Design and Implementation of a Millirobot for Swarm Studies – *mROBerTO*

Justin Y. Kim, Zendai Kashino*, Tyler Colaco, Goldie Nejat, and Beno Benhabib

Department of Mechanical and Industrial Engineering, University of Toronto
5 King's College Road, Toronto, ON, Canada M5S 3G8

(Accepted MONTH DAY, YEAR. First published online: MONTH DAY, YEAR)

SUMMARY
The use of millirobots, particularly in swarm studies, would enable researchers to verify their proposed autonomous cooperative-behavior algorithms under realistic conditions with a large number of agents. While multiple designs for such robots have been proposed, they, typically, require custom-made components, which make replication and manufacturing difficult, and, mostly, employ non-modular integral designs. Furthermore, these robots' proposed small sizes tend to limit sensory perception capabilities and operational time. Some have resolved few of the above issues through the use of extensions that, unfortunately, add to their size.

In contribution to the pertinent field, thus, a novel millirobot with an open-source design, addressing the above concerns, is presented in this paper. Our proposed millirobot has a modular design and uses easy to source, off-the-shelf components. The *m*illi-*r*obot-*T*oronto (*mROBerTO*) also includes a variety of sensors and has a 16×16 mm^2 footprint. *mROBerTO*'s wireless communication capabilities include ANT™, Bluetooth Smart, or both simultaneously. Data-processing is handled by an ARM processor with 256 KB of flash memory. Additionally, the sensing modules allow for extending or changing the robot's perception capabilities without adding to the robot's size. For example, the swarm-sensing module, designed to facilitate swarm studies, allows for measuring proximity and bearing to neighboring robots and performing local communications.

Extensive experiments, some of which are presented herein, have illustrated the capability of *mROBerTO* units for use in implementing a variety of commonly proposed swarm algorithms.

KEYWORDS: Swarm Robotics; Design; Multi-Robot Systems; Control of Robotic Systems; Robot Localization.

1. Introduction

Milli-, micro-, and nano-robots are categories of robots that are small in size, ranging from mm^3 to μm^3.[1] They tend to be expandable, easy to construct, and simple to maintain and set up. They can operate individually, but also in swarms.[2-4]

Other examples of millirobots include ones that are capable of pulling objects several times their body weight[5], biologically inspired multi-legged robots,[6, 7] and robots capable of deploying themselves through folding, inspired by origami.[8] They have been suggested for a variety of application areas, such as dynamic wireless sensor networks (WSNs) with mobile nodes,[9, 10] micro-manufacturing with small scale robots[11], and minimally invasive medical treatments.[12, 13] Future applications also include urban[14-18] and wilderness[19-23] search and rescue, and surveillance[24-29].

Millirobots with footprints greater than 75×75 mm^2 [30-33] often have greater capabilities and capacity compared to smaller units[34-44] – e.g., larger battery capacity, more powerful processors, and increased sensory capabilities. The *E-Puck*[30] and *SwarmBot*,[31] for example, have Linux operating systems and

* Corresponding author. E-mail: zendkash@mie.utoronto.ca

cameras. Another large millirobot, the *R-One*,[32, 33] is equipped with grippers for manipulating objects and docking to other *R-One* robots.

There exist numerous challenges in the development of millirobots. One such challenge is the trade-off between size and capability (e.g., sensing, battery, processing). Another challenge is that as the millirobots become smaller, their assembly becomes more complex. Additionally, custom-made components are often required to achieve better capabilities at small sizes. These issues can reduce the millirobot's advantage of easy setup and manageability. At the present time, few millirobots are commercially available.[45, 46]

Swarm-behavior research has been a main application for millirobots, including *Alice II*,[35] *Jasmine*,[36] *AMiR*,[37] *Wanda*,[38] *TinyTeRP*,[39, 40] and *GRITSBot*.[41] *Alice II* is capable of operating for up to ten hours on nickel-metal hydride batteries and can be equipped with cameras and wireless communication components using extension modules. *Jasmine* is capable of IR sensing and communication using six IR emitter/receiver pairs located at the edge of the robot's chassis. Both *Wanda* and *AMiR* have software development toolkits that include both high- and low-level control functions. *TinyTeRP* can be fitted with a set of all-terrain tracked wheels for mobility on rough terrains. *GRITSBot* comprises three customizable module layers and has autonomous self-charging capabilities.

Kilobot,[42] *Colias*,[43] and *Zooids*[44] are three other examples of millirobots designed for swarm research, but with limited modularity. *Kilobot*s have been reported to operate in swarms of more than one thousand units and make use of vibration motors for mobility.

Most millirobots utilize IR for communications as it allows them to communicate locally with neighboring robots, while also providing a means of localizing these neighbors. Although wireless communication is often easier to work with due the low quality of IR-based communications (e.g., proneness to errors, limited range, and slow data-transfer speeds), it would be difficult to make space for additional modes of communication on such millirobots due to their size constraint. Additionally, the limited space makes it a challenge to provide the robot with enough portable power to fully use all communication modes. Some resolve this issue through the use of attachable radio communication modules allowing for alternative communication modes, such as the use of a radio extension module in *Alice II*.[35]

In order to obtain higher performance at smaller sizes, several past millirobots have employed custom-made or difficult-to-source components. For example, *Alice II* uses a 3D-printed chassis and modified watch motors to achieve effective mobility at small sizes. However, the need for custom-made components has been diminishing as many industries move towards miniaturization of electronic and mechanical components, while maintaining and/or even improving performance. These trends have resulted in smaller and lighter motors, controllers, actuators, and drives. Additional industry trends towards wireless capabilities can also be noted, ranging from wireless communications to wireless charging.

In this paper, we detail a new modular millirobot with a 16×16 mm^2 footprint, ***m***illi-***r***obot-***T***oronto (*mROBerTO*). Our robot is novel in that it (*a*) has a design that enables modifying and expanding robot capabilities without making significant changes to the robot size or structure, (*b*) makes exclusive use of off-the-shelf components, and (*c*) is one of the smallest millirobots that exists with extensive sensing, communication and processing capabilities. A preliminary hardware prototype concept of the robot was first presented in ref. [47]. Herein, we further present a comprehensive and extended description of the hardware and software design of *mROBerTO* with special an emphasis on the millirobot's swarm capabilities, including the introduction of a new hardware module to allow for swarm sensing.

In Section 2 below, we first describe the overall design of *mROBerTO*. Subsequently, in Section 3, we demonstrate *mROBerTO*'s potential for swarm-behavior research, illustrated via detailed experiments. Concluding remarks are presented in Section 4.

2. *mROBerTO* Design

The proposed millirobot for swarm studies was designed based on three primary objectives: to achieve a high degree of modularity, to maximize use of off-the-shelf components for ease of assembly, production, and maintenance, and to occupy a minimum possible footprint area, while maximizing processing power and sensing capabilities. Modularity is desirable to simplify the expansion of sensing capabilities and allow the introduction of improved circuitry in future revisions. Using commercially available, off-the-shelf components enables easy replication and high-volume production at low cost.

mROBerTO, as shown in Fig. 1, has four modules within its 16×16×32 mm^3 envelope: processing and communication module, locomotion module, proximity-sensing module, and swarm-sensing module. Each module consists of a PCB with surface mounted electrical and mechanical components. The modules are all structurally and electrically connected through soldering, except for the swarm-sensing module, which is connected through header pins for rapid exchange. *mROBerTO* has no chassis and uses the PCBs as mechanical structural components instead.

The overview of *mROBerTO*'s architecture is given in Fig. 2, where the detailed specifications are provided in Appendix A. A brief cost comparison of *mROBerTO* to *Kilobot*, which is in the same size class, is provided in Appendix B.

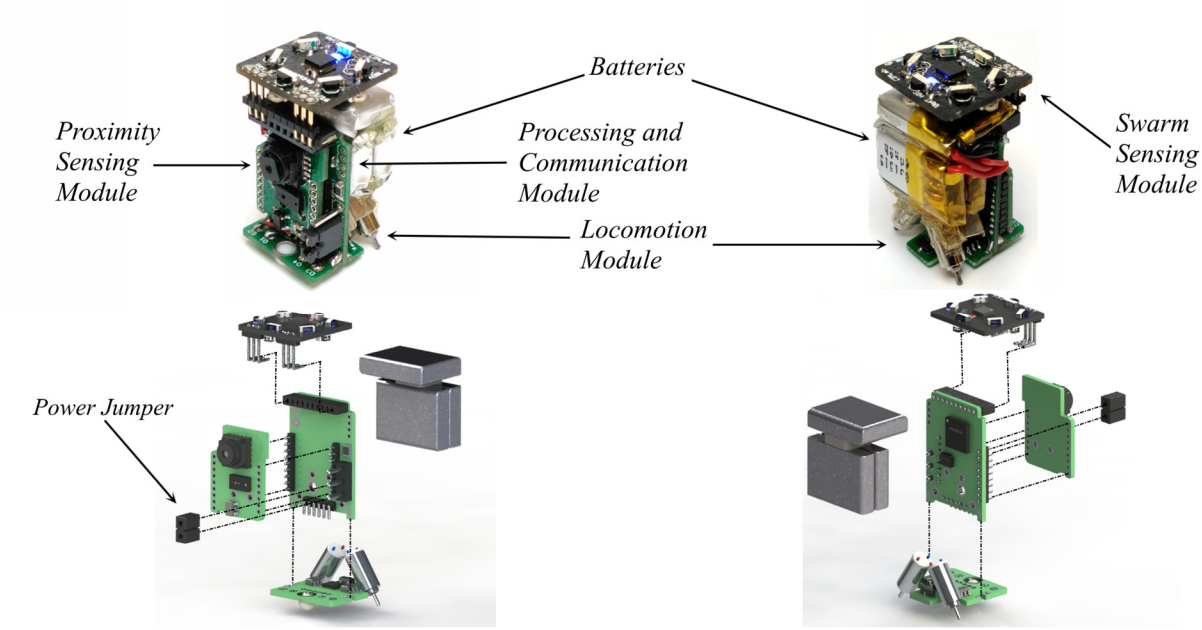


Fig. 1. Two views of *the hardware for mROBerTO*.

2.1. Processing and Communication Module

The processing and communication module, is the central hub of *mROBerTO*. All other modules connect to this module. The processing and communication module has on it the Nordic nRF51422 system-on-chip (SoC), a 32-bit ARM Cortex-M0 clocked at 16 MHz with 32 KB RAM, 256 KB flash memory, and built in wireless networking capabilities[48]. Wireless protocols that the SoC can use are ANT™ and Bluetooth Smart. ANT™ is a wireless networking protocol designed and marketed by ANT Wireless. The protocol runs in the 2.4 GHz ISM band and is designed to need low power, easy to use, and handle a variety of network topologies (e.g., mesh, tree, point-to-point, star, etc.). It has been compared to other low-bitrate, low-power protocols, such as ZigBee and Bluetooth Low Energy, both of which are based on the IEEE 802.15 technical standard. Bluetooth Smart (also known as Bluetooth Low Energy) is a wireless protocol with considerably reduced power consumption and cost compared to the Classic Bluetooth protocol. It should be noted that the Bluetooth Low Energy protocol is not backwards compatible with the

Classic Bluetooth protocol. The signal power of both ANT™ and BLE can be controlled through software and dynamically be changed during runtime. This power can range between -40dBm and +4dBm. This can be useful if the communication range of robots needs to be adjusted for testing a specific swarm scenario. Furthermore, it can be used to save power during deployment when the robots are idle. In addition, its wireless network capabilities, the nRF51422 has several additional features that make it particularly suitable for achieving the design objectives. One such feature is the flexible General-Purpose Input/Output (GPIO) mapping feature, which enables any set of GPIOs to be set as one of two-wire interface (TWI or I2C) or serial peripheral interface/universal asynchronous receiver and transmitter (SPI/UART). This feature provides users with some flexibility when designing new sensing modules. Herein, as will be detailed below, the proximity-sensing module has access to up to 15 GPIO pins, while the swarm-sensing module has access to up to 8 GPIO pins.

