thank Dr. Kevin DeMarco for his help with the use and adaptation of his VideoRay ROS packages.

REFERENCES

[1] I. Ben-Gal. Outlier detection. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 131–146. Springer, 2005.

[2] K.J. DeMarco, M.E. West, and A.M. Howard. Sonar-based detection and tracking of a diver for underwater human-robot interaction scenarios. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 2378–2383, 2013.

[3] A. Gelb, editor. *Applied Optimal Estimation*. The MIT Press, 1974.

[4] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, 2013. Software available at `http://octomap.github.com`.

[5] H. Johannsson, M. Kaess, B. Englot, F. Hover, and J. Leonard. Imaging sonar-aided navigation for autonomous underwater harbor surveillance. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 4396–4403, 2010.

[6] R.R. Murphy, E. Steimle, M. Hall, M. Lindemuth, D. Trejo, S. Hurlebaus, Z. Medina-Cetina, and D. Slocum. Robot-assisted bridge inspection. *Journal of Intelligent Robotic Systems*, 64:77–95, 2011.

[7] M. Quigley, K. Conley, B. Gerkey, J. Faust, T.B. Foote, J. Leibs, R. Wheeler, and A.Y. Ng. ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[8] P. Ridao, M. Carreras, D. Ribas, and R. Garcia. Visual inspection of hydroelectric dams using an autonomous underwater vehicle. *Journal of Field Robotics*, 27(6):759–778, 2010.

[9] R.B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1–4, 2011. Software available at `http://pointclouds.org`.

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

[11] C. White, D. Hiranandani, C.S. Olstead, K. Buhagiar, T. Gambin, and C.M. Clark. The malta cistern mapping project: Underwater robot mapping and localization within ancient tunnel systems. *Journal of Field Robotics*, 27(4):399–411, 2010.