

Aqua Swarms: Design and Implementation of Water Surface AUV

Mustafa Ozkan Daglioz and Aladdin Ayesh¹

Abstract. Autonomous marine vehicles are remarkably useful for tasks such as shallow water surveying, environmental data collecting and military operations such as surveillance and weapon delivery. Because of the necessity for autonomous marine vehicles and the challenges they impose, they have gained a widespread interest among the research groups around the world. This paper presents a water surface autonomous unmanned vehicle which moves from a starting point to any desired destination while avoiding obstacles. Unlike many water surface unmanned vehicles this robots relies on oars for its mobility. This gives a greater control and precision in manoeuvrability while keeping the size of the robot as small as possible. This paper presents the hardware and software design and implementation of a dual-oar water surface autonomous vehicle. Comprehensive experimentation was conducted and results of the experiments are presented.

1 INTRODUCTION

The work presented here is to build a water surface autonomous unmanned vehicle (Aquabot) that uses two oars for its mobility. The main reason and a requirement is that the robot has great manoeuvrability and precise control. This Aquabot has been built using Lego Mindstorms Robotic Invention Kit. This robotic kit allows fast prototyping and flexibility in the robot structure. To present an alternative solution to traditional propelling methods used in autonomous marine vehicles, oars have been used to propel the Aquabot. The design presented here focuses on the performance, cost, environment, size and weight. The result is a blue print for a building block that can use to build and develop water swarms that are able to navigate in small spaces with restricted resources, i.e. light, maneuverability dimensions, etc.

The navigation and control system is designed to be a safe, effective and precise navigation system. To operate the Aquabot autonomously, software has been developed using Not Quite C (NQC) language, which is a language for programming a RCX microcontroller which control the movement of the Aquabot according to the signals received from light sensors fixed at the front of the robot (figure 1). The effect of the stimuli received through the light sensor lead to the deployment of one or both of the ora mechanisms.

The paper starts with a background section reviewing some of the Aqua and water surface robots developed and/or being researched. We then present the hardware design of our robot and show how it meets the requirement for small but efficient water surface unmanned vehicle. Software design follows. Details of experiments conducted and their results are then presented to conclude.

2 BACKGROUND

Autonomous surface vehicles (ASV) are often used in marine tasks where the use of large maned ships is either impractical or dangerous such as the case in shallow water surveying, environmental data collecting, coordinating with autonomous underwater vehicles and military operations such as surveillance and weapon delivery. The variety of applications and the interesting technological challenges that building aqua robots imposes, led to an increased interest among research groups in developing marine robots often by multidisciplinary groups of marine researchers, mechatronics, and mobile robots.

ARTEMIS [1] was the first ASV developed by MIT. The vehicle was too small for open sea applications and it had some performance problems. In order to solve performance problems and increase the size, a catamaran hull was selected for ACES [1], which was the second ASV developed by MIT. Some changes were made to the hull, propulsion systems and power. After these modifications the ASV was renamed Autocad [2]. In 2004, four ASV named SCOUT [3] were developed by MIT. The vehicles consist of polyethylene kayaks equipped with lead acid batteries, communication systems and single board computer.

The MESSIN project [4], which was sponsored by the German Federal Ministry of Education Research and Technology, has been carried out from 1998 to 2000 to develop and test the ASV Measuring Dolphin for high accuracy positioning and measuring devices carrying in shallow water. In the period 1997-2000, the Instituto Superior Technico of Lisbon developed Delfim [5] [6] to develop an ASV for establishing a fast direct acoustic communication link between the AUV and a support vehicle. Lisbon IST is also developing Caravela [6] in addition to Delfim. Cavela is a long range autonomous research vessel, for testing advanced concepts in vehicle/mission control and radar based obstacle avoidance and demonstrating technologies for the marine science community [7].

Charlie, an autonomous catamaran, developed in 2002-2004 by CNR-ISSIA. Charlie initially designed for supporting sensors and samplers for the study of the sea-air interface in Antarctica, where it was exploited in 2004 [8]. In 2005, original steering system was upgraded and the vehicle is currently used for testing of mission control, navigation and guidance algorithms and for evaluating the possibility of using this technology for civil harbour protection in the presence of marine traffic [7]. University of Plymouth, UK, is developing an ASV named Springer for tracing pollutants. The unmanned Springer will be around 3m long and 1.5m high [9].

