www.openswarm.eu

Call: HORIZON-CL4-2022-DATA-01

Type of action: RIA

Grant agreement: 101093046

Deliverable 4.1 - Performance Results from Combining Flexible Charge Storage Driven by Energy-Aware Scheduling

Work Package n°: WP2 - Orchestration of Collaborative Smart Nodes

Task Lead: KUL

WP Lead: UoS

This project has received funding from the European Union's Horizon Europe Framework Programme under Grant Agreement No. 101093046.

Funded by the European Union

Document information	
Author(s)	Danny Hughes, Jonathan Oostvogels, Mengyao Liu, Mohmmadsadegh Mokhtari, Dragan Subotic, Ritesh Kumar Singh, Maarten Weyn, Jeroen Famaey
Reviewers	Elsa Nicol
Submission date	31-Oct-2024
Due date	31-Oct-2024
Type	Report
Dissemination level	PU

Document history

Date	Version	Author(s)	Comments
13-Aug-2024	01	Danny Hughes	Preview
31-Oct-2024	02	Danny Hughes	Deliverable

Table of Contents

1. Executive Summary

This deliverable summarizes the technical outcomes of *Task 4.1 of WP4 (Energy Aware Hardware/Software co-design)*, which first develops a unique battery-free and energy-aware swarm robotics platform [4] and then implements energy awareness on top of this platform to better understand and predict energy usage [20]. This has resulted in two research papers as outlined below.

- **The CapBot Battery-Free Robotics Platform** is a hardware and software platform for swarm robotics with a highly novel, battery-free design. The CapBot reduces maintenance effort and eliminates toxic waste due to battery disposal by using long-life supercapacitors for charge storage. This design supports fast charging (under 12s for a full recharge) and hence extremely high 'duty cycles' (i.e. operational time / down-time of over 99%). It is published as a short paper in SenSys'23 [4] and has been extended to a full paper which is currently under submission to ICRA'25.
- **An Energy Management Module (EMM) for Ambiently Powered Robotic Swarms** has been developed, which provides a low power mechanism to control energy supply and demand for swarms of energy harvesting swarm robots. This EMM integrates with a scheduler that orchestrates task execution rates to provide flexible management of this new class of robot. Specifically, this module prevents individual robots from exhausting their energy reserves, extends mission duration and increases operational efficiency. This paper is under submission to IEEE Robotics and Automation Letters (RA-L) [20].

Considered in sum, these two research initiatives tackle critical problems for swarm robotics. On the one hand, CapBot [4] solves the *battery problem* of swarm robotics testbeds, wherein battery replacement is a significant maintenance headache and generates large amounts of toxic pollution. On the other hand, the EMM [20] provides support for energy harvesting from the environment, increasing opportunities for devices like the CapBot to acquire charge form the environment and thereby extend its operational life. As both contributions are highly complementary, in our future work, we will explore the combination of both approaches as well as their applicability to the proposed 1000 robot OpenSwarm testbed.

Published papers:

- M. Liu, F. Yang, S. Michiels, T. Van Eyck, D. Hughes, S. Alvarado-Marin, F. Maksimovic, and T. Watteyne, "Demo abstract: Freebot, a battery-free swarm robotics platform," in Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems, ser. SenSys '23. New York, NY, USA: Association for Computing Machinery, 2024, p. 488–489. [Online]. Available: https://doi.org/10.1145/3625687.3628401

Papers under submission:

- Mohmmadsadegh Mokhtari, Parham Haji Ali Mohammadi,, Dragan Subotic, Ritesh Kumar Singh, Bram Vanderborght, Maarten Weyn, and Jeroen Famaey. **Low-power Energy Management Module for Ambiently Powered Robotic Swarms**, under submission to Robotics and Automation Letters (IEEE RA-L).
- Mengyao Liu, Lowie Deferme, Tom Van Eyck, Sam Michiels, Said Alvarado-Marin, Filip Maksimovic, Genki Miyauchi, Jessica Jayakumar, Mohamed S. Talamali, Thomas Watteyne, Roderich Groß, Danny Hughes, **CapBot: Enabling Battery-Free Swarm Robotics**, under submission to the International Conference on Robotics and Automation (ICRA), 2025.

2. Key Performance Indicators (KPIs)

This task concerns the application of flexible charge storage and energy-aware scheduling. The KPI listed in the DoA is "energy awareness" with a target accuracy of <= 10%. We also consider key metrics such as charging time and autonomous operation.

Table 1: Key Performance Indicators. These KPIs summarize the results presented in this document, i.e. they complement the more high-level KPIs defined in the Grant Agreement DoA (shown in bold).

KPI	CapBot	Energy Management Module
Energy Metering Accuracy	> 95% accuracy in energy measurement.	> 87% accuracy in energy prediction
Charging time	< 16 seconds	NA
Autonomy	> 51 minutes at 80 RPM	NA

As can be seen from Table 1, the technologies described in this deliverable offer a compelling performance envelope. The CapBot offers a full recharge in under 60s, with over 51 minutes of autonomous operation and carge metering with a worst-case accuracy of over 95%. On the other hand, the energy management module can predict the energy used by a robot with an accuracy of over 87%. Considered together, this provides dependable and energy aware robotics support.

3. Technology Readiness Level (TRL)

Table 2 summarizes the TRL level of the two core technologies described in this deliverable. All technologies were developed during the course of the project.

Table 2: Technology Readiness Level

	TRL Level	Description
CapBot	TRL 5	An open-source pre-prototype has been developed (hardware + software) and tested in the lab an in education at small scale (up to 30 devices).
Energy Management Module	**TRL 2**	Analytical studies and initial validation have been performed through simulation and lab-based benchmarking.

4. Introduction

Work Package WP4 focuses on the design and implementation of tools that enable operators to intuitively program energy-aware swarm level behaviour that is fully adaptive, while respecting formal specifications. Within this broader goal, Task 4.1 focuses on designing and developing a power module for collaborative smart nodes that supports:

- **Adaptive energy harvesting**: accepting a range of energy sources, such as rapid charging, solar and RF harvesting. This will be accomplished using a re-configurable capacitor array with intelligent power management software.
- **Accurate energy monitoring** and benchmarking of software tasks by nodes and back-end models by exploiting the fact that a capacitors charge can be accurately inferred from its voltage.
- **Energy aware scheduling** to reserve appropriate charge and guarantee that enough energy is available for a task. This will be accomplished using OS scheduler extensions which enable applications to reason over available energy and tailor their behaviour to the current execution context.

The following chapters detail the research performed for Task 4.1. Section 5 further positions our contributions relative to the state of the art and provides background information necessary to understand the concepts relied on throughout this document. The research done for Task 4.1 has resulted in two papers, which are presented in Section 6 through 7. Section 8 discusses the use of these technologies in teaching. Finally, Chapter 9 concludes this document and discusses directions for future work.

5. Background

In this section, we give a brief introduction to battery-free system design, energy harvesting and the complexity of energy management in battery-free systems.

Battery-free System Design aims to eliminate the use of batteries in the design of low-power electronics, thereby addressing the fundamental disadvantages of using batteries, which are as follows:

- ***High cost.*** The cost of batteries varies according to their chemistry and manufacturing quality. However, modern battery chemistries which offer long lifetimes and high charge density, such as lithium thionyl chloride (primary, e.g. LiSO2) or lithium iron phosphate (rechargeable, e.g. LiFePO4) cells often comprise a large proportion of the bill of materials for an IoT device. Furthermore, the lifetime of these batteries is fundamentally limited, requiring manual maintenance interventions in order to replace them when charge is exhausted (primary cells) or recharge cycles are exceeded (secondary cells). This is problematic in the context of the OpenSwarm testbed, which will support swarms of 1000s of robots. In contrast, correctly specified supercapacitors have lifetimes of several tens of years, essentially allowing them to be recharged an unlimited number of times, thereby reducing maintenance costs due to battery replacement while having a cost of acquisition in line with traditional batteries.
- ***Low duty-cycles.*** In the case of rechargeable batteries, robots must periodically suspend their operations to return to a charging station and replenish the battery.

This ratio of up-time (i.e. operations) to down-time (i.e. recharging) is known as a duty-cycle. By maximizing duty cycle, the time required to complete a task with a fixed number of robots decreases and the number of robots that are required to ensure continuous operation of the swarm is reduced. Conventional swarm robotics platforms [2,6,7,10,11] offer nearly symmetric charging vs operational times, resulting in an approximate duty-cycle of 50%. In this case, supercapacitors offer significant advantages over batteries, as they are able to source and sink several orders of magnitude larger currents. This enables capacitors to be recharged in seconds, as opposed to hours in the case of batteries and thereby dramatically higher duty cycles than conventional battery-powered swarm robotics platforms.

