A General-Purpose Hardware Robotic Platform for Swarm Robotics

Nureddin Moustafa[1], Akemi Gálvez[1,2], Andrés Iglesias[1,2,†]

[1]Dpt. of Applied Mathematics & Comp. Sci., E.T.S.I. Caminos, Canales y Puertos
Universidad de Cantabria, Avda. de los Castros, s/n, 39005, Santander, Spain
[2]Department of Information Science, Faculty of Sciences, Narashino Campus
Toho University, 2-2-1 Miyama, 274-8510, Funabashi, Japan
[†]Corresponding author. Email: iglesias@unican.es
Website: http://personales.unican.es/iglesias

Abstract. Swarm intelligence is based on the recently-acquired notion that sophisticated behaviors can also be obtained from the cooperation of several simple individuals with a very limited intelligence but cooperating together through low-level interactions between them and with the environment using decentralized control and self-organization. Such interactions can lead to the emergence of intelligent behavior, unknown to the individual agents. One of the most remarkable applications of swarm intelligence is swarm robotics, where expensive and sophisticated robots can be replaced by a swarm of simple inexpensive micro-robots. In this context, this paper introduces a general-purpose hardware robotic platform suitable for swarm robotics. With a careful choice of its main components and its flexible and modular architecture, this robotic platform provides support to the most popular swarm intelligence algorithms by hardware. As an illustration, the paper considers four of the most popular swarm intelligence methods; then, it describes the most relevant hardware features of our approach to support such methods (and arguably many other swarm intelligence approaches as well) for swarm robotics.

Keywords: swarm intelligence, swarm robotics, general-purpose robot, hardware robotic platform, intelligent behaviors

1 Introduction

Swarm intelligence (SI) has been regarded as one of the most exciting new avenues of research in artificial intelligence (AI) during the last few decades. Unlike many other areas in AI, individuals in SI do not need to be actually *intelligent* in its most canonical sense. Instead, the intelligence in SI is typically obtained from the aggregation of very simple behavioral patterns by unsophisticated agents collaborating together to solve a complex problem. Amazingly, this kind of collective behavior had already been observed in some natural groups for centuries. Consider, for instance, the dynamics of colonies of social insects (ants, termites, bees, fireflies), where the group as a whole is able to construct complex nests and carry out many different sophisticated tasks unattainable for its individuals members. Another typical example is the behavior of a flock of birds when moving all together following a common tendency in their displacement. Other examples from nature include animal herding, fish schooling, and many others. Furthermore, these examples have been used as metaphors for some of the most popular SI methods, such as ant colony optimization (ACO) or particle swarm optimization (PSO). In SI methods, there is not a centralized intelligence controlling the swarm, taking decisions, and sending orders to the individual members about how to behave. In fact, such individual agents follow simple rules and have a very limited knowledge and intelligence. However, as a whole, the social group is capable of complex collective behaviors, which emerge from a small set of simple behavioral rules exploiting only low-level interactions between individuals and with the environment (stigmergy) using decentralized control and self-organization. The interested reader is referred to [4, 7, 16] for a comprehensive overview about the field of swarm intelligence, its history, main techniques, and applications. See also Section 2 for a succinct description of some popular swarm intelligence methods.

A major reason to explain this increasing interest on swarm intelligence is its potential application in several fields. An illustrative example is given by *swarm robotics*, a field where swarms of simple and generally inexpensive self-organizing micro-robots are used to replace sophisticated and expensive robots to accomplish complex tasks [2, 5, 9, 10, 12, 13]. As remarked by several authors [1, 3], swarm robotic systems offer several interesting advantages, such as:

