Towards Gaussian Multi-Robot SLAM for Underwater Robotics

Dave Kroetsch
davek@alumni.uwaterloo.ca

Christoper Clark
cclark@mecheng1.uwaterloo.ca

Lab for Autonomous and Intelligent Robotics
University of Waterloo
Waterloo, ON N2L 3G1
Phone: 519-888-4567 x5220

July 25, 2005

Abstract

This paper presents initial steps towards developing autonomous navigation capabilities for cooperating underwater robots. Specifically, Simultaneous Localization and Mapping, or *SLAM*, capabilities are investigated for a group of micro vehicles each equipped with a single downward facing camera and an Inertial Measurement Unit (IMU). To verify the approach, simulations of the multi-robot SLAM running in a 3D environment were conducted, where vehicles in close proximity of one another exchange maps to improve localization.

1 Introduction

This research is motivated by applications involving the use of cooperating underwater robots for biological sampling in near-shore water environments. Lakes and oceans provide us with some of our most valuable resources. To manage and conserve these resources requires understanding them, which can only be accomplished through directed sampling studies. In particular, near-shore water environments are complex systems - both in their diversity and dynamics - that require spatial and temporal surveys over large areas. Multi-robot systems offer several potential advantages, including the ability to simultaneously sample such larger areas.

Enabling autonomous navigation in multi-robot systems is key to making them practical. Simultaneous Localization and Mapping (SLAM), provides a means for autonomous vehicles to navigate in previously unknown environments. SLAM constructs a map of the environment, while at the same time providing a position estimate of the robot within the map.

To enable autonomous navigation, the SLAM algorithm must be scalable and real-time capable. Constant-time implementation is critical, so that an arbitrary number of robots and landmarks can be added to the map without the implementation growing to be intractable. Furthermore, the SLAM algorithm cannot be run on a single centralized server or robot. Underwater communication is unreliable and limited in range, forcing decentralized control and SLAM, with only periodic exchanges of map information.

In this paper, the proposed approach is to merge world models from multiple vehicles using a method similar to fusing multiple measurements with the Kalman filter. This approach was originally demonstrated with data from land vehicles [1] [2], which extended the Sparse Extended Information Filter (SEIF) [3] from single robot implementation to multiple robots. The approach was shown to be scalable, be real-time capable, and function well when decentralized within ad hoc communication networks.

To validate this approach, simulations were conducted in which multiple underwater vehicles successfully carried out 3D SLAM and map merging. The simulation included a full modelling of the vehicle dynamics, but assumed landmarks were easily identified. In order to implement this functionality on a robot, a vision system is required to select landmarks, identify previously observed landmarks and compute their position relative to the craft.

What follows is a brief review of related literature, an explanation of the SLAM implementation, a description of how landmarks are identified for the SLAM algorithm, results including simulations, conclusions and future work.

2 Background

SLAM - Simultaneous Localization and Mapping - addresses the problem of using a robot to map an environment, while at the same time localizing the robot within that map. For the most part, SLAM has been addressed for single ground-based robot systems and is traditionally implemented using a Kalman Filter approach [4].

2.1 Underwater SLAM

Unlike ground-based mobile robots, SLAM for underwater vehicles has only recently been investigated. The first instance of running SLAM on underwater robots appears in [5], where point features, or landmarks in from the natural environment, were extracted through sonar. Sonar is also used in [6] to verify a constant time SLAM algorithm. The SLAM implementation in [7], uses a sensor fusion (sonar and vision) before feature extraction to make the algorithm more robust.

Another approach is to drop transponders at unknown locations, and use these transponders as landmarks in the SLAM algorithm. In [8], ranges to transponders were used to estimate the vehicle and transponder locations.

2.2 Multi-robot SLAM

This research concerns SLAM for multi-robot systems, where robots can cooperatively map the environment and localize themselves.

For ground-based rovers, several approaches have been taken to this problem. Some approaches assume known starting positions [9]. This assumption was not required for the approach taken in [1], which was also shown to be scalable, be real-time capable. In related work [10], the issue of low-bandwidth communication is taken into consideration by only exchanging those landmarks that result in the highest information gain.

Other approaches include [11], where the Set Membership SLAM, or SM SLAM, is extended to multi robot case. In this case, measurement noise and motion error are not assumed to be Gaussian distributions, but are instead viewed as unknown but bounded errors. An example of performing multi-robot SLAM using vision is found in [12].

