Optimal Path Planning for AUV Terrain Relative Navigation

1st Dylan Goff
Aerospace Engineering
Stanford University
Stanford, CA
dgoff@stanford.edu

I. Motivation

The exploration and monitoring of Earth's oceans is of great importance to our understanding of environmental changes and ecosystem health. However, carrying out these missions is no simple task, as the pressure at the sea floor is far too high for any human diver. Robotic vehicles have been created to remedy this, but they are plagued with a variety of issues due to the difficulties inherent in communicating through hundreds, or even thousands, of meters of water. For this reason (among others), many of the solutions used for localization of ground vehicles are unfeasible for underwater vehicles. A large body of scientific work exists that is dedicated to the localization and navigation of underwater vehicles, but many of these solutions are either prohibitively expensive or inaccurate. As such, novel solutions to underwater perception and navigation that mitigate or altogether avoid these issues are of great value to oceanic research.

II. Literature Review

Modern undersea exploration missions utilize two classes of underwater vehicles: remotely-operated vehicles (ROVs) and autonomous underwater vehicles (AUVs). Whereas ROVs are piloted with a fiber-optic cable tether attached to a ship, AUVs are designed to operate wirelessly and autonomously, allowing them much more freedom in terms of depth of exploration and mission length while also being operable with smaller crews [10].

Localization is vital for the operation of these systems. For example, scientific missions that involve visiting a site on the sea floor several times and any other mission that uses terrain relative navigation of the ocean floor will require some form of localization. Unfortunately, both ROVs and AUVs suffer from a lack of reliable, accurate, and cost-effective localization solutions. This is in no small part due to the inability of global positioning system (GPS) signals to penetrate through seawater enough to be useful in most cases [12]. While acoustic ranging beacon systems exist for ROV and AUV localization, these systems tend to suffer from a variety of problems including high costs, slow performance, and limitations on area of coverage [10]. Since ROVs are always in close proximity to a sea vessel, these acoustic systems are commonly used for ROV missions. AUVs instead often opt to use dead-reckoning for localization solutions since close proximity to a beacon is not required, allowing the vessel to move around more freely [10]. Because dead-reckoning integrates measurements from a variety of onboard sensors to generate state estimates, it offers improvements in mobility compared to beacons. However, over time dead-reckoning builds up an error in the position estimate known as drift, which cannot be eliminated if using dead-reckoning alone as a localization solution [10].

Because of these issues, a variety of methods have been proposed and tested for performing localization of underwater vehicles. For beacon-based systems, it is often the case that several beacons with sufficient positional variety and known locations are required for localization, which can be difficult. One study investigated the efficacy of instead using beacons without a priori beacon position estimates for AUV localization [8]. Their method involved the placement of beacons (i.e. by an AUV or airplane) and then the estimation of beacon positions by obtaining ranging measurements from the environment. The main result of this work was a range measurement outlier rejection algorithm that allowed the beacon positions to be estimated and then used for AUV localization. A simultaneous localization and mapping (SLAM) filter was then created from this, which was then shown to be effective in estimating the trajectory for an AUV [8].

In a similar vein, other work has investigated the use of AUV swarms for localization [7]. Instead of requiring a set of ranging beacons for localization, the authors proposed a system wherein several AUVs move together, with one AUV acting as a stationary landmark for the other AUVs in the swarm. Each AUV in the swarm then rotates into this stationary landmark role as time passes, with the moving AUVs keeping track of their own state and the state of the stationary AUV through the use of a particle filter. As communication between AUVs is a requirement for this approach, the authors also developed a method to facilitate sufficiently fast state communication through a compression algorithm. The authors were able to demonstrate robust position tracking for a pair of AUVs in simulation and verify its efficacy on hardware later on [6].

