Design and Development of an Inexpensive Aquatic Swarm Robotics System

Vasco Costa, Miguel Duarte, Tiago Rodrigues, Sancho Moura Oliveira and Anders Lyhne Christensen

BioMachines Lab, Lisbon, Portugal
Instituto de Telecomunicações, Lisbon, Portugal
Instituto Universitário de Lisboa (ISCTE-IUL), Lisbon, Portugal
vasco_craveiro_costa@iscte.pt

***Abstract*—Swarm robotics is a promising approach characterized by large numbers of relatively small and inexpensive robots. Since such systems typically rely on decentralized control and local communication, they exhibit a number of interesting and useful properties, namely scalability, robustness to individual faults, and flexibility. In this paper, we detail the design and development process of a swarm robotics platform composed of autonomous surface robots, which was designed in order to study the use of robotic swarms in real-world environments. Our aquatic surface robots where manufactured using digital fabrication techniques, such as 3D printing and CNC milling, and all hardware and software has been made available as open-source, thus allowing third-parties to customize and further improve our platform.**

I. Introduction

Swarm robotics is a bio-inspired approach to the design of multirobot systems composed of large numbers of simple, autonomous robots with decentralized control [1]. Each robot makes control decisions based on sensors readings and by coordinating with nearby robots. Swarm robotics systems (SRS) have a number of potential advantages when compared with traditional multirobot systems, such as inherent scalability, flexibility, and robustness to faults [1], [2], [3]. Certain constraints have to be taken into account in order to develop a SRS. For instance, in order to make deployment of large swarms viable, the cost of each individual unit must be kept low, which implies that robots must be kept relatively simple. Demonstrations on such robotic platforms usually focus on basic swarm behaviors, such as aggregation, flocking, foraging, clustering, sorting, and path formation [2], and experiments are usually conducted in controlled laboratory environments, as opposed to real-world conditions [3].

In this paper, we provide a technical overview of the design and development process of a SRS platform composed of small and inexpensive autonomous surface vessels (see Figure 1). The aim of the platform was to enable swarm robotics experiments outside of a laboratory environment [4]. The design of our system was based on the following four objectives:

1) The solution should be a low-cost robotic platform. This was achieved through the use of inexpensive off-the-shelf and widely available components, as well as through the use of digital fabrication processes.
2) The solution should allow for easy logistics, namely transportation and deployment. This was achieved through the design of small and compact units (65 cm length by 40 cm wide).
3) Each robotic unit should be capable of autonomous decision-making. This was achieved through the inclusion of onboard processing, communication, and sensing.
4) The system should provide a human-machine interface that allows an operator to monitor and supervise a swarm of aquatic robots. This was achieved through the development of an easy-to-use command and control console.

The developed robotic platform is versatile and customizable, and all hardware specification and designs, as well as all software modules, are made available as open-source under the GNU LGPLv3 license, enabling replication and extension by third parties. The total cost of each unit is approximately 300 EUR in materials. To facilitate studies on control synthesis and swarming behavior for real-world robotic systems [5], we combined the robotic platform with our simulation framework, JBotEvolver [6].

In the following sections, we provide an overview of the design and manufacturing of our robotic units (Sec-

Figure 1: A swarm of eight robots (out of a total of ten developed) at Parque das Nações, Lisbon, Portugal.

tion II), a description of the onboard hardware (Section III), and software (Section IV). We further provide an overview of how control is synthesized, and present recent experiments conducted on the swarm robotics system presented in this paper (Section V). Finally, Section VI contains concluding remarks.

II. Hull Design

For our robotics units, we opted for a monohull-shaped vessel (see Figure 2), which is machinable from a single block of raw material. The robots are relatively small (L 65 cm × W 40 cm × H 15 cm), and light (3 Kg). While we have used low-cost Computerized Numeric Cut (CNC) and 3D-printing fabrication processes and materials, the open-source nature of the platform allows for different fabrication processes, such as casting. Our platform can furthermore be adapted to support different sensors payloads and actuators. The design of all components have been made publicly available and can be found on our research group's website.[1]

A. Fabrication Process

We designed the hull and support parts in computer-aided design (CAD) software (*Rhinoceros 3D*), which were then produced using digital fabrication techniques. The hulls were milled using an *Ouplan 3020* CNC machine, and 12 support parts were produced using a *BQ Prusa i3 Hephestos* 3D printer. The use of digital design, modeling, and fabrication processes allowed us to quickly iterate and optimize the hull and the support parts designs, and to have a short and inexpensive design-to-product cycle. In total, we produced 19 different hulls, nine of them prototypes, and 10 operational units.