3 SLAM

The technique presented in [1] enables the merging of multiple world maps that consist of landmark state estimates and associate covariance. The technique is an extension of the Sparse Extended Information Filter (SEIF) work presented in [2], which was designed for a single robot implementation and then extended to multiple robots in cite [1]. In both [1] and [2], the SLAM algorithms were implemented for a 2D environment, (i.e. using a truck in a city park). Here the system has been extended to 3D, an obvious requirement for operating in the underwater environment.

Much like the typical Kalman Filter, this approach uses a Motion Update to predict the new location of the vehicle and a Measurement update to correct this predicted estimate at every time step. Unlike the Kalman Filter approach, a Sparsifiction step is used to reduce the algorithm run time. Also, additional map merging step is taken if vehicles have the ability to communicate. In summary, each individual vehicle iterates on Algorithm 1 shown below.

Algorithm 1 Multi-Robot SLAM Algorithm for each individual vehicle.

```
1.  Loop on $t$
2.      Motion Update
3.      Measurement Update
4.      Sparsification
5.      If communication with other vehicle exists
6.          Merge Maps with other vehicle
7.  end Loop
```

Within Algorithm 1 landmarks are not described with position mean μ and variance σ, but with a combination of their inverses. That is, at some time step t, landmarks are defined by the state information matrix H_t and information vector b_t.

$$H_t = \Sigma_t^{-1} \tag{1}$$

$$b_t = \mu_t^T H_t \tag{2}$$

If m and n are the number of robots and features respectively, each with 6 degrees of freedom, then state vector $b \in \Re^{6m+6n}$ and corresponding information matrix $H_t \in \Re^{(6m+6n)x(6m+6n)}$.

3.1 Motion Updates

Robot motion updates the robots current position. This step differs from standard SLAM, which treats the environment as static (i.e. only the robots position changes). Here, links between features are established. Re-observation strengthens links between these features, while noise reduces the strength of the link between the robots pose and feature positions.

The estimated robot motion $\hat{\Delta}_t$ is combined with the previous estimate of the information vector b_{t-1} to calculate the predicted state vector $\bar{b}_t$. As shown in Equation 4, the predicted state vector is also a function of the previous information matrix H_{t-1}, the motion error covariance matrix U_t, and the Jacobian of the pose transition function A_t. Similarly, the information matrix $\bar{H}_t$ is also updated.

$$\bar{H}_t = f(H_{t-1}, U_t, A_t) \tag{3}$$

$$\bar{b}_t = f(b_{t-1}, \hat{\Delta}_t, H_{t-1}, U_t, S, A_t) \tag{4}$$

3.2 Measurement Updates

The measurement update uses current measurements z_t with variance Z to correct the predicted state estimates as follows:

$$H_t = \bar{H}_t + C_t Z^{-1} C_t^T \tag{5}$$

$$b_t = \bar{b}_t + (z_t - \hat{z}_t + C_t^T \mu_{t-1})^T Z^{-1} C_t^T \tag{6}$$

In equations 5 and 6, $\hat{z}_t$ represents the measurement that is expected given the current state estimate. The measurement Jacobian C_t is defined by:

$$C_t = [\ \frac{\partial h}{\partial x_t}\ 0...0\ \frac{\partial h}{\partial y_t}\ 0...0\ ] \tag{7}$$

In equation 7, h is the measurement function, x_t is the robot pose variable, and y_t is the feature position variable. It is noted that C_t is sparse, which means that updates are only conducted on the fields which are affected by the current robot pose and the currently observed features. This allows for a scalable algorithm that can be run in real-time.

3.3 Sparsification

As more features are observed, the existing links between all previously observed features would remain continuously active. In order to preserve the constant time nature of this algorithm, sparsity constraints are made on the information matrix. For this implementation, features are deactivated as they leave the robots field of view. That is, their links to other features or robots are removed.

3.4 Merging Maps

In the Kalman filter, the inverses of variances σ are additive. In this implementation, information matrices are directly additive. In the Kalman filter, means are additive, but they are weighted by a Kalman gain, which is essentially the ratio of their variances. In this case, b_t is already scaled by variances, which makes information vectors directly additive. With H_t and b_t directly additive, maps from multiple robots are easily merged. As long as feature correspondence can be achieved, measurements of these feature positions can be added directly.

