An economical micro-submarine testbed for validation of 3D cooperative control strategies for underwater robots.

P. Pruitt, G. Dinolov, A. McAuley, W. Ferenc, M. Gonzales, A. L. Bertozzi and R. Levy

***Abstract*— We describe a new aquatic robotic testbed designed to test algorithms for coordination and control of micro remote control submarines. The testbed is implemented with a three-dimensional tracking algorithm utilizing views from two planes. We demonstrate the tracking using two dimensional open loop tests on the surface of the water and three dimensional open loop tests underwater. We compare the results with a mathematical model that describes the motion of the submarines in response to propeller forces.**

I. Introduction

This paper describes the design, construction and testing of a small-scale aquatic testbed for three dimensional swarming algorithms. The goal of the research is to explore novel mathematical problems inspired by mobile sensor applications for cooperative autonomous aquatic vehicles. Applications of interest include static target searching and detection (as in mine countermeasures), detection of mobile agents (boats, marine life, people, submarines), environmental boundary estimation (e.g. isoclines of temperature, salinity, algal bathymetry and coastal measurements), and environmental monitoring. Computational and experimental approaches are developed in tandem.

There is currently extensive research employing multiple vehicle testbeds. Many groups are engaged in this research; we mention a few to put our research in context. Most testbeds are designed for cars, including large-scale systems with costly custom robotics [4], [9], [5], [2], [14] as well as smaller, more economical testbeds, such as Caltech's MVWT [4] and UCLA's Micro-Car Testbed [12]. The smaller testbeds illustrate how scaled testing can be an economical solution for implementing mathematical swarming models on groups of robots. However, these testbeds do not explore swarming in three dimensions. To extend the research into three dimensions, large-scale aquatic systems exist, such as those run by the Matthew Joordens research group at the University of Texas [10], and researchers at Harbin Engineering University in China [15]. Research is also being conducted in the field, sometimes with autonomous vehicles large enough to carry people or payloads [17], [11], [6], [3].

This paper is supported by ARO MURI grant 50363-MA-MUR and NSF grants DMS-0914856, DMS-0907931, DMS-0601395, and EFRI-1024765.

W. Ferenc, A. McAuley and P. Pruitt are with the Department of Engineering, Harvey Mudd College, Claremont, CA 91711, U.S.A. E-mail: {wferenc,amcauley, ppruitt}@hmc.edu.

G. Dinolov and R. Levy are with the Department of Mathematics, Harvey Mudd College, Claremont, CA 91711, U.S.A. E-mail: {gdinolov,rlevy}@hmc.edu.

A.L. Bertozzi is with the Department of Mathematics, University of California, Los Angeles, CA 90059, U.S.A. E-mail: {bertozzi}@math.ucla.edu.

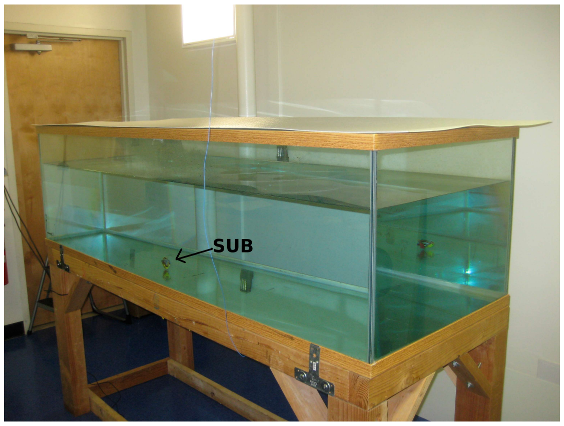

Fig. 1: The aquatic testbed tank and submarine.

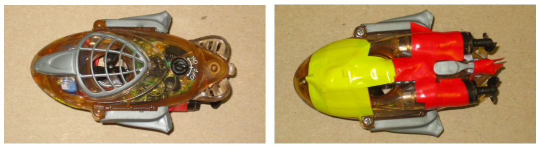

Fig. 2: Micro remote control submarine used in the testbed. Top view on left shows vertical propeller below cage. Bottom view on right shows rear propellers and taping.

Our goal is to design and develop a small-scale aquatic testbed using an aquarium, small remote control submarines (available off-the-shelf and then modified by our lab), and three small cameras. The problem of real-time control and tracking in a 3D aquatic testbed is significantly more difficult than the 2D land-based problems. For example, in 3D there is rotation about three axes as opposed to one. This increases the complexity of the videotracking hardware and software as well as the mathematical model describing the motion.