In all of these projects, issues of hardware and software design were identified often leading to a revision in the design, e.g. [1]. The dynamics of water surface, shape of the robot, and the emerging responses of the interaction of the different forces make hardware design an important issue. The hardware design impact on the software,

¹ De Montfort University, email: aayesh@dmu.ac.uk

which has to respond to the sensors in use. The software and hardware of the Aquabot proposed here have been designed considering the problems highlighted by the reviewed projects and in response to requirements of this project.

The main hardware problems that ASVs have can roughly be gathered under three titles; buoyancy, size and weight. Projects mentioned here have different hull shapes and propulsion methods in order to solve these three problems. We studied the drawbacks of previous designs and reflected proposed solution in our design. Many ASVs reviewed here have a very complicated software to control and operate the hardware. Due to the limited capacity of the software platform of the Aquabot, complicated navigation methods of the projects were simplified using swarm technology and thus a simple but efficient reactive controller was implemented.

3 HARDWARE DESIGN

3.1 Design Considerations

Hardware design is probably the most important part of designing an Aquabot. Without proper hardware design, the Aquabot cannot float on the water or perform necessary actions to reach the destination point. The Aquabot has been designed to meet these 5 requirements; cost, performance, environment, size and weight. The design explained here gives the Aquabot precise and efficient propulsion ability. Since the Aquabot has been designed considering overall weight and size, it does not need much power to propel itself. This is also important because RCX microcontroller, which is the power source of Aquabot, uses 6 AA size batteries as a power source.

Another important consideration is buoyancy. With this design, Aquabot has an equal distribution of weight, which provides well balanced buoyancy. At the end, a small, lightweight and yet efficient Aquabot has been designed.

The working of the Aquabot can be summarised as follows; when Aquabot is running, to detect the location of the destination point, it waits for the signal from the destination point. Aquabot has been adjusted to measure the distance between the source of the signal and itself. The best way of sending signals from the destination point is to use a laser pointer. Therefore Aquabot measures the intensity of the laser that has been sent from the destination point and using that information Aquabot calculates the distance between the destination point and itself. After detecting the location of the destination point, Aquabot starts moving towards it. Software has been designed to keep the Aquabot away from obstacles. When Aquabot hits an obstacle, it moves around the obstacle and keeps going to the destination point.

3.2 Mechanical Structure

The Mechanical structure of Aquabot consists of 5 parts. These are; the hull, oars, oar mechanism, obstacle sensing mechanism and microcontroller. These 5 parts have been designed individually, then fixed together to form the Aquabot. Every one of these 5 parts has its own design considerations. These considerations and detailed mechanical design of each part will be explained individually in this section. Design of the hull will be explained first, then the other parts will be explained.

The hull is the body of the Aquabot, and selecting an appropriate hull form is important in terms of stability, capacity of payload transport and floatability. A catamaran design has been selected as a hull form in this piece of work. This is because waterplane area between two hulls reduces rolling motions and increases displacement. Since

catamarans consists of two hulls, failure of one hull does not end up in a complete loss of buoyancy. To build a catamaran hull, two model ship hulls have been used. Two hulls fixed to each other with 2 cm. space between them. The hull can be seen in the figure 1.

3.3 Oar Mechanism

The oar is the part of the Aquabot that steers, accelerates and stops the boat. Four factors have been taken into consideration while designing oars. These are weight, size, strength and stiffness. To meet these factors, plastic material has been used to build oars. In terms of making more efficient oars, the blades of oars have been curved. With this design, 94 percent of the oar surface touches water. Therefore the oars transfer enough power from motors to the water to propel Aquabot. The Oars can be seen in figure 1.