Bluetooth Smart (also known as Bluetooth Low Energy, or BLE) is a feature of the nRF51422 that allows *mROBerTO* to utilize high data-throughput sensors, such as an onboard camera, without excessive energy costs. BLE has a high data-transfer rate (up to 1.0 Mbps) with low power consumption. Additionally, the nRF51422 can be programmed over-the-air through its BLE interface. This simplifies the setup of the robots, especially, when there are multiple units to program, and adds to overall user-friendliness.

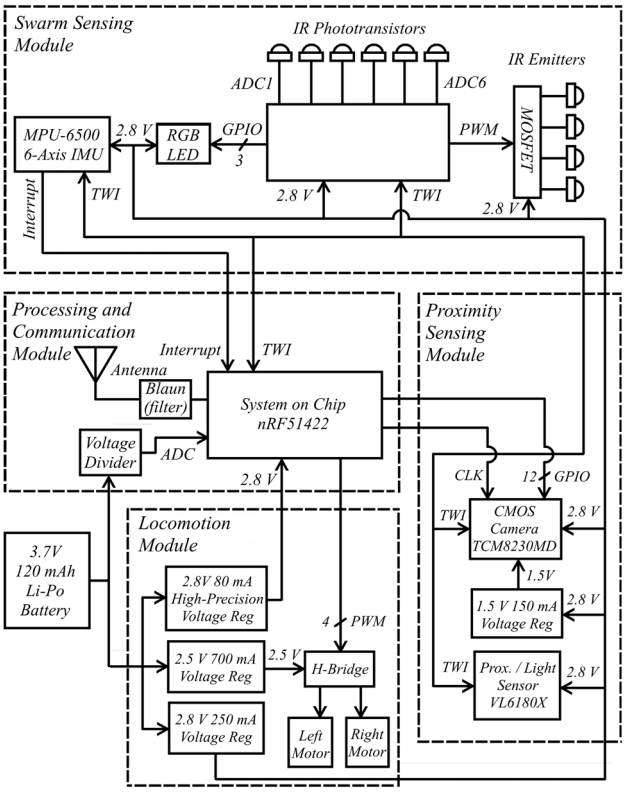


Fig. 2. *mROBerTO*'s architecture.

2.2. Locomotion Module

Placed below the processing and communication module is the locomotion module, serving as the structural base of the robot, Fig. 1. An H-bridge is used for controlling two 4-mm Nano Coreless motors manufactured by Precision Microdrives.[49] These motors have a rated load speed of 39,000 rpm at 0.01 mN·m and a typical maximum output power of 61 mW. The H-bridge is connected to the motors to provide differential drive. The motor shafts have no wheels and are directly in contact with the surface, removing the need for custom made wheels and drive trains. During development, it was noted that the

small contact area between the motor shafts and workspace surface made it difficult to translate motor torque into robot motion. This was addressed, however, by placing the heaviest components over the wheel contact points to increase friction and by adding adhesive to the wheels such that they would grip the surface better. The third point of contact for *mROBerTO* is a small (1/8 inch≈3.175 mm diameter) polytetrafluoroethylene ball on the locomotion module – chosen for its small weight and low coefficient of friction. With its current specifications, detailed in Appendix A, *mROBerTO* can move at speeds ranging from 1 mm/s up to 150 mm/s in a straight line and turn at approximately 500 deg/s.

There exist alternative locomotion methods to achieve differential drive without wheels. One example is the vibration system used in *Kilobot*. However, our experience with these systems shows that such methods of locomotion are, typically, not suitable for precise movement over large distances, especially when making use of commercially available, low-end motors. This is due to the motor and locomotion method's non-linear behavior and random noise added from slippage in various directions.

A comparison of *mROBerTO's* and *Kilobot's*[42] locomotions was carried out through a set of closed-loop control experiments. The closed-loop control system used a firmware PID controller with an overhead camera to provide position feedback.[50] Millirobot motion was recorded to illustrate how much each robot deviated from the desired target path. *mROBerTO* clearly outperforms *Kilobot*, which is to be expected: namely, the *Kilobot* lacks IMUs to use as a form of odometry and moves using stick-slip based locomotion, which causes slippage in undesirable directions. On *mROBerTO*, the IMU provides high temporal resolution estimates of robot motion. Additionally, *mROBerTO*'s differential drive provides it with superior maneuverability, allowing it to outperform *Kilobot* on the curved path.

Shown in Fig. 3 are some representative test results for path following – a curved path with an envelope approximately 250 mm by 250 mm. Path following errors for *mROBerTO* were distributed with a mean of approximately 2.0 mm and a standard deviation of 1.8 mm. At most, *mROBerTO* deviated from the target path by about 8.0 mm. In comparison, *Kilobot* errors had a mean of approximately 7.0 mm with a standard deviation of 6.6 mm, and a maximum error of about 34.0 mm. Additional results for the robots following a straight-line for approximately 500 mm and circular path with an approximate diameter of 500 mm are summarized in Table I below.

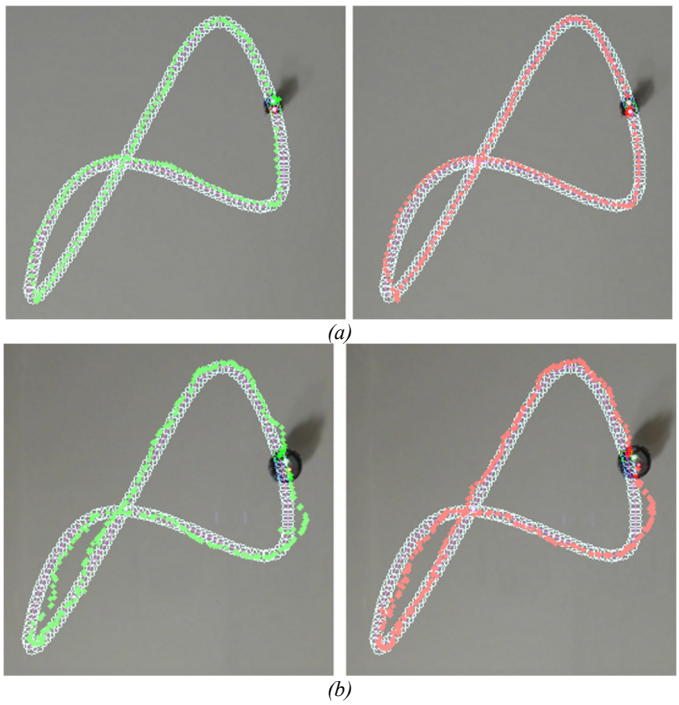


Fig. 3. Curved path tests for (a) *mROBerTO* and (b) *Kilobot*.

Table I. Path error data for path following experiments where a total of 10 runs were completed for each robot and each experiment.

	Straight-line	Circular Path	Curved Path
	mROBerTO / Kilobot	***mROBerTO / Kilobot***	***mROBerTO / Kilobot***
Average Error (mm)	1 / 1	3 / 4	2 / 7
Max. Error Value (mm)	5 / 11	13 / 26	8 / 34
Standard Deviation (mm)	0.9 / 1.0	2.5 / 3.7	1.8 / 6.6

2.3. Proximity-Sensing Module

The proximity-sensing module, mounted on the processing and communication module, was designed to allow the robot to identify obstacles during operation. The module has access to 15 GPIO pins on the SoC and is outfitted with a CMOS camera, the Toshiba TCM8230MD, and a 3-in-1 (proximity-sensing, ambient light sensing, and laser source) time of flight proximity sensor, VL6180X. *mROBerTO* can acquire 7-bit grayscale images of up to 128 × 96 pixels in size at up to 10 frames per second from the TCM8230MD using 8 parallel data pins. Furthermore, up to 10 of these images could be stored locally on the robot at a time.

2.4. Swarm-Sensing Module

The newly developed swarm-sensing module allows *mROBerTO* to exhibit cooperative swarm behavior. It has access to 8 GPIO pins on the SoC and comprises four wide angle IR emitters (Kingbright APA3010F3C-GX) and six wide-angle IR phototransistors (Kingbright APA3010P3BT-GX), placed on the edges of the module for all-around coverage. These are mainly used for measuring the relative distances and bearings of nearby robots, but can also be used for communication purposes. Communication is established using amplitude-modulated IR communication with data transfer rates of up to 105 bits per second on two channels. Communication can be established with robots within 125 mm. Further details on the development of the module, including hardware design, can be found in Appendix C. The noise characteristics of proximity and bearing measurements are presented in Table II. When compared to a similar system, the *Droplet*,[51] our approach for measuring proximity and bearing to neighboring robots, as detailed in Appendix C, had standard deviations that were up to an order of magnitude less. For example, the maximum standard deviation of our bearing measurements was 3° while the compared system had a maximum standard deviation of approximately 12°. Similarly, the maximum standard deviation of our proximity measurements was 2.8 mm while the compared system had a maximum standard deviation of approximately 30 mm.

Table II. Error data for both bearings and proximities at distances from 50 to 125 mm, from 0° to 360° directions, at increments of 15°, for the wide-angle IR detectors.

Distance (mm)	Average Bearing Error (deg)	Standard Deviation of Bearing Errors (deg)	Max Bearing Error (deg)	Average Proximity Error (mm)	Standard Deviation of Proximity Errors (mm)	Max Proximity Error (mm)
50	3	1.8	10	4	2.4	10
75	4	1.7	7	2	1.5	7
100	6	1.9	11	5	2.5	12
125	10	3.0	18	6	2.8	14

The module also has a 6-axis inertial-measurement unit (IMU), one RGB LED, and has 4-pin GPIO access to the processor. The main processor on the swarm module is the ATmega328P, which interacts with the main SoC via TWI. The swarm-sensing module is attached to the processing and communication module through a set of header pins for easy exchange. This allows sensing capabilities of the robot to be modified without having to rebuild the entire unit.

The proximity- and swarm-sensing modules share two GPIO pins from the SoC, allowing for a single instance of SPI/UART or TWI to communicate with both sensing modules simultaneously. The locomotion module provides both sensing modules with a 2.8 V power supply from the voltage regulators located on it.

2.5. Power Management

mROBerTO is powered by three 3.7 V Li-Po batteries connected in parallel such that the total current capacity of the batteries is 120 mAh, Fig. 1. *mROBerTO* can operate for a minimum of one hour at full operation, defined herein as operating both motors under a 30% duty cycle pulse-width modulation (PWM). This results in a speed of approximately 100 mm/s. In addition, the camera and proximity sensor are turned on/off every second, LEDs are continuously on, the IMU is used with a 10 ms update period, the IR emitter/receivers are used to send and receive data packets at 4 Hz, and data is broadcasted/received at 16 Hz through ANT.