In Section II we present the hardware choices used in the lab. In Section III we describe our custom tracking algorithm. In Section IV we describe a mathematical model for the motion of the submarines in response to forces from the propellers, and in Section V we present examples of preliminary open loop tests.

II. Testbed Setup and Hardware

A. System Overview

The testbed, constructed for under \$2500, consists of a with 2.081m long x .608m high x .581m deep aquarium and five major hardware components coordinated through the control loop shown in Figure 4. The hardware components include micro submarines, cameras, an arduino micro-

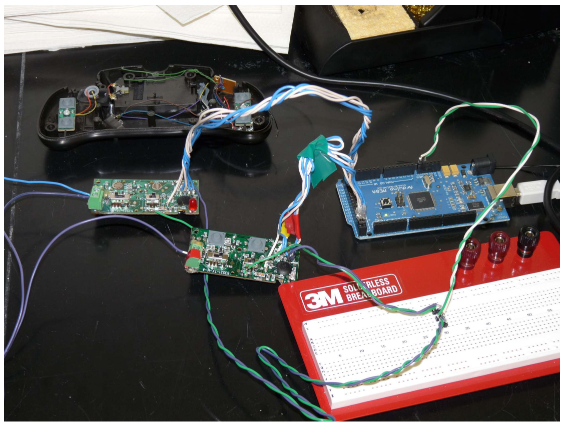

Fig. 3: Hardware linking computer to controller through the Arduino. Controller from original black case at top is removed. Below the case are two controllers connected to Arduino (top right) and breadboard (bottom right) used for testing.

controller, transmitter, and computer. The laboratory space required for the testbed has a small footprint (3.187m x 5.156m), therefore the platform design could be implemented at smaller undergraduate institutions as well as universities with more extensive laboratory facilities.

B. Submarines

The remote control submarines are modified Sub-Sonic XP microsubs (see Figure 2, [7]). The submarines are 26mm wide, 80mm long, and are rated to submerge to three meters. They offer three-dimensional control using two propellers placed at the stern for for horizontal motion (left, right, forward and backward), and one propeller in the center oriented for vertical motion. The submarines receive signals at 27 MHz on one of three channels. Generally each channel corresponds to one submarine, but it is possible to issue the same command to multiple submarines. The bottom of the submarines are marked with colored electrical tape so that the bow is yellow and the stern is red. This facilitates tracking of the orientation of the submarines. The submarines with tape have mass 51.10g and are positively buoyant. These submarines are used for the 2D surface tests. With 3.160g of additional weight, the submarines are negatively buoyant and have a mass of 54.26g. These submarines are used for the 3D underwater tests.

C. Cameras

The testbed employs three Logitech Webcam Pro 9000 cameras. The cameras were chosen for low cost, a high frame rate (30 fps), compatibility with the Linux operating system, and high resolution [13]. Two cameras placed below the tank record its entire length in the x-y plane. This is necessary because the distance from the floor of the lab to the bottom of the tank is not large enough to use only one camera. One camera placed to the side of the tank records the entire x-z plane. Combining the views from the two planes provides 3-D position and orientation information.

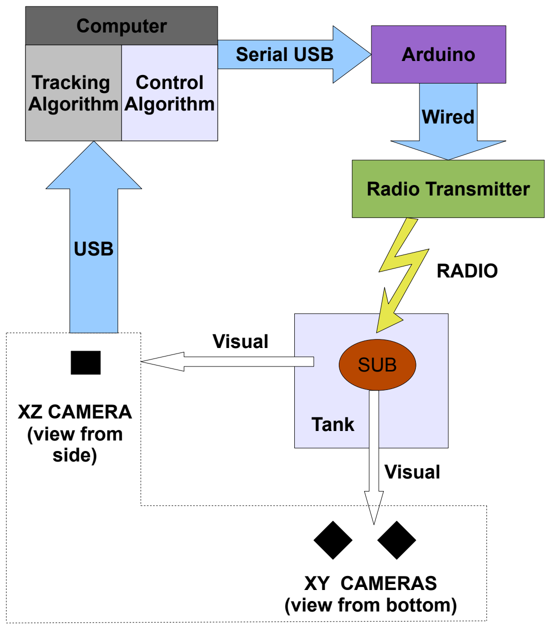

Fig. 4: The control loop for the testbed.

