

## GRASPion: an Open-Source, Programmable Brainbot for Active Matter Research

F. Novkoski,<sup>1, a)</sup> M. Mélard,<sup>2</sup> M. Delens,<sup>2</sup> F. Wéry,<sup>2</sup> M. Noirhomme,<sup>2</sup> J. Pande,<sup>3</sup> A. Maier,<sup>4</sup> A. S. Smith,<sup>1,5</sup> and N. Vandewalle<sup>2</sup>

<sup>1)</sup>*PULS, Institute for Theoretical Physics, FAU Erlangen-Nurnberg, 91058, Erlangen, Germany*

<sup>2)</sup>*GRASP, Institute of Physics B5a, University of Liège, B4000 Liège, Belgium*

<sup>3)</sup>*Department of Physical and Natural Sciences, FLAME University, Pune, India*

<sup>4)</sup>*Pattern Recognition Lab, Department of Computer Science, FAU Erlangen-Nurnberg, Erlangen, Germany*

<sup>5)</sup>*Group for Computational Life Sciences, Division of Physical Chemistry, Ruđer Bošković Institute, Zagreb 10000, Croatia*

(Dated: 3 April 2026)

We present the GRASPion, a compact, open-source bristlebot designed for the controlled study of active matter systems. Built around an Arduino-compatible board and modular 3D-printed components, the GRASPion combines ease of use, programmability, and mechanical versatility. It features dual vibrating motors for self-propulsion, integrated sensors for local interaction, and customizable firmware enabling various motion modes, from ballistic to diffusive regimes. The robot is equipped with onboard IR communication, color and proximity sensors, and a magnetometer, allowing for real-time interaction and complex collective behaviors. With a runtime exceeding 90 minutes and reproducible fabrication, the GRASPion provides a robust and scalable platform for both educational and research applications in out-of-equilibrium physics. This article details the mechanical and electronic design and software architecture of the GRASPion, and illustrates its capabilities through prototypical experiments relevant to active matter, such as diffusive motion, synchronization and phototaxis.

---

<sup>a)</sup>Corresponding author, email: filip.novkoski@fau.de

## I. INTRODUCTION

The field of active matter studies the properties and dynamics of large numbers of individual agents that draw energy from their environment and convert it into self-propulsion or interactions with others. This typically leads to out-of-equilibrium states, most notably collective motion, some common examples of which are observed in biological systems such as flocks of birds or schools of fish. The field has drawn large theoretical attention, but in recent years, controlled experimental studies have also developed, particularly in the contexts of dry granular matter, colloidal suspensions, and robotics.

Among the various experimental platforms, centimeter-scale bristlebots have emerged as a versatile and accessible model system for active matter research. Locomotion in these robots is typically achieved through a vibrating motor that transfers momentum to elastic bristles attached to the body, converting vibration into directed motion. Early implementations relied on 3D-printed asymmetric bodies placed on vibrating plates, producing chiral motion via frictional asymmetries, but required external driving from a mechanical shaker<sup>1</sup>. The introduction of self-contained commercial units, such as the Hexbug®, made it possible to conduct autonomous experiments at low cost<sup>2,3</sup>, while custom-built variants<sup>4-6</sup> enabled specific designs for physics-oriented studies. However, because active matter physics focuses not only on individual propulsion but also on interactions between agents, the need arose for programmable bristlebots capable of sensor-mediated interactions and reproducible behaviors. Several platforms, including the Swarmodroid<sup>7</sup>, the bug-like M5Stack BugC<sup>8</sup>, and the brainbot<sup>9</sup>, addressed some of these requirements, but limitations persisted in achieving fully controlled, directed and fast motion.

Today, a broader technological need is emerging in the field of active matter: experimental capabilities must keep pace with increasingly sophisticated theoretical and numerical models. Figure 1a highlights a trend among existing bristlebot platforms, showing a positive correlation between achievable speed and available onboard computational power. This reflects a growing demand for faster and more intelligent active particles, capable of extensive data collection, high responsiveness, and adaptive behaviors, including onboard machine learning and real-time decision-making. Such advances would open new avenues for studying complex collective dynamics and emergent phenomena at the interface of physics, robotics, and computation.