- *Improved performance by parallelization*: swarm intelligence systems are very well suited for parallelization, because the swarm members can perform different actions at different locations simultaneously. This feature makes the swarm more flexible and efficient for complex tasks, as individual robots (or groups of them) can solve different parts of a complex task independently.
- *Task enablement*: groups of robots can do certain tasks that are impossible or very difficult for a single robot (e.g., collective transport of too heavy items, dynamic target tracking, cooperative environment monitoring, autonomous surveillance of large areas).
- *Scalability*: inclusion of new robots into a swarm does not require reprogramming the whole swarm. Furthermore, because interactions between robots involve only neighboring individuals, the total number of interactions within the system does not increase dramatically by adding new units, making the system highly scalable.
- *Distributed sensing and action*: a swarm of simple interconnected mobile robots deployed throughout a large search space possesses greater exploratory capacity and a wider range of sensing than a sophisticated robot. This makes the swarm much more effective in tasks such as exploration and navigation (e.g., in disaster rescue missions), nanorobotics-based manufacturing, microbotics for human body diagnosis, and many others.
- *Fault tolerance*: due to the decentralized and self-organized nature of the swarm, the failure of a single unit (or a small group of them) does not affect the completion of the given task.

All these advantages motivated a great interest in swarm robotics during the last two decades. The interested reader is referred to [8, 11–13] and references therein for a brief description about previous work in the field.

1.1 Aims and structure of this paper

In this paper, we introduce a first prototype of a hardware robotic platform for swarm robotics designed to meet some important differential features:

- *It is affordable.* Since the robots must operate in swarms, it is important to keep their individual price as low as possible. In our proposal, we avoid expensive components, so that each robot costs about 50∼60US$.
- *It is general-purpose.* Instead of a specialized goal-oriented robot, our design is general-purpose. This is possible thanks to its flexible design and modular architecture, avoiding fixed parts so that different components (e.g., sensors, holders, frames) can readily be added or removed to meet different goals.
- *It is suitable for swarm robotics.* In spite of its low-cost design, the robot CPU and memory are powerful enough to support some of the most popular swarm intelligence techniques running locally at software level. Furthermore, the robots of the swarm are highly interconnected with one another via standard communication interfaces such as *Wifi* and *Bluetooth* (in our case, they are built-in in our microprocessor single-board).

The structure of this paper is as follows: Sect. 2 provides a very brief description of some popular SI methods. Sect. 3 describes our design of the hardware robotic platform, including its architecture and main components, and the programming framework. Then, Sect. 4 discusses the main features of our proposal that provide support to the methods in Sect. 2 for swarm robotics applications. The paper closes with the conclusions and some future work in the field.

2 Some Popular Swarm Intelligence Algorithms

In this section, some of the most typical SI algorithms are briefly revisited. Since they are very popular and widely reported in the literature, we restrict our discussion to their main features without further detail, and refer the interested reader to some bibliographic entries for a fully informed description. Relevant information about these methods will be used for our discussion on our robotic platform approach in Sect. 4.

2.1 Particle swarm optimization

Particle Swarm Optimization (PSO) is a global stochastic optimization algorithm for dealing with problems where potential solutions (called *particles*) can be represented as vectors in a search space [6, 7]. Particles are distributed over such space and provided with an initial velocity and the capacity to communicate with other neighbor particles, even the entire swarm. Particles “flow” through the solution space and are evaluated according to some fitness function after each instance. Particles evolution is regulated by two memory factors: memory of their own best position and knowledge of the global or their neighborhood's best. Particles of a swarm communicate good positions to each other and adjust their own position and velocity based on these good positions. As the swarm iterates, the fitness of the global best solution improves so the swarm eventually reaches the best solution. To this aim, each particle modifies its position P_i along the iterations by storing the coordinates P_i^b associated with the best solution (fitness) achieved so far. These values account for the *memory* of the best particle position. In addition, members of a swarm can communicate good positions to each other, so they can adjust their own position and velocity according to this information. To this purpose, we also collect the best global position P_g^b from the initial iteration. The evolution for each particle i is given by:

$$\begin{array}{l} V_i(k+1) = w\, V_i(k) + \gamma_1 R_1[P_g^b(k) - P_i(k)] + \gamma_2 R_2[P_i^b(k) - P_i(k)] \\ P_i(k+1) = P_i(k) + V_i(k) \end{array} \quad (1)$$

where $P_i(k)$ and $V_i(k)$ are the position and the velocity of particle i at time k, respectively, w is called *inertia weight* and decide how much the old velocity will affect the new one and coefficients γ_1 and γ_2 are constant values called *learning factors*, which decide the degree of affection of P_g^b and P_i^b. This procedure is repeated several iterations until a termination condition is reached. The reader is referred to [4, 6, 7, 16] for further details on this very popular algorithm.