Similar projects have also investigated the use of autonomous surface vehicles (ASVs) in the localization of AUVs [11]. In theory, the use of GPS signals for underwater navigation would be highly valuable for AUV state estimation. As such, this study (among others) investigated the use of an ASV that operates on the water's surface, allowing the ASV to receive GPS positional measurements. The ASV then travels with its partnered AUVs and communicates with them through acoustic ranging to generate localization estimates. This study investigated some of the trade-offs with different state estimation algorithms for the ASV-AUV teaming problem. It was shown that linear state estimation techniques such as the extended Kalman filter (EKF) have subpar performance and that nonlinear state estimation techniques such as particle filters and nonlinear least squares (NLS) have much better performance, with NLS attaining the most accurate state estimates in this case. However, these nonlinear methods have their drawbacks. For particle filters, there is no guarantee on the number of particles needed to effectively converge to a solution, and computational costs rise as more particles are introduced. For NLS, the number of states increases continuously over time, meaning that the problem eventually becomes computationally intractable.

The computational complexity issue of NLS has been addressed through implementation of an online state estimation algorithm [3]. This work incorporated an on-line incremental optimizer called iSAM [4] in order to avoid the issues that NLS has with increasing computational complexity as the AUV trajectory evolves. By incorporating iSAM with NLS, the authors were able to obtain real-time state estimates with NLS while avoiding increasingly high computational costs. This allows for NLS with iSAM to be run on AUV missions for far longer than with NLS alone while still avoiding the need to resurface for GPS measurements.

Regarding the issue with linearization, work has been done to improve EKF estimates by introducing ocean current models into the state estimator [9]. The main idea for this research was to use an EKF to estimate both the ocean currents and AUV state. The authors were able to demonstrate good localization performance even in the case of changing currents, though initialization was slow. The authors proposed that improved covariance estimates for the EKF and field testing in future work could help to improve this system.

For particle filters, prior work has investigated making terrain-relative navigation robust to ocean floor terrain that lacks identifying features [1]. Particle filters rely on measurements of the terrain floor at several points. In comparing these measurements with a known topographical map of the ocean floor, a probability distribution for the current AUV state can be obtained. However, when an AUV encounters non-unique features such as flat terrain, it becomes difficult to pinpoint an exact state estimate. The result of this research was an adaptive filter which mitigated issues associated with state convergence in these environments with little in the way of unique terrain features.

Another study investigated the possibility of avoiding uninformative ocean floor terrain altogether by developing a path planning algorithm that actively seeks to travel over varied topography [5]. The authors defined a problem wherein an AUV seeks to travel from a starting way point to a goal way point. The AUV starts off with a probability distribution for its initial state. Monte Carlo Tree Search (MCTS) is then used to generate an optimal path to the goal, where the optimal path is defined to be the path that results in the highest probability of reaching the goal state. That is to say the optimal path is the one with the most varied terrain, as this will allow the particle filter to obtain the most accurate localization solutions over the trajectory. MCTS was used due to the computational intractability of brute force search over all possible paths, but other methods could potentially be applied to this path finding problem. Furthermore, this problem explored only a single start and goal way point. The author mentions the addition of more way points by increasing the computational scalability of the path finding algorithm as future work.

III. Approach

A. Simulation

The architecture for this project's simulation is built upon the MATLAB UAV package, and more specifically, from information provided in an AUV demo by MATLAB [2]. While the UAV package is normally reserved for planes and drones, AUVs can also be simulated by modifying the appropriate parameters for sensors and dynamics and by providing a custom CAD model. For this project, basic models of a seafloor map were generated using circular polygon meshes. In order to test the estimation algorithm on a variety of test cases, maps were generated with flat terrain, patterned terrain, and terrain with unique features. The final map used for testing is displayed in Figure 1.