3.5 Landmark Extraction

SLAM algorithms require stationary landmarks within the environment to compute relative position estimates. Landmark position estimates are obtained by fusing the relative sensor measurements with inertial sensor measurements and control input information. In the system presented, the relative sensor measurements will be obtained via a vision system. A downwards facing camera selects interesting features as landmarks and outputs their position relative to the vehicle. Matching of each feature to a list of previously observed features in the database is conducted to allow correlated updates.

4 Results

4.1 Simulations

To verify the approach taken in [1] to underwater multi-robot SLAM, a Matlab Simulation was conducted. This simulation allowed for an environment containing a variable number of observable marine features and a variable number of identical robots to explore this region. Each robot was run independently, through a series of waypoints.

4.1.1 Underwater Vehicle

The ANGUS002 Remotely Operated Vehicle (ROV) was used for this simulation due to the availability of a Matlab Dynamic Model. The ROV is equipped with 6 thrusters. The model takes into consideration:

- Vehicle thrusters - including the non-linear forces generated by propellers when they are driven in reverse.
- Buoyancy and center of gravity
- Hydrodynamic drag
- Ocean current

4.1.2 Vehicle Controller

Within each simulation, the Autonomous Underwater Vehicle (AUV) was commanded to navigate through a series of way-points. A multi-modal controller, that switches between a forward travel mode and a station-keeping mode, was implemented to track the way points. It is important to note that the controller was not operating using the output of the SLAM system. In order to be fully autonomous, a robot would be required to generate its trajectory based on its map. Though the method of operation here is different, we are only concerned with the errors between the estimated and true trajectories, in order to evaluate the operation of the selected SLAM algorithm.

4.1.3 Vision System

To simulate landmark recognition, positions of targets on the sea floor were provided to the simulator as a list. As the robot travels through its environment, a model of a downward facing camera system was used to detect these targets. The camera model takes into account:

- range to target
- orientation of the camera
- field of view of the camera

The measurement function h was simply a coordinate transformation which translated the landmark positions in the world-frame into positions in the vehicle frame. When merging models it was assumed that landmarks were easily identified and distinguished from one another, thus eliminating the correspondence problem for the sake of simplicity. The simulation used a noise model with a constant standard deviation $\sigma_m^2 = 0.1$ for measurement error.

4.1.4 Positioning Sensors

Each vehicle was equipped with a Global Positioning System (GPS) unit and an Inertial Measurement Unit (IMU). Vehicles were able to take a GPS reading prior to submerging, so their initial positions were available. Though this is not required for this algorithm, it simplified the map merging step.

Once below the surface, the readings of an IMU were simulated and used as the sole method to estimate vehicle motion. The IMU provided linear accelerations and orientation angles. The orientation angles are relative to the gravity vector and magnetic North, so these readings we not subject to drift. The accelerations, however, were subject to an additive White Gaussian noise with a standard deviation of 7mg or $\sigma_a^2 = 4.5806 \times 10^{-5}$. A depth pressure sensor enables the elimination of drift in the Z-axis, but both the x- and y-axes of the vehicle are subject to long term drift.

An algorithm which is functional without a dynamics model can be beneficial in applications where a model is unavailable. Dynamics models can be difficult to obtain accurately, and external disturbances are often unmodeled. For this simulation, the vehicle dynamics were not used for the motion prediction stage of the SLAM algorithm. In future experiments, using this prediction with the acceleration measurements will most likely improve performance.

4.1.5 Communication

For the purposes of this simulation, it is assumed that vehicles are only able to communicate with one another when in close proximity. When vehicles are in range of one another, they are able to swap information regarding the position and description of landmarks in the environment. The availability of communication prompts a map merging in the simulation system for all robots in range. The simulations conducted during this research used a communication radius of $r_c = 2m$.