[1] http://biomachineslab.com/

B. Materials

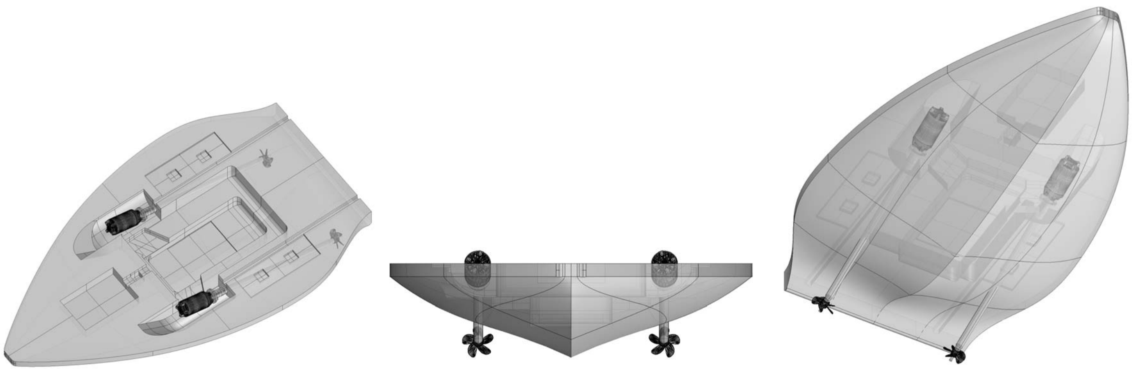
We used extruded polystyrene foam (XPS) for the hull production since it is buoyant, easily machinable, and inexpensive. This material can also be hand worked, allowing for manual shaping and finishing. The 12 support parts were 3D-printed in Polylactic Acid (PLA), which is an inexpensive biodegradable thermoplastic. The 3D-printed parts were installed in the hull using silicon-based glue in order to support the different hardware component, such as motors, shafts, enclosures, and sensors. The shaft support design (see Figure 3) allows for a quick motor and shaft replacement, reducing the repair time in case of motor breakdown. The final batch of robots were coated in black epoxy resin and fiberglass in order to increase strength and robustness, and to waterproof the hull.

III. Electronics and Propulsion

Maritime environments represent a challenge for roboticists: the vessel's exposure to harsh environmental elements, such as solar UV-light, heat, and salt water, requires a high degree of isolation for sensitive components. Most of the electronic components were therefore housed in one of two enclosures.

A. Enclosures

We used 2.5 L (*main enclosure*) and 0.24 L (*secondary enclosure*) plastic containers to house all the electronic components and circuitry. We found that this inexpensive and flexible solution presents a degree of protection similar to IP67 standard, therefore fitting our needs. The main enclosure contained the power source, along with main processing and sensing components. The secondary enclosure contained two diagnostic LEDs to facilitate immediate status reporting, and sensors that needed to be isolated from electromagnetic interference from the motors or other components in the main enclosure. The connections between enclosures, and between components inside and outside enclosures, were made



(a) Top view (b) Front view (c) Bottom view

Figure 2: CAD model

through IP68-rated cable glands. In order to minimize equipment overheating, the main enclosure was covered with aluminium tape to reflect sunlight.

B. Propulsion

Several experiments with different propulsion options were conducted, including experiments with turbines and inboard motors. The turbine system, based on *EDF Ducted Fan Unit 6 Blade 66 mm*, despite fast and efficient, proved prone to motor oxidation and debris entanglement. We therefore opted for a differential propulsion system composed of two motors coupled to a 4 mm drive shaft with a 3-blade 28 mm propeller. The drive shaft ran on a 255 mm length shaft sleeve filled with lithium-based grease. This solution was chosen for the final batch of operational units. Two different motor models were used: (i) *NTM Prop Drive Series 28-30A 750 kv/ 140 w* and (ii) *Emax 2215/25 950 kv 2-3S*. Each motor is driven by a *HobbyKing 50 A Boat ESC 4 A UBEC* electronic speed controller (ESC), which present a current limit nearly twice the one necessary, therefore decreasing chances of equipment overheating while providing good compatibility with the motors used. The ESCs were installed outside on the bottom of the main enclosure. This propulsion setup enabled the final batch of robotic units to move at speeds up to 1.7 m/s (3.3 kts), to achieve turning rates of 90 °/s, and to accelerate to full speed in one second.