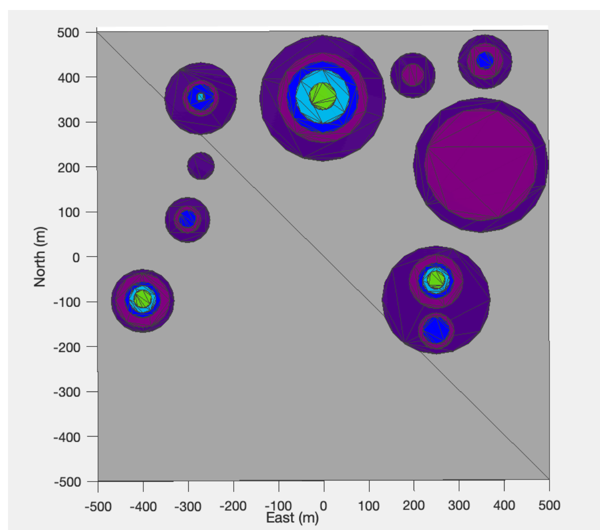

Fig. 1. Sea floor map used for simulations. Grey indicates an altitude of 0, while colors ranging from dark purple to light green indicate altitude increases of 10 meters. Light green corresponds to the highest points, which are located 50 meters above the floor.

In order to accurately model the AUV physical properties, a CAD model of an AUV was borrowed from the MATLAB AUV demo [2] and used in this simulation. To simulate the on-board sonar, the the lidar sensor that come built-in to the MATLAB UAV package was used. Modifications were made to this sensor according to the MATLAB AUV demo in order to realistically simulate the on-board sonar that would be present on an AUV. In order to simulate an on-board IMU, noisy linear and angular velocity measurements were used with recommended noise distributions. Note that, due to limits on computation time, the resolution on the simulated sonar was reduced from the recommended value in order to generate a less dense measurement point cloud.

B. State Estimation

As found in prior literature, linear estimators tend to perform poorly in underwater environments. Many of the issues with linearization stem from difficulty in modeling the effects of ocean currents on the system dynamics. As such, estimators that utilize linearizations of the system dynamics were not considered for this project. Instead, a nonlinear estimator–a particle filter–was used for state estimation. Measurements were taken to be the point cloud received by simulated sonar measurements. The algorithm as implemented in this project is shown in Algorithm 1. Note that the values of γ = 15 and Σ_{new} = $50^2 * eye(2)$ were chosen empirically after some trial and error. Note that if the mean squared error (MSE) of the particles is below the threshold γ, the particles are resampled from a new distribution defined in the algorithm.

Algorithm 1 Particle Filter

Initialize N particles from distribution μ_0, Σ_0
Initialize uniform particle weights $w_i = 1/N$
R = (lidar measurement variance) * eye(number of lidar measurements)
while goal not reached **do**
  for each particle **do**
    Propagate particle according to IMU
    Generate predicted sonar point cloud
    Calculate predicted measurement error vector
    Calculate measurement probability:
    p(x) = $exp(-1/2 * (error)^T * R * (error))$
  end for
  Calculate mean particle position estimate
  Calculate MSE = mean squared error of particles
  if MSE $>= \gamma$ **then**
    Reweight particles as normalized probabilities
    Resample particles according to new weights
  end if
  if MSE $< \gamma$ **then**
    Calculate μ_{new} = weighted mean of particles
    Resample particles with μ_{new}, $\Sigma_{new} = 50^2 * eye(2)$
  end if
  Set estimated state to weighted particle mean
end while

In order to test the performance of terrain relative navigation of the particle filter against a baseline that is known to perform well, a particle filter with four beacon ranging measurements placed at the four corners of the map was also used for one test case. System dynamics were propagated according to IMU measurements, while sea floor sonar measurements were not used.

C. Path Planning

For this project, dynamic programming (DP) was used for path planning due to the guarantee of convergence to a global minimum and its relatively low computational complexity. In order to use dynamic programming, a node cost metric must be defined. Since the goal of this project is to plan paths optimally with respect to state estimation, it was necessary to define a cost other than distance to the goal that would facilitate accurate state estimation. As is shown in the results, accurate state estimation for a particle filter with sea floor sonar measurements requires unique terrain features. In an attempt to quantify this metric, a simulated AUV with noiseless measurements was run over a set of discretized points of the map. The variance σ_i^2 of the sonar point cloud measurements was then calculated for each point. Node cost c_i for each discretized point was then set to be

$$c_i = -\sigma_i^2 \tag{1}$$

A grid of node costs was pre-computed and stored. DP was then run on a map with the bottom left corner corresponding to the starting point and the top right corner corresponding to the goal point. An optimal path was then generated by generating a trajectory according to the DP algorithm in Algorithm 2 with steps constrained to be either up or to the right in order to force the path to converge to the goal while optimizing for unique terrain features along the way. The terminal cost is defined as