- ***Pollution.*** The best performing modern battery chemistries are, sadly, based upon extremely toxic chemicals, which take a long time to break down in the environment and require specialized disposal technologies. Studies [21] predict that, by 2025, battery-driven IoT devices will contribute 130 million empty batteries per day, each containing hazardous chemicals like lithium, nickel, and cadmium, which are expensive to properly dispose of. Only 40% of these batteries are currently being recycled. Not only do correctly specified supercapacitors offer nearly unlimited recharge cycles, but they can also be manufactured from environmentally safe materials such as carbon aerogels which do not require specialized disposal, eliminating a major environmental problem for the swarm.

Energy Harvesting offers an attractive means to supplement available energy and thereby maximize operational lifetime. Specific energy sources of interest include solar power, harvesting energy from ambient light, wireless power transfer wherein radio frequency signals are used to beam power to robots, and trophallaxis [3], wherein robots forage for energy collaboratively and then harvest energy from each other (named for the phenomena of insects feeding each other in the wild). The latter is particularly attractive when combined with the fast-charging features of supercapacitors as it enables robots to work continuously on a task beyond their inherent operational life, while their peers recharge them.

Energy management for battery-free systems is complicated by a number of factors. Firstly, as capacitors have lower charge density than batteries, their voltage must be carefully monitored to avoid charge depletion. It is also an open question to what degree a capacitor-based design can provide a useful operational life in the context of robotics scenarios. This is explored in Chapter 2 and [1]. Secondly, management techniques are required that tailor the rate at which energy is consumed to match required operational life and/or available energy harvesting. This topic is explored in Chapter 3 [20].

6. The CapBot Battery-Free Swarm Robotics Platform

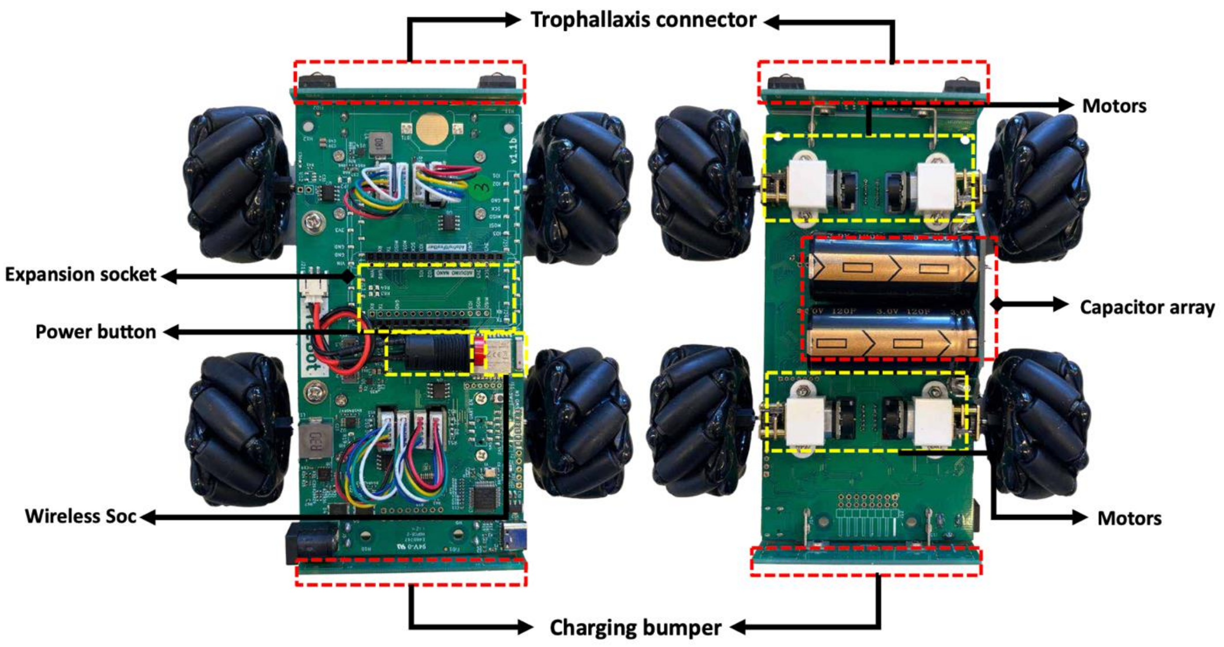


Figure 1: The CapBot as seen from above (left) and below (right)

As swarm robotics moves out of the lab and into the field, power is becoming a major problem. Rechargeable batteries have an average lifetime of 3.5 years. As swarm deployments scale, the cost and complexity of swarm-wide battery replacement is becoming a major threat to the feasibility of many deployments. Furthermore, battery waste accounts for about 10% of global e-waste [1]. This is problematic when batteries are recycled at low rates, and contain toxic chemicals such as cadmium, lead, mercury and lithium, all of which pose a significant risk to the environment. Rechargeable batteries are also expensive, especially in the case of simple robots, where battery cost may be a large part of the Bill of Materials (BoM).

Recharging batteries is equally problematic. Conventional cells often take hours to recharge, resulting in low operational duty cycles, i.e. the proportion of time that a robot spends doing useful work, vs. being offline for recharging. Drive-in charging stations provide an infrastructure solution that can be automated to reduce manual interventions. However, the fixed position of these chargers in combination with slow charging times limits the flexibility of swarm robot behavior. Here, trophallaxis (wherein robots can recharge their peers in the field) offers an exciting alternative, but current solutions [2,3] are either slow or complex.

This paper introduces the CapBot [4], a prototype of which is shown in Fig. 1. The CapBot eliminates batteries as a charge storage medium in favour of supercapacitors, which can source and sink large currents. This feature enables the CapBot to recharge or perform trophallaxis in seconds, as opposed to hours. Furthermore, the lifetime of supercapacitors is not limited by charging cycles, enabling them to achieve lifespans of many decades. Notably for swarm testbeds, the operational vs charging duty-cycle of CapBots is significantly higher than battery-based platforms, at over 99%. More accurate charge metering is also possible as available energy can be measured with high accuracy based upon capacitor voltage. Finally, in contrast to batteries, capacitors can be manufactured using a wide range of non- toxic materials, thereby reducing environmental impact.

As shown in Fig. 1, we have created a prototype of the CapBot with a 240F capacitor array, four drive motors, trophallaxis charging connector, Bluetooth Low Energy net-working and Mecanum wheels for omni-directional movement. Evaluation of the CapBot shows 51 min of operation running at a top speed of 0.73 km/h, 2.5 kg carrying capacity, 100% recharge in 16 s from a mains charger and similar charge times via trophallaxis. All hardware and software materials are released under an open-source license at: https://github.com/openswarm-eu/ ICRA2025_BatteryFreeRobot.

The scientific contributions of this paper are twofold. Firstly, we introduce a practical battery-free swarm robotics platform that is constructed from Commercial Off The Shelf (COTS) components. Secondly, we demonstrate that such a design can exceed the capability of conventional battery- powered robots in terms of: recharge time, duty cycle, trophallaxis support and accuracy of charge monitoring. The reference design has a BoM cost of under $50 (US) and is available under an open-source license.

The remainder of this paper is structured as follows. Section II discusses related work. Section III describes the hardware design of CapBot. Section IV describes the software design of CapBot. Section V evaluates the platform. Finally, Section VI concludes and discusses directions for future work.

4.1 Related Work

Farooq et al. [5] survey how mobile robots are powered and argue that their widespread application remains *"limited due to the lack of efficient power systems"*. Combustion engines provide long autonomy and rapid refuelling, though they cause pollution and cannot be used indoors. Fuel cells can also be rapidly refuelled, while reducing pollution and being safe indoors, though they are complex and expensive. In this context, batteries remain the dominant power source for swarm robots. We discuss popular battery powered swarm robotics platforms in Section II-A. In Section II-B, we discuss novel strategies to reduce charging time. This is followed by a discussion of supercapacitor-based robots in Section II-C, before highlighting requirements for the design of CapBot in Section II-D.

4.1.A. Battery Powered Swarm Platforms

The Kilobot [6] is a small and low-cost swarm robotics platform based around an 8-bit MicroController Unit (MCU) running at 8 MHz, short-range InfraRed (IR) communication (10 cm at 30 kbps) and vibration motors for locomotion. It is powered by a 400mW rechargeable coin-cell battery, which offers 3 hours of autonomy at a top speed of 0.04

km/h. Kilobots support manual recharging of large groups of robots in parallel, which takes 4 hours.

The e-puck [7] is based around a 32-bit MCU running at 168 MHz with Bluetooth and WiFi radios and two wheeled motors. It is powered by a 5Wh rechargeable battery, which offers an autonomy of 3 hours at max 0.9 km/h. Batteries can be recharged in 2 hours and drive-in power stations are available. e-pucks have a range of built-in sensors and can be expanded via an open expansion port.

Thymio II [8] is an educational swarm robotics platform which can be programmed in Scratch [9] and integrates with Lego Mindstorms. It is based around a 16-bit MCU running at 16 MHz equipped with IEEE 802.15.4 radio, two wheeled motors and various sensors. It is powered by a 5.5 Wh rechargeable battery, offering 3 hours of autonomy at 0.72 km/h and recharges in 2 hours.