2.2 Bat algorithm

The *bat algorithm* (BA) is a bio-inspired SI method proposed in 2010 to solve optimization problems [18, 19]. It is based on the behavior of microbats, which use a type of sonar called *echolocation*, with varying pulse rates of emission and loudness, to detect prey, avoid obstacles, and locate their roosting crevices in the dark. The idealization of the echolocation of microbats is as follows:

1. Bats use echolocation to sense distance and distinguish between food, prey and background barriers.
2. Each virtual bat flies randomly with a velocity $\mathbf{v}_i$ at position (solution) $\mathbf{x}_i$ with a fixed frequency f_{min}, varying wavelength λ and loudness A_0 to search for prey. As it searches and finds its prey, it changes the frequency of their pulses and adjust the rate of pulse emission r, depending on the proximity of the target.
3. It is assumed that the loudness will vary from an (initially large and positive) value A_0 to a minimum constant value A_{min}.

Under these idealized rules, the algorithm considers an initial population of bats, each representing a potential solution of the optimization problem and having a location $\mathbf{x}_i$ and velocity $\mathbf{v}_i$. The algorithm initializes these variables with suitable random values. Then, the pulse frequency, pulse rate, and loudness are computed for each individual bat. The swarm evolves in a discrete way over generations until the maximum number of generations is reached. For each g and each bat, new frequency, location and velocity are computed as:

$$f_i^g = f_{min}^g + \beta(f_{max}^g - f_{min}^g) \tag{2}$$

$$\mathbf{v}_i^g = \mathbf{v}_i^{g-1} + [\mathbf{x}_i^{g-1} - \mathbf{x}^*]\, f_i^g \tag{3}$$

$$\mathbf{x}_i^g = \mathbf{x}_i^{g-1} + \mathbf{v}_i^g \tag{4}$$

where $\beta \in [0, 1]$ follows the random uniform distribution, and $\mathbf{x}^*$ represents the current global best location (solution), which is obtained through evaluation of the objective function at all bats and ranking of their fitness values. The best current solution and a local solution around it are probabilistically selected according to some given criteria. Then, search is intensified by a local random walk. For this local search, once a solution is selected among the current best solutions, it is perturbed locally through a random walk. If the new solution achieved is better than the previous best one, it is probabilistically accepted depending on the value of the loudness. In that case, the algorithm increases the pulse rate and decreases the loudness. The reader is referred to [18–20, 23] for further details and in-depth analysis of the method and its implementation.

2.3 Firefly algorithm

The *firefly algorithm* (FFA) is a SI algorithm introduced in 2009 for optimization problems [14]. It is based on the social flashing behavior of fireflies in nature. The key ingredients of the method are the variation of light intensity and formulation of attractiveness. In general, the attractiveness of an individual is assumed to be proportional to their brightness, which in turn is associated with the encoded objective function. In the firefly algorithm, there are three particular idealized rules, based on some of the major flashing characteristics of real fireflies:

1. All fireflies are unisex, so that one firefly will be attracted to other fireflies regardless of their sex;
2. The degree of attractiveness of a firefly is proportional to its brightness, which decreases as the distance from the other firefly increases due to the fact that the air absorbs light. For any two flashing fireflies, the less brighter one will move towards the brighter one. If there is not a brighter or more attractive firefly than a particular one, it will then move randomly;
3. The brightness or light intensity of a firefly is determined by the value of the objective function of a given problem. For optimization problems, the light intensity can simply be proportional to the value of the objective function.

For a full description of this method, the reader is kindly referred to [14–17].