The oar mechanism is the mechanism that moves the oar in vertical and horizontal directions. In other words it is the main mechanism that drives the Aquabot. It provides power and transfers the power to the oars. Aquabot has one oar mechanism for each oar. Mainly, the oar mechanism consists of two motors, two gears, two bars, one oarlock and one touch sensor. To move the oars horizontally, a slide mechanism has been used. The body of oar mechanism has been attached to a ring, and the ring has been attached to two bars. Both ends of the bars have been fixed to the hull. Therefore the body of oar mechanism can slide in a horizontal direction. One motor and one gear have been used to slide the body of the mechanism. To move oars vertically, the gear mechanism has been used. A Long black beam has been used to connect the oar to the gear. The idea behind the gear mechanism is this; every half cycle of the gear, the black beam will move up or down, therefore the oar will move up or down. To limit the movement in vertical direction, 1 Degree Of Freedom (DOF) joint has been used. To stop the motor at the end of every half cycle, a touch sensor has been used. To propel Aquabot, vertical and horizontal motions of the oar mechanism are performed in order. This mechanism can be seen in figure 1.

Each oar mechanism has been placed on each hull. To propel Aquabot straight forward, two oars must move simultaneously. To achieve this aim, two oar mechanisms have been connected to each other using two beams. The obstacle sensing mechanism has been placed in front of the Aquabot. The RCX microcontroller has been fixed to the middle of the hull in terms of balance. One light sensor has been fixed on top of the RCX microcontroller. This light sensor has been used for navigation. Connection cables between the microcontroller and the motors or sensors have been fixed to the hull. The oars have been fixed to the hull using the oarlock, which lets the oar rotate around. Dimensions of the Aquabot; Width = 21 cm.(without oars), Length = 38 cm, Depth = 11 cm. (4,5 cm of 11 cm. is under water.)

3.4 Sensing and Control

The obstacle sensing mechanism is a specially designed mechanism that is used to sense the obstacles in front of the Aquabot. The obstacle sensing mechanism consists of 3 main parts. These are; touch sensor, spring and bar. Two bars have been connected to the front bumper. The bars have been fixed to the hull with help of two springs. The touch sensor has been placed near the other end of the bars. Every time Aquabot hits an obstacle, the front bumper pushes the two bars, the two bars push the touch sensor and with the help of springs the two bars and the front bumper go back to its former position. With help of this mechanism, every time Aquabot hits an obstacle,

the touch sensor sends a signal to the microcontroller. This mechanism can be seen in figure 1.

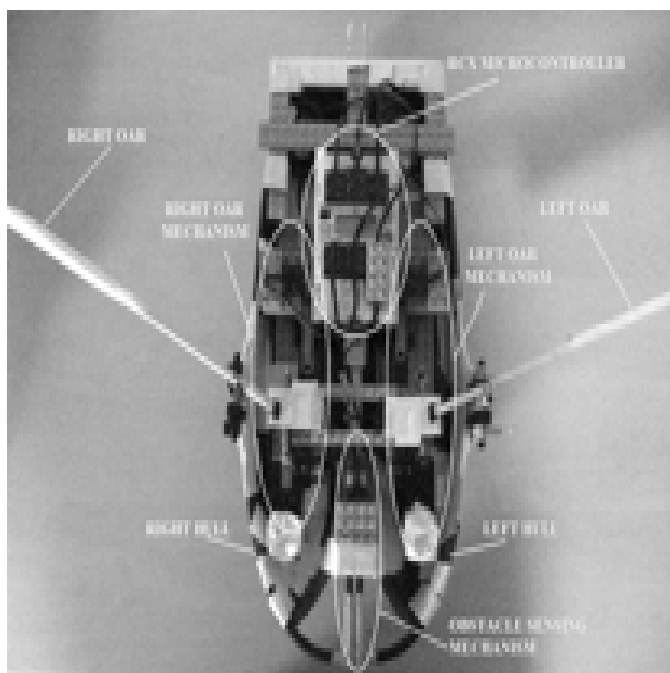


Figure 1. Aquabot: dual-oars water surface swarm robot

RCX microcontroller is the essential part of the Aquabot. RCX microcontroller contains three sensor inputs, three actuator outputs, an IR transceiver, Serial I/O (input/output), ADC (analog digital converter), 8 bit CPU at 16 MHz and Hitachi H8 microcontroller with 32 kilobytes of RAM (4 kilobytes of which is used for interrupt vectors and other low level data.) User programs are stored in 16kb of internal ROM and 32kb of static RAM. An ADC (Analog to Digital Converter) lets the RCX read sensor inputs, and special driver chips makes controlling motors and electrical devices possible. As an user interface, microcontroller has four user buttons, LCD display and an internal speaker. Microcontroller is the heaviest part of the Aquabot. Because of this reason, it was essential to find an appropriate place to fix it. After some calculations, microcontroller has been fixed to the 3 cm away from the middle point of the Aquabot. Therefore, microcontroller does not relocate the centre of gravity of the Aquabot and Aquabot does not lose its balance.

