Testing Controllers on ALE III: A Low Cost mini Autonomous Underwater Vehicle

Savas Piperidis and Nikos C. Tsourveloudis

***Abstract*— This paper presents ALE III, a low cost Autonomous Underwater Vehicle and its potential for experimentation inside a typical research laboratory. ALE III is equipped with sensors and actuators that facilitate real, not simulated testing. The vehicle uses custom made as well as 'off the self' components and minimalistic design solutions suitable for experimentation, as evidenced from various control methodologies.**

I. INTRODUCTION

Autonomous Underwater Vehicles (AUV) emerged from the need to explore and survey the underwater environment but also for the potential to undertake tasks below the water surface. Their capability of underwater operation without being tethered to a surface mother ship or earth control station, is the main comparative advantage against other kind of underwater vehicles. At present, fully functional AUVs are available as commercial products [1], [2], [3]. Though, the majority of the AUV models are the result of ongoing academic research. In both cases the available vehicles are designed to dive in the open sea, navigable lakes and rivers or at Olympic sized swimming pools [4], [5]. The dimensions magnitude of contemporary AUVs may be demonstrated by the size and the power demands of one of the smallest thrusters, available as 'off the shelf' component: $23 \times 8cm$ and $125Watts$ respectively [6]. The product range of today's AUVs is useless for experimentation indoors, at a tank inside the premises of a typical research laboratory's facilities. Only some fish-like biomimetic AUVs [7], [8], [9], [10], [11] offer the opportunity of indoors operation, yet the limitations concerning their embedded computational power and energy storing, restrict their operational capabilities.

This paper describes the main characteristics and the control strategies of ALE III, an AUV compact enough for experimentation inside a small tank of $1m^3$. Apart from the small dimensions, the vehicle's design strategy aims at two additional main targets: low, construction and maintenance, cost and use of open source software and hardware components.

Section II contains an outlook of the vehicle's design solutions, the hardware and software architecture. Section III describes the experimentation area and the development of controllers. Finally the conclusions of the, so far, experimentation along with the plans for future work are depicted at Section IV.

Savas Piperidis and Nikos C. Tsourveloudis are with the Department of Production Engineering and Management, Intelligent Systems and Robotics Laboratory, Technical University of Crete, Chania GR 73132, Greece. savas at dpem.tuc.gr and nikost at dpem.tuc.gr

II. SYSTEM DESCRIPTION

A. Design Solutions

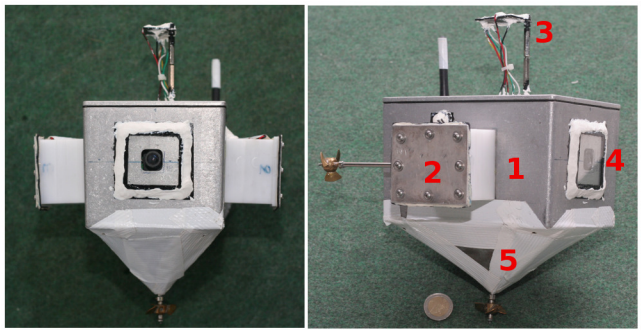

Fig. 1. ALE III autonomous underwater vehicle, (1) water tight hull, (2) custom made thruster motor, (3) inertial measurement unit, (4) vision sensor module, (5) ballast chamber

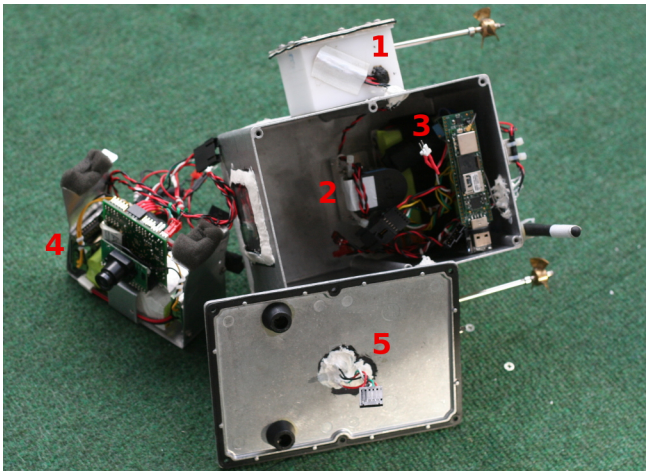

Fig. 2. Parts of Ale III, (1) custom made thruster, (2) heave thruster placed inside the vehicle's hull (3) processing unit, batteries and power supply circuit, (4) vision sensor module, batteries and motor drivers, (5) cover sealing the hull box