2.4 Cukoo search algorithm

The *cuckoo search algorithm* (CSA) is a SI method proposed in 2009 [21] and inspired by the obligate interspecific brood-parasitism of some cuckoo species that lay their eggs in the nests of host birds of other species to escape from the parental investment in raising their offspring and to minimize the risk of egg loss to other species. In CSA, the eggs in the nest represent the pool of candidate solutions while the cuckoo egg represents a new coming solution. The method uses these new (potentially better) solutions associated with the parasitic cuckoo eggs to replace the current solution associated with the eggs in the nest. This replacement, carried out iteratively, will eventually lead to a very good solution. For computational reasons, CSA is also based on three idealized rules [22]:

1. Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;
2. The best nests with high quality of eggs (solutions) will be carried over to the next generations;
3. The number of available host nests is fixed, and a host can discover an alien egg with a probability $p_a \in [0, 1]$. For simplicity, this assumption can be approximated by a fraction p_a of the n nests being replaced by new nests (with new random solutions at new locations).

More details on this algorithm can be found in [16, 17, 21, 22].

3 Hardware Robotic Platform for Swarm Robotics

3.1 Hardware architecture and components

The main components of our robotic platform are shown in Figure 1. They are:

1. the *chassis*: the robot is mounted on a rigid chassis, in orange in that figure. The chassis and other mechanical parts such as the holders have been generated by 3D printing from a PLA (polylactic acid) filament by using a domestic desktop 3D printer. The chassis hosts the battery with its board connectors and the electronics of our robotic unit.
2. the *wheels* and *servomotors*: robot movement is provided through two side wheels with power supplied by two continuous rotation servomotors, displayed in the picture. Unlike ordinary motors, servomotors can be individually controlled; they only require the angle of rotation for motion. Rotation is supported through an omni-directional ball caster wheel able to swivel in any direction.
3. a *battery*: a rechargeable power-efficient 3.7V lithium-ion polymer battery is used in our implementation, with the advantage that it has higher specific energy than other lithium batteries.
4. a *boost step-up power*: for the battery to be voltage-compatible, we also include the module *Lithium 134n3p* charger, a built-in charge and discharge power MOS operating at an input voltage in the range 3.7V∼5.5V with output voltage 5V and providing charge and discharge management, temperature control and protection against over-temperature, output over-voltage, short circuit, heavy load over-charge and over-discharge.