III. Tracking

Several tracking methods for submarine position have been used for swarming testbeds, such as onboard gyroscopes [4] and barcode-reading [12]. Since our micro submarines are too small to implement a gyroscope payload, we developed a custom tracking algorithm.

Tracking of the submarines is automated via computer. Motion commands are sent to the submarine from a computer through a serial connection to an Arduino micro-controller and then to the transmitter (a modified version of the original hand-held controller). The submarine's position is recorded by the cameras which transmit video back to the computer. The computer, running Ubuntu, then executes a custom tracking algorithm using OpenCV 2.0 [16] and C++ to locate the position and orientation of each of the submarines.

The tracking method relies on known initial locations. In such a system, Sub A in frame 1 is said to be the same submarine as Sub B in frame 2 if the difference between the location of Sub A and Sub B is small. The drawback of this method is that the algorithm might fail when multiple submarines are very close to one another. The advantage is that the submarines do not have to be individually marked for identification. Identification markings, such as barcodes, are feasible for 2D testbeds with video through air, but are not ideal in a 3D aquatic setting where distortion can arise from the water itself and from the orientation of the submarine.

After the tracking algorithm receives video input from

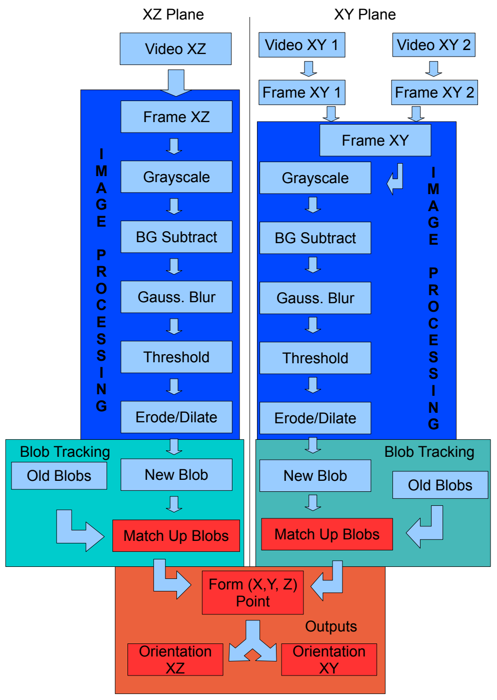

Fig. 5: A flowchart of the tracking algorithm used to detect the submarines.

the cameras it outputs 3D position and orientation data for the submarines. A flowchart representation of the tracking algorithm is shown in Figure 5. Note that most of the processing is performed on the x-y and x-z planes separately. In the x-z plane only one camera is needed to view the entire tank, whereas the x-y plane has two camera feeds that must be merged together. Frames from both x-y are combined with a transformation that corrects for keystoning that results from the camera angles. Information from the x-y and x-z planes combine to provide three dimensional information.

A. *Initializing Tracking Points*

A tracking point consists of a location in 3D and a submarine identification number. The points must be initialized so that the starting positions of the submarines are known. Tracking points are initialized in two different ways. The first initialization occurs during the first iteration of the tracking loop, when any blobs returned by a blob-finding algorithm from both the x-y and x-z planes are designated as tracking points. The second method of initializing tracking points allows the user to add points after the first iteration. Each iteration corresponds to one frame from the camera (30 fps). The user clicks on a location in either the x-y or x-z plane display where a submarine is apparent and the point is designated as an additional tracking point. The program identifies the corresponding object in the other plane by taking the x-coordinate of the clicked-on point and searching all nearby x-coordinates in the other plane for suitable blobs.

B. *Image Processing*

The first step of the tracking algorithm is to extract a baseline background image of the tank from each plane of the video feed. The frames are converted to grayscale in order to facilitate quicker processing. Then after the tracking points are initialized, frames with and without submarines are background subtracted to isolate the submarines. The background subtraction can result in some noise due to slight lighting variations and ripples in the water, so the algorithm incorporates Gaussian blurring, thresholding and erode/dilate to reduce the noise. Remaining noise consisting of small groups of pixels in contrast to the larger solid blocks representing the submarine locations can be eliminated by manually deselecting the noise-induced tracking points.