Battery voltage level is directly monitored by utilizing a voltage divider. The positive end of the battery pack is connected to a voltage divider, which divides the voltage by two. The reduced voltage is, then, monitored through an ADC port. This allows the robot to warn the user when its battery is almost depleted and requires a recharge.

2.6. Software

Software level control of *mROBerTO* is performed by programming the on-board nRF51422 SoC. Programs are written in C++ and processor functions are accessed through the nRF51 software development kit (SDK) provided by Nordic Semiconductors.[52] Versions 9 and 10 of the SDK are used with the nRF51422. Version 9 provides access to BLE functionality and Version 10 provides access to both BLE and ANT™ functionality. Each SDK includes example code for programming the SoC. Documentation for the SDKs can be found in ref. [53].

High-level control functions for the abovementioned sensors and motors were implemented and packaged into header file libraries to serve as the first *mROBerTO* library. For example, motors can be controlled by providing wheel angular velocities to a motor control function, which handles low-level interactions with the motors. A motor self-calibration process, PID motion control, and wireless debugging are all pre-programmed into the robots' firmware.

ANT™ and Bluetooth capabilities are programmed into Nordic's SoftDevices (precompiled, linked binary software). An advantage of using the nRF51422 is that the cross ARM GCC can be used for compiling source code. Most integrated development environments (IDE), such as the free and popular Eclipse, can make use of this free software compiler. Additionally, by using Eclipse, one can perform on-board debugging using the Serial Wire Debug interface for ARM included in the SEGGER J-Link programmer.

The microcontroller used in the swarm-sensing module, the ATmega328P, is programmable using a low-cost (<$20) programmer (e.g., Polulu's USB in-system programming (ISP) module). Atmel Studio 7.0 and AVR Libc can be used as an IDE and compiler respectively. Both are free to download and use.

3. Swarm Capabilities of *mROBerTO*

Swarm robots need to be simple in design and use a distributed approach, with regards to both processing and decentralized control, in order to complete a global task collectively.[54, 55] Collective behaviors studied in the past have been broadly categorized into[56]: *(i)* spatially organizing behavior, where formation

control is involved, *(ii)* navigational behavior, utilized to explore an unknown environment, and *(iii)* collective decision-making, using wireless communication between robots, to reach a consensus.

The main emphasis of running the collective behavior experiments was to showcase *mROBerTO*'s ability to:

a. Measure relative distances and bearings of nearby robots and obstacles.
b. Differentiate nearby robots with their unique IDs.
c. Wirelessly communicate with other robots using point-to-point and mesh network topologies.
d. Move with enough accuracy and precision to exhibit formation control in the collective behaviors.

All of above features are essential in swarm robotics.

In order to showcase *mROBerTO*'s swarm capabilities, specifically, four different collective behaviors were chosen and implemented in experiments herein as examples of the three common collective behavior categories listed above: aggregation, chain formation, collective exploration, and dynamic task allocation. Aggregation and chain formation demonstrate a *spatially-organizing* behavior, where formation control is employed by the robots. Collective exploration illustrates *mROBerTO*'s *navigation* capabilities in an unknown environment, where obstacle avoidance is employed. Lastly, the dynamic task allocation illustrates *mROBerTO*'s *wireless communication* capabilities using ANT.

3.1. Algorithms

As detailed in refs. [54-56], there exist three different primary methods to implement swarm behavior: probabilistic methods, artificial-physics methods, and evolutionary methods. In probabilistic methods, the robot's next behavior is partially determined in a random manner and dictated by interactions between the robot and its environment, which may include other robots. These methods are often used with finite state machines (FSMs) including probabilistic FSM (PFSMs). They accurately model the behaviors of social insects, such as honeybees and cockroaches.

In artificial-physics methods, specifically, for multi-robot systems (also referred to as physicomimetics), virtual forces are utilized by the robots to determine their next movement vectors.[57] Formations observed in nature, such as with flocks of birds, schools of fishes, and swarms of insects, can be modeled by attractive and repulsive forces.

In evolutionary methods, robot controllers are designed based on artificial evolution techniques[58], such as genetic algorithms[59, 60] and q-tournament selection.[61] Fitness functions are often employed, where robots are assigned a higher (better) fitness value if the desired behavior is exhibited. The evolution cycle continues where the fitness functions re-evaluate the fitness levels of all robots until the global task given to the swarm is completed.

The specific algorithms for our four scenarios were chosen based on their popularity in the literature. Furthermore, most have been implemented and verified by other millirobots.[62-65] Appendix D provides more detailed descriptions of the selected swarm scenarios.

3.2. Constraints

Many of the collective behaviors in swarm robotics, especially, for formation control, become trivial if all robots in the workspace are aware of each other's global positions and orientations, and work with a centralized controller. However, by definition, swarm robots may only interact locally (decentralized) and are assumed to have no global knowledge regarding any of the robots or their environment at any given moment.[54-56, 66, 67] Thus, the four swarm scenarios discussed in the sections below are subjected to the following common constraints:

a. No robot has global knowledge of any other robot in the workspace. Furthermore, it has no information regarding the size of the workspace or the total number of robots operating within it.
b. A robot's perceptual knowledge of other robots and the environment is limited to within its sensing radius. Though, even this perception is neither absolute nor reliable.

c. There is no centralized controller. Communications between robots is limited (e.g., up to 128 bytes of data for *mROBerTO*, using advanced ANT™ burst mode), and only locally broadcasted in either a mesh type or point-to-point network. Furthermore, the pertinent signals are not reliable.

d. Non-holonomic differential-drive robots that do not have odometry, typically, rely on sensors susceptible to noise, such as IMUs and magnetometers for motion planning.

3.3. Swarm Scenario 1: Aggregation

In nature, self-organizing aggregation can be observed in schools of fish, flocks of birds, insects such as bees, cockroaches, and even in bacteria.[68-73] Aggregation allows animals to protect themselves from predators and increase their sensing capabilities as a whole by huddling into one unit. Aggregation type behavior is often employed in swarm robotics for simulating such scenarios, allowing interactions between robots at close proximities.[54-56] Aggregation scenarios have been simulated in the literature via the use of FSM approaches,[62, 71, 72] artificial physics,[74-79] and evolutionary algorithms.[59, 80, 81]

For *mROBerTO*s, we chose to use an FSM approach for implementing aggregation behavior due to its simplicity as well as being one of the more popular methods in the literature. The main objective of this scenario is to showcase our robot's ability to measure nearby robots' relative distances and bearings and use this information in order to complete the self-organized aggregation task.

Adapting the FSM approach from ref. [62], a behavior-based model[82] was implemented. The robots were programmed to *(i) approach* other robots, where the robot looks for any nearby robots and create a movement vector using the summed vectors of nearby robots' relative positions, and *(ii) wait*, where if one or more robots are within a user specified d_{close} relative distance, then, the robot stops moving.

Fig. 4 shows the results of an Aggregation experiment where a green LED indicates the *approach* state while a blue LED indicates the *wait* state. Initially, the robots have no knowledge of other nearby robots' positions and they start at a minimum of 80 mm distance away from each other. Fig. 5 shows a corresponding plot of average minimum distances between the robots over time.

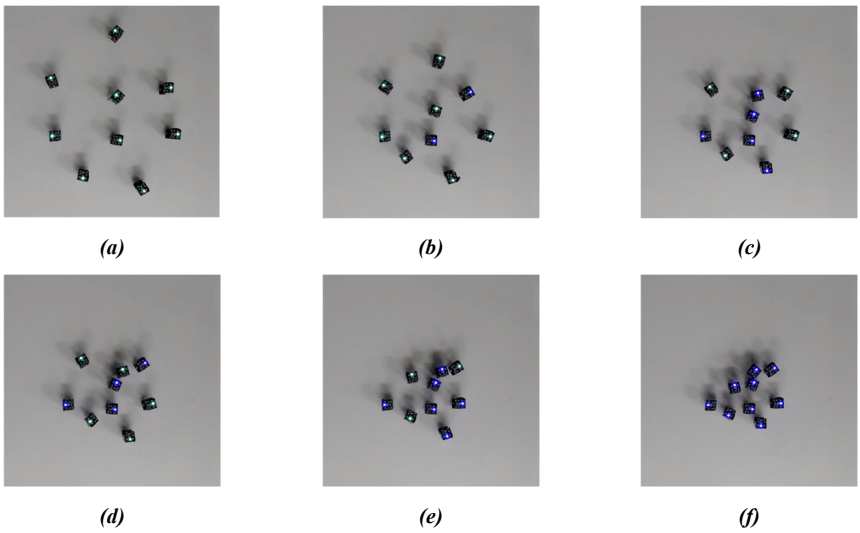


Fig. 4. Aggregation behavior, (*a*) to (*f*), demonstrated by *mROBerTO*s. Robots start at a minimum distance of 80 mm amongst themselves.

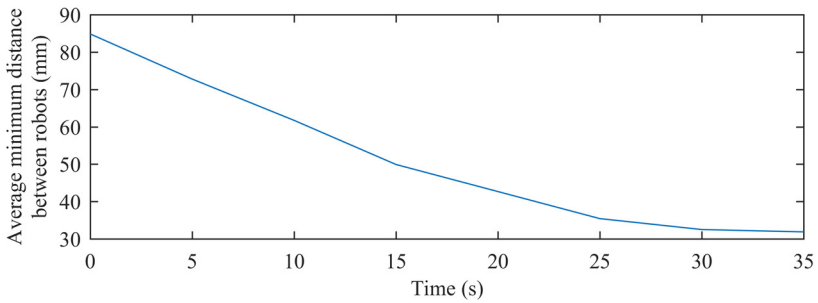


Fig. 5. Average minimum distance between robots over time.

3.4. Swarm Scenario 2: Chain Formation

Social insects, such as ants, form paths between two locations[54] – for example, from a food source to their nest in a foraging scenario. In swarm robotics, chain formation can be used for surveillance and navigation purposes. Same as for aggregation, chain formation can also be achieved via FSM approaches,[63, 83] artificial-physics,[84] or evolutionary methods.[85]

Herein, a FSM approach was used to implement the chain formation collective behavior.[63] However, instead of having the robots that are not part of the chain perform random walks around the entire arena, as done in ref. [63], *mROBerTO*s stayed stationary near the nest since the main objective was to demonstrate their ability to form a straight chain. The chain, starting from the nest outward, was achieved with the use of relative distance and bearing information of nearby robots as well as with the use of the ANT™ mesh communication network.

Fig. 6 shows an experimental result for chain formation. The green LED indicates the current tail of the chain and the blue LED indicates the robot in motion to get ahead of the tail, until it stops and becomes the new tail. Fig. 7 shows a plot of chain lengths over time.

Similar to the aggregation behavior, the chain formation method adapted from ref. [63] did not explicitly provide the technique used for determining the next movement vector of each robot. Thus, the artificial potential approach was adopted herein to move the robots accordingly in order to achieve chain formation (as detailed in Appendix D).