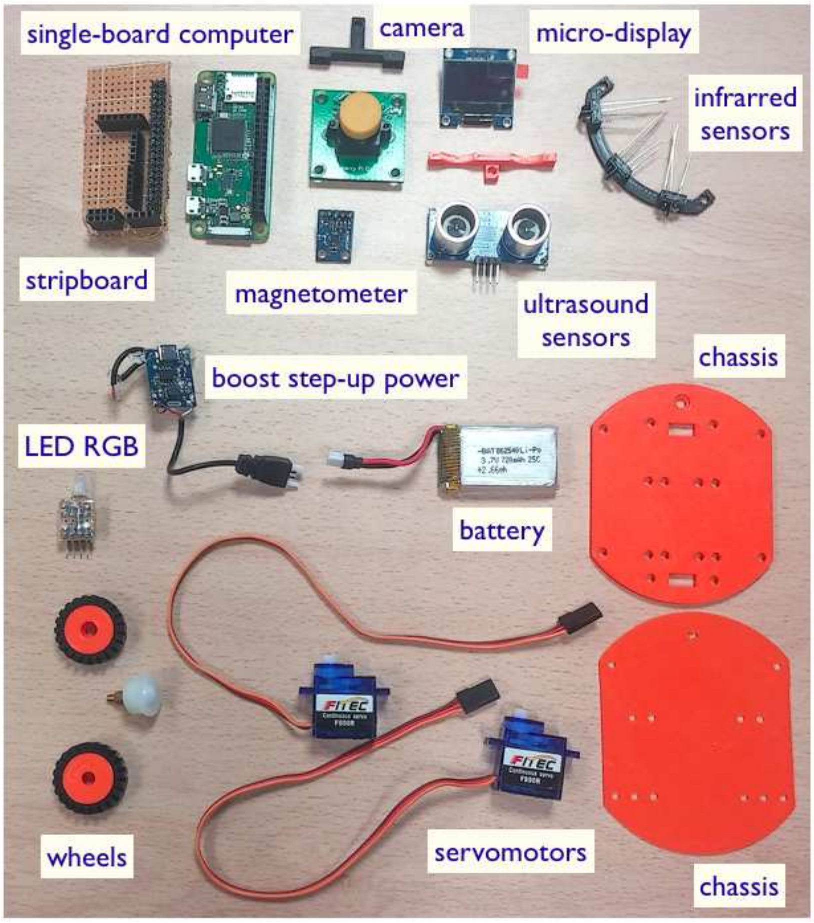

Fig. 1. Main components of our hardware robotic platform for swarm robotics.

5. a *single-board micro-computer*: in our implementation we use the popular micro-computer *Raspberry Pi Zero W*, one of the most affordable and cost-effective micro-computers in the market, with a price of 10US$. This micro-computer comes with a 32-bit *RISC ARMv6Z* architecture, featuring a *Broadcom BCM2835* system on a chip application processor. Its CPU is the *ARM1176JZF-S* core by ARM, running at 1GHz. It also includes a graphical processor unit *Broadcom Video Core IV* running at 250 MHz, with support to Open GL and featuring a H.264/MPEG-4 AVC high-profile decoder and encoder with support to 1080p (high-definition video mode). The system comes with 512 MB (shared with the GPU), 1 micro-USB (direct from the *BCM2835* chip), MIPI camera interface for video input, mini-HDMI at 1080p resolution and composite video for video output, 2 boards via the serial bus

I^2S for audio input, a stereo audio through PWM on GPIO for audio output, and a MicroSDHC non-volatile memory card for data storage.

6. *built-in communication interface*: our single-board chip also provides support for communications via *Bluetooth 4.1* for very short distances (about a range of 10 meters or less) , *802.11n* wireless LAN for wider areas (up to 100 meters), and a FM receiver working in the range 65 MHz to 108 MHz FM bands, all through the *Cypress CYW43438* wireless chip. It also has an unpopulated HAT-compatible 40-pin GPIO header, and composite video and reset headers. These wireless communication options are used for communication and data exchange over short distances among the robots of the swarm and with a central server for tracking purposes.
7. a *stripboard*: used for further connectivity of all electronic components. It is located close to the board chip to gain access to all micro-computer pins.
8. an *ultrasound sensor*: in this work, we use the ultrasound sensor *HC-SR04*, manufactured by *ElecFreaks*. It is an ultrasonic sensor operating at 5V DC that uses sonar to compute the distance to an object. Each *HC-SR04* module includes an ultrasonic transmitter, a receiver and a control circuit, with 4 pins for power, trigger (transmitter), echo (receiver), and ground. Each ultrasound pulse of our sensors operates at a constant frequency of 40 kHz, sending an 8 cycle burst of ultrasound pulses. The sensor captures its echo with signals lasting in the order of milliseconds. The accuracy range of the sensor is about 3 millimeters, with a traveling range of pulses between 2–500 centimeters. The ultrasound sensor is used for collision avoidance with static and dynamic objects (including other robots in the swarm) as well as with the boundaries of the physical 3D environment.
9. a LED RGB: a hand-made diode with RGB lights used to indicate different robot states, such as *active*, *idle*, *sleep*, and others.
10. a *magnetometer*: in this work, we use the triple-axis magnetometer board *HMC-5883L*. This user-friendly compass is a 3.3V max chip with added circuitry to make it 5V-safe logic and power, so that it can be connected to either 3 or 5V microcontrollers. It uses I^2S serial bus for easy interface to communicate. Its internal functioning is based on the anisotropic magnetoresistive (AMR) technology by *Honeywell*, with AMR directional sensors having a full range of ± 8 gauss an a resolution of up to 2 milligauss. The magnetometer is used in this work for global spatial orientation of the robotic units of the swarm regardless their physical environment.
11. *infrared sensors*: used for collision avoidance, scene exploration and navigation throughout the 3D environment. In this work we consider a set of three infrared sensors deployed on a semi-ring holder located in the front of the robot to cover a wider exploration area. One of the IR sensors is located in the middle, and the other two near the corners in the front. They can detect obstacles in the range of 30 centimeters in the sun light. They also come with a variable resistance to adapt the sensors to different short distances.
12. a *mini-camera*: used for image capture and navigation. Our model provides a resolution of 3280×2464 pixels, and support video capture at a frame