In this paper, we present an easy-to-use, compact in size Arduino-based bristlebot, with easily replaceable 3D-printed parts and open-source firmware. The robot, termed the GRASPion, can

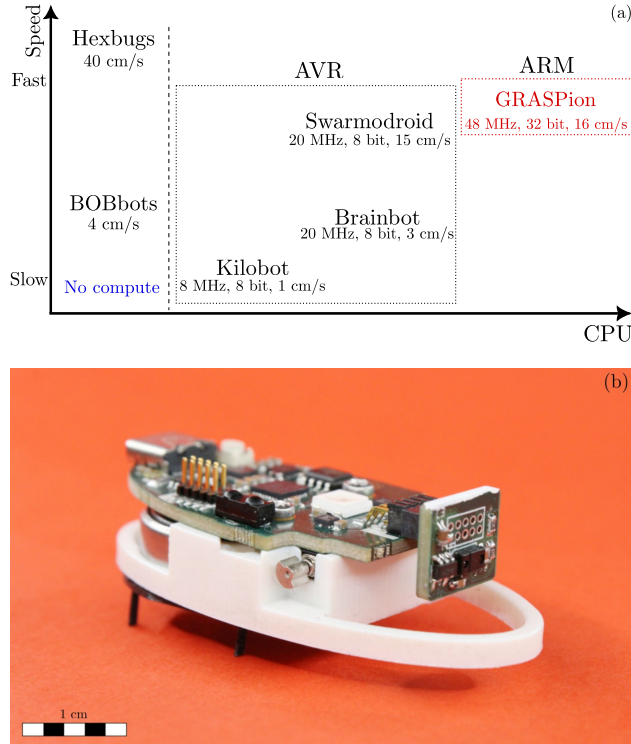


FIG. 1. Top: The evolution of speed and processing power of different varieties of bristlebots. Bottom: Photo of the GRASPion, a fully programmable bristlebot. Photo credit: GRASP, University of Liège

be remotely controlled, can execute preprogrammed motion (including random diffusive behavior), and is easily rechargeable, with a runtime exceeding 90 minutes. It is designed for rapid deployment in both teaching and research, enabling controlled experiments in active matter while supporting flexible extensions via its modular mechanical, electronic, and software architecture.

In the following sections, we will describe the main components of the bot, including a technical description of the mechanical and electronic components, as well as provide several use cases of the bots, with the accompanying code.

## II. BOT DESCRIPTION

A representative image of the GRASPion is presented in Fig. 1b. The robot features an elliptical footprint, with overall dimensions of  $60 \times 30 \times 15 \text{ mm}^3$  and a total mass of 17 g, including all components. The chassis (white in the figure) serves as the main structural element, onto which the electronic circuit board is mounted, and the legs are attached underneath. The legs are a critical functional component, as they enable self-propulsion through frictional interaction with the

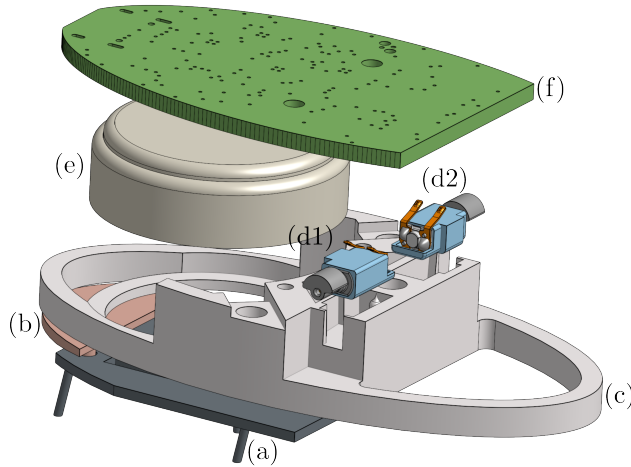


FIG. 2. Exploded view of all bot parts: (a) the PLA leg plate, (b) wedge plate, (c) the ABS printed body, (d) the two vibrating motors with spring contacts, (e) the lithium ion battery, (f) the bot circuit board.

substrate. Their geometry and material properties play a central role in determining the efficiency and reliability of the robot’s locomotion. The only electromechanical elements of the GRASPion are two vibration motors, embedded directly within the chassis, which serve as the actuation mechanism driving the motion.