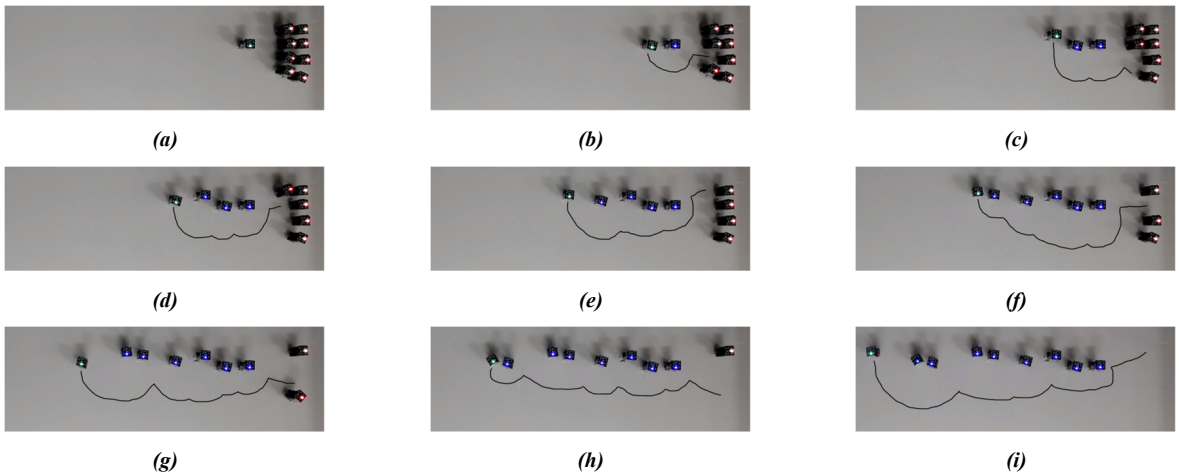


Fig. 6. Chain formation demonstrated by *mROBerTO*s.

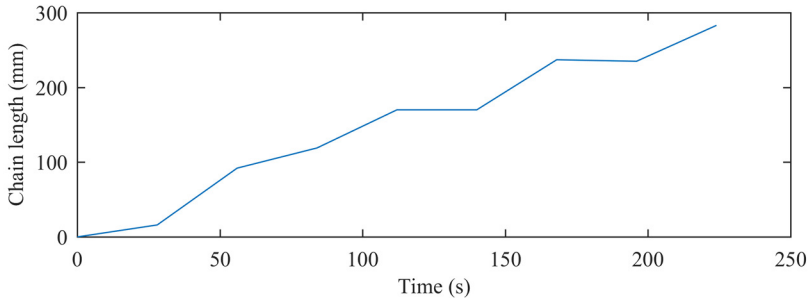


Fig. 7. Chain length over time.

3.5. Swarm Scenario 3: Collective Exploration

Collective exploration of an unknown environment is often used in area coverage problems for monitoring and surveillance, as well as for swarm-guided navigation. For example, robots that cover an area can guide others to get to specific positions within the area for mapping purposes.[9, 56] In general, exploration must be carried out while performing obstacle avoidance.[86] Collective exploration involves self-deployment of robots without a centralized controller, with only local sensing capability.

Several different approaches have been proposed for solving the deployment and collective exploration problem in an unknown environment. In refs. [87-91], taking inspiration from social insects, such as ants, a stigmergic strategy was proposed, where artificial pheromones were dropped by robots to guide and communicate with other robots and for mapping purposes. In refs. [64], [92], [93], artificial physics was used to model the robots as particles and consider the walls of the unknown environment as obstacles in order for robots to disperse evenly in any sized and shaped environment while achieving full area coverage.

The method in ref. [64] for collective exploration was implemented using *mROBerTO*s (detailed in Appendix D). The arena for the experiments was designed to have a wall in the middle of the workspace as shown in Figs. 8 and 9. Experiments, typically, took 240 to 480 s to reach equilibrium, as shown in Fig. 8f and Fig. 9b for two experiments, respectively.

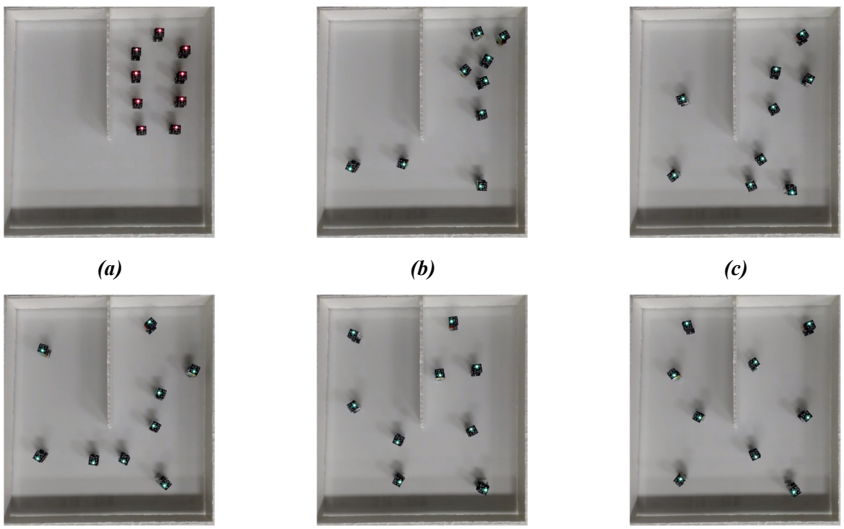


Fig. 8. Collective exploration demonstrated by *mROBerTO*s using artificial potential physics.

(a) ***(b)***

Fig. 9. Second collective exploration behavior experiment demonstrated by *mROBerTO*s: (*a*) start and (*b*) end.

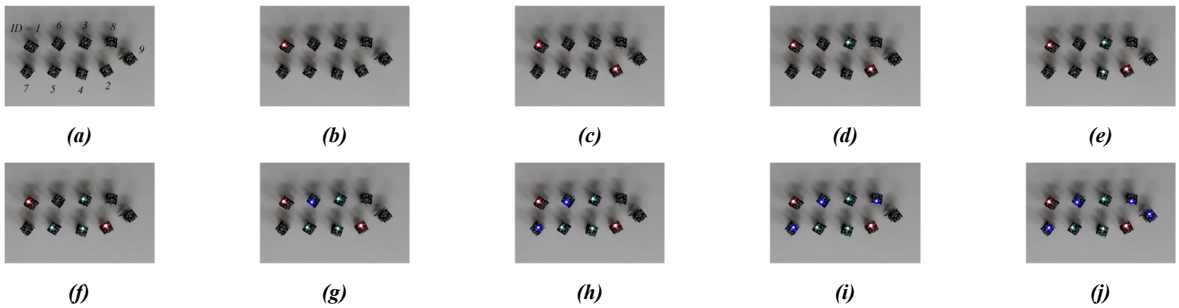
3.6. Swarm Scenario 4: Dynamic Task Allocation

In swarm robotics, a group of robots can be asked to occasionally share/distribute several different tasks amongst themselves by forming subgroups. For example, in a foraging scenario, one subgroup of robots could be tasked to bring resources back to the nest, another subgroup is assigned to actively scout for additional resources in an unknown environment, while a third subgroup is tasked with defending the nest.[94-97]

For the dynamic task allocation experiment, herein, the *Card Dealer's* algorithm in ref. [65] was chosen for implementation. The algorithm begins with all the robots having the desired target distribution vector that describes how the group should be divided into subgroups. The algorithm works iteratively through a series of stages, where at the end of each stage a robot without an assigned task is dealt a task. Further detailed explanation can be found in Appendix D.

One of the prerequisite for the *Card Dealer's* algorithm is that the robots are required to be within proximity of each other in order to measure the relative distances of nearby robots as well as for relaying wireless communication. Relative distances are measured for the purpose of choosing the amount of time period assigned per stage. For example, each stage should have a longer period of broadcasting if there are more robots than a situation where there are fewer robots on the workspace in order to ensure that the messages were relayed successfully to all robots. The communication links between each robot define a directed graph $\mathcal{G}$, where the nodes represent the robots themselves and edges represent the communication links.

For our experiment, we chose the desired task distribution vector as $\boldsymbol{p}$ = (2/9, 3/9, 4/9), where the first task (for two robots) is displayed by red LEDs, the second task (for three robots) is displayed by green LEDs, and the third task (for four robots) is displayed by blue LEDs, Fig. 10. Communication between the robots was achieved using the ANT™ wireless communication in a mesh type network. The robots remain stationary during task allocation. In the experiments performed, robots made a decision every 20 s such that all nine robots had an assigned task according to the desired task distribution vector within 180 s after start.



(a) ***(b)*** ***(c)*** ***(d)*** ***(e)***

(f) ***(g)*** ***(h)*** ***(i)*** ***(j)***

Fig. 10. Dynamic task allocation demonstrated by *mROBerTO*s using *Card Dealer's* algorithm and ANT™ wireless communication.

4. Conclusions

In this paper, the development and implementation of the novel small-sized (16×16 mm^2) modular *mROBerTO* millirobot is presented. The robot is designed using only commercially available and easily sourced components for a simplified design and easy maintenance. In contrast to other millirobots, *mROBerTO* offers a rich range of sensor capabilities while being one of the smaller existing millirobots. Its modular design allows the user to easily change many of its hardware features with little disruption to other modules. This also allows independent development of separate modules to occur and is made to be easily exchangeable among research teams, according to their specific needs.

Various communication network topologies can be utilized on *mROBerTO*, such as point-to-point, mesh, and tree topologies using its Bluetooth Smart and ANT™ communications. Decentralized formation control for swarm behavior studies, where the robot moves autonomously with respect to other robots, can be achieved via the use of the swarm-sensing module.

Extensive experiments have verified the applicability of *mROBerTO* to swarm studies – being capable of exhibiting different collective behaviors, such as aggregation, chain formation, collective exploration, and dynamic task allocation.

Future work is expected to include improvements to the processing power to facilitate the use of higher performance sensors (such as the ESP32 or an updated nRF52 series SoC), improving the locomotion capabilities of the robot further, and developing further sensing modules that could allow for specialization of millirobots in heterogeneous swarm experiments.

Acknowledgements

This research was funded in part by the Natural Sciences and Engineering Research Council of Canada, and the Canada Research Chairs Program.

References

1. I. Paprotny and S. Bergbreiter, "Small-Scale Robotics: An Introduction," *Small-Scale Robotics from Nano-to-Millimeter-Sized Robotic Syst. Applicat.*, New York, NY: Springer, 2014, ch. 1, pp. 2.
2. Y. K. Lopes, A. B. Leal, and T. J. Dodd, "Application of Supervisory Control Theory to Swarms of e-puck and Kilobot Robots," *Swarm Intell*., New York, NY: Springer, 2014, pp. 62–73.
3. N. Correll, S. Rutishauser, and A. Martinoli, "Comparing Coordination Schemes for Miniature Robotic Swarms: A Case Study in Boundary Coverage of Regular Structures," *Springer Tracts in Advanced Robots*, vol. 39, O. Khatib, V. Kumar, and D. Rus, Eds. Berlin, Germany: Springer, 2008, pp. 471–480.
4. D. Fyler, B. Sullivan, and I. A. Raptis, "Distributed Object Manipulation Using a Mobile Multi-Agent System," *IEEE Int. Conf. Technologies for Practical Robot Applicat.*, Woburn, MA, 2015, pp. 1–6.
5. D. L. Christensen, E. W. Hawkes, S. A. Suresh, K. Ladenheim, and M. R. Cutkosky, "μTugs: Enabling Microrobots to Deliver Macro Forces with Controllable Adhesives," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 4048–4055.
6. R. Bruhwiler, B. Goldberg, N. Doshi, O. Ozcan, N. Jafferis, M. Karpelson, and R. J. Wood, "Feedback Control of a Legged Microrobot with On-board Sensing," *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Hamburg, Germany, 2015, pp. 5727–5733.
7. D. W. Haldane and R. S. Fearing, "Running Beyond the Bio-inspired Regime," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 4539–4546.
8. S. Yim and S. Kim, "Origami-Inspired Printable Tele-Micromanipulation System," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 2704–2709.
9. F. El-Moukaddem, E. Torng, G. Xing, E. Torng, G. Xing, and G. Xing, "Mobile Relay Configuration in Data-Intensive Wireless Sensor Networks," *IEEE Trans. Mob. Comput.*, vol. 12, no. 2, pp. 261–273, Feb. 2013.
10. K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme, "Robomote: Enabling Mobility in Sensor Networks*," 4th Int. Symp. on IPSN*, Piscataway, NJ, 2005, pp. 404-409.
11. P. Vartholomeos, K. Vlachos, and E. Papadopoulos, "Analysis and Motion Control of a Centrifugal-Force Microrobotic Platform," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 3, pp. 545–553, Jul. 2013.
12. A. W. Mahoney and J. J. Abbott, "Five-degree-of-freedom Manipulation of an Untethered Magnetic Device in Fluid Using a Single Permanent Magnet with Application in Stomach Capsule Endoscopy," *Int. J. Robot. Res.*, 2015, doi:10.1177/0278364914558006.
13. S. Yim, E. Gultepe, D. H. Gracias, and M. Sitti, "Biopsy using a Magnetic Capsule Endoscope Carrying, Releasing, and Retrieving Untethered Microgrippers*," IEEE Trans. Biomed. Eng.*, vol. 61, no. 2, pp. 513–521, Feb. 2014.

