Université Libre de Bruxelles

Institut de Recherches Interdisciplinaires
et de Développements en Intelligence Artificielle

MoCA: a modular RGB color arena for swarm robotics experiments

D. Garzón Ramos, M. Salman, K. Ubeda Arriaza, K. Hasselmann, and M. Birattari

IRIDIA – Technical Report Series

Technical Report No.
TR/IRIDIA/2022-014

November 2022

IRIDIA – Technical Report Series
ISSN 1781-3794

Published by:
IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2022-014

MoCA: a modular RGB color arena for swarm robotics experiments

David Garzón Ramos, Muhammad Salman, Keneth Ubeda Arriaza, Ken Hasselmann, and Mauro Birattari*

IRIDIA, Université libre de Bruxelles, Belgium

Abstract

We present MoCA: an open-source modular system to perform experiments with robot swarms. Robot swarms are self-organized and highly redundant groups of robots that act collectively to achieve a shared goal. The collective behavior of a robot swarm emerges from the interaction that individual robots have with their peers, and with objects in the scenario where they operate. MoCA provides tools to create, simulate, and physically deploy scenarios where robots react to the color displayed by programmable RGB modules. Users of MoCA can create these scenarios—also known as robot arenas—by assembling the RGB modules in polygonal structures that enclose the workspace of the robots. We describe the hardware and software architecture of MoCA, and we point to the repositories that have the resources to build and operate the system; as well as a plug-in to simulate the system in ARGoS3. MoCA has been used to conceive scenarios and missions in two swarm robotics studies: (i) swarms of e-puck robots that operate with control software produced via optimization; and (ii) swarms of e-puck robots that coordinate using indirect communication. The robot platform we target is the e-puck robot and the implementation of MoCA we present here is made to work with this robot.

Keywords: Swarm robotics; modular arena; RGB LEDs; open-source hardware; MoCA.

1 Hardware in context

Swarm robotics [1] research is mainly conducted within controlled laboratory conditions [2]. Researchers design swarms of small educational robots [3, 4, 5] that exhibit specific collective behaviors in simplified *ad hoc* scenarios. These scenarios comprise mostly convex bounded spaces with objects that the robots can perceive, identify, and interact with. The characteristics of the scenario where the robots operate partially condition the set of collective behaviors that a swarm can exhibit. Indeed, the scenario must provide the elements that enable the emergence of a desired collective behavior—for example, a robot swarm that navigates its workspace using color cues requires colored objects to operate correctly.

In this technical report, we present MoCA: an open-source modular system that provides tools to create, simulate, and physically deploy scenarios to experiment with swarms of robots that can react to colored objects. MoCA comprises a series of interconnected small cuboid modules with RGB LEDs that are controlled via a computer—see Figure 1. The size of the RGB modules has been selected to match the size of the e-puck [6]. The RGB modules enable MoCA for displaying a wide variety of color patterns that can be configured and changed in runtime. Users of MoCA can create scenarios to perform experiments with robot swarms by assembling the RGB modules in polygonal structures that enclose the workspace of the robots—also known as *robot arenas*. Thanks to the modularity of the system, the number of RGB modules can be scaled to create arenas of various sizes and shapes. In addition to the physical system, we provide a simulated version of MoCA for ARGoS3 [7]—a state-of-the-art

*Contact: david.garzon.ramos@ulb.be (D.G.R.); mauro.birattari@ulb.be (M.B.)

Figure 1: MoCA: a modular RGB color arena for swarm robotics experiments.

multi-robot simulator largely used in swarm robotics research. We release MoCA—both the hardware and software components—under a GNU General Public License (GPL).

Robot platforms used in swarm robotics research often have cameras that give the robots the capability of perceiving colored objects [8]. Designers of robot swarms commonly conceive experiments in which these objects represent information that is relevant to the task that a swarm must perform—for example, objects that display different colors could indicate regions of the scenario in which the robots should act differently. We identify two types of objects that are used to this purpose: (i) non-programmable simple objects like prints, wooden, or plastic items [9, 10, 11, 12, 13, 14]; and (ii) programmable devices that can modify their characteristics in runtime and/or actively interact with the robots [15, 16, 17, 18]. On the one hand, simple objects are cost-effective and easy to fabricate. However, it is difficult to use them beyond the context of the specific experiment for which they are created. On the other hand, programmable devices requires additional configuration steps in preparing an experimental setup. Yet, they are more versatile, reusable, and enable the design of experiments with time-varying scenarios.