$$J(x_{goal}) = h_{goal}(x_{goal}) = 0 \tag{2}$$

And the cost $J(x_i)$ at node i is defined to be

$$J(x_i) = h_{goal}(x_{goal}) + \sum_{k=i}^{N-1} g(x_k, \pi_k(x_k), k) \tag{3}$$

Where $g(x_k, \pi_k(x_k), k)$ is the cost-to-go at a future node k along the path generated by following policy π. DP is implemented as shown in Algorithm 2.

Algorithm 2 Dynamic Programming

Start from goal node: x_{k+1} = x_{goal}
while Not converged **do**
  Find possible child nodes x_k of x_{k+1}
  for child node **do**
    $J_k^*(x_k) = \min_{u_k} g(x_k, u_k, k) + J_{k+1}^*(x_{k+1})$
  end for
end while
Start from start node x_0
Greedily generate path from costs $J_i(x_i)$ generated by DP.

D. Test Cases

In order to motivate the use of a path planning algorithm, small test cases for flat terrain, repeated terrain, and terrain with unique features were generated and tested with the particle filter on a fixed, straight-line trajectory. Following this, DP was run on the full map shown in Figure 1 and a particle filter with 500 particles was tested on the generated trajectory. These results were then compared with a particle filter running on a straight-line trajectory from the start point to the end point to demonstrate the improvement in filter performance when run on an optimal path. The optimal path results were also compared with the case where the AUV uses a particle filter with ranging measurements from beacons of known location in order to compare performance to a baseline that is known to generally perform well.

IV. Results

A. Test Cases

1) Flat Terrain: In test case 1, a particle filter is run on a short path over terrain that lacks any distinguishable features. Results show that the particle filter remains incapable of converging to the true trajectory, as predicted measurements and observed measurements are identical everywhere. See figures 2 and 3.

2) Repeated Terrain: In test case 2, a particle filter is run on a short path over terrain with a repeating pattern of features. Results show that the particle filter is yet again incapable of converging to the true trajectory, as predicted measurements and true measurements are identical for the estimated and true paths due to the repeating terrain. This result demonstrates the fact that having terrain features is not in and of itself a sufficient condition for particle filter state convergence. See figures 2 and 3.

3) Unique Terrain: In test case 3, a particle filter is run on a short path over terrain that contains a single unique feature. Results show that the particle filter converges rather nicely to the true trajectory after the AUV passes over the unique feature in the terrain. This demonstrates the fact that having sufficiently unique terrain features can be a sufficient condition for the particle filter to converge to the true state. See figures 2 and 3.

B. Full Path Planning and Estimation

In this section, we discuss results for different path and measurement combinations on a full trajectory for the map shown in Figure 1. The initial true state is [-450, -450] and the true final state is [450, 450]. The particle filter is initialized with particles distributed according to $\mu_0 = [-470, -405]$ with $\Sigma_0 = 50^2 * eye(2)$ and 500 particles.

C. Full Path: Particle Filter with Sonar, Straight Trajectory

In this case, we take a straight-line path from the start to end point. This involves traversing over much flat terrain, and as a result, the particle filter (which uses sea floor sonar data) performs rather poorly over the course of the trajectory. See figures 6 and 7.

D. Full Path: Particle Filter with Sonar, DP Trajectory

In this case, we follow the path generated by the DP algorithm. A particle filter estimates the trajectory using sea floor sonar data. We can see a significant increase in state estimation accuracy along the trajectory compared to when a straight trajectory from the start to goal point is taken, as the average euclidean distance error for this path is 19.05 meters compared to 31.24 meters for the straight-line trajectory. Note, however, that while this path is optimal for state estimation, it does cover twice as much distance as the straight-line path. See figures 4 and 5 for results for this DP path particle filter.