The intelligence of the GRASPion stems from its onboard electronic circuit. The uploaded software enables full control over the robot’s motion by independently addressing each motor via dedicated drivers. Additionally, the microcontroller processes data from the onboard sensors, allowing the bot to perform real-time decision-making based on environmental inputs. The entire system is powered by a rechargeable lithium-ion battery, ensuring autonomous operation over extended durations.

With a general overview of the bot given, we will first focus on the mechanical parts in Section II A, followed by a more in-depth discussion of the circuit in Section II B, and finally concluding with the final layer, the bot firmware in Section II C.

## A. Mechanical parts

The body of the bot is of an ellipsoidal shape with major and minor axes of length 60 and 30 mm, respectively, with the total height of the body being 9 mm, without the legs. The body is 3D printed out of ABS plastic, with the STL files and print settings for all parts available through a

public repository<sup>10</sup>. The body can be freely modified to accommodate experimental requirements, such as to enable physical connections between bots, break body symmetry, add chirality, or transform into a purely circular shape (the corresponding circular circuits can be made available upon request). The complete view of all parts of the bot is shown in Figure 2. The body has a total mass of 6 g, and has been optimized to reduce weight as much as possible while retaining structural integrity. The only required post-printing step is tapping the holes that hold the circuit and legs to the body.

In order to induce motion, the bot relies on four legs, printed out of PLA plastic, with a diameter of 0.8 mm, and a  $12^\circ$  angle with respect to the vertical. The leg angle has a strong impact on the locomotion of the bot, and can be varied in order to serve different purposes. The legs are printed as part of a leg-plate (part a in Fig. 2) which attaches to the body of the bot through two M2 screws, making them easily replaceable in case of breaking. We have found that it is crucial that the leg plates are printed out one at a time (not multiple legs in parallel layer by layer) in order to avoid weak layer adhesion on the legs. Between the leg-plate and the body, a wedge (part b in Fig. 2) is added in order to provide a forward pitch to the bot, allowing for a more directed and controlled forward motion. The parameters of the legs have been carefully chosen in order to achieve reproducible and well-controlled bot trajectories. In addition to cylinder legs, a rectangular form is also provided, which enhances the durability of the legs as well as bot speed, at a slight cost of trajectory control.

As mentioned above, the bot is driven by two vibrating motors (VZ3TH8B171700L), which are enclosed in two corresponding compartments in the body of the bot, as displayed in Fig. 2. The motors use spring contacts to receive power, meaning that they are detachable from the circuit, allowing for easy replacement. The applied voltage and its polarity to each motor, and thus the speed and rotation direction, can be controlled directly through the firmware (see below), with the motors being independent from each other.

## **B. Electronics**

The control over the bot's motion, as well as all its other capabilities, comes from the onboard circuit. The onboard circuit is based around an Adafruit QtPy SAMD21 clone, which relies on the ARM Cortex M0+ processor. A custom firmware is flashed a single time onto a low-power AVR coprocessor (AVR32DD20), which handles all low-level processes, including charging, IR recep-

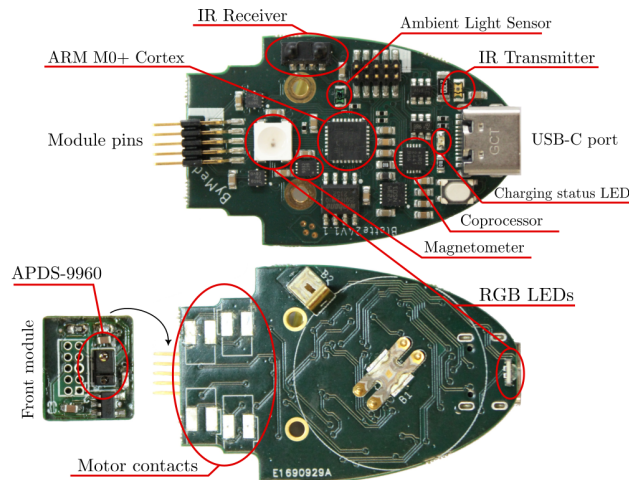


FIG. 3. Top and bottom of the GRASPion circuit board. On the top circuit, we find most of the components, including the M0+ Cortex processor, an RGB LED, and the USB-C socket. On the bottom are located the contacts for the motors, which are connected through their spring contacts once the board is mounted. Additionally, we find the battery contact, along with an additional rear RGB LED. On the other end, we find the pins for the detachable front module, which contains the APDS-9960 color and proximity sensor.