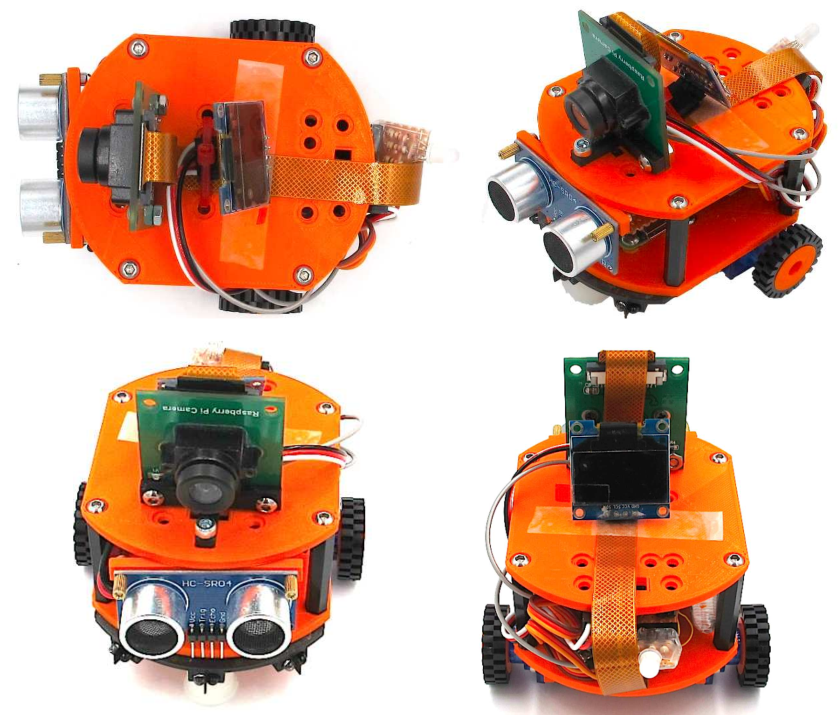

Fig. 2. Four views of the proposed general-purpose hardware robotic platform: (top-left) top view; (top-right) side view; (bottom-left) front view; (botom-right) rear view.

rate of 30 FPS (frames per second) for a resolution of 1080p, 60 FPS at 720p and 90 FPS at 480p.

13. an *OLED micro-display*: a 0.96 inches *SDD1306* chipset with serial connection I^2C, and power consumption of 20mA. It provides a resolution of 128×64 pixels and vision angle of 160 degrees and is mainly used to display relevant information for tracking purposes and as user-interface with the board.

All these components are connected to different I/O pins in a rather standard way. The detailed description of these connections is out of the scope of this paper and will be omitted here to keep the paper to a manageable size.

3.2 Programming framework

There are several operating systems and programming frameworks that can be used for the *Raspberry Pi Zero* micro-computer. Among all possibilities for the operating system, we recommend to use *Raspbian*, a free operating system based on *Debian*, optimised for the *Raspberry Pi* hardware. *Raspbian* comes with over 35,000 packages and is the recommended operating system for normal use on a *Raspberry Pi*. Other popular options are *Snappy Ubuntu*, *Pidora*, *Arch Linux ARM*, *Gentoo Linux*, *FreeBSD*, or *RISC OS Pi*.

Regarding the programming framework, the choice is strongly dependent on the programming language used for coding. In our implementation, we propose to use *Phyton* running on *Wing IDE*. We remark that *Wing IDE* is not directly supported by the *Raspberry Pi*, although it is possible to set up *Wing IDE* on a computer connected to the *Raspberry Pi* to generate and debug *Python* code remotely. Other possibilities include *Kivy*, *GTK+*, *PyQt*, and *Glade*.