E. Full Path: Particle Filter with Beacons, DP Trajectory

In this case, we follow the path generated according to the DP algorithm while using beacon ranging measurements with a particle filter for localization. Sea floor sonar readings are not used. This is done to have a baseline to compare to state estimation with a particle filter using sea floor sonar readings. We see that, as expected, having ranging measurements from known beacon locations allows us to localize well over the course of the trajectory. Note that this would be the case for any trajectory on this map since we assume the beacons are always within range, thus making state estimation with beacon measurements invariant to the surrounding terrain. Normally, beacons are too costly and/or impractical for missions; the results show that running a particle filter with sea floor sonar readings on an optimal trajectory could be a viable alternative to using stationary beacons. See figures 4 and 5 for the comparison between the sea floor sonar readings particle filter and the implementation with beacon ranging measurements.

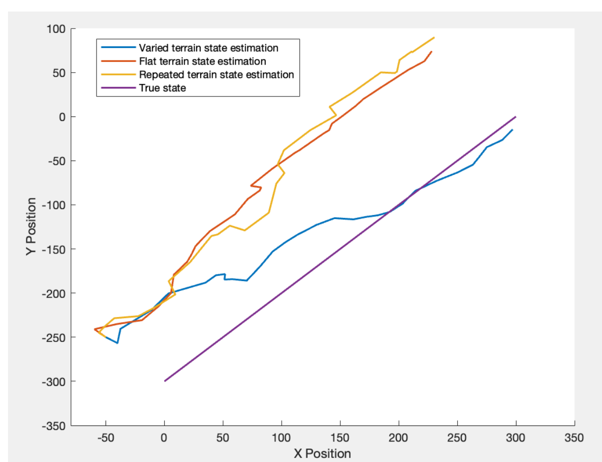

Fig. 2. Trajectories for the simple test cases from subsection A. Note that only the path that navigates over a unique feature in the terrain approaches the true state estimate.

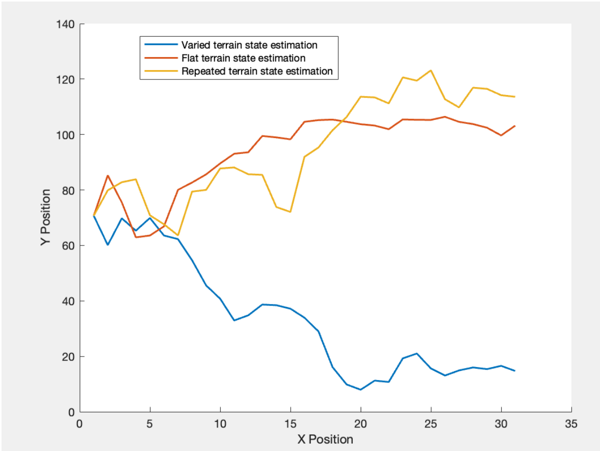

Fig. 3. Euclidean distance error for the simple test cases outlined in subsection A. Note how the trajectory that passes over a unique feature achieves a far lower state estimate error after passing over the feature.

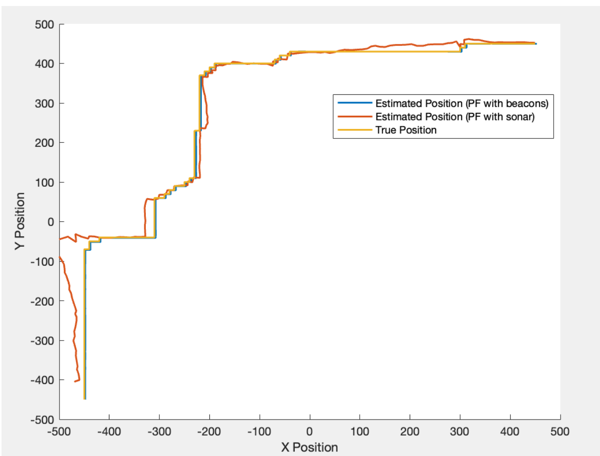

Fig. 4. Generated trajectories from following the path computed by dynamic programming. State estimation using beacons is much more accurate, as expected. However, the particle filter is able to track the trajectory relatively well once it reaches unique features on the terrain (see the map in Figure 1).