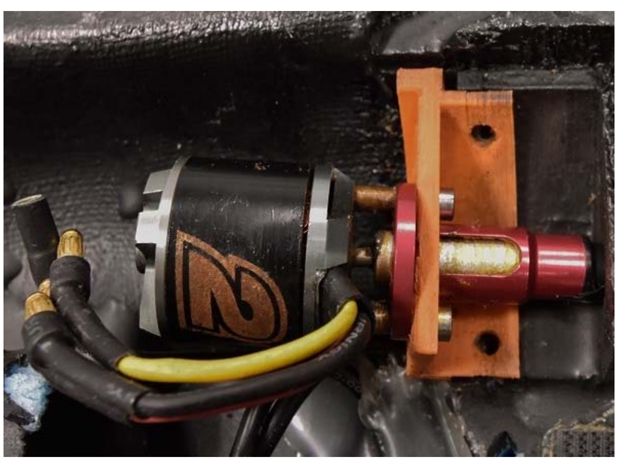


Figure 3: Detachable motor support. The orange piece supports both the motor and the shaft, and allows the module to be detached from the hull support (in black).

C. Energy

Energy was provided by two batteries, both located in the main enclosure: (i) a unit that powers all the equipment related with motors and propulsion (*motor battery*), and (ii) a unit to power control, processing and sensing components (*control battery*). We conducted experiments with both lithium-polymer (LiPo) and lithium-iron-phosphate (LiFePo4) batteries. LiPo batteries were chosen for the final iteration of the platform due to their lower price and relatively higher power density. For the motor battery, we chose a *ZIPPY Flightmax 8000 mAh 3S1P* battery, which provided an autonomy between 1h30m and 4h30m depending on motor usage. The control battery used was a *ZIPPY Flightmax 5000 mAh 3S1P*, which supplied power to all the remaining components through a *Turnigy 5A* (8-26 V) switched battery eliminator circuit (SBEC), that regulates and stabilizes the battery voltage to 5 VDC. The control battery provided a run time of approximately 4h30m.

D. Computation & Communications

Onboard computation was provided by a Raspberry Pi 2 single-board computer (SBC). The Raspberry Pi 2 is composed of a quad-core ARM Cortex-A7 CPU clocked at 900 MHz, 1GB RAM, 4 USB ports and 40 general purpose input/output (GPIOs) pins supporting diverse protocols such as UART, I^2C, SPI and One-Wire, which facilitates integration with different electronic components and modules. The SBC is located in the *main electronics enclosure* and is connected to the remaining components through a custom breakout cable. In order to enable communication between neighboring robots, we included a wireless communication system using a TP-Link TL-WN722N High-Gain Wi-Fi adapter, connected to the SBC through an USB interface. The adapter was coupled to a monopole 4 dBi gain antenna, providing an effective communication range between neighboring robots of 40 m on the water surface.

E. Sensors

Various sensors were included in each robot, namely a Global Positioning System (GPS) receiver, a digital compass unit, and a temperature sensor. Global position information was provided by an *Adafruit Ultimate GPS Breakout*, based on *GlobalTop FGPMMOPA6H GPS Standalone* module [7], which was placed in the main enclosure. This module is a 66 channel GPS receiver providing position updates with a 5 Hz frequency, and interfaced with the SBC through the UART protocol. It was coupled with an active 26 dB gain GPS antenna, increasing the received signal quality and providing positioning information with a ± 3 m accuracy.

Heading information was provided by a *STMicroelectronics LSM303D* magnetometer, which interfaced with the SBC through a standard I^2C protocol. This unit contains both a triple-axis magnetometer and a triple-axis accelerometer, allowing for the compensation of the magnetic readings according to the pose of the robot. The location of the sensors in the vessel was also subject to experimentation, since we verified that high current wires, motors, and batteries interfered with the magnetic field readings. Therefore, we installed the magnetometer in the secondary enclosure, which was located in the prow of the vessel.