4.2 Simulation Results

In order to evaluate the performance of the selected algorithm, several simulation scenarios were constructed. It was desired to look at the performance:

- without SLAM
- with SLAM
  - without map merging
  - with map merging
    - at the end of the mission
    - once, each time within range
    - frequently, when within range

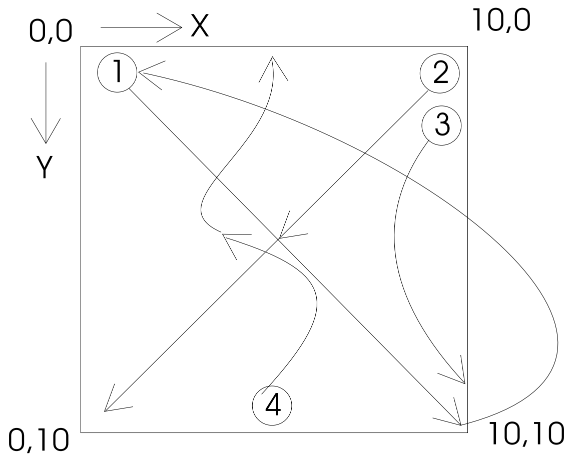


Figure 1: Simulation: Trajectories of the 4 robots tracking waypoints

The metric used to evaluate the positioning performance under these situations is position error with respect to time. In the case of the robot position, the trajectory error is examined, and, for landmark location, the measured location is compared to the actual position.

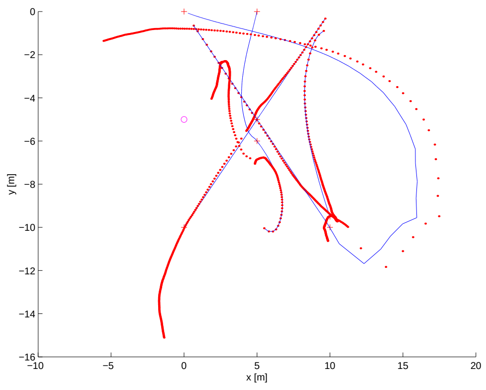
A set of waypoints was created for each of the robots. The approximate desired trajectories for each of the 4 robots used for this simulation are shown in Figure 1. The robots' trajectories are at different depths to avoid collision.

4.2.1 Without SLAM

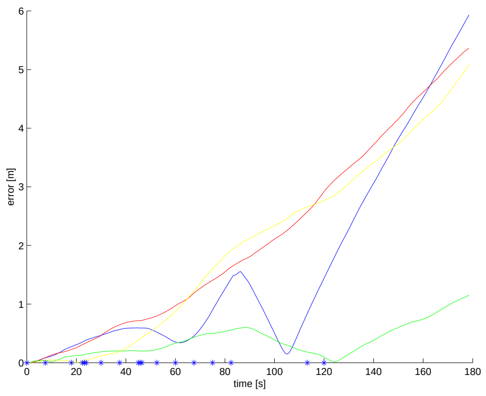
Without a SLAM algorithm running, the vehicles have only the readings of their IMUs to go on. Due to the double integration of IMU readings, and the noise present with the acceleration readings, vehicle positions can drift quite severely and position errors only grows with time. Figure 2(a) shows the actual position as a solid line and the estimated position as dots for the 4 robots. The error for each of the position estimates is shown in Figure 2(b).

4.2.2 SLAM Without Merging

Next, several landmarks are placed on the bottom of the marine environment. Figure 3 depicts the paths



(a) Actual and Measured Trajectories



(b) Position Error wrt Time

Figure 2: The robot trajectory and position error with time for a all robots using only IMU measurements for position determination. The actual trajectory is shown as a solid line in the first plot, with measurements as '.'. The second plot shows an error measurement for each robot with respect to time.

followed by robots and the landmarks below from a 3D viewpoint. As can be seen in Figure 4, the positioning performance is greatly increased. Robot position error does not grow unbounded with time, and the return to a known location eliminates errors which have accumulated.

When landmarks are not available, for example during times of low visibility in a marine environment or in regions where a descriptive scene is not present, position uncertainty will grow. Figure 5 shows the