4 SOFTWARE DESIGN

Navigation has always been a difficult task in robotics. Autonomous navigation means that a vehicle can move to a desired destination without a user interference. To design a good autonomous navigation system, every bit of information should be collected from all available sources. In this piece of work, these sources are the sensors. At this point, there is a limitation on the number and types of sensors that can be used with the RCX microcontroller. There are only three sensor inputs on the RCX microprocessor. Two sensor inputs have been used to control the oar mechanisms. Therefore only one sensor has been used to navigate the Aquabot.

The software has been programmed using Not Quite C (NQC). A NQC program is composed of code blocks and global variables.

There are three distinct types of code blocks: tasks, inline functions, and subroutines. Each type of code block has its own unique features and restrictions, but they all share a common structure. Tasks can be run simultaneously. Because of this advantage, most of the program has been composed of tasks. The main task is started whenever the program is run. Other tasks can be started from inside the main task. Two tasks have been written to control the vertical motion of each oar. Since the two oars have been connected to each other, one task has been written to control the horizontal motion of the two oars. Propulsion of the Aquabot is done by running these tasks continuously. Another task has been used to detect obstacles. This task is run simultaneously with the main task when the program starts. As it was mentioned earlier, the main task waits for the signal from the destination point to detect the location of said destination point. After this signal, Aquabot starts moving. If Aquabot does not receive a signal for 60 seconds, it turns around to wait for a signal from another direction.

Navigation system approaches can roughly be divided into two groups, absolute positioning and relative positioning [10]. Absolute positioning is when the robot uses landmarks to determine its position. Since landmarks can not be placed on water, the only solution for the Aquabot is to use relative positioning. Relative positioning does not require landmarks. In relative positioning, Aquabot computes its position from the starting point. This is done by counting strokes that Aquabot takes. To count strokes, the program assigns a variable which equals zero at the beginning. Every time Aquabot takes a stroke, the program adds one to the variable, if Aquabot goes back because of an obstacle, the program subtracts one. With one stroke, Aquabot moves 45 cm away. Using this information, the program converts the distance to number of strokes. Therefore the program knows how many strokes that Aquabot has to take in order to reach the point of destination. When the stroke counter variable equals the number of strokes that Aquabot has to take, Aquabot knows that it is at the destination point and stops.

To avoid obstacles, the program does the following; four variables have been used for four directions. These directions are; North, South, East and West. The program assumes that Aquabot always starts to move northward. When Aquabot starts moving, the program assigns a number of strokes that it has to take in order to reach the point of destination to the north variable. Therefore the program knows that Aquabot has to go to the north to reach the destination point. Aquabot starts going north until it reaches the destination or hits an obstacle. When Aquabot hits an obstacle, the program assigns two more strokes to every variable of every direction. Therefore, Aquabot knows that it has to go two strokes south, then two strokes east, then two strokes north and then two strokes to the west to avoid obstacles. With the help of this algorithm, Aquabot moves around the obstacle. After avoiding the obstacle, Aquabot keeps going to the point of destination. To monitor which direction Aquabot is going in, another variable has been used. Every time Aquabot turns right the program adds one to the variable, or subtracts one if it turns left. Therefore the program always monitors to which direction Aquabot moves. The program repeats the same set of sequences every time Aquabot hits an obstacle. Therefore, Aquabot can avoid very complicated obstacles.

The flowchart of the navigation controller can be found in figure 2. To keep flowchart simple, each task is represented by a square. Therefore simultaneous run of the tasks can be seen clearly.