Finally, temperature information was provided by both the onboard SBC temperature sensor and by a waterproof *Maxim DS18B20* sensor [8]. The first sensor was used to monitor the conditions inside the main enclosure. The second sensor, positioned in the bottom of the vessel, was used to measure the water temperature. This latter is a digital 12-bit resolution temperature sensor, which gives readings in 0.0625°C increments and has an error of ± 0.5 °C. This unit has an update frequency of approximately 1.25 Hz and interfaced with the SBC through a One-Wire standard protocol.

The location of the electronic and propulsion components on board each of the robots can be found in Figure 4, and a summary of all the components can be found in Table I.

IV. SOFTWARE

The software that enables the control and monitoring of the robotic platform is divided into three different elements:

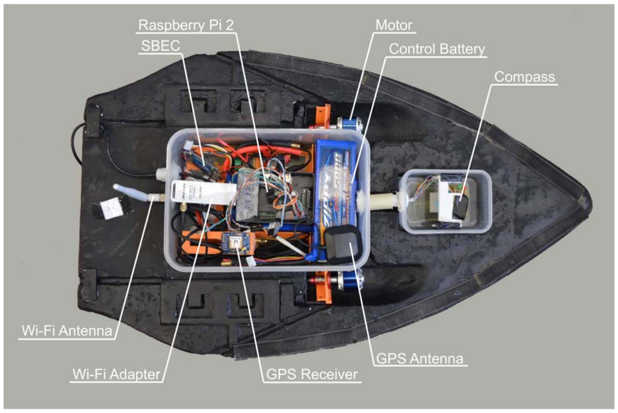
- An onboard software component, responsible for the control and management of each robotic unit (*Raspberry Controller*);
- A console that enables command and control of the swarm by a human operator (*Control Console*);
- An API layer, which makes the use of simulation or the real robotic hardware transparent to the robotic controller (*Common Interface*).

A. Onboard Software

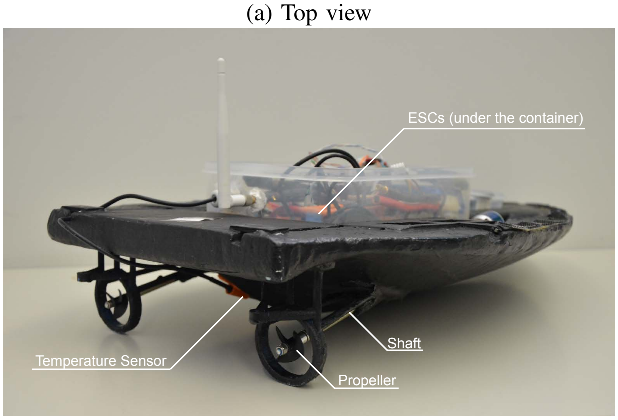
The Raspberry Pi 2 SBC runs a Raspbian Wheezy Linux operative system, which is based on Linux Debian Wheezy distribution compiled for ARM architecture and with hard-float support. In order to interact with the different hardware components, we used several existent open-source software components. A guide on how to

Table I: Components

Component	Make and Model
Enclosures	
Main enclosure	2.5 L watertight plastic box
Secondary enclosure	0.24 L watertight plastic box
Propulsion	
Motor (A)	NTM Prop Drive Series 28-30A 750 kv/ 140 w
Motor (B)	Emax 2215/25 950 kv 2-3S
Shaft	4 mm drive shaft
Shaft Sleeve	255 mm length shaft sleeve
Propeller	3-blade 28 mm propeller
ESC	HobbyKing 50 A Boat ESC 4 A UBEC
Power	
Motor battery	ZIPPY Flightmax 8000 mAh 3S1P
Control battery	ZIPPY Flightmax 5000 mAh 3S1P
SBEC	Turnigy 5A (8-26 V)
Computation & Communications	
Single board computer	Raspberry Pi 2
Wi-Fi Adapter	TP-Link TL-WN722N
Sensors	
GPS	Adafruit Ultimate GPS Breakout
Compass	STMicroelectronics LSM303D
Water Temperature Sensor	Maxim DS18B20