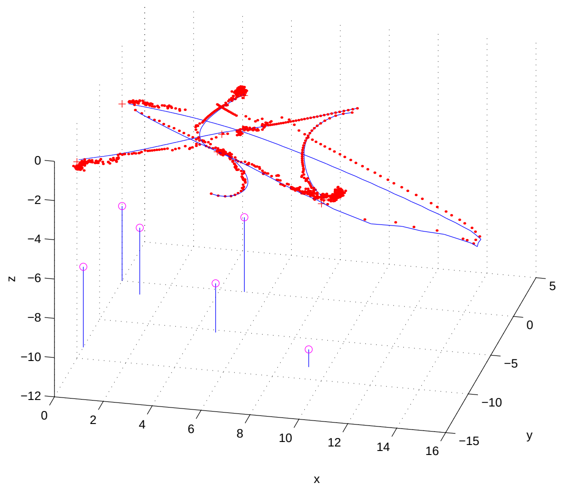


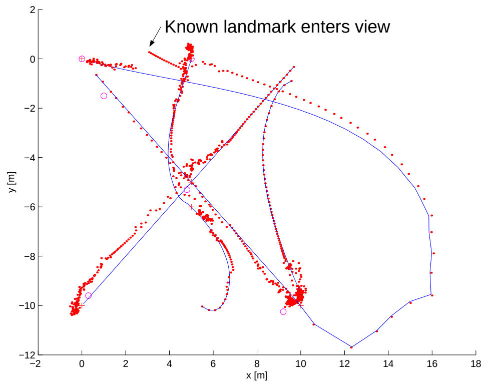
Figure 3: Simulation: 3D view of 4 robot trajectories above simulated landmarks (shown as "o").

confidence ellipses for robot 1. The uncertainty and error grow as the robot looses visibility of all landmarks, and decreases once a known landmark is located.

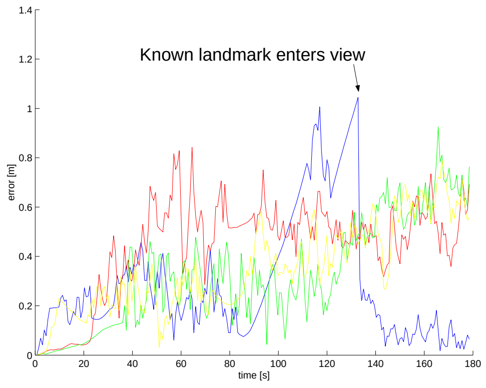
4.2.3 Merging After Completion

Thus far, only the robot position has been examined. Turning to the environment features, it can be seen that combining the measurements of multiple robots is better than each robot on its own. An example of a set of measurements from 3 robots running independently is shown in Figure 6. Here, each robot determines its best guess as to the location of the landmark. Using the certainty of each robot's estimate, the measurements are fused at the completion of the run, reducing the overall error.

In the example shown in Figure 6, the error is reduced to approximately 15cm from the actual position. Although robot 2 (shown with '+') was able to take many measurements of the position of the landmark, with no landmarks along the approaching segment of its path, it had a large uncertainty in its position estimate. This is taken into consideration during the merge and this estimate is not weighted strongly during the final merge.



(a) Actual and Measured Trajectories



(b) Position Error wrt Time

Figure 4: The robot trajectory and position error with time for all robots using running SLAM, using IMU readings and the measurements to landmarks in the environment.

4.2.4 Merging At First Visibility

In this simulation, robots are equipped with an underwater communication system, so they can exchange map information while exploring, as well as at the end of the mission. These exchanges occur when robots come within a given range of one another. Figure 7 gives an overview of this situation.

Several beneficial characteristics resulting from communication can be observed. First, consider the communication between robot 1 and robot 4. It can be seen that measurements of the landmark at $(5.0, 0.8)$ are exchanged at $t = 113.25s$, leading to a

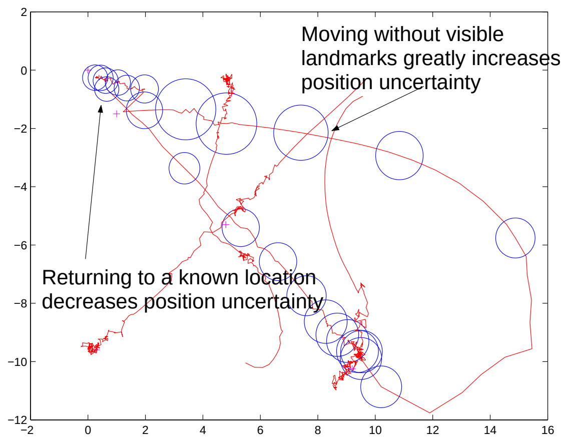


Figure 5: Simulation: The estimated trajectory of all 4 robots, and the confidences ellipses for robot 1.