The GRITSBot is used in the Georgia Tech 'Robotarium' testbed [10]. It is based around an 8-bit MCU running at 16 MHz. GRITSBots support swarm-wide WiFi commu- nication as well as local optical communication. They are powered by a 5.5Wh rechargeable battery offering 30 min of autonomy at max. 0.9 km/h. Drive-in recharging taking around 30 min.

Considering the robotics platforms discussed above, we note that contemporary swarm robots provide sufficient au- tonomy, with battery lives (30 min to 3 hours) but relatively low duty-cycles (43% for the Kilobot to 60% for the e-puck).

4.1.B. Novel Charging Solutions

The MarXbot [11] is a mobile robot that reduces recharge times by automating the hot-swapping of depleted batteries for charged ones. During this process, the robots maintains power for up to 15s using supercapacitors. MarxBot's 38Wh battery enables 4 hours of autonomy at 1.26 km/h. It is based around a 32-bit processor running at 533 MHz with WiFi radio and a range of location sensors. This approach enables a very higher duty cycle (99.9%), though the cost and complexity of this charging infrastructure is higher than traditional recharging.

Arvin et al. [12] propose the use of powered surfaces with inductive charging to continually recharge swarms of battery- powered robots. The two-wheeled robot ‘Mona’ is based on an 8-bit microcontroller running at 16 MHz and is equipped with an inductive charging unit and an 888 mW rechargeable battery. It recharges as it traverses the charging surface at speeds of up to 0.05 km/h. While this approach achieves a 100% duty cycle, it limits the speed of locomotion and requires extensive environmental modification.

Trophallaxis is a process whereby insects share nutrients in order to enable collective goals. This idea has also been applied in the context of peer-to-peer charging for autonomous mobile robots. For example, Evo-bots [2] take over two hours to charge a peer to a level that can support 30 min of operation. Similarly, FreeBots [13] enable trophallaxis between battery powered robots which, due to a custom case design can form complex networks. However, as with the evo-bot, battery-based Trophallaxis takes hours to complete. Schioler et al. tackle this problem by introducing the CISS- bot [3], which is capable of hot-swapping batteries with peer robots. Unlike MarXbot [11], which uses supercapacitors to provide power during battery swapping, the CISSbot uses an array of multiple batteries. As with the MarXbot, this achieves a near 100% duty cycle, though it significantly increases the cost and complexity of the robots.

4.1.C. Capacitor-based Mobile Robots

Muffoleto et al. [14] introduce a supercapacitor based mobile robot which can be recharged in 72 s, while offering an autonomy of up to 7 min. This results in a higher duty cycle than battery-powered robotics platforms (85%), however, it does so at the expense of autonomy. Johnson et al. introduce MilliMobile [15], a 1 cm^3 mo- bile robot that uses supercapacitors for charge storage in combination with solar panels and wireless power transfer for energy harvesting. Through the design of a novel motor controller, MilliMobile drives down locomotion power to 50μW, enabling the robot to move sustainably using harvested energy. Millimobile has a maximum speed of 0.020 km/h, roughly half that of the Kilobot. As with Mona [12], energy harvesting achieves 100% autonomy at low speed.

4.1.D. Requirements

Based upon our analysis of related work, we highlight the following requirements for the design of the CapBot:

1. Enhanced duty cycle. Exploiting the high current capability of supercapacitors, charging time can be reduced by orders of magnitude. However, system-wide power optimisation is also required to ensure that CapBots preserve comparable autonomy to today's swarm computing platforms [6,7,10,8] when using a lower energy-density charge storage medium.
2. Trophallaxis: As demonstrated by the Evo-bot [2], trophallaxis provides a compelling mechanism to increase the flexibility of when and how robots can recharge. We aim to exploit the high current capability of supercapacitors to significantly increase the speed of trophallaxis.
3. Energy awareness: Given the small amount of energy that supercapacitors store in comparison to batteries, it is essential to accurately track how much energy is being consumed to inform recharging decisions and energy-adaptive behaviour.
4. Simple COTS design: While work such as MilliMobile [15] are inspirational, they require exotic components and advanced manufacturing techniques. A COTS design using standard manufacturing techniques is required for widespread adoption in swarm applications.

4.2 Hardware Design

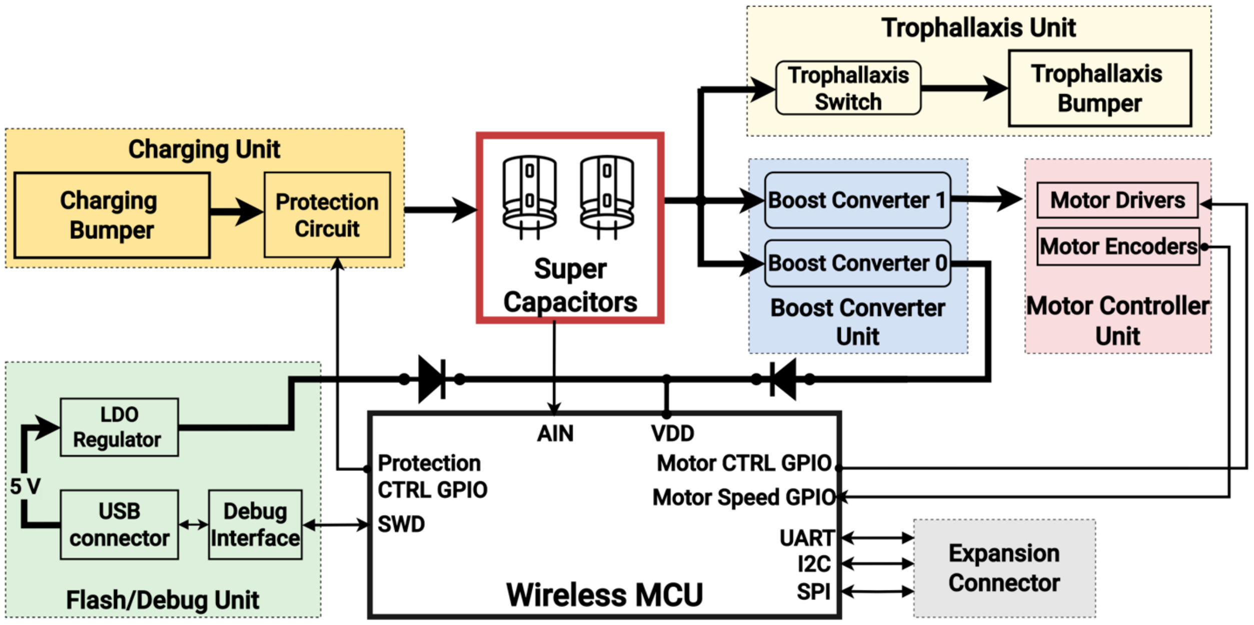


Fig. 2: High level block diagram showing the hardware components of the CapBot.

Fig. 2 illustrates the high-level architecture of the CapBot hardware platform. Motors are mounted on the PCB. The assembled weight of the robot is 285 grams. Each element of the architecture is described below:

Wireless MCU: The nRF52840 [2] offers a Cortex M4F running at 64 MHz with 256kB RAM, 1MB Flash and a 2.4GHz radio supporting Bluetooth Low Energy (BLE), IEEE 802.15.4 and ANT. This processor executes all firmware and application code; monitoring supercapacitor voltage, regulating the robot's speed and direction, managing energy transfer between robots and supporting communication.

Flash/Debug Unit: This unit allows program downloading and debugging via a standard USB connection, eliminating the need for an external debug interface.

Supercapacitor Array: Two 120F supercapacitors connected in parallel provide a reliable power supply capable of meeting high energy demands.

Trophallaxis Unit: This unit integrates a software- controlled high-current relay (Trophallaxis Switch) to enable or disable energy transfer. The Trophallaxis Bumper, equipped with spring terminals facilitates peer-to-peer energy transfer between robots while maintaining electrical and mechanical stability.

Charging Unit: The protection circuit provides over- current and over-voltage prevention during charging. The charging bumper connects to a drive-in charger or the Trophallaxis Bumper of a peer robot. The CapBot may also be charged using a standard barrel jack.

Boost Converter Unit: Consists of two TPS61021 convert- ers, supplying up to 3A and operating down to 0.6V. One powers the MCU, the other supplies the motors, ensuring a stable voltage throughout the charging cycle.

Motor Controller Unit: The CapBot uses four 3V DC motors with 1:100 gearing and rotary encoders to monitor speed, enabling omnidirectional movement through differen- tial control of the 48 mm Mecanum wheels.

Expansion Connector: The CapBot expansion connector follows the Adafruit feather standard3, enabling the use of 100's of compute, network, sensing and actuator 'wings'.

4.2.A. Fast Charging

The following sections discuss how CapBots rapidly recharge using infrastructure or trophallaxis.