(a) Top view



(b) Side view

Figure 4: Components

replicate the robot's software system configurations can be found in our team's GitHub page.[2]

The Raspberry Controller is the Java-based software running onboard each robot. This software is responsible for interacting with all sensors and actuators, executing the behavioral control logic, and for communicating with nearby robots and the control console. It relies on the Pi4J library to interact with the hardware components, except for the interaction with ESCs, which is achieved using the ServoBlaster kernel module. The source code for our Raspberry Controller software is available under open-source license.[3]

ServoBlaster[4] is a kernel module that enables the generation of pulse position modulated (PPM) signals through the Raspberry Pi's GPIOs. This modulation enables the transmission of position information encoded in temporal pulses [9], the signal necessary to control the ESCs used in our robots.

We use WiringPi v2.25 C library[5] to manipulate the GPIO and to interact with the different sensors. To the access the WiringPi C library's methods from the onboard software, we use Pi4J 1.1-SNAPSHOT library.

The communication between a human experimenter and the swarm is performed through an ad-hoc *IEEE 802.11g* wireless network. An *Ubiquiti BULLET-M2-HP* running OpenWrt Chaos Calmer 15.05 r46133 firmware[6] with LuCI Configuration Interface coupled to a 12 dBi gain monopole antenna is installed at the base station. The setup provided a communication range of 150 m between the base station and the robots operating on the water surface. Two pieces of information are broadcast on the network using UDP messages, namely the robot's GPS position and keep-alive messages. When reliability is required, such as when a robotic unit is tele-operated by an operator or when new control logic is uploaded, TCP/IP connections are used.

B. Control Console

For command and control, we developed a stand-alone multi-platform desktop application (see Figure 5). This application[7] enables the experimenter to control and monitor a swarm of aquatic robots. Each unit's location and heading is displayed on a map. Additional telemetry information can be displayed when required, along with data collected by the onboard sensors. The robots' onboard control logic can furthermore be updated through the console, and various spatial entities can be configured and deployed to specific robots, such as waypoints, geo-fences, and the location of obstacles to avoid. The software generates log files of the commands sent to individual units along with all broadcasted messages that enable off-line replay of the experiments and facilitate off-line debugging and data extraction. Multiple instances of the control console can be executed simultaneously, providing control redundancy and allowing for multiple operators.

C. Common Interface

We developed a *common interface* API layer, which provides source code level compatibility between control logic executed in simulation and on the real robots. This component sits between the high-level control logic and the low-level hardware interface, facilitating the

[2]https://github.com/BioMachinesLab/drones/wiki
[3]https://github.com/BioMachinesLab/drones/tree/master/RaspberryController
[4]https://github.com/richardghirst/PiBits/tree/master/ServoBlaster
[5]http://wiringpi.com/
[6]https://openwrt.org/
[7]https://github.com/BioMachinesLab/drones/tree/master/DroneControlConsole