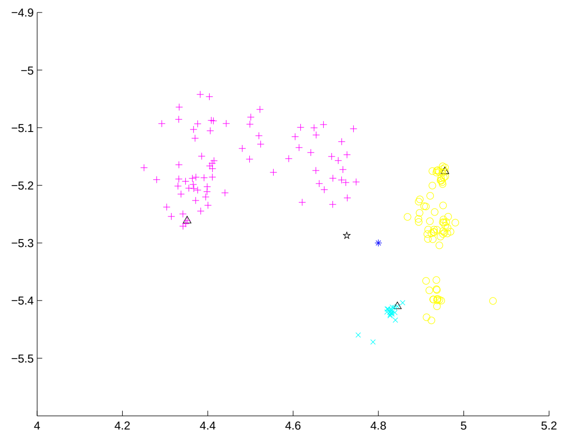
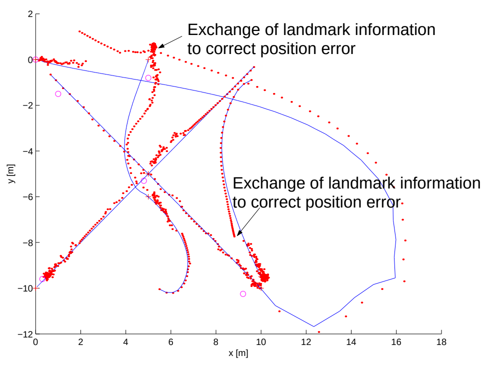
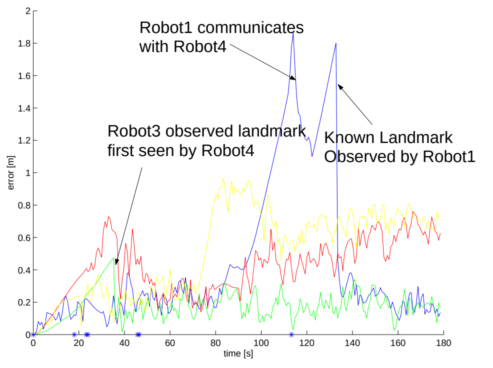


Figure 6: Simulation: The results of combining maps at the completion of the simulation. The actual feature location is shown as an '*', while measurements from robot 1 are shown as 'x', robot 2 as '+' and robot 4 as 'o'. Each robot's final estimate is shown as a triangle and their combined estimate as a star.



(a) Actual and Measured Trajectories



(b) Position Error wrt Time

Figure 7: The robot trajectory and position error with time for all robots using running SLAM, with robots merging each time a robot first comes within range. '*'s along the x-axis represent a merge between 2 robots

correction of robot 1's position. This does not reduce the position error to zero, since the estimate of this landmark location by robot 4 contains some error.

Secondly, a benefit of indirect communication can be seen. As robot 2 begins to travel, it observes the location of a landmark at $(9.2, 10.25)$. At $t = 18.0s$, it exchanges maps with robot 1. At $t = 24s$, robot 1 and robot 3 exchange their maps, which now informs robot 3 of the position of the landmark at $(9.2, 10.25)$. As this landmark comes into view, robot 3 is able to correct its position to reflect the observation initially made by robot 2, even though it has never directly communicated with robot 2.

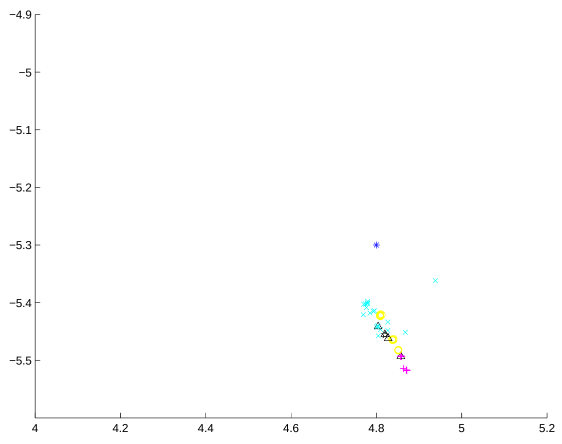


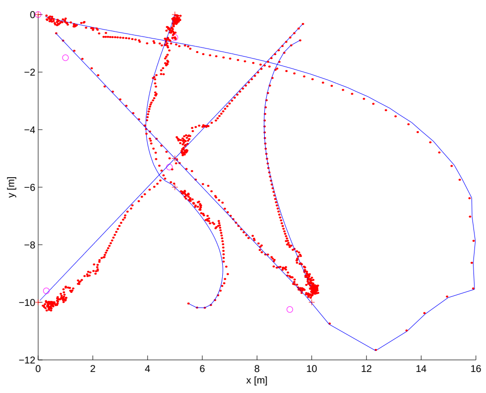
Figure 8: Simulation: The results of merging maps each time a new robot comes within communication range. The actual feature location is shown as an '*', while measurements from robot 1 are shown as 'x', robot 2 as '+' and robot 4 as 'o'. Each robot's final estimate is shown as a triangle and their combined estimate as a star.