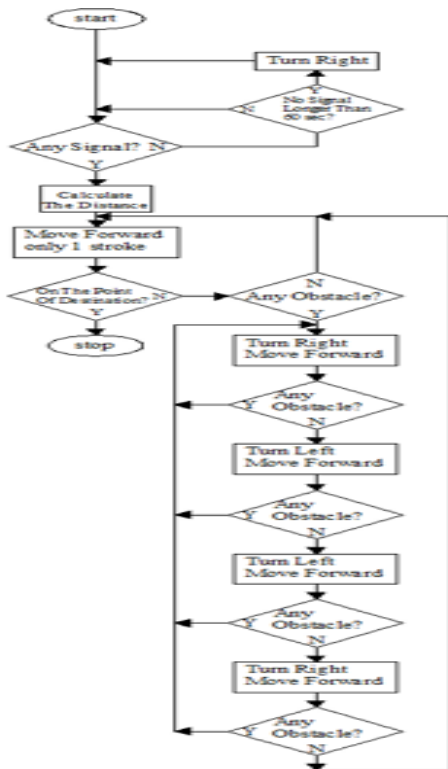


Figure 2. Flowchart of the navigation controller

5 EXPERIMENTS

5.1 Experiment Settings

The first group of experiments has been done to test the obstacle avoidance system. To test this system, four different obstacles have been used with different start and destination points. Schematic representations of these experiments can be found in figure 3. In the first experiment a small simple obstacle has been used. As it can be seen in figure 3, Aquabot followed the route that has been indicated with black line. When Aquabot hit the obstacle, it went back then moved around the obstacle and reached the destination point. In terms of avoiding obstacle, the first experiment was successful. In the second experiment, more complicated obstacle have been used. Because of the size of the obstacle, Aquabot hit the obstacle three times, but it managed to manoeuvre around the obstacle and reach its destination point. A bigger obstacle has been used in the third experiment. As can be seen on the figure 3, Aquabot hit the obstacle four times. But in the end, the experiment was successful; Aquabot reached its destination point. The fourth experiment has been done to find the answer of the question of what happens if the destination point has been surrounded by obstacles. As can be seen in figure 3, this experiment was also successful, the program managed to lead Aquabot to its destination even though the destination point has been surrounded.

The second group of experiments have been done to determine the range of the signal receiving system. Experiments show that the program successfully receives signals in the range of 50 cm to 7 m. As has been mentioned in previous sections, the light sensor has been used to receive signals. The light sensor measures the intensity of light and according to intensity, it sends a value to the RCX microcontroller. Even if there is no signal from destination point, the light

sensor measures the intensity of daylight. Because of this, the RCX microcontroller cannot receive signals from a distance greater than 1,5 m in very bright daylight. In average daylight it can receive signals up to 3 m. And it works best in dark environments, it can receive signals from 7 m.

5.2 Experiment Results

Aquabot has shown good performance in most of the experiments. Aquabot has successfully received signals from the destination point, accurately measured the distance between the destination point and itself, propelled itself with oars, avoided obstacles and successfully reached the destination point. These experiments show that the aims mentioned previous sections were accomplished.