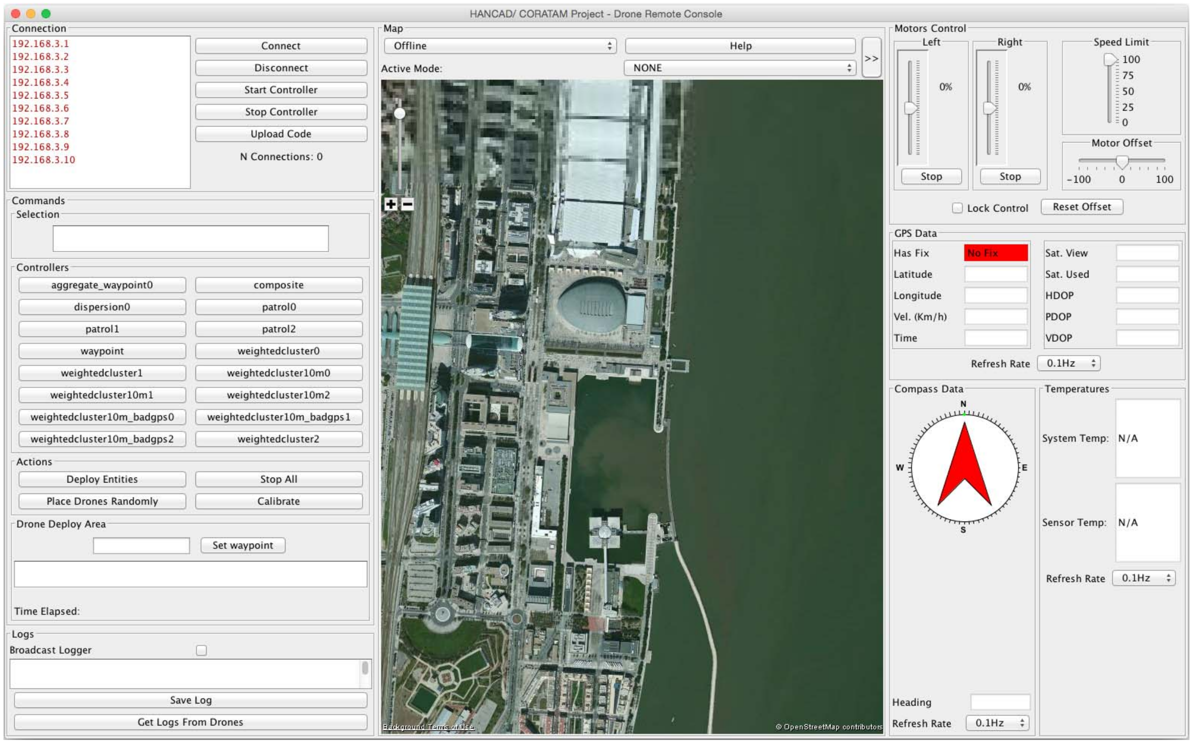


Figure 5: A screenshot of the control console.

synthesis of control and its transfer from simulation to the real robots. The common interface was integrated with our simulator JBotEvolver [6] in order to synthesize self-organized swarm control, which was then transferred successfully to the real robotic swarm.

V. Studies Conducted with the Platform

We applied evolutionary robotics (ER) techniques [10] in order to synthesize control for our robots. ER is a promising approach for SRS since it allows for the automatic synthesis of control based only on a description of the task's goal. Through this technique, swarm and self-organized behaviors emerge [11], avoiding the need for manual specification of the low-level behavior of each individual in the swarm [12].

We demonstrated the successful transfer of evolved control from simulation to real hardware in a series of experiments [4]. In a first study, we synthesized control for four canonical swarm behavior tasks: (i) homing, (ii) dispersion, (iii) clustering and (iv) area monitoring. Afterwards, we experimented with a sequential composition of the different behaviors in an environmental monitoring task, where the robots had to navigate to a predefined area, disperse, cover the area while continuously collecting water temperature measurements, and finally aggregate and collectively navigate back to the base station.

We also studied the application of *hierarchical control synthesis* for SRS [13]. This method enables complex tasks to be solved by combining different types of control synthesis techniques. We tested this approach on an intruder detection task with realistic constraints [14]. The robots had to monitor an area, detect, and follow any intruder that attempted to cross it and periodically recharge their batteries at a base station.

In both of the two studies discussed above, the performance and behavior observed on the real robots was similar to those observed in simulation [4], [14]. In this way, the hardware platform presented in this paper has facilitated novel contributions to the field of swarm robotics, and most notably, was used in the first successful demonstration of evolved control outside of strictly controlled laboratory conditions [3], [4].

VI. Conclusions

We advocate that the highly distributed and autonomous nature of SRS can be advantageous in many real-world maritime missions, and potentially enable completely new classes of tasks to be addressed. In order to bring this vision closer to fruition, we developed a robotics platform for SRS experiments in marine environments. This platform has the following key features:(i) each unit is relatively simple and inexpensive, enabling large numbers of robots to be manufactured and tested (ii) each unit is relatively small and compact, allowing for an easy deployment logistics, (iii) each robot is capable of autonomous decision-making, and (iv) the system provides an intuitive and easy-to-use command and control interface.

In this paper, we provided an overview of our robotic platform, demonstrating how our four key design objectives were achieved. All of our designs and the source code necessary to replicate and control the swarm were made available under the GNU LGPLv3 license. In summary, our solution represents a simple, inexpensive, flexible, and open platform for maritime swarm robotics studies, which can be extended and improved by third parties.