Infrastructure Charging: Infrastructure charging for the CapBot is performed by a mains-connected power supply. In our experiments we used a 40A constant voltage supply, enabling the charging of CapBots at up to 120W. However, when charging a capacitor with a constant voltage supply set at its maximum rated voltage, charging rate decreases logarithmically as the stored charge approaches the maxi- mum rated voltage [16,17]. We circumvent this problem by using a supply voltage that is higher than the maximum rated voltage of the capacitor and monitoring the internal voltage of the capacitor in real time, cutting off the power supply when the measured capacitor voltage approaches its rated maximum. This maximises charging speed while minimising charger cost and complexity. The hardware protection circuit described above provides another layer of protection against over-voltage or over-current conditions.

Charging via Trophallaxis: Building upon the ability of the CapBot to safely source and sink high currents, we provide support for *trophallaxis* as a way to flexibly share energy between CapBots. Two advantages are achieved by this system: Firstly, CapBots can share charge with a peer in order to prolong its mission or even to revive a depleted robot. Secondly, this approach enables a single CapBot to execute tasks that might outlast its individual autonomy. Both infrastructure and trophallaxis charging use the same spring-loaded bumper system that allows to CapBots to make and break a connection simply by driving up to and away from each other. The CapBot contains a simple detection circuit to sense when a connection has been made reliably and the trophallaxis switch must be explicitly closed by the firmware in order for current to flow and charging to occur. Figure 3 shows nine CapBots under test along with two robots performing trophallaxis.

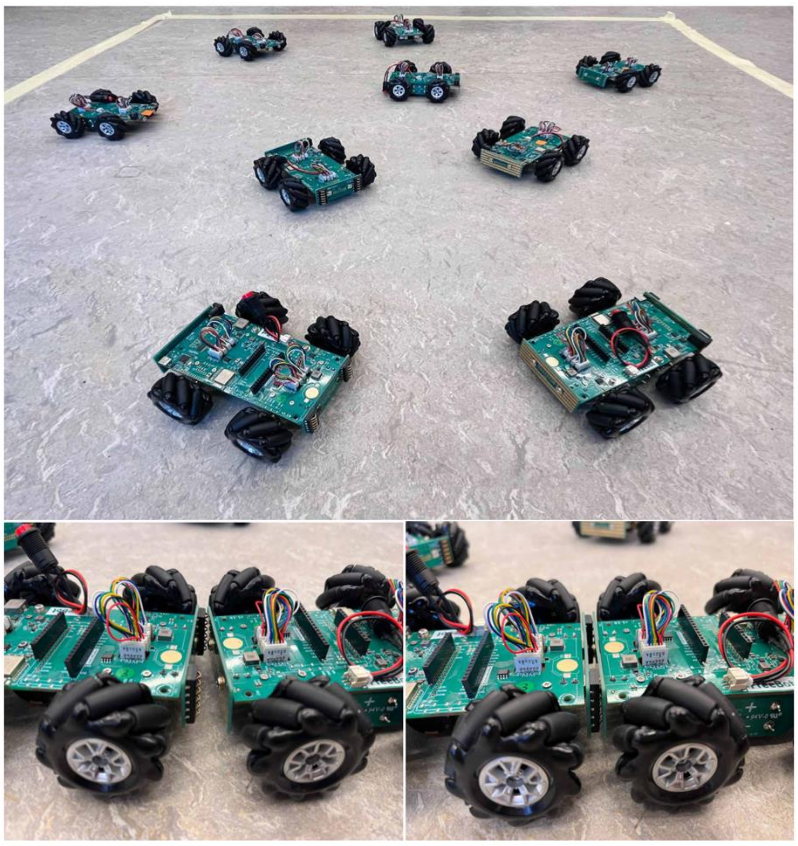


Fig. 3: A group of CapBots under test (top), two of which perform trophallaxis by connecting bumpers (bottom)

4.3 Software Design

Fig. 4 provides a high-level overview of the CapBot software stack which is provided as a library on top of Zephyr[4], a popular Real-Time Operating System (RTOS) for embedded devices. Zephyr provides priority-based multitasking, which is useful for concurrently executing application code and background tasks such as motor control.

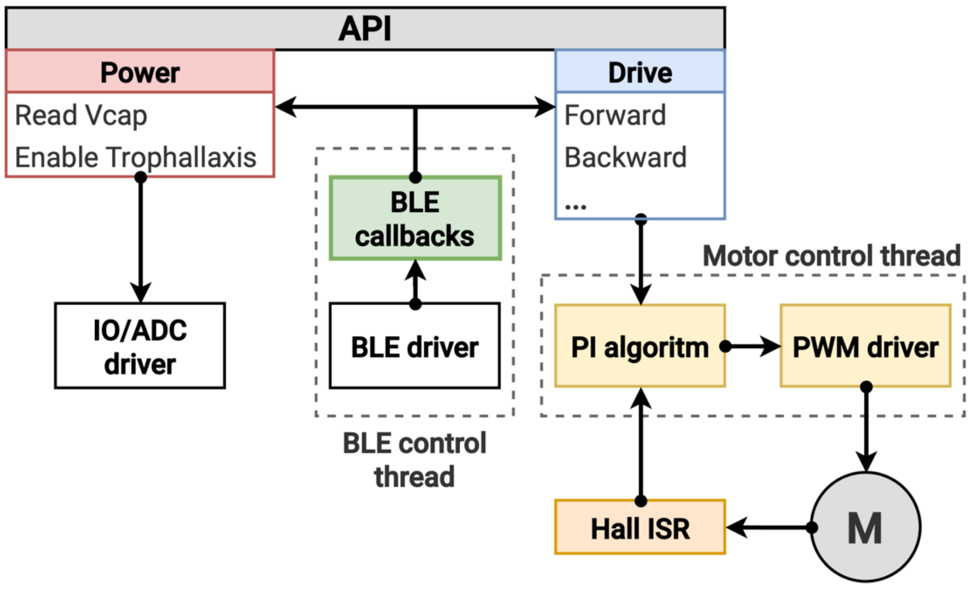


Fig. 4: High level overview of the CapBot software stack running on the Zephyr OS

The CapBot library API can be split in three parts based on functionality:

On-board IO: Exposes functions to interact with on-board IO like LEDs and buttons. The proof-of-concept mainly uses this to show the bot's status through the LEDs. Furthermore, some of pins on the Adafruit feather expansion connector can be controlled via this API component.

Power management: The power management API allows to monitor the the charge available on the supercapacitor by measuring its voltage using the onboard Analog to Digital Converter (ADC) of the nRF52840 and converting this to charge using the formula E = 0.5C V2, where E is energy in Joules, C is the capacitance in farads, and V is the voltage in Volts. Second, it enables the activation and deactivation of the Trophallaxis bumper using the associated switch.

Network interface: Enables a serial-port like connec- tion implemented on top of Bluetooth Low Energy (BLE), exposing a simple management API for controlling robot actuators (e.g. motors) as well reading relevant sensors (e.g. state of charge). At the current time, CapBots cannot be programmed wirelessly and implementing scalable 'Over The Air' programming is an important element of our future work [18].

Motor control: Enables omnidirectional movement and speed control using a Proportional Integral (PI) Controller, which realises a feedback loop between the PWM motor control and hall sensor motor rotation sensor.

4.4 Evaluation

4.4.A. Autonomous Operation

Fig. 5 show the autonomy of the CapBot with the four motors running at different 20, 40, 60 and 80 rpm, resulting in a speed of 0.18 to 0.73 km/h. Robot behaviour becomes unpredictable once the voltage of the capacitor array falls below 0.6 V ('brown-out'), resulting in an autonomy of between 51 and 65 min. This is competitive with existing battery powered swarm robots [6,7,10,8] and over four times longer than reported for previous supercapacitor-based robots [14].

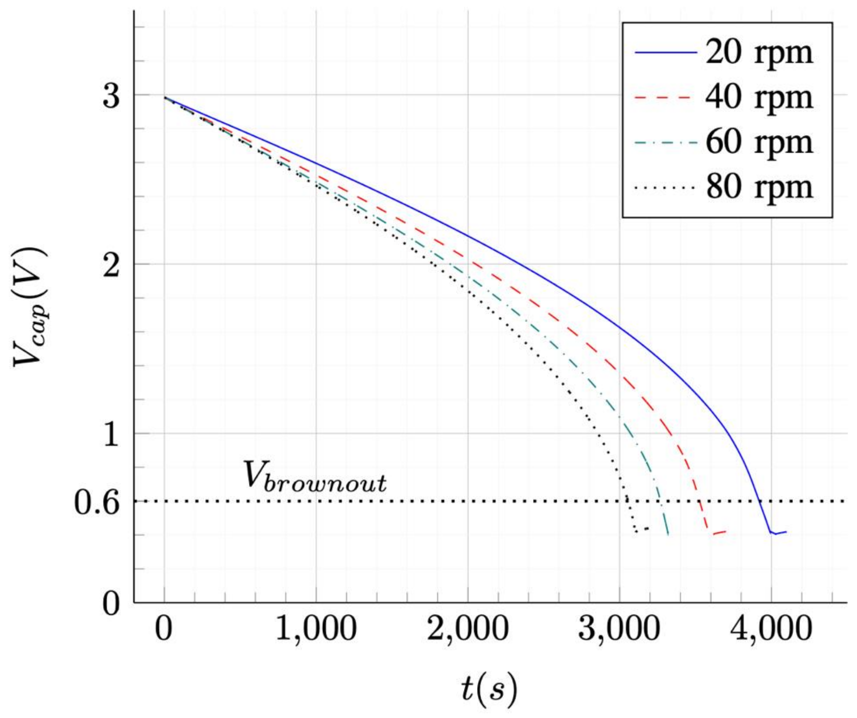


Fig. 5: CapBot operational autonomy at 20, 40, 60, and 80 rpm. The robot is fully charged at 3V and becomes inoperable at 0.6 V.