In the swarm robotics literature, authors commonly do not give prominence to the components of the experimental arenas and rather focus on the robots. Indeed, few system have been formally released that enable the fast creation of experimental arenas for robots that perceive colored objects—we focus only in systems that comprise programmable devices. For example, Allwright *et al.* developed SRoCS [16]—a multi-robot construction system in which robots assemble small programmable RGB cubic blocks. The blocks in SRoCS are modules that embed a micro-controller and can communicate with robots by using Near-field Communication (NFC) modules, Xbee radio transceivers, and by changing the color of RGB LEDs. Similarly, Brutschy *et al.* [15] introduced the Task Abstraction Module (TAM). The TAM is a programmable booth-shaped smart device in which robots can enter to interact with it. The TAM can communicate with the robots using infrared transceivers and by displaying different colors using RGB LEDs. It embeds Xbee radio transceivers that enable the communication between TAMs. Systems like SRoCS and the TAM exemplify the usefulness of introducing objects that can display colors in the design of swarm robotics experiments. By enabling an easy and flexible configuration of the workspace of the robots, these devises be relevant tools to devise and perform high impact experiments [19].

Devices like SRoCS and the TAM typically are not intended to create the structures that enclose the robots in their workspace—rather, they focus on enabling individual interaction points for the robots. With MoCA, on the contrary, we aim to facilitate the creation of complete robot arenas that display color patterns, which can varied during the execution of an experiment. MoCA has the attributes to fulfill the objective for which we conceived it: (i) it is modular and can be assembled in various shapes

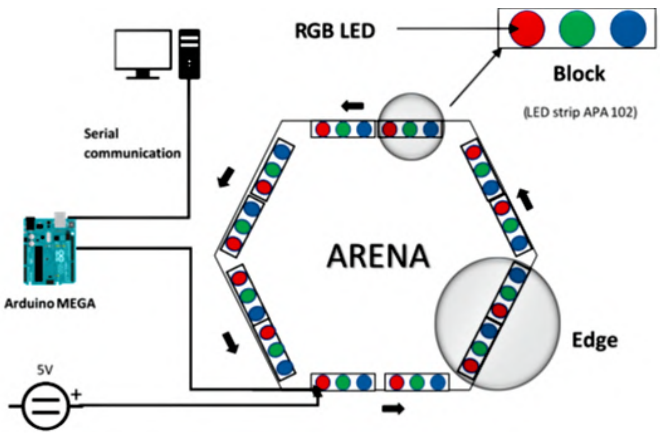

Figure 2: `MoCA`'s hardware components.

and sizes; (ii) it can display a wide diversity of color patterns that can vary in runtime. We designed `MoCA` to be relatively less complex than systems like SRoCS and the TAM—both in the functional specifications and hardware. By doing so, we provide a solution that is cost-effective and easy to reproduce, configure, and extend.

In the rest of the technical report, we describe first the hardware and software components of `MoCA`. After, we detail the instructions to build and operate the system. In the end, we highlight two studies that demonstrate the functioning of `MoCA`.

2 `MoCA`: hardware and software

`MoCA` comprises three major hardware components: (i) the RGB modules; (ii) an Arduino MEGA microcontroler; and (iii) a desktop computer. Figure 2 shows a graphical representation of the hardware architecture of `MoCA`.

The RGB modules are entities whose function is to display color lights in the workspace of the robots. A module is, in practice, a wooden box with electronic connectors that encloses a segment of 12 RGB LEDs from a LED strip APA102. The front of the module has a Plexiglas diffuser that homogenizes the light emitted by the RGB LEDs and creates a surface with uniform color. In this sense, the diffuser avoids high saturation light that could possibly saturate the camera of the robots. `MoCA`'s RGB modules can be produced with a laser cut machine—the design files are available in `MoCA`'s hardware repository—see Section 6. Figure 3 shows an assembly schematic of the modules.

The Arduino MEGA microcontroller is the interface between the control software running in the computer and the LED strip APA102. The selection of the MEGA line is arbitrary, and in practice, it can be replaced by any other Arduino line. The Arduino MEGA microcontroller receives desired configurations for individual RGB modules and translates these configurations into commands for the LED strip. The firmware of the microcontroller has been developed using the libraries FastLED and ArduinoJson. FastLED has the required commands to communicate with the LED strip and ArduinoJson is an interpreter for the JSON string with which the user communicates the desired configuration for `MoCA`. The firmware for the Arduino microcontroller is available in `MoCA`'s software repository—see Section 6.