In ongoing work, we are studying potential improvements to the platform. In large-scale swarms, different robots might be equipped with different types of communication capabilities and serve as gateways for the rest of the swarm [5], or a few of the robots may be equipped with different sensors payloads and share information with the neighboring robots [15]. Such approaches can allow for the increase of the swarm's capabilities, while keeping the cost of the average robot low. In our ongoing work, we are furthermore integrating our software stack with the Robot Operating System (ROS).

Acknowledgments

This work was partly supported by FCT – Foundation of Science and Technology under grants SFRH/BD/76438/2011, EXPL/EEI-AUT/0329/2013, and UID/EEA/50008/2013.

References

[1] E. Sahin, "Swarm Robotics: From Sources of Inspiration to Domains of Application," in *Swarm Robotics* (E. Sahin and W. M. Spears, eds.), vol. 3342 of *Lecture Notes in Computer Science*, pp. 10–20, Springer, Berlin, Heidelberg, 2005.

[2] L. Bayındır, "A review of swarm robotics tasks," *Neurocomputing*, vol. 172, pp. 292–321, 2015.

[3] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[4] M. Duarte, V. Costa, J. Gomes, T. Rodrigues, F. Silva, S. M. Oliveira, and A. L. Christensen, "Evolution of Collective Behaviors for a Real Swarm of Aquatic Surface Robots," 2016. Preprint, http://arxiv.org/abs/1511.03154.

[5] A. L. Christensen, S. Oliveira, O. Postolache, M. J. de Oliveira, S. Sargento, P. Santana, L. Nunes, F. Velez, P. Sebastiao, V. Costa, M. Duarte, J. Gomes, T. Rodrigues, and F. Silva, "Design of communication and control for swarms of aquatic surface drones," in *Proceedings of the International Conference on Agents and Artificial Intelligence*, pp. 548–555, SCITEPRESS, Lisbon, Portugal, 2015.

[6] M. Duarte, F. Silva, T. Rodrigues, S. M. Oliveira, and A. L. Christensen, "JBotEvolver: A Versatile Simulation Platform for Evolutionary Robotics," in *Proceedings of the International Conference on the Synthesis and Simulation of Living Systems (ALIFE)*, pp. 210–211, MIT Press, Cambridge, MA, 2014.

[7] GlobalTop Technology Inc., "FGPMMOPA6H GPS Standalone Module Data Sheet." https://www.adafruit.com/datasheets/GlobalTop-FGPMMOPA6H-Datasheet-V0A.pdf, 2012. original document from GlobalTop Technology Inc.

[8] Maxim Integrated Products, Inc., "DS18B20." https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf, 2015.

[9] J. Hamkins, "Pulse Position Modulation," in *Handbook of Computer Networks: Key Concepts, Data Transmission, and Digital and Optical Networks, Volume 1* (H. Bidgoli, ed.), pp. 492–508, Hoboken, NJ, USA: John Wiley & Sons, Inc, 2007.

[10] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-organizing Machines*. MIT Press, Cambridge, MA, 2000.

[11] S. Nolfi, "Evolutionary Robotics: Exploiting the Full Power of Self-organization," *Connection Science*, vol. 10, no. 3-4, pp. 167–184, 1998.

[12] D. Floreano and L. Keller, "Evolution of adaptive behaviour in robots by means of Darwinian selection.," *PLoS biology*, vol. 8, no. 1, p. e1000292, 2010.

[13] M. Duarte, S. M. Oliveira, and A. L. Christensen, "Evolution of hybrid robotic controllers for complex tasks," *Journal of Intelligent and Robotic Systems*, vol. 78, no. 3–4, pp. 463–484, 2015.

[14] M. Duarte, J. Gomes, V. Costa, S. M. Oliveira, and A. L. Christensen, "Hybrid control for a real swarm robotic system in an intruder detection task," in *Proceedings of the 18th European Conference on the Applications of Evolutionary Computation (EvoStar)*, Springer, Berlin, Germany, 2016.

[15] T. Rodrigues, M. Duarte, M. Figueiró, V. Costa, S. M. Oliveira, and A. L. Christensen, "Overcoming limited onboard sensing in swarm robotics through local communication," in *Transactions on Computational Collective Intelligence XX*, LNCS, pp. 201–223, Springer, Berlin, Germany, 2015.