4.4.B. Accuracy of Energy Measurement

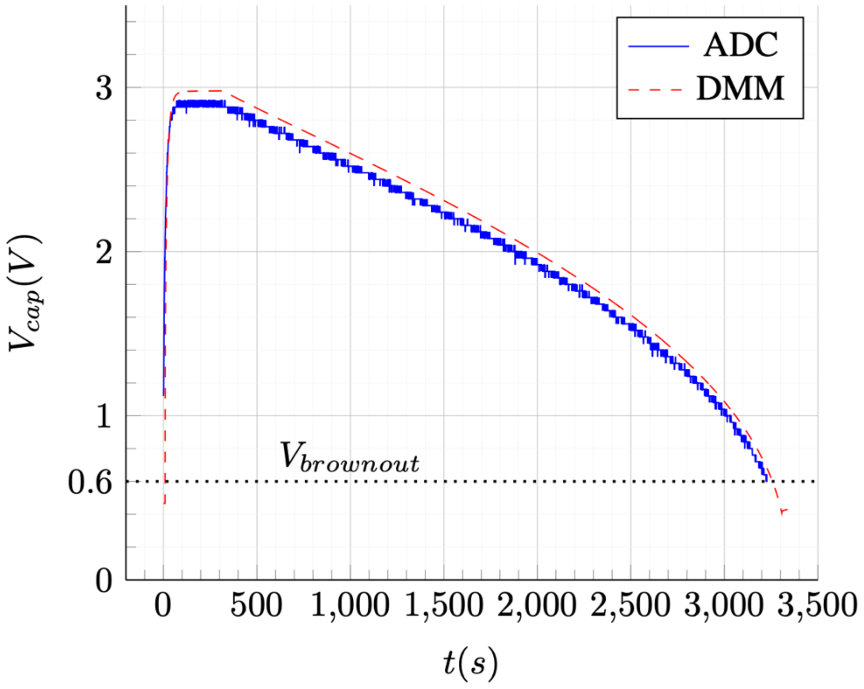
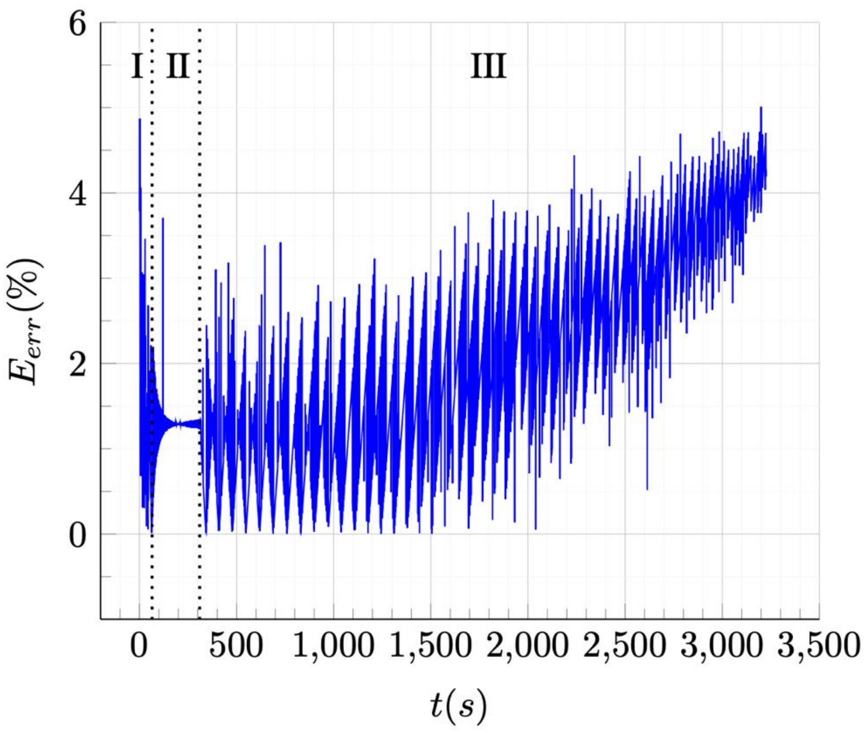


Fig. 6: Comparison between V_{cap} measured with on-board ADC and external digital multimeter at 80 rpm

As described previously, the energy stored in a capacitor can be easily inferred from its measured voltage. Nevertheless, due to inaccuracies in the ADC and its associated measurement circuit some error is inevitable. Fig. 6 compares the voltage recorded by the onboard ADC against 'ground truth' as provided by a digital multimeter over one charge/discharge cycle at 80 rpm. As can be seen from the figure, the measured voltage closely tracks ground truth. Fig. 7 quantifies the accuracy of charge measurement using equation (1) and a fixed offset K that is calibrated at manufacture time.

$$E_{err} = \frac{|\, E_{cap,DMM} - (E_{cap,ADC} + K) \,|}{E_{cap,max}} \cdot 100 \qquad (1)$$



=

Fig. 7: Relative error for E_{cap} from the on-board ADC compared to an external multimeter (raw data in Figure 6)

The initial stage (I) of this graph shows charging, during which V_{cap} rapidly increases, causing an error of up to 5%. The second stage (II) starts when the capacitor is fully charged and ends when the charger is disconnected. This stage demonstrates that the error is around 1.5% whenever the voltage slope is zero. In the final stage (III), the capacitor is discharged at the maximum operational rate by running the wheels at 80 rpm. During this phase, the error climbs as capacitor voltage falls due to the limited resolution of the ADC. Nevertheless, over a complete charging cycle, the error of energy measurement remains under 5%, which is considerably less than the state of practice for lithium batteries, which depends upon a load resistor and results in a typical error of 10-20%.

4.4.C. Infrastructure Charging

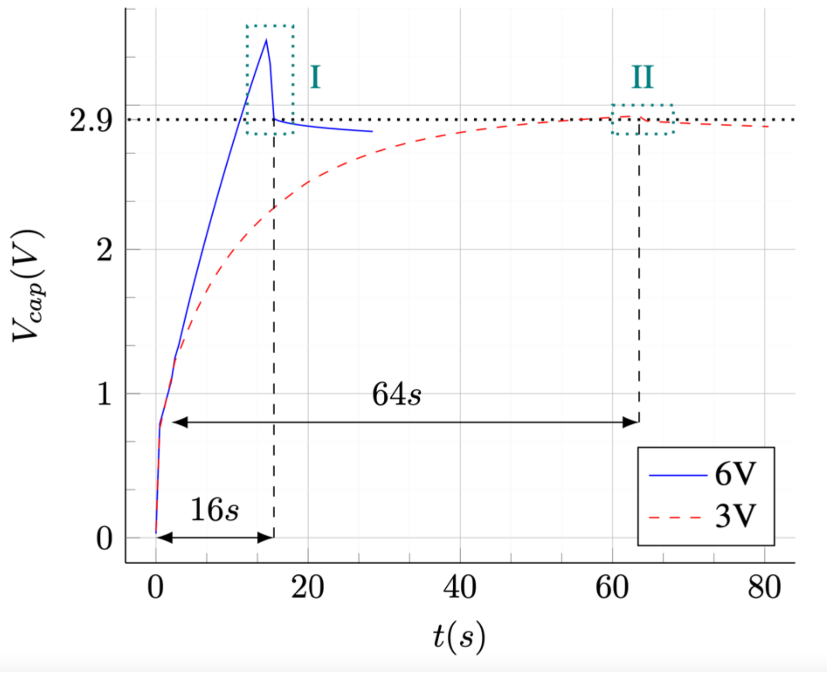


Fig. 9: Super charging the capacitors at 6 V ensures that the current stays constant and that the total charge increases linearly until the charging process is stopped.

As shown in Fig. 9, charging the CapBot using a 40A power supply at the capacitor's rated voltage of 3V charges the supercapacitor to 2.9V in 64 s. Using the quick charging approach described in Section III-A, charging time is reduced to 16 s. The voltage drop that occurs at point I, when fast charging is turned off, is related to the amount of current flowing into the supercapacitors. The same drop occurs when standard charging is turned off at point II, but it is smaller due to the relatively lower current. Infrastructure charging of CapBot is significantly quicker than any of the robotics platforms discussed in Section II, including those that use supercapacitors. CapBot achieves a duty-cycle of over 99%, which is significantly higher than reported for both battery-powered robots (43-60%) or capacitor-based platforms (85%).