tion, and transmission. This allows a higher-level programming of the bot through the standard Arduino IDE and the accompanying libraries, using the onboard USB-C port. The GRASPion is thus currently the only Arduino-based bristlebot, making its deployment and control fast and easy. The bot is powered by a rechargeable lithium-ion coin battery, while an onboard battery charger covers battery charging, power management, full battery protection, and power PATH. The battery is charged through the USB-C port, allowing it to be connected to a docking station. The onboard battery charger is fully accessible via the coprocessor (chosen for easy use on the user end), which is then accessible from the M0+ Cortex, allowing access in the Arduino code.

The circuit also comes equipped with:

- an IR receiver and transmitter,
- a 3-axis magnetometer,
- two Neopixel RGB LEDs, one on top and one at the back,
- a Flash memory (2M-byte),
- an ambient light sensor mounted on top, and

- a front module that can be exchanged and which currently contains:
  - a proximity sensor,
  - a color detector (RGB), and
  - a gesture sensor (left, right, up, down).

The last of the listed components, the module, is a small add-on circuit board, as seen in Figure 3. While it currently contains the above-mentioned detectors, this can be customized and exchanged for other types of sensors, such as an accelerometer, for example. Being based on the I2C protocol, it can reach kilohertz speeds of sensing. This add-on interface also serves as the contact point for a Bluetooth module that is currently under testing, and which will allow for a more robust communication protocol between the bots as compared to IR. Specifically, it will allow communication of large numbers of bots at the same time with easy identification of individuals, and large spatial range ( $\sim 10$  m). The complete pinout and design of the circuit are available on the public repository<sup>10</sup>.

### C. Software

As mentioned previously, the bot is completely controlled through standard Arduino code, meaning it requires only knowledge of C++. A complete guide on which libraries are needed for the programming of the board, as well as which configuration should be used in the Arduino IDE in order to do it, is given in the repository. An additional advantage is that, through the use of a USB charging hub, it is possible to bulk flash large numbers of bots at once, with the code needed for this being made available as well.

Within the main loop of the code, access to all of the sensor data is available, as well as direct control over the two motors. This makes it extremely simple to direct the bot with a pre-programmed motion, make it react to its environment, or simply remote control it. It is also possible to induce random motion (through the use of the random C function) – however, for this a seed has to be given, which is done through the use of the analog pin on the IR sensor. Below, several examples are given for the bot motion, with references to the accompanying code.

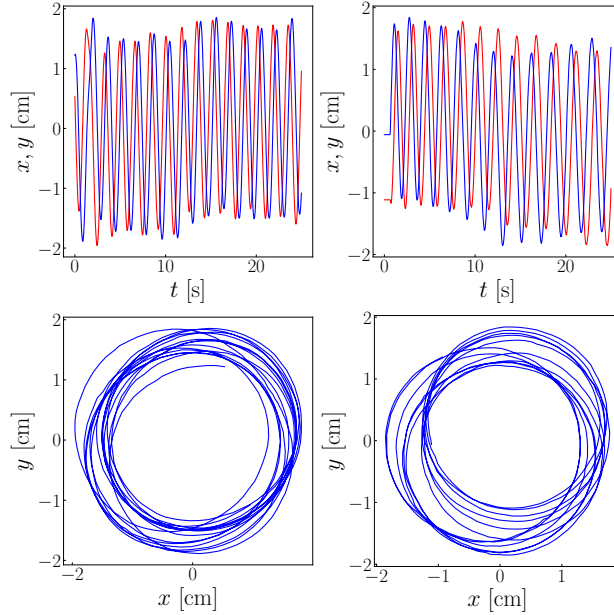


FIG. 4. Top: The  $x$  and  $y$  coordinates of the bot center of mass as it turns clockwise (top left) and counter-clockwise (top right). Bottom: The trajectory in the  $xy$  plane for clockwise (left) and counter-clockwise (right) motion.