14. T. Suzuki, R. Sugizaki, K. Kawabata, Y. Hada, and Y. Tobe, "Autonomous Deployment and Restoration of Sensor Network using Mobile Robots," *Int. J. Advanced Robotic Syst.,* 2010, doi:10.5772/9696.
15. Y. Liu and G. Nejat, "Multirobot Cooperative Learning for Semiautonomous Control in Urban Search and Rescue Applications," *J. Field Robotics*, vol. 33, no. 4, pp. 512–536, Jun. 2015.
16. B. Doroodgar, Y. Liu, and G. Nejat, "A Learning-Based Semi-Autonomous Controller for Robotic Exploration of Unknown Disaster Scenes While Searching for Victims," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2719–2732, Dec. 2014.
17. Z. Zhang, G. Nejat, H. Guo, and P. Huang, "A novel 3D sensory system for robot-assisted mapping of cluttered urban search and rescue environments," *Intel. Serv. Robotics*, vol. 4, no. 2, pp. 119–134, Apr. 2011.
18. R. Arezoumand, S. Mashohor, and M. H. Marhaban, "Efficient terrain coverage for deploying wireless sensor nodes on multi-robot system," *Intel. Serv. Robotics*, vol. 9, no. 2, pp. 163–175, Apr. 2016.
19. R. S. Fearing, "Challenges for Efffective Millirobots," *Int. Symp. on Micro-NanoMechatronics and Human Sci.*, Nagoya, 2006, pp. 1–5.
20. Z. Kashino, J. Y. Kim, G. Nejat, and B. Benhabib, "Spatiotemporal Adaptive Optimization of a Static-Sensor Network via a Non-Parametric Estimation of Target Location Likelihood," *IEEE Sensors J.*, vol. 17, no. 5, pp. 1479–1492, Mar. 2017.
21. A. Macwan, J. Vilela, G. Nejat, and B. Benhabib, "A Multirobot Path-Planning Strategy for Autonomous Wilderness Search and Rescue," *IEEE Trans. Cybern*., vol. 45, no. 9, pp. 1784–1797, Sep. 2015.
22. A. Macwan, G. Nejat, and B. Benhabib, "Target-Motion Prediction for Robotic Search and Rescue in Wilderness Environments," *IEEE Trans. Syst., Man, and Cybern., Part B*, vol. 41, no. 5, pp. 1287–1298, Oct. 2011.
23. M. S. Couceiro, D. Portugal, R. P. Rocha, and N. M. F. Ferreira, "Marsupial Teams of Robots: Deployment of Miniature Robots for Swarm Exploration under Communication Constraints," *Robotica*, vol. 32, no. 7, pp. 1017–1038, Oct. 2014.
24. S. Bergbreiter, "Effective and Efficient Locomotion for Millimeter-Sized Microrobots," *IEEE/RSJ Int. Conf. on Int. Robots and Syst.*, Nice, France, 2008, pp. 4030–4035.
25. M. D. Mackay, R. G. Fenton, and B. Benhabib, "Multi-camera active surveillance of an articulated human form – An implementation strategy," *Computer Vision Image Understanding*, vol. 115, no. 10, pp. 1395–1413, Oct. 2011.
26. A. Bakhtari, M. Mackay, and B. Benhabib, "Active-Vision for the Autonomous Surveillance of Dynamic, Multi-Object Environments," *J. Intell. Robot Syst*., vol. 54, no. 4, p. 567, Apr. 2009.
27. A. Bakhtari and B. Benhabib, "An Active Vision System for Multitarget Surveillance in Dynamic Environments," *IEEE Trans. Syst., Man, and Cybern., Part B,* vol. 37, no. 1, pp. 190–198, Feb. 2007.
28. A. Bakhtari, M. D. Naish, M. Eskandari, E. A. Croft, and B. Benhabib, "Active-vision-based multisensor surveillance - an implementation," *IEEE Trans. Syst., Man, and Cybern., Part C*, vol. 36, no. 5, pp. 668–680, Sep. 2006.
29. V. Shinde, A. Dutta, and A. Saxena, "Experiments on Multi-Agent Capture of a Stochastically Moving Object Using Modified Projective Path Planning," *Robotica*, vol. 31, no. 2, pp. 267–284, Mar. 2013.
30. W. Liu and A. F. T. Winfield, "Open-hardware E-puck Linux Extension Board for Experimental Swarm Robotics Research," *Microprocess. Microsyst.*, vol. 35, no. 1, pp. 60–67, Feb. 2011.
31. R. Gross, M. Bonani, F. Mondada, and M. Dorigo, "Autonomous Self-Assembly in Swarm-Bots," *IEEE Trans. Robot.*, vol. 22, no. 6, pp. 1115–1130, Dec. 2006.
32. J. McLurkin, A. McMullen, N. Robbins, G. Habibi, A. Becker, A. Chou, H. Li, M. John, N. Okeke, J. Rykowski, S. Kim, W. Xie, T. Vaughn, Y. Zhou, J. Shen, N. Chen, Q. Kaseman, L. Langford, J. Hunt, A. Boone, and K. Koch, "A Robot System Design for Low-cost Multi-Robot Manipulation," *IEEE/RSJ Int. Conf. on Intelli. Robots and Syst.*, Chicago, IL, 2014, pp. 912–918.
33. G. Habibi, Z. Kingston, W. Xie, M. Jellins, and J. McLurkin, "Distributed Centroid Estimation and Motion Controllers for Collective Transport by Multi-Robot Systems," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 1282–1288.
34. K. Lembke, L. Kietlinski, M. Golanski, and R. Schoeneich, "RoboMote: Mobile Autonomous Hardware Platform for Wireless Ad-hoc Sensor Networks," *IEEE Int. Symp. on Ind. Electron.*, Gdansk, Poland, 2011, pp. 940–944.
35. G. Caprari and R. Siegwart, "Mobile Micro-Robots Ready to Use: Alice," *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Edmonton, Canada, 2005, pp. 3295–3300.
36. S. Kornienko and S. Kornienko, "IR-Based Communication and Perception in Microrobotic Swarms," ArXiv11093617 Cs, Sep. 2011.
37. K. S. Farshad Arvin, "Development of a Miniature Robot for Swarm Robotic Application," *Int. J. of Comput. Electr. Eng.*, vol. 1, 2009, doi:10.7763/IJCEE.2009.V1.67.
38. A. Kettler, M. Szymanski, and H. Wörn, "The Wanda Robot and Its Development System for Swarm Algorithms," *Advances in Auton. Mini Robots*, U. Rückert, S. Joaquin, and W. Felix, Eds. Berlin, Germany: Springer, 2012, pp. 133–146.
39. H. B. Jang, R. D. Villalba, D. Paley, and S. Bergbreiter, "RSSI-based Rendezvous on the Tiny Terrestrial Robotic Platform (TinyTeRP)," *Inst. Syst. Research and Tech. Rep.*, Univ. Maryland, Aug. 2013.
40. A. P. Sabelhaus, D. Mirsky, L. M. Hill, N. C. Martins, and S. Bergbreiter, "TinyTeRP: A Tiny Terrestrial Robotic Platform with Modular Sensing," *IEEE Int. Conf. on Robot. and Autom.*, Karlsruhe, Germany, 2013, pp. 2600–2605.
41. D. Pickem, M. Lee, and M. Egerstedt, "The GRITSBot in Its Natural Habitat - A Multi-Robot Testbed," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 4062–4067.
42. M. Rubenstein, C. Ahler, N. Hoff, A. Cabrera, and R. Nagpal, "Kilobot: A Low Cost Robot with Scalable Operations Designed for Collective Behaviors," *Robot. Auton. Syst.*, vol. 62, no. 7, pp. 966–975, Jul. 2014.
43. F. Arvin, J. Murray, C. Zhang, and S. Yue, "Colias: An Autonomous Micro Robot for Swarm Robotic Applications," *Int. J. Adv. Robot. Syst.*, vol. 11, no. 1, 2014.