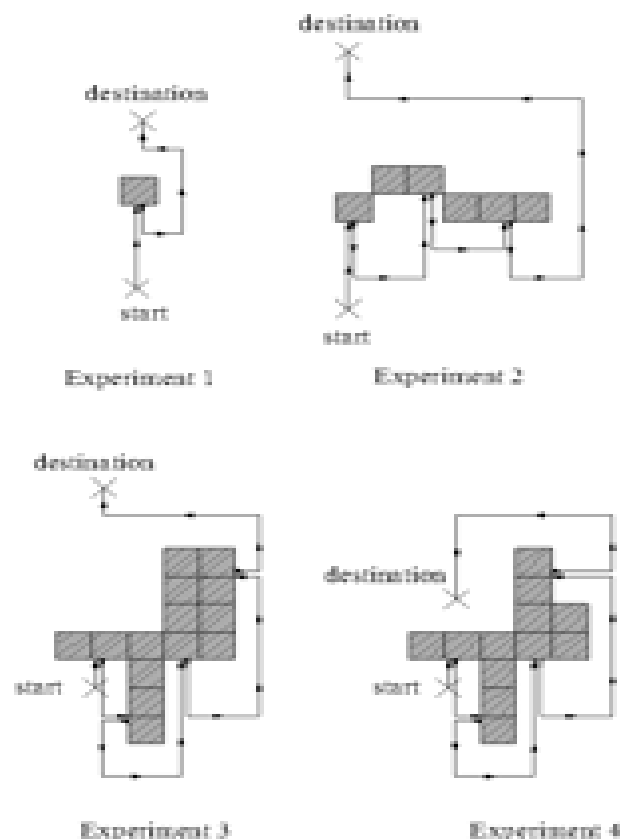


Figure 3. Experiment Settings

The summary of the results of the four experiments that can be seen in figure 3, can be seen in table 1. These results show that the length of the route followed by Aquabot changes depending on the area of the obstacle. In addition to the area of the obstacle and the length of the course, the number of hits occurred during experiments and the distance between start and destination point are also mentioned in the said table.

Two pictures taken during experiments can be seen in figure 4 and 5. In figure 4, Aquabot is approaching the obstacle. Figure 5 was taken seconds after the first picture while Aquabot avoiding obstacle.

The full length video of one of the experiments is uploaded on

Table 1. Summary of the Experiments

No. of Exp.	◇	□	△	★
1	0.5 m	0.92 m	81 cm^2	1
2	0.82 m	3.03 m	486 cm^2	3
3	0.64 m	2.74 m	1134 cm^2	4
4	0.28 m	3.10 m	972 cm^2	4

◇ Distance □ Course Length △ Obstacle Area ★ Hits Occurred

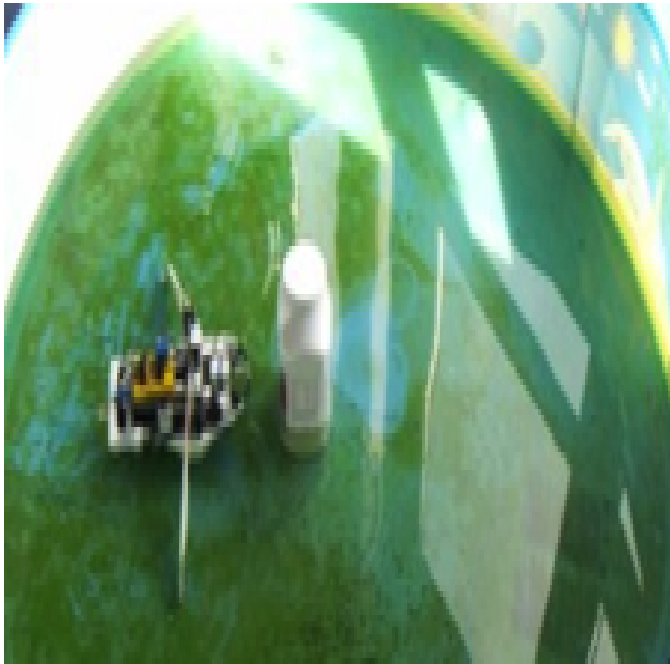


Figure 4. Still Picture from Experiment



Figure 5. Another Picture from Experiment

6 CONCLUSION

The above experiments have proven the reliability and successful working of the Aquabot. The first lesson that can be learned from the work described above is that with Lego Mindstorm Robotic Invention Kit, efficiently working robots can be built. It was easy to build hardware part of the Aquabot with Lego parts, but on the other hand this easiness brought some limitations with it. For example, because of the location of the obstacle sensing mechanism, the front part of the hull of Aquabot hits the obstacle. Otherwise Aquabot can not detect obstacles. In experiments it was observed that when one of the oars of the Aquabot hit the obstacle, Aquabot loses its balance. This limitation can be eliminated using another design for the obstacle avoidance mechanism.

Another limitation of the Aquabot happened because of the Lego light sensor. The range of the Aquabot depends on the daylight. Since the components that we can use are limited with Lego parts, it is very hard to eliminate this limitation. A more sensitive light sensor could be used in the Aquabot, but there is not another light sensor in the Lego Mindstorm kit.