### III. EXAMPLES OF USE CASES

In this section, we present three simple use cases of bot motion. All of the code can be found in the example directory of the repository.

#### A. Simple remote control

Using the IR receiver on the bot board, we are able to remotely control it using a standard remote, as is used for television sets, for example. The remote control emits an IR signal, which the bot decodes into hexadecimal values and can then easily be handled in the Arduino code.

The essential example of this is remotely controlling the motion of all the bots at the same time. In order to turn the bot, we need to power only one of the motors (turning on the left motor makes the bot turn right, and vice versa). In the given code, the function `buzCw` takes two arguments, the first one corresponding to a power level, with allowed values from 0 to 127, and the second to the pin of the motor we wish to turn on, either `BUZL_PIN` for the left, or `BUZR_PIN` for the right motor, which are defined at the beginning of the code base on the pinout. For completeness, a secondary

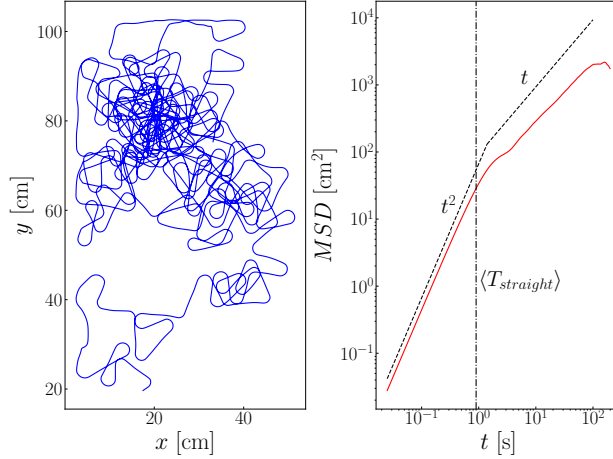


FIG. 5. Left: Sample of the trajectory of a bot running the diffusive algorithm. Right: The mean square displacement of the full trajectory, with a clear transition from the ballistic to the diffusive regime.

function `buzCcw` is also present, turning the motor in the counter-clockwise direction; however, this has the same physical consequence as `buzCw`.

Thus, to turn, we first have to call the function `buzStop(0)` to turn off all motors, and then simply call `buzCw` with the appropriate arguments. Already, this type of basic motion allows for accessible studies of chiral motion and chiral active particles, providing immediate experimental relevance, as well as opening up the possibility of controlled and programmable chiral particles.

In Figure 4, we plot the measured  $x$  and  $y$  coordinates of the bot's center of mass both for the left and right turns for multiple periods of rotation. Essentially, if only one motor is constantly on, the bot will simply rotate around the corresponding front leg. As we can see, this rotation is consistent over multiple periods, with very slight drift ( $\sim 5$  mm), and potentially allows for studies of chiral particles.

## B. Diffusive motion

Using the above functions, we can easily implement diffusive motion. The algorithm for this is to move straight for a time  $T_{straight}$ , which is chosen at random uniformly in the interval of [400, 1200] ms, after which the bot turns either left or right with a 50% chance of either, and it once again turns for a random amount of time  $T_{turn}$  chosen in the same interval as  $T_{straight}$ . It is important for the random motion that we seed it properly. This is done through the use of the native Arduino `randomSeed` function, on the pin A0, which corresponds to the light sensor. In

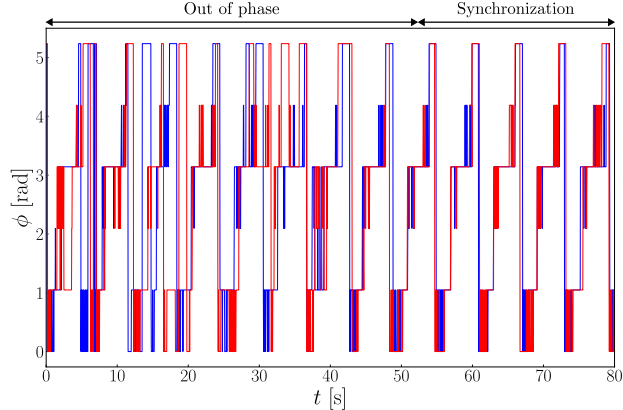


FIG. 6. The phases of two bots with 6 discrete states interacting through the prescribed discrete Kuramoto procedure. While they are out of sync initially, we see that the two oscillators synchronize both in phase and in frequency.

order to achieve straight motion, we simply call `buzCw` for both driver pins to turn both motors on.