Ale III, as shown at the photos of Fig. 1,2, comprises of a water proof, die cast aluminium, box shaped hull. It should be noted that in the current version of the vehicle there was no study for hydrodynamic covers design. Such a work was beyond the scope of this paper. The hull contains the vehicle electronics, a vision sensor module and the vertical thruster motor. Two lateral thruster motors are attached to the starboard and the portside of the hull. The bottom thruster is apt to the heave movement while the two lateral are

apt to the surge movement and the yaw turning. Vehicle's buoyant force was almost double of its weight. Inside a chamber, underneath the hull, the appropriate amount of ballast was added to accomplish a slightly positive buoyancy. Thus, vehicle's centre of mass is located quite lower than its centre of buoyancy, maximizing the righting moment and offering passive roll and pitch stability, while diving underneath the water surface. Several experiments were conducted to assure vehicles underwater stability: the AUV recovers from an upside down position to a stable near-zero roll and pitch orientation in just a few seconds, Fig. 9. The thruster motors are custom made, using components from radio controlled scale models. In this version of ALE III the Inertial Measurement Unit is positioned on the top side of the vehicle, away from the interferences caused by the other electronic devices inside the closed hull box. The vehicle uses a Linux powered Computer On Module (COM), specifically designed for low power, embedded applications [12]. The COM, the IMU and the vision sensor module are open source components, fully customized according to vehicle's operational demands. Software resources used for the controllers development are valid through open source licences and thus free of charge. The overall dimensions of ALE III are $17 \times 21 \times 25cm(L \times W \times H)$ and its total weight, including ballast, is $2.6kg$. The electronics power consumption does not exceed the amount of $2.5W$ while each thruster consumes $2.2W$ when spinning at half of the maximum speed. Using two separate, one for electronics and one for the motors, inexpensive NiMH battery packs at $(3 + 3)Ah$ each, an autonomy of at least two hours is achieved in real experimentation conditions.

TABLE I
ALE III TECHNICAL SPECIFICATIONS

dimensions	$17 \times 21 \times 25cm(L \times W \times H)$
hull	die cast aluminium
thrusters	3, custom made
sensors	vision sensor module inertial measurement unit
processing unit	Ångström Linux computer
electronics power consumption	$2.5W$
thruster power consumption	$2.2W$ at half speed
electronics battery	NiMH at $6Ah$
thrusters battery	NiMH at $6Ah$

B. Hardware and Software Architecture

Electronics hardware was carefully chosen to accomplish low cost, small dimensions, minimal power consumption and easy customization. In this version ALE III is equipped with the following sensors:

- vision sensor module, including frame grabber and embedded microprocessor with reprogrammable firmware, enabling the implementation of custom image processing algorithms [13]. Image grabbing and processing is a task undertaken by the module, leaving COM resources intact for the control process. It serves each COM's request with a data packet containing the image processing results.
- 9 degrees of freedom IMU, incorporating triple-axis gyro, triple-axis accelerometer and triple-axis magnetometer [14]. Sharing the same operation philosophy with the vision module, IMU is an embedded microprocessor system with reprogrammable firmware. It serves each COM's request with a data packet containing the roll, pitch and yaw orientation of the vehicle.

The vehicle's COM, running an Ångström Linux distribution operating system [15], handles the input/output communication with the sensors through serial protocol ports and executes the controller program. It also directs the controller signals to the electronic motor drivers that actuate the thrusters.

The controller software was developed using C++ programming language and object oriented environment. It comprises from two parts:

- the *Robot* class, that uses operating system modules to communicate with the system's input and output devices, i.e. the sensors and the motor drivers respectively. It accesses and manipulates the sensors readings and forwards the control signals to the motor drivers. *Robot* class is modular in the sense that for every new device connected to the system a new class member has to be implemented.
- the *Controller* class, encapsulating vehicle's behaviour. This class receives an input from the *Robot* class containing the sensor readings. It processes these data according to the control algorithm and produces an output that, in turn, is fed back to the *Robot* class. As soon as *Robot* class receives the controller output it produces the proper signal to drive the actuators.

Several behaviours may be developed as different members of the *Controller* class. In fact, this architecture permits several behaviours to be programmed and initialized, so that the system user may choose which one of them to use, depending on the circumstances. All these different behaviour objects of the *Controller* class use the same communication protocol to implement the input/output operations with the *Robot* class. In the following section we emphasize on the experimental modularity of software architecture and in particular of the *Controller* class. Two different controllers, namely, a classic Proportional Derivative (PD) and a fuzzy PD are implemented and tested.