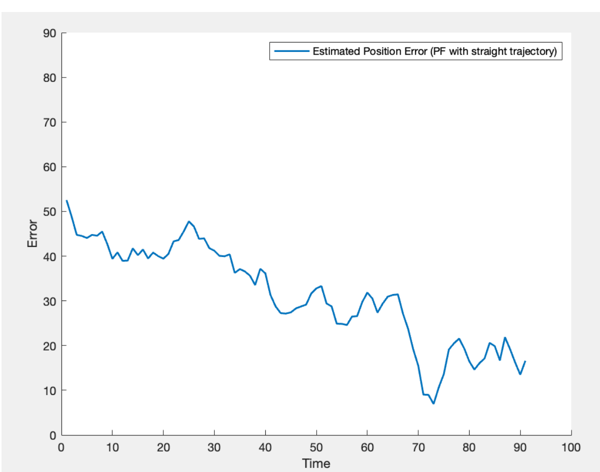

Fig. 7. Euclidean distance error in the estimated state for a straight-line trajectory from Figure 6. Notice how error remains high throughout most of the trajectory, shrinking significantly only when it reaches unique features near the goal point.

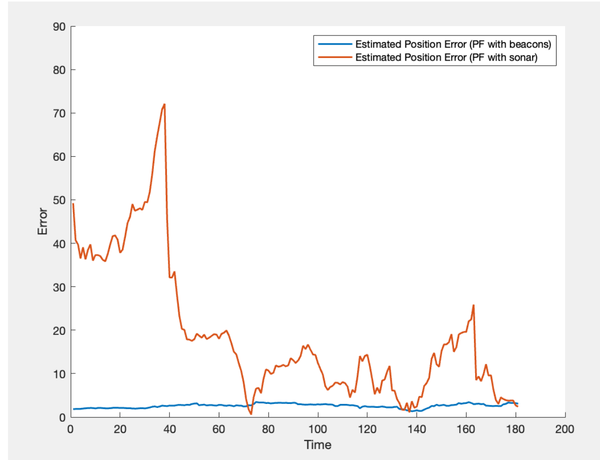

Fig. 5. State estimate errors while following the path planned by DP. Note that beacon error remains low since it is not affected by the terrain. For the solution using sea floor sonar data, we can see error dip significantly once the AUV reaches unique features in the terrain (see the map in Figure 1).

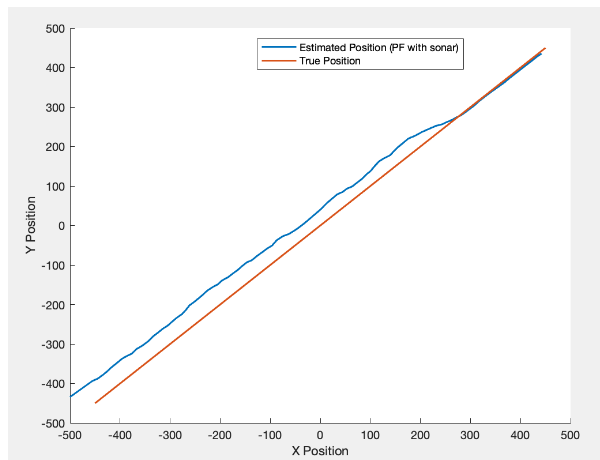

Fig. 6. Estimated path taken by a simple straight-line trajectory from the start to goal point. The AUV does not pass over any unique features (see Figure 1) until the very end of the trajectory, causing the state estimate to have a constant error along most of the trajectory.