44. M. Le Goc, L. H. Kim, A. Parsaei, J.-D. Fekete, P. Dragicevic, and S. Follmer, "Zooids: Building Blocks for Swarm User Interfaces," in *Annu. Symp. User Interface Software Technology*, New York, NY, 2016, pp. 97–109.
45. Gctronic, "Mobile Robot Products." [Online]. Available: http://www.gctronic.com/products.php. [Accessed: 17-Feb-2016].
46. K-Team Corporation, "K-Team Mobile Robot Products." [Online]. Available: http://www.k-team.com/mobile-robotics-products. [Accessed: 17-Feb-2016].
47. J.Y. Kim, T. Colaco, Z. Kashino, G. Nejat, B. Benhabib, "mROBerTO: A Modular Millirobot for Swarm-Behavior Studies," *IEEE/RSJ Int. Conf. on Intelli. Robots and Syst.*, Daejeon, Korea, 2016.
48. Nordic Semiconductor. (2018). *nRF51422 / ANT™ / Products / Home - Ultra Low Power Wireless Solutions from NORDIC SEMICONDUCTOR*. [online] Available at: https://www.nordicsemi.com/eng/Products/ANT/nRF51422 [Accessed 5 Jan. 2018].
49. Precisionmicrodrives.com. (2018). *4mm DC Motor - 8mm Type | Precision Microdrives*. [online] Available at: https://www.precisionmicrodrives.com/product/104-001-4mm-dc-motor-8mm-type [Accessed 5 Jan. 2018].
50. R. Drisdelle, Z. Kashino, L. Pineros, J. Y. Kim, G. Nejat, and B. Benhabib, "Motion Control of a Wheeled Millirobot," in *Proc. Int. Conf. Control, Dynamic Systems, Robotics*, Toronto, ON, Canada, 2017, pp. 124-1–124-6.
51. N. Farrow, J. Klingner, D. Reishus and N. Correll, "Miniature six-channel range and bearing system: Algorithm, analysis and experimental validation," *2014 IEEE Int. Conf. on Robot. and Autom.*, Hong Kong, China, 2014, pp. 6180-6185.
52. Developer.nordicsemi.com. (2018). *download.recurser.com*. [online] Available at: http://developer.nordicsemi.com/nRF5_SDK/ [Accessed 5 Jan. 2018].
53. Developer.nordicsemi.com. (2018). *nRF5 SDK Documentation*. [online] Available at: http://developer.nordicsemi.com/nRF5_SDK/doc/ [Accessed 5 Jan. 2018].
54. L. Bayındır, "A Review of Swarm Robotics Tasks," *Neurocomputing*, vol. 172, pp. 292–321, Jan. 2016.
55. J. C. Barca and Y. A. Sekercioglu, "Swarm Robotics Reviewed," *Robotica*, vol. 31, no. 3, pp. 345–359, May 2013.
56. M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm Robotics: A Review from the swarm Engineering Perspective," *Swarm Intell.*, vol. 7, no. 1, pp. 1–41, Jan. 2013.
57. W. M. Spears, D. F. Spears, J. C. Hamann, and R. Heil, "Distributed, Physics-Based Control of Swarms of Vehicles," *Auton. Robots*, vol. 17, no. 2–3, pp. 137–162, Sep. 2004.
58. Y. Jin and Y. Meng, "Morphogenetic Robotics: An Emerging New Field in Developmental Robotics," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 41, no. 2, pp. 145–160, Mar. 2011.
59. V. Trianni, R. Groß, T. H. Labella, E. Şahin, and M. Dorigo, "Evolving Aggregation Behaviors in a Swarm of Robots," *Advances in Artificial Life*, W. Banzhaf, J. Ziegler, T. Christaller, P. Dittrich, and J. T. Kim, Eds. Springer Berlin Heidelberg, 2003, pp. 865–874.
60. G. Francesca, M. Brambilla, V. Trianni, M. Dorigo, and M. Birattari, "Analysing an Evolved Robotic Behaviour Using a Biological Model of Collegial Decision Making," *Animals to Animats 12*, T. Ziemke, C. Balkenius, and J. Hallam, Eds. Springer Berlin Heidelberg, 2012, pp. 381–390.
61. M. Gauci, J. Chen, T. J. Dodd, and R. Groß, "Evolving Aggregation Behaviors in Multi-Robot Systems with Binary Sensors," *Distrib. Auton. Robot. Syst.*, M. A. Hsieh and G. Chirikjian, Eds. Springer Berlin Heidelberg, 2014, pp. 355–367.
62. O. Soysal and E. Sahin, "Probabilistic Aggregation Strategies in Swarm Robotic Systems," *IEEE Symp. on Swarm Intell.*, Pasadena, CA, 2005, pp. 325–332.
63. S. Nouyan, R. Groß, M. Bonani, F. Mondada, and M. Dorigo, "Teamwork in Self-Organized Robot Colonies," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 695–711, Aug. 2009.
64. A. Howard, M. J. Matarić, and G. S. Sukhatme, "Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem," *Distri. Auton. Robot. Syst. 5*, H. Asama, T. Arai, T. Fukuda, and T. Hasegawa, Eds. Springer Japan, 2002, pp. 299–308.
65. "Dynamic Task Assignment in Robot Swarms | Multi-Robot Systems Lab - Rice University, Houston TX." [Online]. Available: http://mrsl.rice.edu/papers/dynamic-task-assignment-robot-swarms. [Accessed: 21-Apr-2016].
66. R. Doriya, S. Mishra, and S. Gupta, "A Brief Survey and Analysis of Multi-Robot Communication and Coordination," *Int. Conf. on Comput., Commun. Autom.*, Uttar Pradesh, 2015, pp. 1014–1021.
67. T. AbuKhalil, T. Sobh, and M. Patil, "Survey on Decentralized Modular Robots and Control Platforms," *Innovations and Advances in Comput, Informat., Syst. Sci., Netw. and Eng.*, T. Sobh and K. Elleithy, Eds. Springer International Publishing, 2015, pp. 165–175.
68. S. Camazine, J. Deneubourg, N. Franks, J. Sneyd, G. Theraula, and E. Bonabeau, *Self-Organization in Biological Systems*, Princeton University Press, 2003.
69. A. Okubo, "Dynamical Aspects of Animal Grouping: Swarms, Schools, Flocks, and Herds," *Adv. Biophys.*, vol. 22, pp. 1–94, Jan. 1986.
70. J. K. Parrish, S. V. Viscido, and D. Grünbaum, "Self-Organized Fish Schools: An Examination of Emergent Properties," *Biol. Bull.*, vol. 202, no. 3, pp. 296–305, Jun. 2002.
71. O. Soysal and E. Şahin, "A Macroscopic Model for Self-organized Aggregation in Swarm Robotic Systems," *Swarm Robot.*, E. Şahin, W. M. Spears, and A. F. T. Winfield, Eds. Springer Berlin Heidelberg, 2006, pp. 27–42.
72. J. Hereford, "Analysis of BEECLUST swarm algorithm," *IEEE Symp. on Swarm Intell.*, Paris, France, 2011, pp. 1–7.
73. V. G. Santos and L. Chaimowicz, "Cohesion and segregation in swarm navigation," *Robotica*, vol. 32, no. 2, pp. 209–223, Mar. 2014.
74. "Formations of Robotic Swarm: An Artificial Force Based Approach." [Online]. Available:

http://connection.ebscohost.com/c/articles/37378693/formations-robotic-swarm-artificial-force-based-approach. [Accessed: 25-May-2016].
75. S. S. Ge and Y. J. Cui, "Dynamic Motion Planning for Mobile Robots Using Potential Field Method," *Auton. Robots*, vol. 13, no. 3, pp. 207–222, Nov. 2002.
76. K. Derr and M. Manic, "Extended Virtual Spring Mesh (EVSM): The Distributed Self-Organizing Mobile Ad Hoc Network for Area Exploration," *IEEE Trans. Ind. Electron.*, vol. 58, no. 12, pp. 5424–5437, Dec. 2011.
77. W. M. Spears and D. F. Spears, *Physicomimetics: Physics-Based Swarm Intelligence*. Heidelberg, NY: Springer, 2012.
78. D. V. Dimarogonas and K. J. Kyriakopoulos, "Connectedness Preserving Distributed Swarm Aggregation for Multiple Kinematic Robots," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1213–23, Oct. 2008.
79. V. Gazi, "Swarm Aggregations Using Artificial Potentials and Sliding-Mode Control," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1208–1214, 2005.
80. H. Hamann, H. Worn, K. Crailsheim, and T. Schmick, "Spatial Macroscopic Models of a Bio-Inspired Robotic Swarm Algorithm," *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Nice, France, 2008, pp. 1415–1420.
81. B. Yang, Y. Ding, Y. Jin, and K. Hao, "Self-Organized Swarm Robot for Target Search and Trapping Inspired by Bacterial Chemotaxis," *Robot. Auton. Syst.*, vol. 72, pp. 83–92, Oct. 2015.
82. R. C. Arkin, *Behavior-Based Robotics*. Cambridge, MA: A Bradford Book, 1998.
83. S. Nouyan, A. Campo, and M. Dorigo, "Path Formation in a Robot Swarm," *Swarm Intell.*, vol. 2, no. 1, pp. 1–23, Dec. 2007.
84. P. M. Maxim, W. M. Spears, and D. F. Spears, "Robotic Chain Formations," *Proc. IFAC Workshop on Networked Robotics*, Golden, CO, pp. 19–24, 2009.
85. V. Sperati, V. Trianni, and S. Nolfi, "Self-Organised Path Formation in a Swarm of Robots," *Swarm Intell.*, vol. 5, no. 2, pp. 97–119, Apr. 2011.
86. S. M. Lee and H. Myung, "Receding Horizon Particle Swarm Optimisation-Based Formation Control with Collision Avoidance for Non-Holonomic Mobile Robots," *IET Control Theory Applications*, vol. 9, no. 14, pp. 2075–2083, 2015.
87. E. Osherovich, V. Yanovki, I. A. Wagner, and A. M. Bruckstein, "Robust and Efficient Covering of Unknown Continuous Domains with Simple, Ant-Like A(ge)nts," *Int. J. Robot. Res.*, vol. 27, no. 7, pp. 815–831, Jul. 2008.
88. T. Kuyucu, I. Tanev, and K. Shimohara, "Evolutionary Optimization of Pheromone-Based Stigmergic Communication," *Applicat. of Evol. Comput.*, C. D. Chio, *et al.*, Eds. Springer Berlin Heidelberg, 2012, pp. 63–72.
89. I. A. Wagner, M. Lindenbaum, and A. M. Bruckstein, "Distributed Covering by Ant-Robots Using Evaporating Traces," *IEEE Trans. Robot. Autom.*, vol. 15, no. 5, pp. 918–933, Oct. 1999.
90. J. Svennebring and S. Koenig, "Building Terrain-Covering Ant Robots: A Feasibility Study," *Auton. Robots*, vol. 16, no. 3, pp. 313–332, May 2004.
91. B. Ranjbar-Sahraei, G. Weiss, and A. Nakisaee, "A Multi-Robot Coverage Approach Based on Stigmergic Communication," *Multiagent Syst. Technol.*, I. J. Timm and C. Guttmann, Eds. Springer Berlin Heidelberg, 2012, pp. 126–138.
92. S. Poduri and G. S. Sukhatme, "Constrained Coverage for Mobile Sensor Networks," *IEEE Trans. Robot. Autom.*, 2004, vol. 1, pp. 165–171 Vol.1.
93. E. Ugur, A. E. Turgut, and E. Sahin, "Dispersion of a Swarm of Robots Based on Realistic Wireless Intensity Signals," *22nd Int. Symp. on Comput. and Inf. Sci.*, Melbourne, Australia, 2007, pp. 1–6.
94. E. Castello, T. Yamamoto, F. D. Libera, W. Liu, A. F. T. Winfield, Y. Nakamura, and H. Ishiguro, "Adaptive Foraging for Simulated and Real Robotic Swarms: The Dynamical Response Threshold Approach," *Swarm Intell*, pp. 1–31, Jan. 2016.
95. J. P. Hecker, J. C. Carmichael, and M. E. Moses, "Exploiting Clusters for Complete Resource Collection in Biologically-Inspired Robot Swarms," *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Hamburg, Germany, 2015, pp. 434–440.
96. N. R. Hoff, A. Sagoff, R. J. Wood, and R. Nagpal, "Two Foraging Algorithms for Robot Swarms Using Only Local Communication," *IEEE Int. Conf. on Robot. Biomimetics*, Tianjin, RPC, 2010, pp. 123–130.
97. K. Lerman, C. Jones, A. Galstyan, and M. J. Matarić, "Analysis of Dynamic Task Allocation in Multi-Robot Systems," *Int. J. Robot. Res.*, vol. 25, no. 3, pp. 225–241, Mar. 2006.
98. R. Ramaithitima, M. Whitzer, S. Bhattacharya, and V. Kumar, "Sensor Coverage Robot Swarms Using Local Sensing without Metric Information," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 3408–3415.
99. Y. Ou, P. Kang, M. J. Kim, and A. A. Julius, "Algorithms for Simultaneous Motion Control of Multiple T. Pyriformis Cells: Model Predictive Control and Particle Swarm Optimization," *IEEE Int. Conf. on Robot. and Autom.*, Seattle, WA, 2015, pp. 3507–3512.
100. L. Barnes, M.-A. Fields, and K. Valavanis, "Unmanned Ground Vehicle Swarm Formation Control Using Potential Fields," *Mediterranean Conf. on Control Autom.*, 2007, pp. 1–8.
101. R. Oikawa, M. Takimoto, and Y. Kambayashi, "Distributed Formation Control for Swarm Robots Using Mobile Agents," *IEEE 10th Jubilee Int. Symp. on Appl. Computational Intell. Informatics*, Timisoara, Romania, 2015, pp. 111–116.
102. G. Goertzel, "An Algorithm for the Evaluation of Finite Trigonometric Series," *Am. Math. Mon.*, 1958, vol. 65, no. 1, pp. 34–35.
103. J. Pugh and A. Martinoli, "Relative Localization and Communication Module for Small-Scale Multi-Robot Systems," *IEEE Int. Conf. on Robot. and Autom.*, Orlando, FL, 2006, pp. 188–193.
104. A. Gutierrez, A. Campo, M. Dorigo, J. Donate, F. Monasterio-Huelin, and L. Magdalena, "Open E-puck Range & Bearing Miniaturized Board for Local Communication in Swarm Robotics," *IEEE Int. Conf. on Robot. and Autom.*, Kobe, Japan,