C. *Matching Blobs to Tracking Points*

Within the tracking algorithm, when a new frame is taken by the camera, it is necessary to update the submarine identification number. An OpenCV blob-finding algorithm can isolate the submarines based on their larger size and return the location of the blob. The algorithm compares the new blob locations to the locations of blobs in the previous frame.

Determining which blobs correspond to the new locations of the tracking points is more complicated than simply assigning the closest blob to be the updated location of a submarine. This naive approach results in some blobs being missed and others being assigned multiple identification numbers. An improved algorithm is outlined in the steps below and illustrated in Figure 6:

a) For each blob (red circles) in the new frame, identify the closest tracking point location from the previous frame (blue crosses).

b) If the distance between the new blob and old tracking point is greater than a prescribed radius, it is disregarded since the submarine could not have moved that far between frames.

c) Update the tracking points with the position of the nearest remaining candidate blob (green stars) within the prescribed radius. If two tracking points claim the same blob (highly unlikely), then a random tracking point is assigned for that frame. Random tracking points are corrected with information from a subsequent frame.

Using two camera views, each submarine only provides one record of its (y, z) location, taken from the (x, y) and (x, z) tracking points. However, both planes provide information about the x location. The (x, y) and (x, z) measurements must be combined into a single value in order to form a 3D location estimate (x, y, z). A single x value is assigned from a weighted average of the two x estimates. An x value is given more weight if there are no other tracking points near it. The rationale for this weighting scheme is that if there are other tracking points nearby, the likelihood of the blob tracker accidentally returning the wrong point

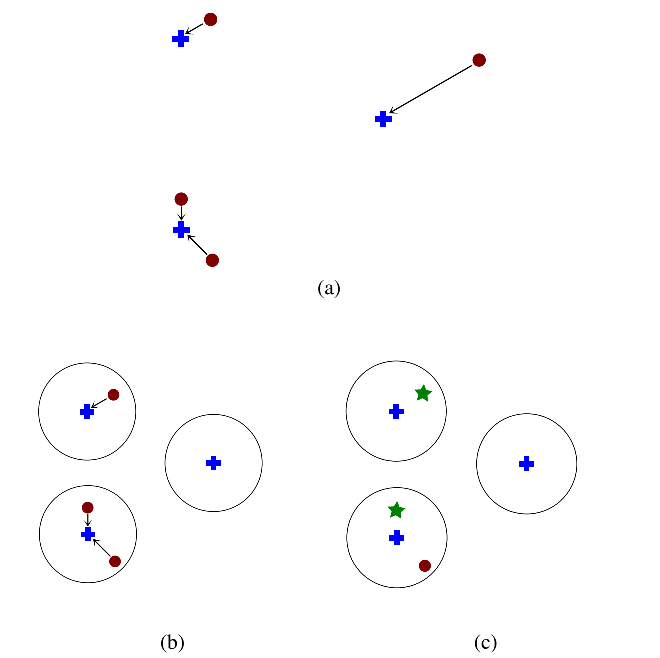

Fig. 6: The three blob tracking steps. (a) Each blob (red circle) identifies the closest tracking point location from the previous frame (blue crosses). (b) If the distance between the blob and tracking point is greater than a prescribed radius, it is disregarded since it could not have moved that far between frames. (c) The tracking points update with the position of the nearest remaining candidate blob within the prescribed radius (green star). If there is no viable update in the current iteration, the tracking point will update in a subsequent iteration.

is increased. If there are no other points nearby, the point returned by the blob tracker can be assumed to be more reliable.

D. Orientation Detection

Once the three dimensional location of each submarine is known, the next step in the tracking algorithm is to determine the orientation of each submarine. The bottom of each robot is marked with yellow on the bow and red on the stern. Using the tracking points, a color frame is extracted for a small region containing the submarine. A custom thresholding operation isolates the red and the yellow portions of the submarine in each small frame. Using the position of the nose and the tail, an angle representing the orientation of the submarine is returned. This angle can be used in a control algorithm to determine which propellers must be activated to achieve a desired motion. An image highlighting the isolated tail is contained in Figure 7.