III. CONTROLLER DEVELOPMENT AND TESTING

A. Experimentation Area

AUV's small dimensions offer the potential for indoors experimentation. As shown in Fig.3, a plastic, opaque, cylindrical tank, $1.3 \times 0.8m(Diameter \times Height)$, was established inside the premises of an academic laboratory. The tank has the proper drainage allowing easy control of the water surface level and maintenance. This experimentation area was used extensively, not only for testing and verifying the AUV seaworthiness during the several design and

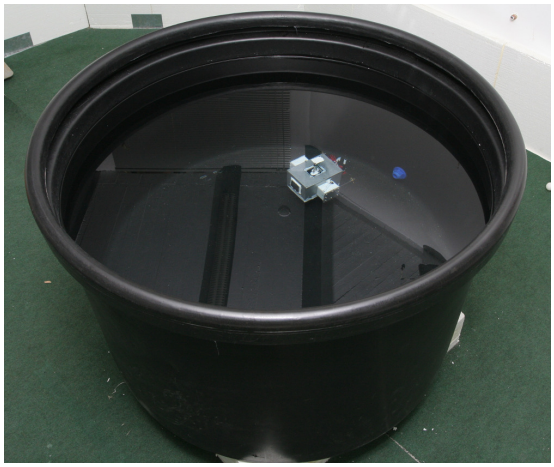

Fig. 3. The experimentation area inside the laboratory

construction levels, but also for the controllers development depicted at next section. More specifically, the following tasks were conducted inside the experimentation area facility, proving its adequacy:

- Thrusters and hull design evaluation, water proof tests, AUV balance and stability verification, energy efficiency measurements and ballast estimation.
- Sensors calibration, sensors firmware development.
- Controllers evaluation and comparison, development of yaw oriented and target-following behaviour.

B. Yaw Controller

The development of a Proportional (P), Derivative (D) controller (PD), achieving and maintaining a given yaw value, was chosen as a first proof of concept for ALE III project. This task uses a *Controller* class member, dedicated to the task corresponding behaviour and a *Robot* class member for the IMU and thrusters utilization. *Robot* class forwards IMU's yaw readings every $0.04sec$ to *Controller* class and after the control process receives the results to drive the thrusters. The proportional control factor is given by:

$$O_p = Kp_{yaw} \times (\phi_e/\phi_{max}) \times P_{max} \quad (1)$$

where O_p and Kp_{yaw} are the output and gain respectively, ϕ_e is the yaw heading error, ϕ_{max} its maximum positive value and P_{max} is thrusters max turning speed. The derivitive factor is:

$$O_d = Kd_{yaw} \times d\phi_e \times P_{max} \quad (2)$$

where $d\phi_e$ is the yaw heading error derivative, O_d and Kd_{yaw} are the D controller output and gain respectively. The controller gain values were estimated using experiments, firstly for the proportional one and afterwards for the derivative gain. Fig. 5 presents the results of an experiment that encapsulates the controller efficiency for different parameter values: the vehicle is underwater, with a yaw heading of approximately $0°$ when, at zero time, it has to rotate to $-115°$ and after that, it has to change its yaw orientation for $90°$ every $12sec$.

C. Target Following

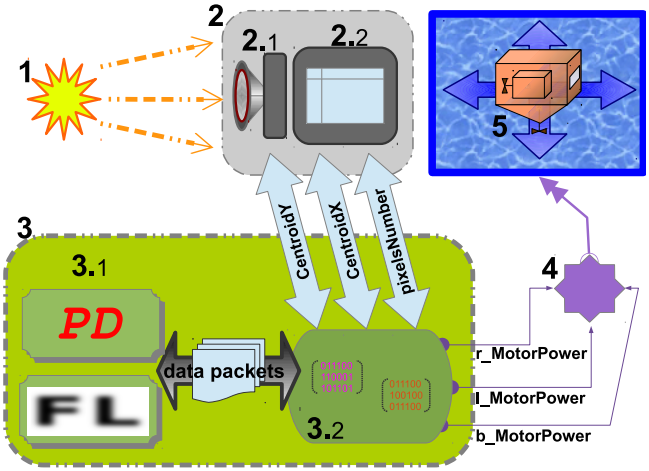

Fig. 4. System's configuration for the target following scenario: (1) Light Target, (2) Vision Sensor Module, (2.1) Image Sensor, (2.2) Frame Grubber and Image Manipulation Program, (3) Controller Software Engine executing at the memory of Computer On Module, (3.1) Proportional Derivative and Fuzzy Logic alternate *Controller* classes, 3.2 *Robot* class, (4) Motors' Drivers Module, (5) AUV's thruster motors

Vehicle's second control test is divided in two parts. In the first part ALE III uses a PD controller similar to the one presented in the previous section. The second part describes robot's behaviour when a fuzzy logic controller is employed. Both controllers are implemented as part of an executable computer program that repeatedly manipulates the motors turning speed, using information collected from the camera sensor. Their task is to locate and follow one white light target inside the tank of the experimentation area, Fig. 4. Both alternate controllers repeat approximately 5 control cycles per second. Every one of these cycles consists of the following parts:

1) The *Robot* class program part accesses the vision sensor module's results *CentroidX*, *CentroidY* and *pixelsNumber*. The module computes the former values using a custom firmware manipulating the white light target pixels scanned by the camera sensor. *CentroidX* and *CentroidY* values depict target's central point (the so called *centroid*) abscissa and ordinate. These values are estimated as the sum of the tracked pixels' abscissas and ordinates, divided by their total number. The centroid point is an indication of the target's position relative to vehicle. *pixelsNumber* is the total number of pixels detected. It is an indication of the distance between vehicle and target.
2) The *Controller* class program part computes the derivatives *dCentroidX* and *dCentroidY* of *CentroidX* and *CentroidY* respectively.
3) The controller software engine, inside *Controller* class, computes the motor control variables *leftMotorPower*, *rightMotorPower* and *bottomMotorPower* and passes

their values to the *Robot* class. These variables correspond to the thrusters rotating speed.

4) The *Robot* class, using COM resources, forwards the output control values to the motors' driver unit. The motor drivers actuate the thrusters and move the vehicle.

Fig. 4 presents the data and signals exchange between the system's modules, i.e. the vision sensor module, the COM and the motor drivers module.

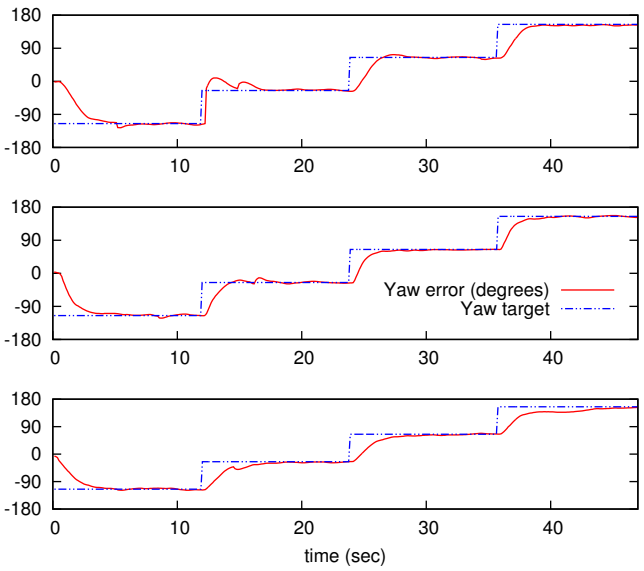

Fig. 5. Yaw PD Controller experiment results for different parameters values. The vehicle is underwater, with a yaw heading of approximately $0°$. At $0sec$ it has to change its yaw heading to $-115°$. Subsequently, it has to change its yaw orientation for $90°$ every $12sec$. The plot at the top shows the results for $Kp_{yaw} = 1.2$, $Kd_{yaw} = 0.005$, $P_{max} = 800$, the plot at the middle for $Kp_{yaw} = 1.2$, $Kd_{yaw} = 0.005$, $P_{max} = 1500$ and the plot at the bottom for $Kp_{yaw} = 0.8$, $Kd_{yaw} = 0.005$, $P_{max} = 1000$

1) Proportional Derivative target following controller: its output is synthesized by the yaw (3a) , heave (3b) and surge (3c) coordinates,

$$O_{yaw} = Kp_{yaw} \times \chi_e \times P_{surgeMax} + Kd_{yaw} \times d\chi_e \times P_{surgeMax} \quad (3a)$$

$$O_{heave} = Kp_{heave} \times \psi_e \times P_{heaveMax} + Kd_{heave} \times d\psi_e \times P_{heaveMax} \quad (3b)$$

$$O_{surge} = Kp_{surge} \times (\rho - \rho_{\oplus}) \times P_{surgeMax} + Kd_{surge} \times d\phi_e \times P_{surgeMax} \quad (3c)$$

where

χ_e: *CentroidX* error, i.e. target's centroid abscissa deviation from the camera sensor frame's x axis centre, $CentroidX - res_h/2$, where res_h is the camera's sensor horizontal resolution.

ψ_e: *CentroidY* error, i.e. target's centroid ordinate deviation from the camera sensor frame's y axis centre, $CentroidY - res_v/2$, where res_v is the camera's sensor vertical resolution.

$d\chi_e$: Derivative of χ_e. It is related to the submarine's yaw rotating speed. If the vehicle is kept stationary, a $d\chi_e$ value of 200 is equivalent to a left heading target course, causing an increase of 25 pixels to the target's centroid abscissa at the camera sensor frame, during one controller time step of approximately 0.13 seconds.

$d\psi_e$: Derivative of ψ_e. It is related to the submarine's heave speed. If the vehicle is kept stationary, a value of 160 is equivalent to an upwards heading target course, causing an increase of 21 pixels (in analogy to the previous paragraph's, higher resolution, x axis movement of 25 pixels) to the target's centroid ordinate at the camera sensor frame.