The above algorithm has been implemented in the `launchRT` function provided in the example code, and is activated remotely through one of the buttons on the control. By recording and tracking the bot, we can recover its trajectory, a part of which is shown in the left part of Figure 5. While it is difficult to see from the trajectory itself that this motion is diffusive, it becomes much clearer once we consider the mean square displacement (MSD) in the right part of Fig. 5. Here we see that at short time scales the motion is ballistic and the MSD grows with  $\sim t^2$ , while at longer scales, after a transition, it grows with  $t$  as expected for classical diffusion.

### C. Color synchronization

Beyond self-propulsion, interaction plays a crucial role in the study of active matter. One of the emergent effects that comes out of interactions of many individual agents is synchronization, as exemplified by the Kuramoto model of interacting phase oscillators<sup>11,12</sup>. Through a sufficiently strong interaction, individual oscillators can synchronize, resulting in a global oscillating frequency. When this oscillatory behavior is coupled with self-propulsion, it leads to so-called swarmalators, which can result in a wide variety of moving patterns<sup>13</sup>.

Inspired by this, we have implemented a phase oscillatory behavior and colour detection in the bot, with the example code found in the public repository. The onboard RGB LEDs are made to go through a given cycle of discrete colours, 6 of them in this example, which can be made to

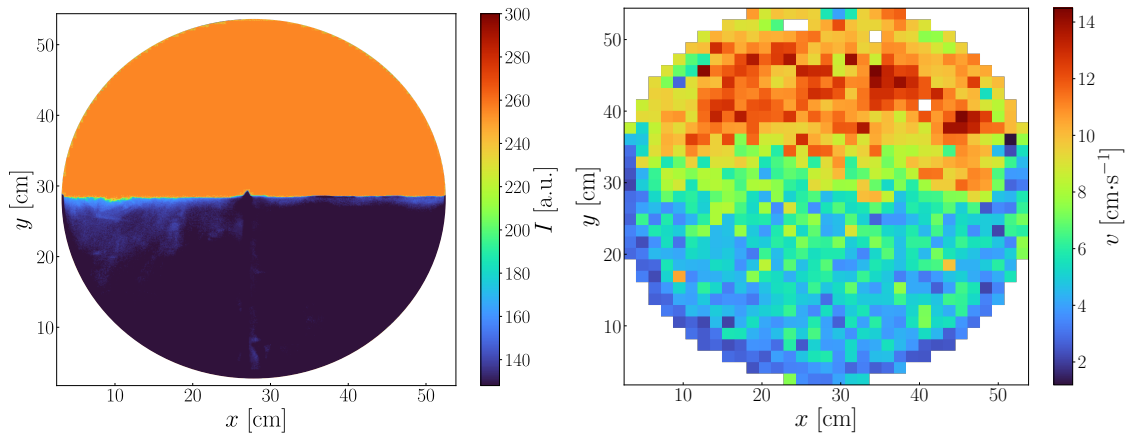


FIG. 7. Left: Light intensity provided by LED panels below the plane of bot motion. The bot is placed within a circular arena. Right: The average bot velocity in the arena for the programmed light-sensitive diffusive motion, measured over a 30 min interval. We can observe that it respects well the distribution of light provided on the left.

correspond to a given phase  $\phi$  in the  $[0, 2\pi)$  interval. This colour is emitted on both LEDs, and switched with a frequency  $\omega_t$ . This frequency is made to vary based on the input coming from the color sensor of the front module, in order to set a basis for synchronization. The frequency changes actively as

$$\omega_{t+1} = \omega_t + K \sin(\phi - \phi_d) \quad (1)$$

where  $\phi_d$  is the detected colour coming from the outside, which is first matched with the closest one in the given list of discrete colours (done by measuring the Euclidean colour difference), and  $K$  is a coupling constant that can be adjusted. Here, we rely on the *Adafruit\_NeoPixel\_ZeroDMA* in order to use DDS (Direct Digital Synthesis) to have an arbitrary frequency resolution.