Figure 8 shows the result of intermediate map merging on landmark measurements. The measurements are more clustered now, since the independence between robot measurements is reduced as they exchange data. This means that errors which one robot accumulates will influence the others. At the same time, this exchange of data will help to reduce drift in the robots' position estimates.

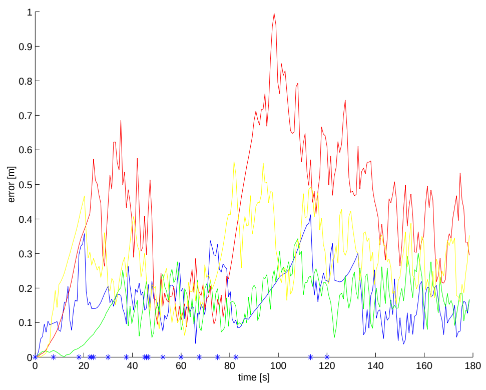
4.2.5 Merging Several Times While Visible

A simulation was conducted to determine the effect of continuously merging data with all other robots in range. A delay of 7.5 seconds was used to simulate data transfer time. The results are shown in Figure 9.

Examining the landmark at $(4.8, 5.3)$, it can be seen in Figure 10, as was discussed in Section 4.2.4, that the communication of the robot tends to cluster measurements, eliminating the independence that was seen in Figure 6. The increased vehicle position accuracy, however, will lead to more accurate landmark estimates.



(a) Actual and Measured Trajectories



(b) Position Error wrt Time

Figure 9: The robot trajectory and position error for all robots running SLAM, with robots merging several times with all robots in range. ’*’s along the x-axis represent a merge between 2 robots

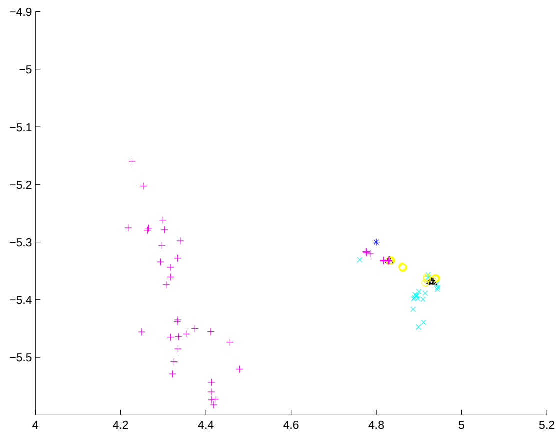


Figure 10: Simulation: The results of merging maps several times with all available robot within communication range. The actual feature location is shown as an ’*’, while measurements from robot 1 are shown as ’x’, robot 2 as ’+’ and robot 4 as ’o’. Each robot’s final estimate is shown as a triangle and their combined estimate as a star. The final estimate is difficult to see, but is located at $(4.93, 5.37)$.

4.2.6 Summary

In order to compare the results from each of the above algorithm modifications, Table 1 has been prepared. This table shows a drastic reduction in error when SLAM is used. Merging map data produces a further improvement. Continuous merging yields and improvement in 2 robots, and reduction in 2 others. Further study should be conducted to determine the optimal merging rate.

Table 1: RMS errors in robot position for each mapping algorithm

	RMS Error [m]			
Algorithm	Robot 1	Robot 2	Robot 3	Robot 4
IMU-only	2.3507	2.6830	1.6003	2.5230
SLAM	0.5832	0.6950	0.4644	0.3299
w/ 1 merge	0.3772	0.2208	0.2461	0.2282
w/ >1 merge	0.2995	0.4036	0.1784	0.3654

5 Conclusions

The proposed SLAM technique is ideally suited for multiple underwater vehicles. The technique requires infrequent data exchange between robots and does not require a central processor or map server, making it ideal for underwater applications, where communication is both limited in range and unreliable. Although, it does benefit from more frequent data exchange. The algorithm is constant-time, allowing for real-time implementations on ROVs or AUVs.