ρ: *pixelsNumber* variable, used by the controller to monitor vehicle's distance from target. The control process aims at keeping this value as close as possible to a $\rho_{\oplus}$ target value determined by the vehicle user, according to the light target and experimentation area characteristics. When $\rho_{\oplus}$ is 600 pixels then the submarine is approximately $30cm$ away from the target used for the tests.

$P_{surgeMax}$ *and* $P_{heaveMax}$: lateral and bottom thrusters maximum rotating speed.

$Kp_{\text{ yaw, heave and surge}}$, $Kd_{\text{ yaw, heave and surge}}$: proportional and derivative gains respectively. Their values were estimated using recurrent experimentation. At first, the proportional gains were figured out from uncoupled motion commands, for surge movement for example. Afterwards, the corresponding derivative gain was added to the uncoupled motion command and tuned in with respect to the proportional one. Finally the all the gains were fine tuned using motion commands for coupled surge, heave and yaw turning movements.

The controller was tested inside the experimentation area using a scenario where the white light target was departing away from the vehicle, following a sinusoidal - up/down course. The vehicle managed to follow the target without significant errors that could lead to target disappearance from camera's field of view, Fig. 6. The plot at figure top shows target's centroid deviation from the camera sensor frame centre. Next plot shows *pixelsNumber* value along with the straight grey lines that enclose the range of acceptable values. Next three plots show the controller outputs for left, right and bottom thruster accordingly.

2) Fuzzy Logic target following controller: uses the same input and output variables with the PD controller but fuzzifies their values according to the membership functions of Fig. 7, converting their crisp values to the following fuzzy logic linguistic variables.

- Input Variables
  - $CentroidX_{fz}$, fuzzifies *CentroidX* variable with range $[0, 176]$, since the horizontal camera resolution is 176.
  - $CentroidY_{fz}$, fuzzifies *CentroidY* variable with range $[0, 144]$, since the vertical camera resolution is 144.
  - $pixelsNumber_{fz}$, fuzzifies *pixelsNumber*.
  - $dCentroidX_{fz}$, fuzzifies the *CentroidX* derivative.
  - $dCentroidY_{fz}$, fuzzifies the *CentroidY* derivative.

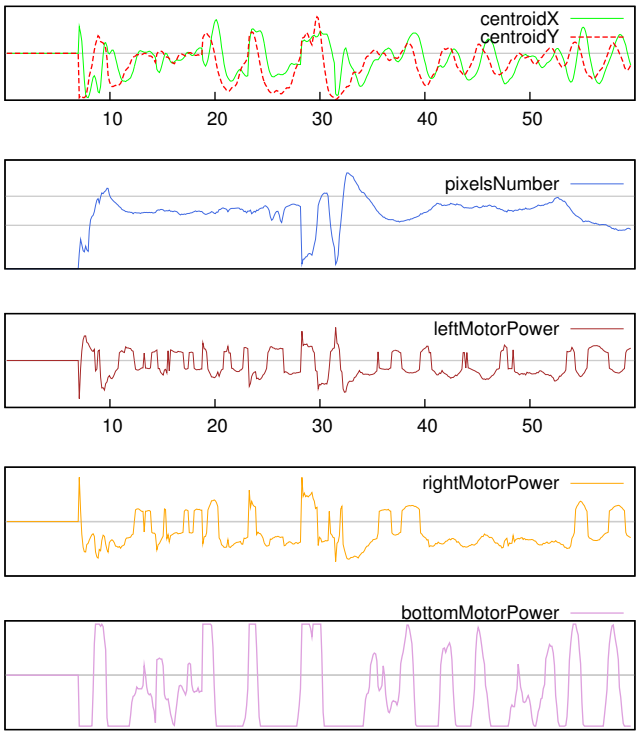

Fig. 6. Proportional Derivative controller for the 'light target following' behaviour, where the maximum errors are $max_{|\chi_e|} = 73$, $max_{|\phi_e|} = 71$ and their mean values are $mean_{|\chi_e|} = 23.2$, $mean_{|\phi_e|} = 19.8$

- Output Variables
  - $left-right-bottomMotorPower_{fz}$, fuzzify *left-right-bottomMotorPower* thrusters power values.