IV. Mathematical Model

In this section we present a mathematical model and computer simulations for comparison with results from the aquatic testbed. The model is based on the equations of motion resulting from all of the forces acting on the submarine. It is closely related to a 2D model of motion by Hsieh, et

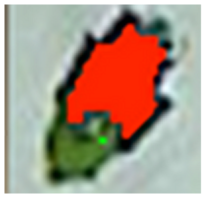

Fig. 7: An image from the videotracking program of the submarine viewed from below with the tail of the submarine identified in the upper right portion of the image.

al. [8]. We assume that the linear and rotational drag are proportional to the linear and angular velocity, respectively. We neglect lift resulting from the shape of the vehicle and external hydrodynamic forces. The model is composed of two differential equations:

$$m\frac{d\vec{v}}{dt} = \vec{F}_L + \vec{F}_R + \vec{F}_V + \vec{\beta} - D\vec{v} \quad (1)$$

$$I\frac{d\vec{\omega}}{dt} = \vec{r}_L \times \vec{F}_L + \vec{r}_R \times \vec{F}_R + \vec{r}_V \times \vec{F}_V - R\vec{\omega} \quad (2)$$

where m is the mass of the vehicle, t is time, $\vec{v}$ and $\vec{\omega}$ are the linear and angular velocities of the vehicle. $\vec{F}_L$, $\vec{F}_R$, and $\vec{F}_V$ are the forces from the left, right, and vertical engines of the vehicle. Similarly, $\vec{r}_L$, $\vec{r}_R$, and $\vec{r}_V$ are the vectors from the center of mass of the vehicle to the respective engines. Since the linear drag $D\vec{v}$ and rotational drag $R\vec{\omega}$ are modeled as proportional to $\vec{v}$ and $\vec{\omega}$, both D and R are diagonal matrices. $\vec{\beta}$ is the buoyant force and tensor I is the inertial tensor of the vehicle. Equations (1) and (2) are both expressed in the Lagrangian reference frame of the vehicle. As we are interested in laboratory measurements, equations in our model must be transformed to the Eulerian reference frame, using the standard time derivative conversion operator.

$$\left(\frac{d}{dt}\right)_{lab} = \left(\frac{d}{dt}\right)_{vehicle} + \vec{\omega}\times \quad (3)$$

V. Open Loop Tests

Open loop tests provide data for the submarines' response to propeller pulses. 2D tests were conducted on the surface of the water, and 3D tests were conducted under the surface. In the tests we refer to the propellers as left and right according to the submarine top view. To conduct each test, we waited for at least 15 seconds for disturbances in the water from previous tests to dissipate. With the tracking program, we obtained the background image. Next a submarine was slowly placed in the starting position at one end of the tank. As soon as the person placing the submarine was out of view of the cameras, the tracking was initiated, and the submarine was given a 0.33 sec pulse. Ten seconds of data were recorded, and then the tracking was stopped, prompting the creation of the output data file.

To conduct the 2D surface tests, a positively buoyant submarine was placed on the surface of the water. Two typical forward tests from simultaneous pulses to the two

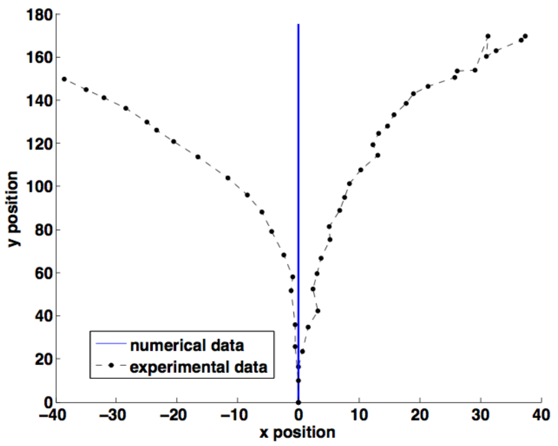

Fig. 8: 2D open loop test with positively buoyant submarine. Videotracking captures a single pulse from both rear propellers. The time between points is 0.03 seconds. The black dashed line represents the trajectory of the submarine with dots at the data points. The blue solid line indicates the trajectory of the mathematical simulation. The drifting of the submarine to the right or left of the model path indicates that frequent pulses should be used to provide the tightest control of the submarine trajectory. Units are in pixels.