4.5.D. Trophallaxis

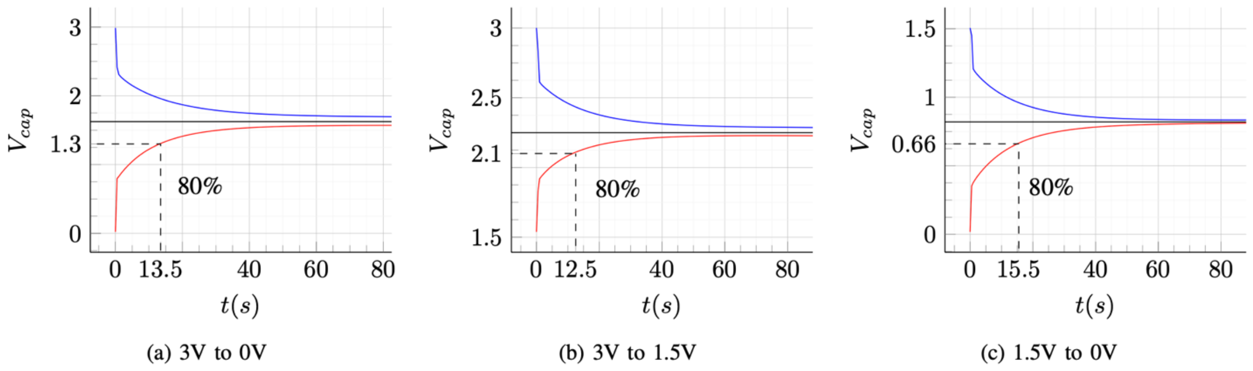


Fig. 8: Trophallaxis at multiple charge levels is faster than battery-based systems. Transferring the first 80% of charge occurs quickly (under 16 s), though reaching equilibrium takes longer (up to 80 s).

Trophallaxis may occur between any two robots where a voltage differential exists and will result in charge being equally distributed across both robots. Fig. 8 shows the trophallaxis process for three voltage differentials: (a) a 3V donor charging a 0V recipient, (b) a 3V donor charging a 1.5V recipient and (c) a 1.5V donor charging a 0V recipient. As can be seen from the figure, it takes between 12.5 and 15.5 s to reach 80% of equilibrium and 80 s for charge levels to fully equalise.

Table 3: Autonomy Before and after Trophalaxis

Trophalaxis Cases	Donor Change (minutes)		Recipient Change (minutes)	
	@ 80 rpm	@ 20 rpm	@ 80 rpm	@ 20 rpm
Case (a)	- 36	- 47	+ 12	+ 14
Case (b)	- 22	- 28	+ 18	+ 25
Case (c)	- 9	- 10.5	+ 1.8	+ 1.8

As can be seen from Table 3, trophallaxis is less than 50% efficient, which is fundamental to capacitor to capacitor charging. Nevertheless, CapBot's approach to trophallaxis is simple and orders of magnitude faster than prior systems [2,13], with the exception of CISSBot which operates by physically exchange batteries [3]. In our future work, we will explore techniques to enable robots with negative voltage differentials to donate charge building on our prior work on reconfigurable capacitor arrays [19].

4.3 Conclusions and Future Work

This paper has presented the CapBot, a battery-free swarm robotics platform that achieves a unique performance profile, including: full recharge in under 20 s, rapid and efficient trophallaxis, 51 min of autonomy, 99% duty cycle and fine- grained charge metering with an accuracy of under 5%. Considered in sum, this feature set is very appealing in the context of large-scale swarm robotics testbeds. To date we have built 50 CapBots, which have been used to teach three masters level courses at KU Leuven (Software for Embedded Systems, Industrial Internet Infrastructure and Software for Real Time Control), each of which have approximately 50 students enrolled. The platform has proven quite reliable in an educational context. Stepping back from the specifics of the CapBot platform, this work points the way to a bright future for battery-free robots, particularly in scenarios where up-time is more important than autonomy.

Our future work is focussed on the creation of a 1000 node testbed of CapBots in the context of the EU OpenSwarm project. This testbed will be open, supporting swarm exper- iments for the global robotics community over the Internet. This necessitates work on a range of topics, including: cost reduction, integration of localisation technologies and support for reliable swarm-wide reprogramming.

7. Energy Management Module for Ambiently Powered Swarms

Optimization of energy consumption in a swarm of robots can reduce operating costs and improve performance. Additionally, batteries are the main energy storage technology for untethered robots [22], and the energy consumption optimization not only improves the robot's long-duration autonomy but also enhances the overall sustainability of the robot by increasing its battery lifespan.

Looking deeply into the energy optimization of robots opens up two main branches: component-based energy optimization and time-base energy optimization [23]. In component-related efficiency, researchers look into the main energy consumer efficiency of the robot, including mechanical mechanisms, motor drives, control systems, and computational units. On the other hand, there are other factors like efficient routing, navigation, scheduling, and swarm planning [24], which are time-related efficiency factors and optimize the time that the swarm uses a specific robot. Indeed time-related objective factors are important since optimizing the time that a swarm uses a specific robot causes energy consumption reduction and in some cases task accomplishment efficiency.

Generally, optimizing a robot's energy consumption involves several key components that are crucial for maximizing efficiency and prolonging the operational lifespan. These can be broadly categorized into three main aspects: active time, idle time, and charging time [25].

Ambiently Powered Swarm Robots (APSRs) are currently gaining traction [26] due to their potential applications in various fields such as mobile sensing, search and rescue operations, agriculture, and surveillance. APSRs are swarm robotic systems designed to harvest energy from ambient sources such as solar energy.

In this paper, an Energy Management Module (EMM) based on an ultra-lower power MicroController Unit (MCU) is proposed that uses a Bluetooth Low Energy (BLE) connection to maintain perpetual connectivity between the central controller and the APSRs, even when the available energy is severely constrained. This enables an energy-aware scheduler to flexibly manage the swarm remotely and brings about advantages for swarm robot management applications. The first advantage of this method is managing energy consumption in idle time.

This plug-in module wakes up the robot when necessary and at the right time and enables the energy-aware scheduler to manage the robot's functionality. The second benefit of the EMM is enabling the scheduler to recover APSRs with dead batteries using a low-power extra connection that the EMM provides. Additionally, the scheduler can calculate the recovery time and recovery energy needed to recover a robot with a dead battery and schedule the tasks more flexibly. The third advantage of this method is to be able to hard reset the robot during malfunctions using the electronic switches which increases the reliability of the robots [27]. To this end, this paper provides a practical research effort to provide a module that enables low-power communication, wake-up, and monitoring of energy harvesting that enables the robot's recovery and facilitates its integration and collaboration with the swarm.

5.1 Related Work

Optimizing the energy consumption of swarm robotics has gained significant attraction due to the futuristic view of this field. Paryanto et al. [28] investigated the power loss points of robots from the hardware layer of mechanical components towards the software layers of planning and management. They stated that existing methods for reducing the energy consumption in robots are energy-efficient motion planning, optimizing operating parameters, and scheduling. This work emphasizes that reducing operation time and idle time is an important method for optimizing energy consumption.

Nilakantan et al. [29] researched minimizing cycle time and total energy consumption in robotic assembly line systems. This study specifically highlighted standby energy consumption as one of the most significant points of energy loss. Li et al. [27] investigated minimizing energy consumption and cycle time.

To overcome the idle mode energy management, Benedikt et al. [30] presented a novel scheduling approach that aims to minimize the energy consumption of a machine during its idle periods. In the scheduling domain, it calls the idle management model of the machine a "common" way by defining a set of machine modes, e.g., “on”, “off” and “standby”. They argue that this model might be too restrictive for some types of machines (e.g. furnaces). For such machines, they propose to employ the complete time-domain dynamics and integrate it into an idle energy function. Since normally swarm robots are not that complicated we considered a "common" model to manage idle mode. Abikarram et al. [31] presented energy cost minimization for unrelated parallel machine scheduling which considered this "common" idle mode scheduling.

The novelty of our work lies in using a low-power module as a redundant communication system, which enables the scheduler and other monitoring units to have reliable additional access to ambiently powered swarm robots. This system is specifically designed for swarm robot applications in hazardous, inaccessible, or hard-to-access use cases. Unlike other works, our research focuses on ambiently powered robots, reducing reliance on mains-powered charging infrastructure.

5.2 System Description

The EMM has important functions that help optimize the operation of swarm robotic systems by monitoring energy consumption, facilitating dead-battery recovery, and ensuring efficient task scheduling. In this regard, the system shown in Figure 5 below.