Triangular symmetric membership functions were chosen to maximize the fuzzy logic library computation performance. Other kind of functions, as Gaussian for example, leaded to slightly longer periods of the controller step. Recurrent experimentation showed that five membership functions were enough for the quantization of the variables' ranges. Adding more functions had no positive influence to the controller's behaviour, while using four or less functions leaded to unstable overshooting behaviour. Mamdani type fuzzy rules with centre of gravity deffuzifier produce the controller output. The fuzzy rules database contains rules associating:

TABLE II

RULES ASSOCIATING $CentroidX_{fz}$ AND $dCentroidX_{fz}$ ANTECEDENT INPUTS WITH $leftMotorPower_{fz}$ AND $rightMotorPower_{fz}$ CONSEQUENT OUTPUTS

$dCentroidX_{fz}$ \ $CentroidX_{fz}$	vR	R	C	L	vL
qtR	⇑⇓	↓↑	⇓⇑	⇓⇑	⇓⇑
tR	⇑⇓	⋆⋆	↓↑	↓↑	⇓⇑
noT	⇑⇓	↑↓	⋆⋆	↓↑	⇓⇑
tL	⇑⇓	↑↓	↑↓	⋆⋆	⇑⇓
qtL	⇑⇓	⇑⇓	⇑⇓	↑↓	⇓⇑

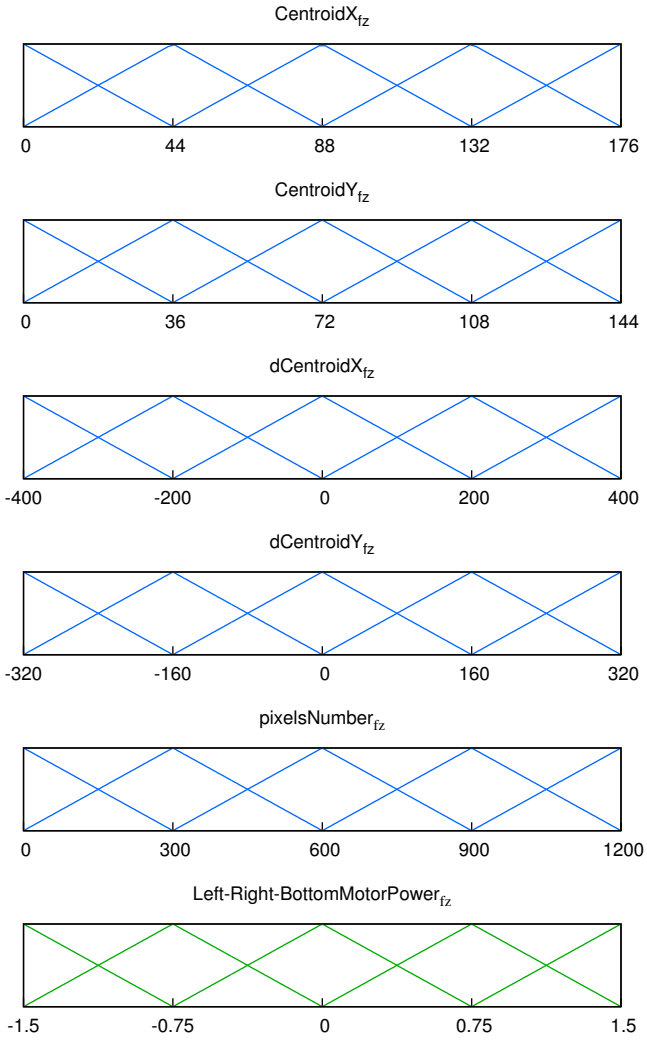

Fig. 7. Membership Functions of linguistic variables

- $CentroidX_{fz}$ and $dCentroidX_{fz}$ antecedent inputs with $leftMotorPower_{fz}$ and $rightMotorPower_{fz}$ consequent outputs, Table II. Each table element comprises of two symbols: the first refers to $leftMotorPower_{fz}$ and the second to $rightMotorPower_{fz}$. Table III explains the symbols used at Table II
- $CentroidY_{fz}$ and $dCentroidY_{fz}$ antecedent inputs with $bottomMotorPower_{fz}$ consequent output, Tables IV, V. The subscript number of a table element denotes rule's firing strength.
- $pixelsNumber_{fz}$ antecedent input with $leftMotorPower_{fz}$ and $rightMotorPower_{fz}$ consequent outputs, Tables VI, VII.

The FL controller was tested with the same scenario as the PD one, producing analogous results shown in Fig. 8. The vehicle successfully follows the light target course without overshooting and significant deviations. The FL control is deemed adequate for achieving this kind of behaviour, requiring yaw, heave and surge control. Vehicle's diving

TABLE III

SYMBOLS INDEX

$l/rightMotorpower_{fz}$		$CentroidX_{fz}$		$dCentroidX_{fz}$	
⇑	full ahead	vR	very right	qtR	quick turn right
↑	ahead	R	right	tR	turn right
⋆	stop	C	centre	noT	no turn
↓	astern	L	left	tL	turn left
⇓	full astern	vL	very left	qtL	quick turn left

TABLE IV

RULES ASSOCIATING $CentroidY_{fz}$ AND $dCentroidY_{fz}$ ANTECEDENT INPUTS WITH $bottomMotorPower_{fz}$ CONSEQUENT OUTPUT