In addition to the firmware of the Arduino microcontroler, the rest of the core software componentes of `MoCA` run in a host computer. The host computer runs three major software components: (i) an HTTP API; (ii) the controller of `MoCA`; and (iii) a serial interface. Figure 4 shows a graphical representation of the software architecture of `MoCA`. All components have been developed and tested using Python 3. Users of `MoCA` interact with the HTTP API via an HTTP client. The HTTP API is a web server based in the aiohttp library and its main purpose is to make available the functionalities of `MoCA` in the form of services. The communication between the HTTP client and the server is done via JSON messages. In this sense, the desired configuration (color setup) of `MoCA` must be structured in a JSON string. The controller of `MoCA`, the second major software component, interprets and translates the desired configuration appropriate instructions for the Arduino microcontroller. Finally, the serial

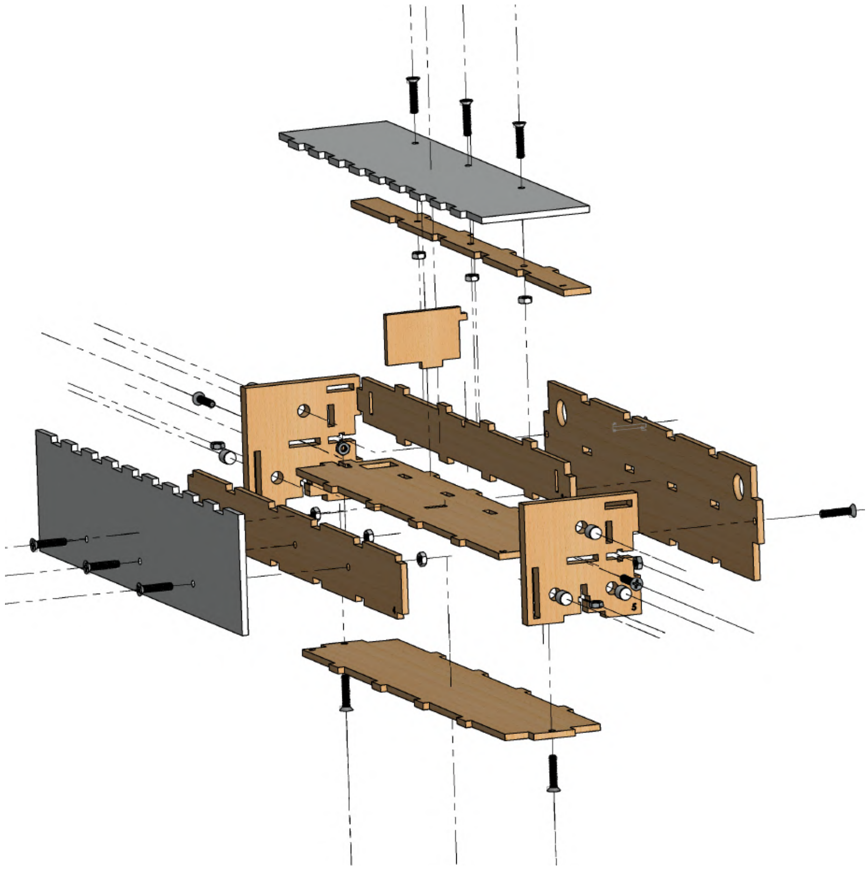

Figure 3: Assembly schematic of MoCA's RGB modules

interface, the third component, interfaces with the Arduino microcontroller and communicates with the device via serial communication. The implementations of the HTTP API, the controller of MoCA, and the serial interface are available in MoCA's software repository—see Section 6.

3 Operation instructions

The modular nature of MoCA enables the assembly and operation of experimental arenas with various shapes. The experimental arenas devised with MoCA typically have a polygonal shape. The RGB modules are arranged in walls. A number of walls then can be arranged in the desired experimental arena. The RGB modules are the unitary blocks that form the edges (the walls) of the polygon. The edges are interconnected by wooden joints that establish the aperture angle between to edges of the polygon. The design files for the wooden joints are also available in the hardware repository of MoCA—see Section 6.

Hardware setup: An experimental arena is then a serial arrangement of interconnected RGB modules. In a convex arrangement, the first and last module are not electronically interconnected. The first block of the arena is that connected to the Arduino microcontroller and to the power supply. The Arduino microcontroller is connected to the computer via a USB port. For an arrangement of 24 RGB modules, we use a 5V power supply with the capacity of delivering 1.5A.