Another limitation happened because of the lack of the relative positioning. In relative positioning, Aquabot computes its position from the starting point. The program assumes that if Aquabot takes one stroke, it goes 45 cm away from the start point. In experiments it was seen that depending on the flow rate and direction of the flow,

² The link of the video is; <http://www.youtube.com/watch?v=1PJGxYXFMsM>

this length was changed. Experiments also show that if something disturbs Aquabot and makes it move without control of the program, Aquabot can not find the destination point. This problem could be solved using a PID controller. To implement this controller, more than three sensors must be used, which is quite impossible because the RCX microcontroller only has three sensor inputs and all of them were used to control the oars or to receive signals.

All limitations mentioned in this paper occurred because of the available hardware. But even with these limitations, experiments showed that the aims of the work were accomplished.

With help of IR transceiver and sensors, Aquabot can easily communicate with other robots around. This communication could help the Aquabot to gather any necessary information required to successfully accomplish missions. This advantage of the Aquabot is very important in terms of extended swarm robotics employing direct communication mechanisms as part of their swarm algorithm. The water surface AUV implemented here has been designed to operate as an autonomous entity individually or in collective producing a complete random swarming behavior. However, the same technical specifications provided in this paper, with adjustment to software implementation, can enable this AUV to work with other robots in a goal-directed fashion.

7 REFERENCES

- [1] Manley, J. (1997) Development of the autonomous surface craft ACES. Proc. of Oceans97, vol. 2, pp. 827 - 832.
- [2] Manley, J., Marsh, A., Cornforth, W., Wiseman, C. (2000) Evolution of the autonomous surface craft AutoCat. in Proc. of Oceans00, vol.1, pp. 403 - 408.
- [3] Curcio, J., Leonard, J., Patrikalakis, A. SCOUT - A low cost autonomous surface platform for research in cooperative autonomy.
- [4] Majohr, J., Buch, T. (2006) Advances in unmanned marine vehicles. IEE Control Series, ch. Modelling, simulation and control of an autonomous surface marine vehicle for surveying applications Measuring Dolphin MESSIN, pp. 329 - 352.
- [5] Pascoal, A., Silvestre, C., Oliveira, P. (2006) Advances in unmanned marine vehicles. IEE Control Series, ch. Vehicle and mission control of single and multiple autonomous marine robots, pp. 353 - 386.
- [6] Pascoal, A. (2000) Robotic ocean vehicles for marine science applications: the european asimov project. in Proc. of Oceans 2000.
- [7] Caccia, M. (2006) Autonomous Surface Craft: Prototypes And Basic Research Issues. Control and Automation. MED '06. 14th Mediterranean Conference. pp. 1 - 6.
- [8] Caccia, M., Bono, R., Bruzzone, G., Spirandelli, E., Veruggio, G., Stortini, A., Capodaglio, G., (2005) Sampling sea surface with SESAMO. IEEE Robotics and Automation Magazine. vol. 12. no. 3. pp. 95 - 105.
- [9] www.plymouth.ac.uk/pages/view.asp?page=9007.
- [10] Horn, M. (2005) Developing safety-security critical systems: A prototype LEGO Mindstorm Detection System. Norwegian University of Science and Technology, Software Engineering Depth Study.
- [11] Breivik, M., Fossen, T.I. (2004) Path Following For Marine Surface Vessels. Mts/Ieee Techno-Ocean '04, 4. pp. 2282 - 2289.
- [12] Encarnacao, P., Pascoal, A. (2001) Combined Trajectory Tracking And Path Following: An Application To The Coordinated Control Of Autonomous Marine Craft. Decision and Control. Proceedings of the 40th IEEE Conference, 1. pp. 964 - 969.
- [13] Naem, W., Xu, T., Sutton, R., Chudley, J. (2006) Design Of An Unmanned Catamaran With Pollutant Tracking And Surveying Capabilities. UKACC Control. Mini Symposia. pp. 99 - 113.
- [14] Martin, F. G. (1995) The Art Of Lego Design. The Robotics Practitioner: The Journal for Robot Builders, 1 (2).
- [15] K.A.Hawick, K., A., James, H., A. (2002) Simulating Swarm Behaviour of Robots. Technical Note.DHPC-118, Submitted to IASTED Conference on Applied Modelling and Simulation.
- [16] Mondada, F., Pettinaro, G., C., Guignard, A. (2004) Swarm-Bot: A New Distributed Robotic Concept. Autonomous Robots. Vol. 17. No. 2-3. pp. 193 - 221.