Furthermore, although untested by this experiment, the ability of this algorithm to fuse maps by correlating features without requiring knowledge of the vehicles starting locations allows arbitrary motion to be conducted, without absolute position measurements. The ease with which maps are fused, requiring only a simple coordinate transformation and matrix addition, significantly reduces computational complexity.

6 Future Work

Currently, the algorithm is being implemented on a VideoRay Micro ROV, (see Figure 11) for real world testing. The vehicle has a color camera that can tilt from horizontal to vertical (downward facing). Orienting the camera in a downward facing configuration provides a means to obtain landmarks on the lake bottom. SIFT features [13] are being investigated as a means to extract landmarks from the vision system. The ROV has been equipped with an O-Navi Falcon-MX IMU to provide inertial measurements.

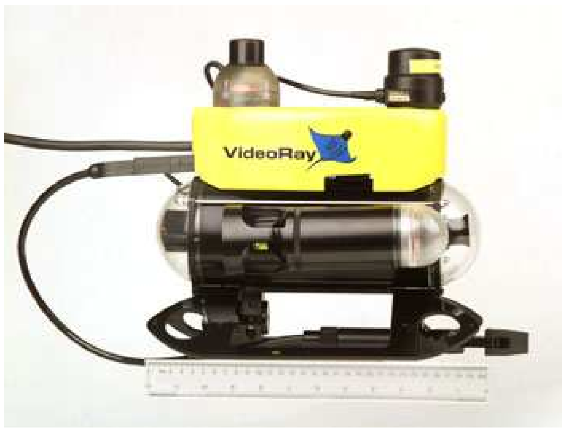


Figure 11: The VideoRay Pro III Micro ROV.

References

[1] S. Thrun and Y. Liu, "Multi-robot SLAM with sparse extended information filers," in *Proceedings of the 11th International Symposium of Robotics Research (ISRR'03)*. Sienna, Italy: Springer, 2003.

[2] Y. Liu and S. Thrun, "Results for outdoor-slam using sparse extended information filters," in *ICRA 2003*, pp. 1227–1233.

[3] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Ng, "Simultaneous mapping and localization with sparse extended information filters: Theory and initial results," in *Proc. of the Int. Workshop on Algorithmic Foundations of Robotics*, 2002.

[4] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," in *International Journal of Robotics Research*, vol. 5, no. 4, 1986.

[5] S. B. Williams, P. Newman, M. W. M. G. Dissanayake, and H. F. Durrant-Whyte, "Autonomous underwater simultaneous localisation and map building," in *Proceedings of IEEE International Conference on Robotics and Automation*, San Francisco CA, USA, 2000, pp. 22–28.

[6] P. M. Newman, J. J. Leonard, and R. J. Rikoski, "Towards towards constant-time slam on an autonomous underwater vehicle using synthetic aperture sonar," in *Proceedings of the Eleventh International Symposium on Robotics Research*, Sienna, Italy, 2003.

[7] S. Majumder, J. Rosenblatt, S. Scheding, and H. F. Durrant-Whyte, "Map building and localisation for underwater navigation," in *Experi-*

mental Robotics VII. London: Springer Verlag, 2001, pp. 511–523.

[8] P. M. Newman and J. J. Leonard, "Pure range-only subsea slam," in *Proceedings of IEEE International Conference on Robotics and Automation*, Tawain, 2003.

[9] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun, "Collaborative multi-robot exploration," in *IEEE International Conference on Robotics and Automation*, 2000.

[10] N. Nettleton, S. Thrun, and H. Durrant-Whyte, "Decentralised slam with low-bandwidth communication for teams of vehicles," in *Proceedings of the International Conference on Field and Service Robotics*, Lake Yamanaka, Japan, 2003.

[11] M. Di Marco, A. Garulli, A. Giannitrapani, and A. Vicino, "Simultaneous localization and mapping for a team of cooperating robots: A set membership approach," in *IEEE Transactions on Robotics and Automation*, vol. 19, no. 2, 2003, pp. 238–249.

[12] H. Hajjdiab and R. Laganiere, "Vision-based multi-robot simultaneous localization and mapping," in *First Canadian Conference on Computer and Robot Vision, CRV04*, Canada, 2004.

[13] S. Se, D. Lowe, and J. Little, "Vision-based mobile robot localization and mapping using scale-invariant features," in *Proceedings of IEEE International Conference on Robotics and Automation, ICRA 2001*, Seoul, Korea, 2001, pp. 2051–2058.