Software setup: The Arduino microcontroller automatically starts and wait for instructions once the device is powered up. The HTTP server must be started in the computer. The user can send HTTP requests to set the color of the RGB modules in the experimental arena. The request is structured in a JSON string that contains the desired configuration. The HTTP server interprets the JSON structure, translates it in a series of instructions that can be interpreted by the Arduino microcontroller, and delivers these instructions via a serial connection. After, the Arduino microcontroller sends the appropriate commands to the RGB modules.

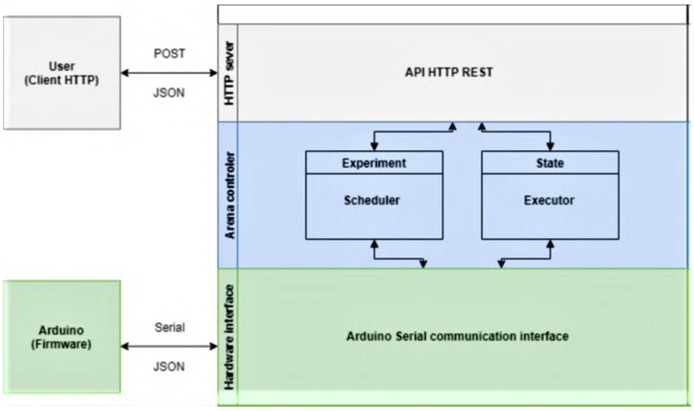

Figure 4: `MoCA`'s sofware architecture.

`MoCA`'s language: We conceived a structured language that enables the definition of the color configuration of `MoCA`. The language comprises six entities that give control of the configuration of `MoCA` with varying granularity: LED—the individual LEDs and the minimum entity to which it is possible to set a color; BLOCK—arrangements of LEDs in the form of RGB modular blocks; EDGE—arrangements of RGB modular blocks in to edges (walls) of the experimental arena; ARENA—arrangements of edges into an experimental arena; STATE—the time-dependent states of an experimental arena; EXPERIMENT—the configuration of a time-regulated experiment in which time-dependent states are processed one after the other. `MoCA`'s language enables the definition of the color configuration of the RGB modules by individual LEDs, RGB modules, edges (walls) of the arena, or the arena as a whole. In this sense, it gives an expressive means of representing the possible configuration of the RGB modules in `MoCA`. While using `MoCA`, it is possible to combine expressions at different levels to indicate a desired configuration. For example, in a single instruction, one could define that a whole experimental arena of twelve modules turns green and only the fourth module turns red. Indeed, the outcome of an instruction built with `MoCA`'s language depends in the way that the different entities are instantiated and combined. A comprehensive instructive on how to use `MoCA`'s language is given in `MoCA`'s software repository; see Section 6.

4 `MoCA` plug-in for ARGoS3

We have produced a software plug-in to simulate `MoCA` in ARGoS3—a widely used swarm robotics simulator. The plug-in enables instantiating ARENA entities in ARGoS3 experiments. Likewise its real counterpart, the experimental arenas simulated in ARGoS3 can be controlled with various degrees of granularity. In particular, the plug-ing provides methods to define and operate regular polygonal experimental arenas by setting the color of the light emitted by LED, EDGE, and ARENA entities. The RGB modules in `MoCA`'s plugin for ARGoS3 are a modified version of the ARGoS3' generic *box entity*—which already contains an ARGoS3 generic *LED entity* by default. `MoCA`'s EDGE and ARENA entities are logical arrangements of ARGoS3' box entities.

Robots in ARGoS3 perceive the LEDs of the `MoCA`'s RGB modules. In this sense, `MoCA` can be perceived by robots that use ARGoS3 generic *omnidirectional camera entity* or a modifed version of it—for example, that of the e-puck plug-in for ARGoS3 [20]. Instructions on how to operate and instantiate `MoCA`'s plug-in for ARGoS3 are given in the plug-in repository—see Section 6. The current version of `MoCA`'s plug-in has been developed and tested with ARGoS3 beta 48. Figure 5 shows a simulated version of `MoCA` in ARGoS3.