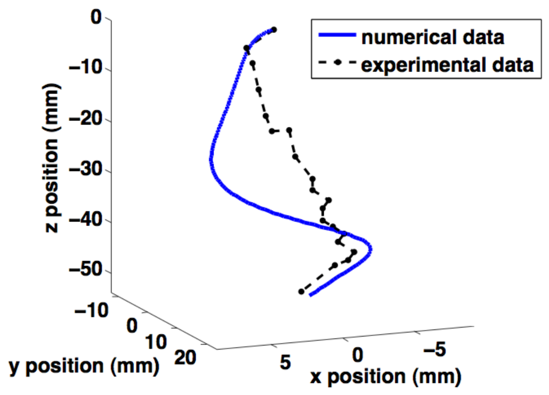

Fig. 9: 3D open loop test with negatively buoyant submarine. Videotracking captures a single pulse from the left rear propeller. The time between points is 0.03 seconds. The black dashed line represents the trajectory of the submarine with dots at the data points. The blue solid line indicates the trajectory of the mathematical simulation.

rear propellers are shown in Figure 8. For comparison, we show the result of the model predicting forward motion in a straight line. Note that the submarine drifts to the right or left after some time, indicating that in a control loop, commands will need to be issued at frequent intervals to achieve tight control of the submarine motion.

To conduct the 3D tests, a negatively buoyant submarine was placed below the surface, but near the top. As the submarine descended, a pulse to the left rear propeller created a clockwise spiral (observed from above). A typical result is contained in Figure 9 with a comparison to the results of the mathematical model in Equations (1) and (2), solved using a Forward Euler numerical scheme. The initial conditions for the model are provided by estimates of the initial position and velocities from the data.

The parameters used for the simulations in Figures 8 and 9 are described below. The initial position of the sub is shifted to the origin, which is provided as initial data to the simulation along with initial velocities from the experimental data: linear velocity $\vec{v} = [0, 106, -70]$ and angular velocity $\vec{w} = [0, 0, -60]$. We input the mass of the submarine $m = 54.26g$. For the 3D simulations the initial Euler angles used in conversion to the Lagrangian frame (based on the experimental data) are the yaw angle $\phi = 0.1525$, pitch angle $\theta = 0$ and roll angle $\psi = 0$.

For the preliminary simulations presented in this paper, we use the vectors $\vec{r}_L = [-1, -1, 0]$ and $\vec{r}_R = [1, -1, 0]$ to indicate that the left and right propellers are separated from the center of mass. The vertical propeller is assumed to be at the center of mass, such that $\vec{r}_V = [0, 0, 0]$. For the 3D simulation we pulse only the left propeller, which causes a right turn: $\vec{F}_L = [0, 5700, 0]$, $\vec{F}_R = [0, 0, 0]$. For the 2D simulation, the two propellers have the same force. In both simulations, the vertical propeller is turned off: $\vec{F}_V = [0, 0, 0]$. The pulse length is $0.33s$. The computational time step is 0.001. The buoyancy is not used in the 2D test, and is taken to be $\vec{\beta} = [0, 0, 0]$. In the 3D test it is taken to be strong and negative: $\vec{\beta} = [0, 0, -2200]$.

The inertial matrix I is related to how the mass is distributed in the vehicle. For the 3D simulation, the results are most sensitive to the lower right component in the matrix because it corresponds to rotation in the x-y plane. For these simulations we use

$$I = \begin{pmatrix} 0.7 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.7 \end{pmatrix}$$

.

The matrices D and R correspond to linear and angular drag and are assumed to be diagonal for the reasons discussed above. They are used as fitting parameters and will be refined in future work. For the simulations of this paper we use the following values:

$$D = \begin{pmatrix} -1900 & 0 & 0 \\ 0 & -1900 & 0 \\ 0 & 0 & -55 \end{pmatrix}$$

,

$$R = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -200 \end{pmatrix}$$

.

Future work will involve refining the parameters to provide better comparison between experimental results and simulations. We will also introducing control to the tracking sequence for closed loop tests. We plan to implement a 3D version of a model using attractive and repulsive potentials to simulate 2D swarming [1]. Preliminary tests indicate that this approach is promising.

VI. Conclusion

We have demonstrated the viability of an economical aquatic testbed using open source software and commercially available small submarines. Open loop tests indicate that to provide desirable control it may be necessary to navigate using combinations of relatively short pulses. Reasonable comparison in 3D between the experimental and mathematical results demonstrates the feasibility of such a model for control of the submarine. Refinement of parameters in the model will be the next focus for the modeling aspect of the project. In the future, the aquatic testbed will be used to implement mathematical swarming algorithms for coordination and control of multiple submarines.

VII. Acknowledgements

We would also like to thank Prof. Z. Dodds of the Computer Science Department at Harvey Mudd College and the DYNAR research group for helpful discussions. This research was supported by ONR grant N000141010641, NSF DMS 0601395, NSF DMS 0907937, NSF EFRI 1024765.