$dCentroidY_{fz}$ \ $CentroidY_{fz}$	vU	U	C	D	vD
qhU	⇑	↓	⇓	⇓	⇓
hU	⇑	↑ 0.3	↓	↓	⇓
noH	⇑	↑	⋆	↓	⇓
hD	⇑	↑	↑	↓ 0.3	⇓
qhD	⇑	⇑	⇑	↑	⇓

stability is indicated by the last plot of Fig. 8, showing roll and pitch orientation during the test: the deviation from the roll and pitch balance position is negligible throughout the diving experiment. The basic difference between the two controllers proved to be a smoother behaviour for fuzzy one, throughout the diving tests inside the experimentation area. The PD controller, though, had better performance during experiments including sudden light target course changes.

IV. CONCLUSIONS

ALE III, a low cost UAV, sharing the open source philosophy, capable of indoors underwater operation, experimentation and testing was developed. Different underwater behaviours emerged as a proof of concept for the vehicle and the experimentation area. The proposed software architecture simplifies the design and programming of control processes. PD and FL controllers were easily evolved and proved to be adequate inside the custom made experimentation area.

After the, so far, project phases of design, implementation, testing and improvement, the following future work plans arose. The *Controller* class should be enriched with an intelligent feature to choose the appropriate behaviour, from a set of available simple behaviours. For example, a set of three simple behaviours, similar to the ones already mentioned at the paper, could be programmed: SEARCH for a light target, TRACK a light target and move to a special RECHARGE facility area. A supervisor *Controller* class, enriched with intelligent behaviour planing, will be able to choose the appropriate, each time, behaviour among the set $\{\text{SEARCH}, \text{TRACK}, \text{RECHARGE}\}$, of the available ones. Thus, this supervised control will be able to evolve more complex and biomimetic behaviours.

The use of a tank with bigger dimensions, e.g. $2.5 \times 0.8(Diameter \times Height)$ will provide the necessary area for experimenting with two vehicles like Ale III. Using the experimentation area described at the previous sections, came

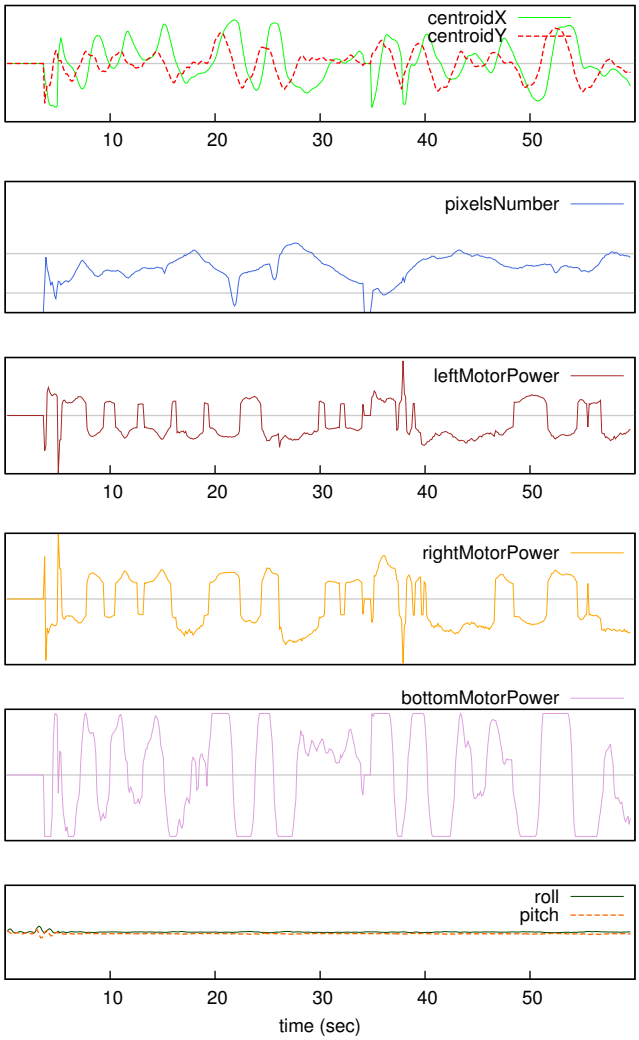

Fig. 8. Fuzzy Logic controller for the 'light target following' behaviour, where the maximum errors are $max_{|\chi_e|} = 83$, $max_{|\phi_e|} = 65$ and their mean values are $mean_{|\chi_e|} = 29.4$, $mean_{|\phi_e|} = 21.8$. The plot at the bottom shows the roll and pitch passive stability throughout the experiment. The maximum roll and pitch errors are $max_{roll_e} = 18^o$, $max_{pitch_e} = 8.3^o$ and their mean values are $mean_{roll_e} = 0,82^o$, $mean_{pitch_e} = -4.18^o$ respectively