5 Demonstrations

The usability of `MoCA` has been demonstrated in experiments on the design of control software for robot swarms. Both the physical implementation and the plug-in for ARGoS3 have been used to conduct two

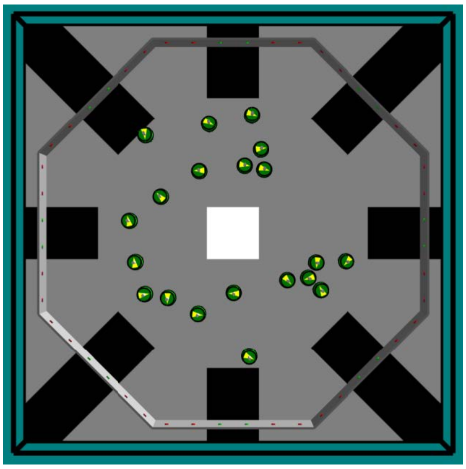

Figure 5: Simulation of MoCA in ARGoS3.

swarm robotics studies: (i) swarms of e-puck robots that operate with control software produced via optimization [8]; and (ii) swarms of e-puck robots that coordinate using indirect communication [21].

In the first study, Garzón Ramos and Birattari [8] used MoCA to devise three experimental arenas for a swarm of twenty e-pucks endowed with an omnidirectional camera: FORAGING, AGGREGATION, and STOP. In FORAGING, the robots carry abstract objects between sources of food and the nest of the robots. The experimental arena comprised twenty-four RGB modules arranged in square shape. In this mission, the color of the RGB modules of MoCA indicated the location and quality of the sources of food and also the location of the nest. In AGGREGATION, the robots differentiate between two possible aggregation areas. The experimental arena comprised twenty-four RGB modules arranged in an hexagonal shape. In this mission, the color of the RGB modules of MoCA indicated the location of the aggregation areas. In STOP, the robots move until a signal is given in their workspace, and they must stop moving and standstill afterward. The experimental arena comprised twenty-four RGB modules arranged in an octagonal shape. In this mission, the change of color of an RGB module of MoCA gives the signal to the robots at a given time. Experiments were condudcted in simulation and with physical robots.

In the second study, Salman *et al.* [21] used MoCA to devise two experimental arenas for a swarm of five e-pucks endowed with an omnidirectional camera and with a module to deploy artificial pheromone: FORAGING and TASKING. In FORAGING, the robots carry abstract objects between sources of food and the nest of the robots. In this mission, contrary to the first study, the color of the RGB modules of MoCA only indicated the location of the nest. In TASKING, the robots performed a number of abstract tasks in a large set of available workstations. In this mission, the color of the RGB modules of MoCA indicated the location of the workstation on which robots could execute the abstract tasks. In the two cases, the experimental arena comprised twenty RGB modules arranged in a rectangular shape.

6 Resources and repositories

We release MoCA—both the hardware and software components—under a GNU General Public License (GPL). MoCA's **hardware repository**[1] provides the design files to laser-cut the pieces to assemble the RGB modules and the joints for the experimental arenas. MoCA's **software repository**[2] provides the firmware for the Arduino microcontroller and the Python 3 implementations of the HTTP API, the controller of MoCA, and the serial interface. It also provides an extensive description of the language to setup and operate MoCA. MoCA's **plug-in repository**[3] provides the implementation of MoCA to be used within the simulator ARGoS3.

[1] https://github.com/demiurge-project/moca-hardware
[2] https://github.com/demiurge-project/moca-driver
[3] https://github.com/demiurge-project/argos3-arena

Acknowledgements

The project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 681872) and from Belgium's Wallonia-Brussels Federation through the ARC Advanced Project GbO–Guaranteed by Optimization. MB acknowledges support from the Belgian *Fonds de la Recherche Scientifique*—FNRS. DGR acknowledges support from the Colombian Ministry of Science, Technology and Innovation—Minciencias.

References

[1] Marco Dorigo, Mauro Birattari, and Manuele Brambilla. Swarm robotics. *Scholarpedia*, 9(1):1463, 2014.

[2] Marco Dorigo, Guy Theraulaz, and Vito Trianni. Reflections on the future of swarm robotics. *Science Robotics*, 5:eabe4385, 2020.

[3] Nadia Nedjah and Luneque Silva Junior. Review of methodologies and tasks in swarm robotics towards standardization. *Swarm and Evolutionary Computation*, 50:100565, 2019.

[4] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[5] Melanie Schranz, Martina Umlauft, Micha Sende, and Wilfried Elmenreich. Swarm robotic behaviors and current applications. *Frontiers in Robotics and AI*, 7:36, 2020.

[6] Francesco Mondada, Michael Bonani, Xavier Raemy, Jim Pugh, Christopher Cianci, Adam Klaptocz, Stéphane Magnenat, Jean-Christophe Zufferey, Dario Floreano, and Alcherio Martinoli. The e-puck, a robot designed for education in engineering. In Paulo Gonçalves, Paulo Torres, and Carlos Alves, editors, *ROBOTICA 2009: Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, pages 59–65, Castelo Branco, Portugal, 2009. Instituto Politécnico de Castelo Branco.