References

[1] N. Bao, Y.-L. Chuang, D. Tung, C. Hsieh, Z. Jin, L. Shi, D. Marthaler, A. Bertozzi, and R. Murray, "Virtual attractive-repulsive potentials for cooperative control of second order dynamic vehicles on the Caltech MVWT," in *American Control Conference*, 2005, pp. 1084–89.

[2] "The Minnow Project," http://www.cs.cmu.edu/˜coral/minnow/, Carnegie Mellon University. [Online]. Available: http://www.cs.cmu.edu/∼coral/minnow/

[3] "Robotic platforms of Michael Benjamin," http://oceanai.mit.edu/mikerb/robots/home.html, Center for Ocean Engineering. [Online]. Available: http://oceanai.mit.edu/mikerb/robots/home.html

[4] T. Chung, L. Cremean, W. Dunbar, Z. Jin, E. Klavins, D. Moore, A. Tiwari, D. van Gogh, , and S. Waydo, "A platform for cooperative and coordinated control of multiple vehicles: The caltech multi-vehicle wireless testbed," *Proc. of the 3rd Conference on Cooperative Control and Optimization*, Dec 2002.

[5] R. D'Andrea, "Robot soccer: A platform for systems engineering," *Computers in Education Journal*, vol. 10, no. 1, p. 5761, 2000.

[6] "Fumin zhang," http://www.isr.umd.edu/news/news_story.php?id=2259, Georgia Tech. [Online]. Available: http://www.isr.umd.edu/news/news_story.php?id=2259

[7] "Silverlit Sub-Sonic XP 3D control RTR RC submarine," http://www.hobbytron.com/SilverlitSub-SonicXP3DControlRCSubmarine.html, Hobbytron. [Online]. Available: http://www.hobbytron.com/SilverlitSub-SonicXP3DControlRCSubmarine.html

[8] C. Hsieh, Y.-L. Chuang, Y. Huang, K. Leung, A. Bertozzi, and E. Frazzoli, "An economical micro-car testbed for validation of cooperative control strategies," in *American Control Conference*, 2006, pp. 1446–51.

[9] Z. Jin, S. Waydo, E. B. Wildanger, M. Lammers, H. Scholze, P. Foley, D. Held, and R. M. Murray, "MVWT-II: The second generation Caltech multi-vehicle wireless testbed," in *Proc. of the 2004 American Control Conference*, 2004.

[10] M. A. Joordens and Matthew, "Design of a low cost underwater robotic research platform," in *SOSE 2008 : IEEE International Conference on System of Systems Engineering*, 2008, pp. 1–6.

[11] N. Leonard, D. Paley, R. Davis, D. Fratantoni, F. Lekien, and F. Zhang, "Coordinated Control of an Underwater Glider Fleet in an Adaptive Ocean Sampling Field Experiment in Monterey Bay."

[12] K. K. Leung, C. H. Hsieh, Y. R. Huang, A. Joshi, V. Voroninski, and A. L. Bertozzi, "A second generation micro-vehicle testbed for cooperative control and sensing strategies," in *Proceedings of the 2007 American Control Conference*, 2007.

[13] "Webcam pro 9000," http://www.logitech.com/en-gb/webcam-communications/webcams/devices/5867, Logitech. [Online]. Available: http://www.logitech.com/en-gb/webcam-communications/webcams/devices/5867

[14] T. W. Mclain and R. W. Beard, "Unmanned air vehicle testbed for cooperative control experiments," in *Proc. of the 2004 American Control Conference*, 2003, pp. pp. 5327–5331.

[15] Z. Ming-jun, Y. Li-ping, W. Yu-jia, D. Qing-zhu, and L. Xiao-bai, "Development and experiment of an underwater vehicle test-bed controlled by rudders and thrusters," in *In Proceedings of the 2009 International Conference on Robotics and Biomimetics*, 2009, pp. 1633–1638.

[16] "Welcome - OpenCV wiki," http://opencv.willowgarage.com/wiki/, Open Computer Vision. [Online]. Available: http://opencv.willowgarage.com/wiki/

[17] L. Whitcomb, "Underwater robotics: Out of the research laboratory and into the field," in *IEEE International Conference on Robotics and Automation*, vol. 1. Citeseer, 2000, pp. 709–716.