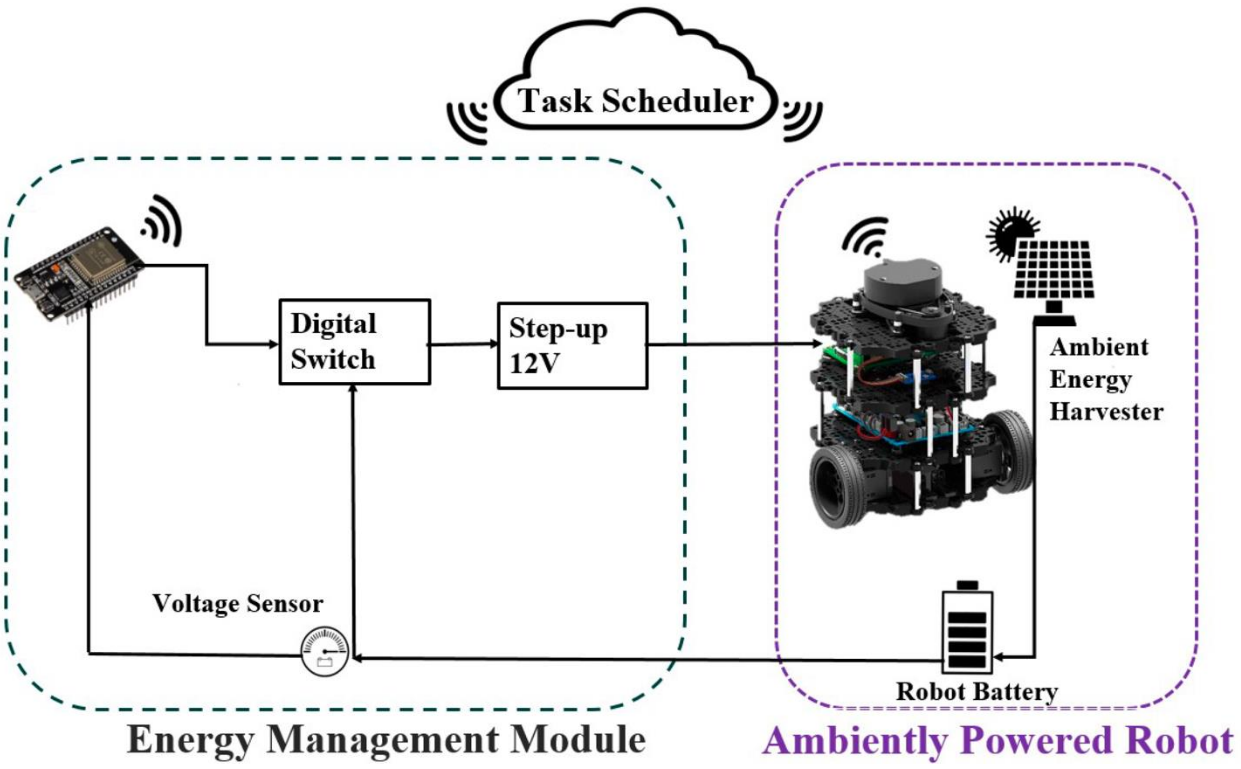


Figure 5: The implementation model of an energy management module integrated with an ambiently powered robot and a scheduler unit.

This system consists of three main components:

1. The EMM, mounted directly on the robot, serves as the primary interface for the scheduler. It includes low-power communication modules for data transmission, energy monitoring, and switches to handle different power states. The EMM supports lower-power communication like BLE and allows the scheduler to wake up and power off the robot during different states.
2. The scheduler is an external system responsible for planning and managing the robot's tasks and charging schedule, optimizing the robot's energy consumption by controlling its operational modes. The proposed scheduler is aware of the robot's energy model and using the EMM it can manage the robot's energy consumption efficiently.
3. The ambiently powered robot, is equipped with various sensors and actuators. It harvests ambient energy. The robot communicates to the scheduler using its own wireless interface (e.g., Wi-Fi based) during "Standstill" and "Active" modes.

The robot's five operational modes can be summarized as follows:

• *Active Mode:* The robot is fully operational and performs tasks as scheduled.

• *Standstill Mode:* The robot is in a standstill state, conserving energy while allowing it to transition back to Active or Off mode when necessary.

• *Off Mode:* The robot is fully powered down, and the EMM enters standby mode. In this state, the scheduler can still monitor the robot via the EMM. The scheduler can activate the EMM to turn the robot into Standstill mode.

• *Dead Battery Mode:* The robot battery is drained and is not able to switch from Off mode into Standstill mode.

• *Malfunction Mode:* The robot is unable to continue in active mode due to software or hardware problems and needs a hard reset.

The idle period is when the robot has no assigned task for a specific time. In this period robots can be in either; "Off", "Standstill" or a combination of these two states. **Figure 6** shows the state flow of the different operational modes of the robot. The arrow's direction shows the state transition flow.

System Definitions and Notations

Let us consider a swarm of robots $R = \{r_{1,}r_{2,} \dots, r_n\}$, a set of tasks $T_r = \{t_{1,}t_{2,} \dots, t_m\}$ assigned to each robot $r \in R$ by the scheduler, and a set of stationary charging stations $P = \{p_1, p_2, \dots, p_n\}$.

The transition time of a robot $r \in R$ from the Off to the Standstill mode is defined as r $T_r^{startup}$ and the energy consumption in this period is $E_r^{startup}$. At time τ, the location of robot r is defined as $L_r[\tau]$, and its energy level as $E_r[\tau]$.

L_p is the location of the charging station p. The energy needed for a dead battery robot to reach the nearest charging station, from a location L, equals $E_r^{recovery}[L]$, while the time equals $T_r^{recovery}[L]$. $P_r^{harvest}[\tau]$ represents the harvested power of the robot r at time τ . Each task $t \in T_r$ has a starting time T_r^{start}, an end time T_r^{end}, a starting location L_t^{start}, and an end location L_t^{end}. The energy consumed while performing task t is defined as E_t and the task completion time is T_t.

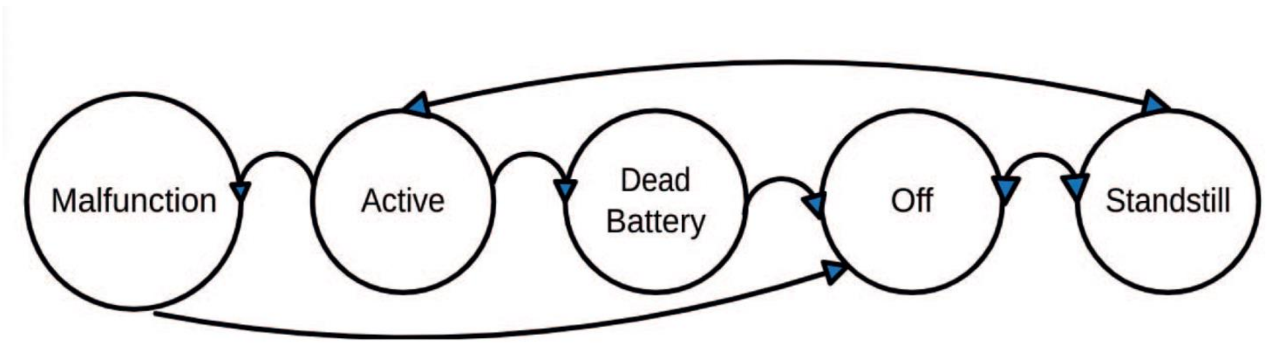


Figure 6: The state flow defined for the robots in the swarm. The direction of the arrows shows the operational state flow

Mathematical Model

Let In this section, the mathematical model of idle mode energy consumption, robot energy model, energy harvesting model, robot's recovery time, and recovery energy is formulated.

1) Idle mode energy model: The robot $r \in R$ has an idle time $T_{r,j}^{idle}$ between executing tasks t_{j-1} and t_j , given by:

$$T_{r,j}^{idle} = T_{t_j}^{start} - T_{t_{j-1}}^{end} \tag{1}$$

The associated energy consumption over a time period $T_{r,j}^{idle}$ can then be calculated as:

$$E_{r,j}^{idle} = \int_{T_{t_j}^{start}}^{T_{t_{j-1}}^{end}} P_r^{idle}[t]dt \tag{2}$$

Where P_r^{idle} is the power consumption during the idle state.