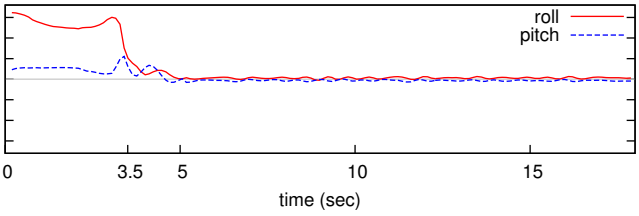

Fig. 9. Experiment denoting vehicle's roll and pitch stability during diving. Initially the AUV is forced to take an upside down position with all the thrusters turned off. At 3.5 seconds it is released to move freely and it recovers to a stable, near-zero roll and pitch orientation after a few seconds

TABLE V

SYMBOLS INDEX

$bottomMotor_{fz}$		$CentroidY_{fz}$		$dCentroidY_{fz}$	
$\Uparrow$	full ahead (up)	vU	very up	qhU	quick heave upwards
$\uparrow$	ahead	U	up	hU	heave up
$\star$	stop	C	centre	noH	no heave
$\downarrow$	astern (down)	D	down	hD	heave down
$\Downarrow$	full astern	vD	very down	qhD	quick heave down

TABLE VI

RULES ASSOCIATING $pixelsNumber_{fz}$ ANTECEDENT INPUT WITH $leftMotorPower_{fz}$ AND $rightMotorPower_{fz}$ CONSEQUENT OUTPUTS

	$pixelsNumber_{fz}$				
$pixelsNumber_{fz}$	vS	S	T	B	vB
$l/rightMotorPower_{fz}$	$\Uparrow\Uparrow$	$\uparrow_{0.3}\uparrow_{0.3}$	$\oslash$	$\Downarrow\Downarrow$	$\downarrow_{0.3}\downarrow_{0.3}$

out the conclusion that for every new vehicle added to the scenario, an addition of $1m^3$ should be made to the tank's volume.

Another issue worth dealing, offering great improvement to the experimentation area capabilities is the addition of special equipment to estimate the vehicle, or vehicles, absolute position inside the tank. For example, the addition of an external camera, at a height above the tank's centre, watching downwards, will be able to track a special light sign located at vehicle top and will estimate its absolute position at the horizontal plane.

REFERENCES

[1] "Bluefin Robotics." http://www.bluefinrobotics.com.

[2] "Gavia Autonomous Underwater Vehicles." http://www.gavia.is.

[3] "Kongsberg Maritime." http://www.km.kongsberg.com/hydroid.

[4] "Autonomous Underwater Vehicles, a collection of groups and projects." http://www.transit-port.net/Lists/AUVs.html.

[5] "AUVSI Foundation and ONR's International RoboSub Competition." http://www.auvsifoundation.org/Competitions/RoboSub.

[6] "Tecnadyne." http://www.tecnadyne.com.

[7] S. Guo, L. Shi, N. Xiao, and K. Asaka, "A biomimetic underwater microrobot with multifunctional locomotion," *Robotics and Autonomous Systems*, vol. 60, pp. 1472–1483, Dec. 2012.

[8] J. Liu and H. Hu, "Biological Inspiration: From Carangiform Fish to Multi-Joint Robotic Fish," *Journal of Bionic Engineering*, vol. 7, pp. 35–48, Mar. 2010.

[9] Q. Yan, Z. Han, S.-w. Zhang, and J. Yang, "Parametric Research of Experiments on a Carangiform Robotic Fish," *Journal of Bionic Engineering*, vol. 5, pp. 95–101, June 2008.

[10] D. Zhang, L. Wang, J. Yu, and G. Xie, "Robotic fish motion planning under inherent kinematic constraints," in *2006 American Control Conference*, p. 6 pp., IEEE, 2006.

[11] K. Zou, C. Wang, G. Xie, T. Chu, L. Wang, and Y. Jia, "Cooperative control for trajectory tracking of robotic fish," in *2009 American Control Conference*, pp. 5504–5509, IEEE, 2009.

[12] "Gumstix, Inc.." https://www.gumstix.com.

[13] "CMUcam: Open Source Programmable Embedded Color Vision Sensors." http://www.cmucam.org.

[14] "Sparkfun Electronics." https://www.sparkfun.com.

[15] "The Ångström Distribution." http://www.angstrom-distribution.org.

TABLE VII

SYMBOLS INDEX

$l/rightMotorPower_{fz}$		$pixelsNumber_{fz}$	
$\Uparrow$	full ahead	vS	very small
$\uparrow$	ahead	S	small
$\oslash$	no rule	T	target value
$\downarrow$	astern	B	big
$\Downarrow$	full astern	vB	very big