4 Support Features for Swarm Intelligence Algorithms

A major ingredient of all SI methods (not only those in this paper) is a population of unsophisticated individuals with the ability to compute their own positions according to some evolution equations and communicate such positions and other information to other members of the swarm. All these relevant features are incorporated in our proposal. We consider a swarm of individuals that are identical replicas of the hardware robotic platform described in previous section. This micro-robot includes all hardware components and features required to implement the previous methods for swarm robotics at full extent:

– Its *Raspberry Pi Zero W* micro-processor is powerful enough to perform all the required computations in real time for all methods described. Although this feature can arguably be accomplished with (cheaper) micro-controllers (e.g., *Arduino*), our proposal is much more powerful. Instead of running one program many times as micro-controllers do, we have a general-purpose full-fledged computer running on Linux and with the ability to run multiple programs in a complex way. Consequently, we have no limitations in terms of the SI algorithms to be implemented, the number of individuals in the swarm (even multiple swarms are easily supported) and the number and complexity of tasks the robotic units are assigned.
– All SI methods require *global positioning*, which is computed with the magnetometer and its built-in digital compass, so that the robot can determine its current position with suitable accuracy for many practical applications.
– All SI methods require *communication capabilities*. Built-in-board *Wifi* and *Bluetooth* interfaces are included in our robots, allowing them to communicate useful information (e.g., position) to other members of the swarm.
– Our robots support *collision avoidance*. This feature is not included in the SI methods, but it is a must for swarm robotics, where robots moving in a real environment can collide with static and dynamic objects. Our robotic units are equipped with several sensors (ultrasounds, infrared, camera), global positioning (magnetometer and digital compass), and communication features (*Wifi*, *Bluetooth*) to avoid collisions with obstacles and other robots.
– Different *sensors* can be used to cope with the specific features and needs required by each SI method, as follows:
  • The PSO method relies on positions, velocities and orientations. Changes in velocity and position can be computed through an accelerometer,

while changes in orientation and rotational velocity can be measured through a gyroscope. These sensors are not currently embedded into the *Raspberry Pi* board, but they can easily be attached to it. In our case, we rely on the servomotors to compute distance and velocity.
- The work in [11] proved that some ultrasound sensors are ideal for the bat algorithm. We propose to use the same sensor in this approach.
- In this paper, we argue that infrared sensors are adequate for the firefly algorithm. Infrared sensors can be used to measure distances, as required by the algorithm. Furthermore, since these sensors can also determine how "bright" the light is, we can go even further with the nature-based metaphor and use them to embed the concepts of brightness and attractiveness by hardware instead of computing them exclusively by software.
- Following the rationale of the previous item, a combination of some of our sensors (possibly including the mini-camera) can also be used for the cuckoo search algorithm.

5 Conclusions and Future Work

This paper introduces a general-purpose hardware robotic platform suitable for swarm robotics. Our approach is based on a careful choice of its main hardware components (computing unit, sensors, communication interfaces) to support the most popular swarm intelligence algorithms by hardware. Our design has also a very flexible and modular architecture so that it can be adapted to different swarm intelligence methods and many applications with minimal (if any) modifications. As an illustration, four examples of popular swarm intelligence methods (particle swarm optimization, firefly algorithm, bat algorithm, and cuckoo search algorithm) are considered in this paper. The most important hardware features of our approach to support such methods (and arguably many other swarm intelligence approaches as well) for swarm robotics are discussed.

Future work includes the consideration of other swarm intelligence approaches with the (possible) addition of extra components to support them. On the other hand, we plan to apply our robotic platform to real-world problems that could benefit from the swarm robotics principles and techniques. Reducing the size while maintaining (or even increasing) the performance of our robotic platform is also part of our plans for future work in the field.

Acknowledgments

This research work has been kindly supported by the project PDE-GIR of the European Union's Horizon 2020 research and innovation programme, Marie Sklodowska-Curie grant agreement No 778035, the Spanish Ministry of Economy and Competitiveness (Computer Science National Program) under grant #TIN2017-89275-R of the Agencia Estatal de Investigación and European Funds FEDER (AEI/FEDER, UE), and the project #JU12 of SODERCAN and European Funds FEDER (SODERCAN/FEDER UE).