2009, pp. 3111–3116.
105. Pan Li, N. Scalabrino, Yuguang Fang, E. Gregori, and I. Chlamtac, "How to Effectively Use Multiple Channels in Wireless Mesh Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 20, no. 11, pp. 1641–1652, Nov. 2009.
106. R. N. Bracewell, "The Fast Hartley Transform," *Proceedings of the IEEE*, vol. 72, no. 8, pp. 1010–1018, Aug. 1984.

Appendix A: Spec Sheet of *mROBerTO*

This appendix provides the detailed specifications of *mROBerTO*, Table A1.

Table A1. *mROBerTO's* Spec Sheet

Size and Weight	
Dimensions	16 × 16 × 32 mm^3 (L × W × H)
Weight	9.6 g
Processor, Sensors, and Peripherals	
Processor	ARM Cortex-M0 (32-bit @ 16 MHz)
Memory	32 KB RAM and 256 KB flash
Inertial Measurement Units	3D gyroscope, 3D accelerometer, and 3D magnetometer
Front Range	Time-of-flight proximity sensor (measures up to 255 mm) and ambient light sensor
Front Camera	CMOS VGA (allows up to 640 × 480 @ 30 FPS)
Proximities and Bearings	Multi-channel IR based communication module
Visual User Interface	RGB LED (top side)
Locomotion and Speed	
Maximum Speed	150 mm/s
Minimum Speed	1 mm/s
Type of Locomotion	Differential drive with 2 wheels
Battery and Power Management	
Operation Time	Approximately 3 hours (ranges from 1 to 6 hours)
Battery Type	Lithium polymer 3.7 V nominal with 120 mAh
Charge Time	1 hour
Wireless Communication	
Communication Type (RF)	Selectable or concurrent operation of BLE 4.2 (up to 1 Mbps) or ANT™ (up to 128 Kbps)
Maximum Range (RF)	< 0.5 m
Communication Type (IR)	Multi-channel infrared based (up to 2 × 105 bps)
Maximum Range (IR)	< 150 mm
Software and Debugging	
IDE	Eclipse with ARM GCC compiler
Programmer/Debugger	SEGGER's J-Link
Programming (Over-the-Air)	Available
Operating Environment	
Temperature	0 °C to 55 °C
Workspace Surface	Flat with no to little roughness (e.g., whiteboard and acrylic glass)

Appendix B: Cost Comparison to *Kilobot*

For *Kilobot*s, the developers state a unit cost of about $14 for parts, when 1000 units or more are purchased.[42] However, the cost per unit for parts would be approximately $50 for a purchase of parts for 100 units. The *Kilobot*s also require an overhead controller that costs $588 in order to communicate with, calibrate, and program the *Kilobot*s (both wired and wirelessly). Assembled and preprogrammed *Kilobot*s are commercially available, today, for about $150 per unit.

The *mROBerTO*'s cost is about $60 in parts per unit, when 25 units' worth of parts are purchased: the processing and communication module costs about $7, the locomotion module costs $14, the proximity-

sensing module costs $14, the swarm-sensing module costs $12, and the battery packs cost $13.

Appendix C: Sensing and Communication for Collective Behavior

The ability to locally measure the relative distances and bearings of neighboring robots is fundamental for collective behavior in swarm robotics.[98-101] For *mROBerTOs*, the swarm-sensing module, with IR emitters and detectors, was specifically designed and implemented for this purpose. *mROBerTO* sends out modulated IR signals via its IR emitters, achieved using the PWM feature on the ATmega328P microcontroller. The modulated IR signals are encoded with the robot's unique ID and are capable of supporting up to 127 unique IDs. The six wide-viewing-angle (120°) IR phototransistors' voltage values are constantly read by the ADC block of the microcontroller. The incoming modulated IR signals are demodulated using the Goertzel Algorithm (GA), through software on the microcontroller, in order to decode the incoming robot's ID and determine the presence of nearby robots.[102]

The bearings of nearby robots are determined by weighting the scaled IR intensities received with the six IR detectors, as described in.[103] The following equation is used to estimate the bearing of the nearby robots:

$$\hat{\varphi} = \arctan\left(\frac{\sum_{i=1}^{6} \rho_i \cdot \sin(\beta_i)}{\sum_{i=1}^{6} \rho_i \cdot \cos(\beta_i)}\right), \tag{C1}$$

where $\hat{\varphi}$ is the estimated bearing value of the nearby robot with respect to its heading, i is the sensor position, i = 1 to 6, β_i is the angular distance between the i^{th} sensor and the robot's heading (i.e., 0°, 60°, 120°, …, 300°), and ρ_i is the scaled IR intensity reading of the i^{th} sensor.

In order to estimate the relative distances of neighboring robots, the scaled IR intensities are summed and used to determine an approximate distance value. A translator between summed scaled IR intensities to relative distance was created through an offline calibration procedure where the summed IR intensities from a detector were recorded at different positions and angles relative to another robot acting as an IR signal source. The physical arrangement of the calibration process follows the same procedure that was used in ref. [104] for a modelling test, where values of IR intensities were recorded at several different positions and angles using one IR emitter and one IR detector in order to characterize the IR based localization system. The calibration procedure for the swarm modules can be conducted up to the maximum distance between the center of the IR emitter to the center of the IR detector, defined to be 125 mm in our case, which is approximately near the end of the detection range of the IR detectors.

mROBerTO's current microcontroller, with the use of GA, can read up to 105 bits per second (bps) on two separate IR channels (210 bps in total). In order to enable the use of multiple IR channels and remove noise from the environment, the IR signals are modulated using amplitude modulation (amplitude-shift keying).

Multiple channels are desired to allow for scalability for communication among robots at close proximities while increasing the data transmission throughput on the same type of network[105]. However, the main means of communication among robots remains BLE and ANT™ wireless communication, for a more reliable and larger transfer of data. Lastly, the power of the wireless transmission for both BLE and ANT™ can be controlled through software and dynamically be changed during runtime, where the power can range from -40dBm to +4dBm. This is useful if the communication radii of one or more robots were needed to be changed for testing a specific swarm scenario and to save power during deployment if the robots were to become idle.

During our research, several alternate approaches and designs were considered before reaching the final configuration of the swarm sensor module for IR communication described above. Firstly, instead of the GA, the Fast Hartley Transformation (FHT)[106] was initially utilized for IR communication demodulation. Although the FHT provides better overall amplitude resolution than would the GA, the achievable update frequency with FHT is only 3 Hz compared to the 105 Hz achievable with the GA. In

order to increase the overall performance of the GA and FHT implementation, fixed-point numbers, 12 bits in size, were used instead of floating point numbers (32 bits long) on the microcontroller, to utilize the 16-bit integer multiplier hardware in the AVR processor. Experiments showed that by making use of the hardware multiplier, the update frequency could almost be doubled.

Secondly, the original processor clock speed on the ATmega328P, located on the swarm-sensing module for swarm purposes and used to carry out IR communication demodulation, was set to 8 MHz. In order to increase performance of this module, without the need for additional components, the processor clock was overclocked from 8 to 12 MHz. However, consequently, a new calibration process needed to be carried out to ensure that the internal clock speed is indeed set to 12 MHz.

Since the voltage sources for the proximity and swarm-sensing modules only provide up to 2.8 V, adding and using a 16 MHz external oscillator for faster clock speed was not an option. A minimum of 4.5 V is required to use the 16 MHz external oscillator. Thus, the internal oscillator was overclocked instead.

Lastly, our first version of the swarm-sensing module for swarm purposes consisted of narrow angle (i.e., viewing angle of less than 70°) IR phototransistors for detecting the IR signals. However, as stated in ref. [36], *dead zone* issues were noted, where IR signals incoming at angles between two IR phototransistors in the center area were not detected due to the low-viewing-angles of the phototransistors. One solution considered to resolve dead zone areas was to add more IR phototransistors for all around coverage to the original six IR phototransistors. However, several problems were noted with this approach. Firstly, the ATmega328P only has up to eight ADC input pins and two out of the eight ADC pins are also configured to be TWI data and clock pins which leaves one with six usable ADC pins unless a different communication interface is utilized between the ATmega328P and the SoC. Secondly, the ATmega328P is restricted to be used with only up to eight IR phototransistors. Thirdly, adding more IR phototransistors would require extra clock cycles from the microcontroller to read the additional ADC pins' voltages as well as analyze more ADC data using GA. Finally, with additional IR phototransistors, the IR data transmission rate would linearly decrease.

In order to overcome the abovementioned problems, wide-angle IR phototransistors with a viewing angle of 120° were used. The tradeoff of using wide angle versus narrow angle IR phototransistor is that the wide angle one, generally, has less sensing range than the narrow angle. However, the wide angle IR phototransistor, in our case, could detect other neighboring robots beyond 125 mm, which is more than enough distance for carrying out meaningful collective behavior.

Appendix D: Swarm Experiment Procedures

This appendix provides more detailed descriptions of the experimental procedures used for the four swarm scenarios introduced in Section III.

D.1. Aggregation Behavior

The aggregation behavior was implemented on *mROBerTOs* using the method from ref. [62] with the behavior-based approach adapted from ref. [82]. However, since there were no explicit formulae for the motor schemas in these papers and other papers, in our work, we adopted an artificial potential method to move the robots. The approach state for aggregating was achieved using an artificial attractive force. The equation below defines the artificial attractive force, $\boldsymbol{F}_{att}$, used for the approach state when a robot looks for nearby robots and move towards the group:

$$\boldsymbol{F}_{att,i}(d_i,\theta_i)=\begin{cases}0 & d_i \le d_{limit}\\ \zeta(d_i-d_{limit})\begin{bmatrix}\cos(\theta_i)\\ \sin(\theta_i)\end{bmatrix} & d_{limit}<d_i\le d_{close}\\ d_{close}\,\zeta\begin{bmatrix}\cos(\theta_i)\\ \sin(\theta_i)\end{bmatrix} & d_{close}<d_i<d_{far}\\ 0 & d_i \ge d_{far}\end{cases}, \tag{D1}$$

$$\boldsymbol{F}_{att}=\sum_{i=1}^{N}\boldsymbol{F}_{att,i}, \tag{D2}$$

Where d_i and θ_i are the i^{th} nearby robot's relative distance and bearing, respectively, ζ is the scaling factor for the attractive force, d_{limit} is distance limit value of how close the robot can get, d_{close} is the transitioning value from conic to parabolic artificial potential well, d_{far} is the IR sensing radius limit of the robot, set to 125 mm in our work, and N is the total number of nearby robots that are within the sensing radius (125 mm).