As an example, we show the synchronization of two stationary bots with this property, placed head to tail in order to be able to observe each other's LEDs. The front bot is made to oscillate with a fixed frequency (see the red line in Figure 6), and we can observe that the back bot synchronizes with the other one both in frequency and in phase (blue line).

## D. Interaction with environment

In this section, we present an example of using the onboard sensors in order to sense and react to the environment. While there is a wide range of possible interactions that can be encoded, that use magnetic fields, colour sensing or proximity for example, we focus on simple reactivity to light. Using the front module, we program the bot so that its velocity, when executing diffusive motion, depends linearly on the amount of light it receives. Thus, the bot will move more slowly in darker regions and faster in light regions.

In order to observe the behaviour of the bot when using this program, we place it within a circular arena, below which LED panels are placed and turned on. As seen in Fig. 7a, only half of the space is bright, while the other half is dark. We measure the bot's responsivity by looking at the time-average velocity of the bot in space, shown in Fig. 7b, displaying that the bot clearly moves faster in the lighter region, with an excellent correlation between light intensity and bot speed. Artificial phototaxis can thus easily be implemented for biologically inspired studies of active matter in complex environments.

## IV. CONCLUSION

The GRASPion offers a rare combination of mechanical simplicity, high programmability, and robust performance. From basic locomotion experiments to the emergence of complex collective behaviors, it provides a reliable and modular platform for probing the full range of active matter phenomena under controlled and reproducible conditions.

The flexibility of the GRASPion opens up unique opportunities to investigate and test frontiers in collective robotics and the emerging field of smart active matter<sup>14</sup>, allowing for a model system of adaptive swarm control<sup>15</sup> and distributed learning in robotic ensembles, as well as an experimental basis for the study of classical concepts such as thermodynamic engines in the context of active matter<sup>16</sup>, swarmalators<sup>13</sup> and predator-prey dynamics. Moreover, thanks to its open hardware and software design, virtually any interaction law, from classical alignment and repulsion<sup>17</sup> to highly unconventional non-reciprocal couplings<sup>18</sup>, can be implemented, thus using programmable interactions to emulate exotic couplings rarely accessible in physical experiments. The versatility, expandability, and accessibility of this bot thus give a range of possibilities for the emulation of animate and active matter, as well as the creation of novel metamaterials, intelligent swarms, and

metamachines<sup>19</sup>.

The GRASPion thus stands as a versatile experimental bridge between statistical physics, robotics, and computational intelligence, paving the way for a new generation of laboratory studies on emergent behavior.

## ACKNOWLEDGMENTS

N.V. thanks the Fondation Francqui for support. F.N. thanks the Alexander von Humboldt Foundation for a postdoctoral fellowship. The authors thank C. Ibanez Gil, D. Manianga and R. Thénon for their contribution to the colour synchronization of the bots.

## AUTHOR DECLARATIONS

### Conflict of Interest

The authors have no conflicts to disclose.

### Author Contributions

**F. Novkoski**: Formal analysis (equal); Investigation (equal); Software (equal); Visualization (equal); Writing - original draft (lead). **M. Mélard**: Conceptualization (equal); Methodology (equal); Software (equal); Writing - review & editing (equal). **M. Delens**: Investigation (equal); Validation (equal); Writing - review & editing (equal). **F. Wéry**: Investigation (supporting); Software (supporting); **M. Noirhomme**: Investigation (supporting); Software (supporting); **J. Pande**: Conceptualization (equal); Methodology (equal); Writing - review & editing (equal). **A. Maier**: Conceptualization (equal); Project administration (equal); Supervision (equal). **A. S. Smith**: Conceptualization (equal); Project administration (equal); Supervision (equal). **N. Vandewalle**: Conceptualization (equal); Funding acquisition (equal); Project administration (equal); Resources (lead); Supervision (equal); Writing - review & editing (equal).

## DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

## REFERENCES

- <sup>1</sup>C. Scholz, M. Engel, and T. Pöschel, *Nat. Commun.* **9**, 931 (2018).
- <sup>2</sup>O. Dauchot and V. Démery, *Phys. Rev. Lett.* **122**, 068002 (2019).
- <sup>3</sup>P. Baconnier, D. Shohat, C. H. López, C. Coulais, V. Démery, G. Düring, and O. Dauchot, *Nat. Phys.* **18**, 1234 (2022).
- <sup>4</sup>A. Deblais, T. Barois, T. Guerin, P. H. Delville, R. Vaudaine, J. S. Lintuvuori, J. F. Boudet, J. C. Baret, and H. Kellay, *Phys. Rev. Lett.* **120**, 188002 (2018).
- <sup>5</sup>S. Li, B. Dutta, S. Cannon, J. J. Daymude, R. Avinery, E. Aydin, A. W. Richa, D. I. Goldman, and D. Randall, *Sci. Adv.* **7**, eabe8494 (2021).
- <sup>6</sup>J. F. Boudet, J. Lintuvuori, C. Lacouture, T. Barois, A. Deblais, K. Xie, S. Cassagnere, B. Tregon, D. B. Brückner, J. C. Baret, and H. Kellay, *Sci. Rob.* **6**, eabd0272 (2021).
- <sup>7</sup>A. A. Dmitriev, A. D. Rozenblit, V. A. Porvatov, M. K. Buzakov, A. A. Molodtsova, D. V. Sennikova, V. A. Smirnov, O. I. Burmistrov, T. I. Karimov, E. M. Puhtina, and N. A. Olekhno, “Swarmodroid 1.0: A modular bristle-bot platform for robotic active matter studies,” (2023), arXiv:2305.13510 [cond-mat.soft].
- <sup>8</sup>M. Pan, Y. Yang, X. Qin, L. Jiang, T. Zhao, Y. Wang, Y. Li, and L. Liu, *IEEE Robotics and Automation Letters* **9**, 7333 (2024).
- <sup>9</sup>M. Noirhomme, I. Mammadli, N. Vanesse, J. Pande, A.-S. Smith, and N. Vandewalle, *Phys. Rev. Appl.* **23**, 064008 (2025).
- <sup>10</sup>F. Novkoski, M. Mélard, M. Delens, and N. Vandewalle, “GRASPion repository,”.
- <sup>11</sup>Y. Kuramoto, *Lecture notes in Physics (LNP)* **30**, 420 (1975).
- <sup>12</sup>A. Pikovsky, M. Rosenblum, J. Kurths, and A. Synchronization, *Self* **2**, 10 (2001).
- <sup>13</sup>K. P. O’Keefe, H. Hong, and S. H. Strogatz, *Nat. Commun.* **8**, 1504 (2017).
- <sup>14</sup>G. Jung, M. Ozawa, and E. Bertin, *Phys. Rev. Lett.* **134**, 248302 (2025).
- <sup>15</sup>A. Ziepke, I. Maryshev, I. S. Aranson, and E. Frey, *Phys. Rev. X* **15**, 031040 (2025).

- <sup>16</sup>P. Pietzonka, E. Fodor, C. Lohrmann, M. E. Cates, and U. Seifert, *Phys. Rev. X* **9**, 041032 (2019).
- <sup>17</sup>T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, *Phys. Rev. Lett.* **75**, 1226 (1995).
- <sup>18</sup>S. A. M. Loos and S. H. L. Klapp, *New J. Phys.* **22**, 123051 (2020).
- <sup>19</sup>G. Volpe, N. A. M. Araújo, M. Guix, M. Miodownik, N. Martin, L. Alvarez, J. Simmchen, R. D. Leonardo, N. Pellicciotta, Q. Martinet, J. Palacci, W. K. Ng, D. Saxena, R. Sapienza, S. Nadine, J. F. Mano, R. Mahdavi, C. Beck Adiels, J. Forth, C. Santangelo, S. Palagi, J. M. Seok, V. A. Webster-Wood, S. Wang, L. Yao, A. Aghakhani, T. Barois, H. Kellay, C. Coulais, M. van Hecke, C. J. Pierce, T. Wang, B. Chong, D. I. Goldman, A. Reina, V. Trianni, G. Volpe, R. Beckett, S. P. Nair, and R. Armstrong, *J. Phys. Condens. Matter* **37**, 333501 (2025).