V. Conclusions and Future Work

Dynamic programming has been shown to be effective at generating a path that will maximize the likelihood in our state estimate over time. As expected for this terrain, our average state estimate over time is improved compared to a straight-line path from the start to goal position. This provides a promising alternative to using measurements from beacons with known locations, as such a setup can be costly and impractical for many missions. In the future, considering alternative path planning algorithms and node cost metrics could be of interest. Testing other nonlinear state estimators such as the unscented Kalman filter could also be of interest to compare performance to the particle filter. Furthermore, the current DP trajectory planner forces trajectory steps to be either up or to the right in order to guarantee convergence to the goal point. Future work could consider modifications to the algorithm presented here that may allow us to relax these requirements while still guaranteeing convergence to the goal in order to generate better paths. Furthermore, it is currently unclear how this algorithm would perform on repeated terrain. Further tests on repeated terrain, as well as modifications to the node cost metric that could more accurately punish repeated terrain, may be of interest. Tests on real bathymetric data would also be of interest, as this simulation only uses low-resolution simulated terrain. Investigating the effects of perturbations on trajectories could also be of use, as the current simulation assumes dynamics that are propagated predictably with the only disturbances coming from noisy measurements. Lastly, for missions concerned with traveling to multiple goal locations, it may be of use to incorporate this work with algorithms such as ant colony optimization, which are effective at solving traveling salesman-type problems.

VI. Project Repository

Code for this project is located on the following github repository:

https://github.com/SpacePanda-42/AA273_Code/tree/66ef78b027fd84c0d8f84f7c13a47a71566eba2b/final_project

VII. References

References

[1] Shandor Dektor. "Robust adaptive terrain-relative navigation". MA thesis. Stanford University, Dec. 2015, p. 188. URL: https://purl.stanford.edu/fp830ts8344.

[2] *Design, Modeling, and Simulation of Autonomous Underwater Vehicles*. https://www.mathworks.com/videos/design-modeling-and-simulation-of-autonomous-underwater-vehicles-1619636864529.html.

[3] Maurice F. Fallon et al. "Efficient AUV navigation fusing acoustic ranging and side-scan sonar". In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 2398–2405. DOI: 10.1109/ICRA.2011.5980302.

[4] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. "iSAM: Incremental Smoothing and Mapping". In: *Robotics, IEEE Transactions on* 24 (Jan. 2009), pp. 1365–1378. DOI: 10.1109/TRO.2008.2006706.

[5] Steven Krukowski and Stephen Rock. "Waypoint planning for Autonomous Underwater Vehicles with Terrain Relative Navigation". In: *OCEANS 2016 MTS/IEEE*. Monterey, CA, Sept. 2016. URL: https://web.stanford.edu/group/arl/sites/default/files/public/publications/KrukowskiR2016.pdf.

[6] Takumi Matsuda et al. "Alternating landmark navigation of multiple AUVs for wide seafloor survey: Field experiment and performance verification". In: *Journal of Field Robotics* 35.3 (2018), pp. 359–395. DOI: https://doi.org/10.1002/rob.21742. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21742. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21742.

[7] Takumi Matsuda et al. "State Estimation and Compression Method for the Navigation of Multiple Autonomous Underwater Vehicles With Limited Communication Traffic". In: *IEEE Journal of Oceanic Engineering* 40.2 (2015), pp. 337–348. DOI: 10.1109/JOE.2014.2323492.

[8] Edwin Olson, John J. Leonard, and Seth Teller. "Robust Range-Only Beacon Localization". In: *IEEE Journal of Oceanic Engineering* 31.4 (2006), pp. 949–958. DOI: 10.1109/JOE.2006.880386.

[9] J. Osborn et al. "AUV state estimation and navigation to compensate for ocean currents". In: *OCEANS 2015 - MTS/IEEE Washington*. 2015, pp. 1–5. DOI: 10.23919/OCEANS.2015.7401906.

[10] Jose Padial. "Underwater robotic terrain-relative navigation using acoustic shadows in sonar imagery". MA thesis. Stanford University, Mar. 2017, p. 183. URL: https://purl.stanford.edu/gv205yj2414.

[11] Georgios Papadopoulos et al. "Cooperative localization of marine vehicles using nonlinear state estimation". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2010, pp. 4874–4879. DOI: 10.1109/IROS.2010.5650250.

[12] Gunnar Taraldsen, Tor Arne Reinen, and Tone Berg. "The underwater GPS problem". In: *OCEANS 2011 IEEE - Spain*. 2011, pp. 1–8. DOI: 10.1109/Oceans-Spain.2011.6003649.