[7] Carlo Pinciroli, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews, Eliseo Ferrante, Gianni A. Di Caro, Frederick Ducatelle, Mauro Birattari, Luca Maria Gambardella, and Marco Dorigo. ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, 6(4):271–295, 2012.

[8] David Garzón Ramos and Mauro Birattari. Automatic design of collective behaviors for robots that can display and perceive colors. *Applied Sciences*, 10(13):4654, 2020.

[9] Gianpiero Francesca, Manuele Brambilla, Arne Brutschy, Vito Trianni, and Mauro Birattari. AutoMoDe: a novel approach to the automatic design of control software for robot swarms. *Swarm Intelligence*, 8(2):89–112, 2014.

[10] Gianpiero Francesca, Manuele Brambilla, Arne Brutschy, Lorenzo Garattoni, Roman Miletitch, Gaëtan Podevijn, Andreagiovanni Reina, Touraj Soleymani, Mattia Salvaro, Carlo Pinciroli, Franco Mascia, Vito Trianni, and Mauro Birattari. AutoMoDe-Chocolate: automatic design of control software for robot swarms. *Swarm Intelligence*, 9(2–3):125–152, 2015.

[11] Miquel Kegeleirs, David Garzón Ramos, and Mauro Birattari. Random walk exploration for swarm mapping. In Kaspar Althoefer, Jelizaveta Konstantinova, and Ketao Zhang, editors, *Towards Autonomous Robotic Systems: 20th Annual Conference, TAROS 2019*, volume 11650 of *Lecture Notes in Computer Science*, pages 211–222, Cham, Switzerland, 2019. Springer.

[12] Gaëtan Spaey, Miquel Kegeleirs, David Garzón Ramos, and Mauro Birattari. Evaluation of alternative exploration schemes in the automatic modular design of robot swarms. In Bart Bogaerts, Gianluca Bontempi, Pierre Geurts, Nick Harley, Bertrand Lebichot, Tom Lenaerts, and Gilles Louppe, editors, *Artificial Intelligence and Machine Learning: BNAIC 2019, BENELEARN 2019*, volume 1196 of *Communications in Computer and Information Science*, pages 18–33. Springer, Cham, Switzerland, 2020.

[13] Simon Jones, Alan Winfield, Sabine Hauert, and Matthew Studley. Onboard evolution of understandable swarm behaviors. *Advanced Intelligent Systems*, 1(6):1900031, 2019.

[14] Eduardo Castello, Tomoyuki Yamamoto, Fabio Dalla Libera, Wenguo Liu, Alan F. T. Winfield, Yutaka Nakamura, and Hiroshi Ishiguro. Adaptive foraging for simulated and real robotic swarms: the dynamical response threshold approach. *Swarm Intelligence*, 10:1–31, 2016.

[15] Arne Brutschy, Lorenzo Garattoni, Manuele Brambilla, Gianpiero Francesca, Giovanni Pini, Marco Dorigo, and Mauro Birattari. The TAM: abstracting complex tasks in swarm robotics research. *Swarm Intelligence*, 9(1):1–22, 2015.

[16] Michael Allwright, Weixu Zhu, and Marco Dorigo. An open-source multi-robot construction system. *HardwareX*, 5:e00050, 2019.

[17] Nithin Mathews, Anders Lyhne Christensen, Rehan O'Grady, Francesco Mondada, and Marco Dorigo. Mergeable nervous systems for robots. *Nature Communications*, 8(1):439, 2017.

[18] Shervin Nouyan, Roderich Groß, Michael Bonani, Francesco Mondada, and Marco Dorigo. Teamwork in self-organized robot colonies. *IEEE Transactions on Evolutionary Computation*, 13(4):695–711, 2009.

[19] Lorenzo Garattoni and Mauro Birattari. Autonomous task sequencing in a robot swarm. *Science Robotics*, 3(20):eaat0430, 2018.

[20] Lorenzo Garattoni, Gianpiero Francesca, Arne Brutschy, Carlo Pinciroli, and Mauro Birattari. Software infrastructure for e-puck (and TAM). Technical Report TR/IRIDIA/2015-004, IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, 2015.

[21] Muhammad Salman, David Garzón Ramos, Ken Hasselmann, and Mauro Birattari. Phormica: photochromic pheromone release and detection system for stigmergic coordination in robot swarms. *Frontiers in Robotics and AI*, 7:195, 2020.