References

1. Arkin, C.R.: *Behavior-Based Robotics*. MIT Press, Cambridge, Mass, USA (1998).
2. Arvin, F., Murray, J.C., Licheng S., Zhang, C., Yue, S.: Development of an autonomous micro robot for swarm robotics. *Proc. of IEEE Int. Conf. on Mechatronics and Automation – ICMA'2014*, 635–640 (2014).
3. Bonabeau, E., Dorigo, M., Theraulaz. G: *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, New York (1999).
4. Engelbrecht, A.P.: *Fundamentals of Computational Swarm Intelligence*. John Wiley and Sons, Chichester, England (2005).
5. Faigl, J., Krajnik, T., Chudoba, J., Preucil, L., Saska, M.: Low-cost embedded system for relative localization in robotic swarms. *Proc. of IEEE Int. Conf. on Robotics and Automation - ICRA'2013*, 993–998 (2013).
6. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. *IEEE International Conference on Neural Networks*, Perth, Australia 1942–1948 (1995).
7. Kennedy, J., Eberhart, R.C., Shi, Y.: *Swarm Intelligence*. Morgan Kaufmann Publishers, San Francisco, CA (2001)
8. Sahin, E.: Swarm robotics: from sources of inspiration to domains of application. In: Swarm robotics. *Lecture Notes in Computer Science*, **3342**, 10–20 (2005).
9. Suárez, P., Iglesias, A.: Bat algorithm for coordinated exploration in swarm robotics. *Advances in Intelligent Systems and Computing*, **514**, 134–144 (2017).
10. Suárez, P., Gálvez, A., Iglesias, A.: Autonomous coordinated navigation of virtual swarm bots in dynamic indoor environments by bat algorithm. *Lecture Notes in Computer Science*, **10386**, 176–184 (2017).
11. Suárez, P., Iglesias, A., Gálvez, A.: Make robots be bats: specializing robotic swarms to the bat algorithm. *Swarm and Evolutionary Computation* (in press).
12. Tan, Y., Zheng, Z.Y.: Research advance in swarm robotics. *Defence Technology Journal*, **9**(1) (2013) 18–39.
13. Wagner, I., Bruckstein, A.: Special Issue on Ant Robotics. *Annals of Mathematics and Artificial Intelligence*, **31**(1-4) (2001).
14. Yang, X.S.: Firefly algorithms for multimodal optimization. *Lectures Notes in Computer Science*, **5792**, 169–178 (2009) .
15. Yang, X.S.: Firefly algorithm, stochastic test functions and design optimisation. *Int. Journal of Bio-Inspired Computation*, **2**(2) (2010) 78-84.
16. Yang, X.S.: *Nature-Inspired Metaheuristic Algorithms (2nd. Edition)*. Luniver Press, Frome, UK (2010).
17. Yang, X.S.: *Engineering Optimization: An Introduction with Metaheuristic Applications*. Wiley & Sons, New Jersey (2010).
18. Yang, X.S.: A new metaheuristic bat-inspired algorithm. *Studies in Computational Intelligence*, **284**, 65–74 (2010).
19. Yang, X.S.: Bat algorithm for multiobjective optimization. *Int. J. Bio-Inspired Computation*, **3**(5), 267–274 (2011).
20. Yang, X.S.: Bat algorithm: literature review and applications. *Int. J. Bio-Inspired Computation*, **5**(3), 141–149 (2013).
21. Yang, X.S., Deb, S.: Cuckoo search via Lévy flights. *Proc. World Congress on Nature & Biologically Inspired Computing, NaBIC*, 210–214. IEEE Press (2009).
22. Yang, X.S., Deb, S.: Engineering optimization by cuckoo search. *Int. J. Mathematical Modelling and Numerical Optimization*, **1**(4) (2010) 330-343.
23. Yang, X.S., Gandomi, A.H.: Bat algorithm: a novel approach for global engineering optimization. *Engineering Computations*, **29**(5), 464–483 (2012).