An additional behavior, referred to as *avoid obstacle* state herein, was implemented in order to avoid collisions with other robots (treated as obstacles). The following artificial repulsive force, $\boldsymbol{F}_{obs}$, was used to achieve this:

$$\boldsymbol{F}_{obs,i}(d_i,\theta_i)=\begin{cases}\left(\frac{1}{d_i}-\frac{1}{d_{threshold}}\right)\begin{bmatrix}\cos(\theta_i)\\ \sin(\theta_i)\end{bmatrix} & d_i < d_{threshold}\\ 0 & d_i \ge d_{threshold}\end{cases}, \tag{D3}$$

$$\boldsymbol{F}_{obs}=\sum_{i=1}^{N}\boldsymbol{F}_{obs,i}, \tag{D4}$$

where η is the scaling factor for the repulsive force and $d_{threshold}$ is the threshold distance value where the robot tries to avoid obstacles if it were to be closer than the threshold distance value. One can note that this state is only active when $d_i < d_{threshold}$ and the magnitude of the repulsive velocity increases as relative distance to nearby robot decreases.

Examining Equations (D1) to (D4), one can note that the approach state is activated when an i^{th} nearby robot's relative distance is in between $d_{limit} < d_i < d_{far}$ and avoid obstacle state is activated when $d_i < d_{threshold}$. If $d_{limit} < d_{threshold}$, then, there exists a range of relative distance values, d_i, when both approach state and avoid obstacle state can be active simultaneously (i.e., when the i^{th} nearby robot is in between $d_{limit} < d_i < d_{threshold}$). This is an invalid state and must be avoided since only one out of the two artificial forces should be active per robot. In order to prevent this invalid state from occurring, we employ the following constraint: $d_{limit} \ge d_{threshold}$.

However, the IR detectors on the swarm-sensing module that are responsible for getting the relative distance values of nearby robots are susceptible to noise. Thus, if the relative distance value, d_i for a nearby robot is close to the values of d_{limit} and $d_{threshold}$, oscillatory motion could be experienced. Namely, the robot could frequently switch from approach state to avoid obstacle state. In order to prevent this, the above constraint can be changed to $d_{limit} > d_{threshold}$ so that the robot stays stationary and the movement vector is zero if the i^{th} nearby robot is within the relative distance range of $d_{limit} > d_i > d_{threshold}$. In addition, one can set the difference between d_{limit} and $d_{threshold}$ to a large value for better robustness to the oscillatory

motion.

D.2. Chain Formation

The second collective behavior demonstrated by *mROBerTOs* is chain formation using the method from ref. [63]. Prior to the start of the chain formation experiment, one designated robot is set to be the nest of the chain (i.e., the very first robot in the chain formation) with the robot illuminating its green LED. The rest of the robots are in the wait state, where the robots stay put indefinitely until chosen to be the next tail of the chain (i.e., next robot to become part of the chain). Before the experiment starts, the current nest of the chain is also the current tail of the chain. The chain formation procedure is as follows,

a. A robot in the *wait* state within the group is randomly chosen, initially from the front column and then the back, to be the next *tail* of the chain.

b. The chosen robot moves towards the existing chain, or towards the nest if chain has yet to be formed.

c. Once the new to be tail robot is close to the chain, it starts moving perpendicular in the clockwise direction along the chain.

d. When the robot reaches the end of the chain, it rotates around the end (*tail*) of the chain until the desired orientation relative to the end (*tail*) of the chain is reached.

e. Once the desired angle is reached, this robot is now the new *tail* of the chain (the new *tail* robot is indicated with green LED). The new *tail* robot will relay a message to the robots that are in the *wait* state in order to randomly select a new robot to continue with increasing the chain size. This procedure repeats until no robots are left in the group in the *wait* state and all robots are part of the chain.

Similar to the aggregation behavior, ref. [63] discusses all the required motor schemas and behaviors in order to achieve chain formation but does not explicitly express which approach is used to create the movement vectors of the robots. Thus, as for the aggregation behavior, the artificial potential approach was adopted in our work to determine the next movement vectors for the robots in order to achieve chain formation.

Reviewing the previous list of procedures for achieving chain formation, one can note that there are two motor schemas we require: move towards a robot (i.e., nest of the chain) and move perpendicular to the robots. Moving towards a robot can be implemented using Equation (C2). As for moving perpendicular to robots in order to move along the chain, we need an artificial force vector that points perpendicular to a robot in the clockwise direction. In addition, this artificial perpendicular force vector needs to keep the robot at a fixed radius to other robots in the chain so that it does not drift away from the chain or get too close and collide into the chain of robots.

When the robot moves along a chain, it should latch onto a target robot and move perpendicular to that robot with a fixed radius until it gets close enough to the next robot on the chain. This can be seen in Fig. 5c, where the moving robot first latches onto the nest robot and moves in a perpendicular direction until it reaches a certain distance to the second robot in the chain. Next, the moving robot latches onto the second robot in the chain and it moves in the perpendicular direction until end of the chain is reached. In summary, the moving robot latches onto different robots in the chain by continuously changing its target robot in order to move along the chain until the end of the chain is reached.

The equation below was used to move the robot perpendicular to a chosen target robot (i.e., a robot to latch onto) with a fixed range of radius between the target robot and the robot itself:

$$\boldsymbol{F}_{perp}(d,\theta)=\begin{cases}\eta\left(\frac{1}{d}-\frac{1}{d_{limit}}\right)\begin{bmatrix}c_\theta\\ s_\theta\end{bmatrix}+\rho\begin{bmatrix}c_{\theta+90^\circ}\\ s_{\theta+90^\circ}\end{bmatrix} & d\le d_{limit}\\ \rho\begin{bmatrix}c_{\theta+90^\circ}\\ s_{\theta+90^\circ}\end{bmatrix} & d_{limit}<d\le d_{perp}\\ \zeta(d-d_{limit})\begin{bmatrix}c_\theta\\ s_\theta\end{bmatrix}+\rho\begin{bmatrix}c_{\theta+90^\circ}\\ s_{\theta+90^\circ}\end{bmatrix} & d_{perp}<d\le d_{close}\\ d_{close}\,\zeta\begin{bmatrix}c_\theta\\ s_\theta\end{bmatrix}+\rho\begin{bmatrix}c_{\theta+90^\circ}\\ s_{\theta+90^\circ}\end{bmatrix} & d_{close}<d\end{cases}, \tag{D5}$$

where $\boldsymbol{F}_{perp}$ is the next force movement vector, d and θ are relative distance and bearing of the target robot, c_θ and s_θ are the cosine and sine functions of θ respectively, η and ζ are the scaling factors for repulsive and attractive forces respectively, and ρ is the weight parameter for the perpendicular movement vector.

In order to prevent collisions, an artificial repulsive force was added into the above equation and is used when the two robots are closer than d_{limit}. In addition, in order to prevent the moving robot drifting away from the target robot and the chain itself, an artificial attractive force was added into the equation and is used when the distance between the moving and target robots are $d > d_{perp}$. Lastly, when the distance between moving and target robots are within the ideal distance range (i.e., $d_{limit} < d \le d_{perp}$), the robot moves only in the perpendicular direction.

D.3. Collective Exploration

The third collective behavior demonstrated by *mROBerTOs* was the collective exploration using the same method in ref. [64]. Initially, the robots are placed relatively close to one another in a corner of the walled arena (Fig. 7a). With the use of artificial repulsive forces, the robots move away from both each other and the walls (obstacles) around them when the experiment begins. The equations, with artificial forces, expressed in ref. [64] were used to implement the artificial repulsive forces used in the collective exploration behavior:

$$\boldsymbol{F}=\boldsymbol{F_o}+\boldsymbol{F_n}, \tag{D6}$$

where $\boldsymbol{F_o}$ and $\boldsymbol{F_n}$ are repulsive forces for obstacles (i.e., walls) and other mobile robots respectively, and

$$\boldsymbol{F_o}=-\nabla U_o=-k_o\sum_i\frac{1}{d_{o,i}^2}\cdot\frac{\boldsymbol{d_{o,i}}}{d_{o,i}}, \tag{D7}$$

$$\boldsymbol{F_n}=-k_n\sum_i\frac{1}{d_{n,i}^2}\cdot\frac{\boldsymbol{d_{n,i}}}{d_{n,i}}. \tag{D8}$$

Above, k_o and k_n are the scaling factors for the repulsive forces of obstacles and other mobile robots, and $d_{o,i}/d_{n,i}$ is the relative distance between the obstacles/robots.

D.4. Dynamic Task Allocation

The fourth collective behavior demonstrated by *mROBerTOs* was the dynamic task allocation using the method in ref. [65], and more specifically, using the *Card Dealer's* algorithm. Before the algorithm can be applied, the robots that need to be assigned a task must be within communication range in a mesh type network where the links between each robot can be represented by a directed graph $\mathcal{G}$, and the nodes represent the robots themselves and edges represent the communication links. Furthermore, the robots

must be able to measure relative distances of nearby robots in order to determine their presence, which determine how long each communication period will last. Namely, if there are more robots present in the group than a situation where there are fewer robots, the communication period between robots should be longer for the case where there are more robots. In addition, all robots in the group must have a unique ID assigned.

The *Card Dealer's* algorithm is completed in a series of stages. At the end of each stage, one of the robots that participated in the stage is dealt a task and the next stage begins with the robot that has just received a new task being eliminated from upcoming future stages. This cycle repeats until all robots within the group are assigned a task.

In each stage, a suppression technique is applied in order to select which robot gets assigned a task at the end of the stage. This technique is achieved by having all the robots without an assigned task broadcast their unique ID in the beginning of a new stage so that everyone in the group is aware of each robot's unique ID. During this stage, when a robot receives a broadcast message from another robot with a lower unique ID value than its own, it stops broadcasting its own ID and begins broadcasting the lower robot ID. This is accomplished for all robots and, at the end of the stage, all robots should be broadcasting the lowest robot ID that was relayed in the entire group. At the end of this stage, the robot with the lowest unique ID that was being broadcasted by all other robots will be assigned a task and the next stage will begin when the newly task assigned robot gives the greenlight to all other robots. This cycle repeats until no robot is left without a task.

One should note that the *Card Dealer's* algorithm is scalable for any number of robots. In addition, the algorithm can be made robust to sudden addition or removal of robots into the group during runtime of the algorithm by adding a timeout period feature. The timeout feature can be explained through a simple example. In a situation where the robot with the lowest ID is about to get a task assigned, it gets removed with all communication links to this robot being cutoff. The stage ends with no robot being assigned a task and all other robots are waiting for the *green-light* to start the next stage from the newly task assigned robot. Since this newly task assigned robot was removed before being assigned a task and is now completely out of the group, all other robots are stuck in the waiting state indefinitely. By adding a timeout period feature, all awaiting robots can count down to a certain timeout period value and automatically start the new stage without having the need to get the command from the newly task assigned robot.