2) Robot energy estimation model: Energy consumption estimation for mobile robots can generally be categorized into three main approaches. The first approach relies on a dynamic model of the robot and its motorized actuators. The second approach estimates power consumption from a physical perspective by modeling the energy required to move the robot and overcome traction resistance and related frictions. The third approach includes a broad range of data-driven and machine-learning models. Moreover, the power consumption of a robot is a function of its subsystems, including sensing, computing, communication, and actuation instance powers. It can be shown

that, in scenarios such as the one considered in this paper, the total sensing, computing, and communication powers remain constant with respect to the robot's task. So the estimated energy consumption of the task $\widehat{E_t}$, can be formulated as follows:

$$\widehat{E_t} = \int \left(\sum_{j=1}^{N} \left(\sigma_0 \dot{\theta}_{w_j}^2(t) + \sigma_1 \, sgn(\dot{\theta}_{w_j}^2(t)) \dot{\theta}_{w_j}^2(t) \right) + \sigma_2 m + \sigma_3 \right) dt \qquad (3)$$

where N is the number of the robot's active wheels and $\dot{\theta}_{w_j}^2$ is each wheel speed, $\sigma_0, \ldots, \sigma_3$ are coefficients that can be estimated using the well-known regression methods and m is the total robot and payload mass. Additionally, to estimate the task time $\widehat{T_t}$ using the robot's path planning information and obtaining the predicted route's waypoints (wp^k) the following equation can be used:

$$\widehat{T_t} = \sum_{j=1}^{k} \left(\frac{d_l(wp^j, wp^{j-1})}{v} + \frac{d_a(wp^j, wp^{j-1})}{w} \right) \qquad (4)$$

where $d_l(.)$, $d_a(.)$ denotes the functions for calculating the linear and angular distance of two waypoints and v, ω are robot linear and angular speeds and k is the number of waypoints.

3) Battery capacity voltage curve model: To be able to have the robot energy level $E_r(t)$ based on the battery voltage level, one approach is to model the capacity voltage curve [13]. Normally in this method, the polynomial model of the battery capacity-voltage curve is modeled as the following equation:

$$E_r(V_r) = \sum_0^n a_n \cdot V_r^n \qquad (5)$$

where n is the polynomial order number. Here, E_r is the robot energy and V_r is the robot battery voltage. By knowing the voltage, we can estimate the approximate energy level of the robot battery.

4) Energy harvesting model: The energy harvested by a robot r over a time period $T_r^{harvest}$ considering a fixed $P_r^{harvest}$ is given by:

$$E_r^{harvest} = P_r^{harvest} \cdot T_r^{harvest} \qquad (6)$$

5) Recovery energy calculation: The energy needed to recover the robot and navigate the robot to the nearest charging station is:

$$E_r^{recovery}[\tau] = E_{t_p} - E_r[\tau] + E_r^{startup} \tag{7}$$

Where t_p is the task of navigating the robot from $L_r[\tau]$ to the nearest charging station L_p.

6) Recovery time calculation: The time needed to charge the robot to $E^{recovery}$ is:

$$T_r^{recovery}[\tau] = \frac{E_r^{recovery}}{P_r^{harvest}} + T_{t_p} + T_r^{startup} \tag{8}$$

5.3 Scheduler Function

This section presents two pivotal algorithms designed for the EMM: the Recovery Calculation Algorithm (RCA) and the **Idle Period Management Algorithm (IPMA).**

Recovery Calculation Algorithm

The RCA computes the energy $E_r^{recovery}$ needed for a robot to reach a charging station and the time $T_r^{recovery}$ required for the robot to reach the nearest charging station. It takes into account the current energy level, the distance to the nearest charging station, the navigation power required, and the energy harvesting rate of the robot. $E_r^{recovery}$ and $T_r^{recovery}$ are two important components that the scheduler needs to plan for future tasks. Algorithm 1 shows the detailed steps for calculating these elements.

```
Algorithm 1   Recovery Calculation Algorithm

Wake up the EMM
Input: Receive the $V_r$
Calculate $E_r$, $P_r^{harvest}$ from Eq.6 and Eq.5
Calculate $E_t$ from Eq.3
Calculate $E_r^{recovery}$ from Eq.7
Calculate $T_r^{recovery}$ from Eq.8
```

```
Return $T_r^{recovery}, E_r^{recovery}$
```

Algorithm 2 Idle Period Management Algorithm

```
Input: Idle time $T_{r,j}^{idle}$, Startup time $T_r^{startup}$
Wake up the EMM
Procedure:
  1) If $T_{r,j}^{idle} > T_r^{startup}$ and $E_{r,j}^{idle}[\tau, \tau + T_{r,j}^{idle}] > E_r^{startup}$:
      a) Switch off the robot to "Off" mode.
      b) Wait until $\tau = T_{t_j}^{startup} - T_r^{startup}$.
      c) Switch on the robot to "Standstill" mode.
  2) Else:
      a) Keep the robot in "Standstill" mode.
```

Idle Period Energy Management

The IPMA determines whether to keep the robot powered on or shut it down during idle times based on the duration of the idle period ($T_{r,j}^{idle}$) compared to a predefined startup time (T_r^{start}). If the idle time exceeds T_r^{start} and the $E_{r,j}^{idle}[\tau, \tau + T_{r,j}^{idle}]$ is bigger than the $E_r^{startup}$, the algorithm shuts down the robot to conserve energy and schedules it to start up shortly before the next task begins. Otherwise, it keeps the robot on to minimize startup delays. Algorithm 2 shows the idle state energy management procedure.

5.3 Evaluation

This section presents the experimental results of the energy management strategies implemented in a TurtleBot3 Burger (TB) robot that is depicted in Figure 5. The robot harvests ambient energy using a FIT0600 6V 1A solar panel with a size of 29 × 15 × 0.135cm and a maximum power of 5W.

The EMM is an ESP32 development board with a frequency of 2.4GHz equipped with BLE communication. The EMM controls the application by turning the power on and off. There is no communication between the ESP32 and the robot. The robot requires a 12 V power supply which is managed by the step-up module on the EMM. Communication with the cloud is done through predefined commands. The cloud scheduler will turn off the robot for a fixed period by sending "off, 'time'" to the robot. The robot will be powered off, and the EMM will enter sleep mode for that period and start a timer.

To track power consumption and energy harvesting, the EMM will periodically (every 60s) perform ADC measurements to monitor the battery level. To be able to use the solar panels we used a set of six 3.7V Lithium Polymer batteries instead of the on-market TB battery. monitor the battery level. To be able to use the solar panels we used a set of six 3.7V Lithium Polymer HW batteries instead of the on-market TB battery.

Energy Consumption During Standstill Mode

As demonstrated in **Figure 7**, we measured the energy consumption of the TB while it was in standstill mode. During this mode, the robot remains stationary, but the Raspberry Pi board, motor drive board, and its mounted sensors, such as LiDAR, remain active, leading to considerable energy consumption. The Standstill mode of the TB is quite energy-intensive, with prolonged periods in this mode causing significant energy use. The results show that the TB consumes an average of 7.5 W in Standstill mode, which is approximately 81% of its energy consumption in no-load Active mode (9.2 W).

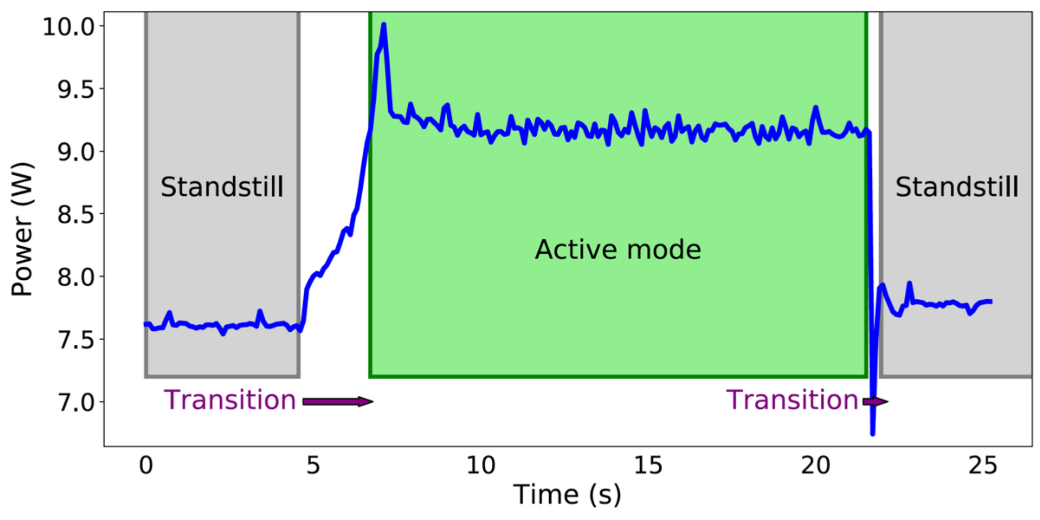


Figure 7: TurtleBot3 power consumption during no-load Active mode and Standstill mode. This figure also shows the transition between the Standstill mode to Active mode and vice-versa.

EMM Energy Consumption

This section presents the energy consumption of the EMM board. Figure 8 and 9 show experimental results of the wakeup signal sent by the scheduler and the power consumption of the EMM during active mode respectively. The power consumption of the EMM board is on average 0.49 W. The average power consumption of the EMM in its sleep mode without any Analog-to-Digital Converters (ADC) measurement is 26.2 µW, and with an ADC measurement every 60 seconds (c.f, Figure 7), it is 2.3 mW. Both values are negligible.

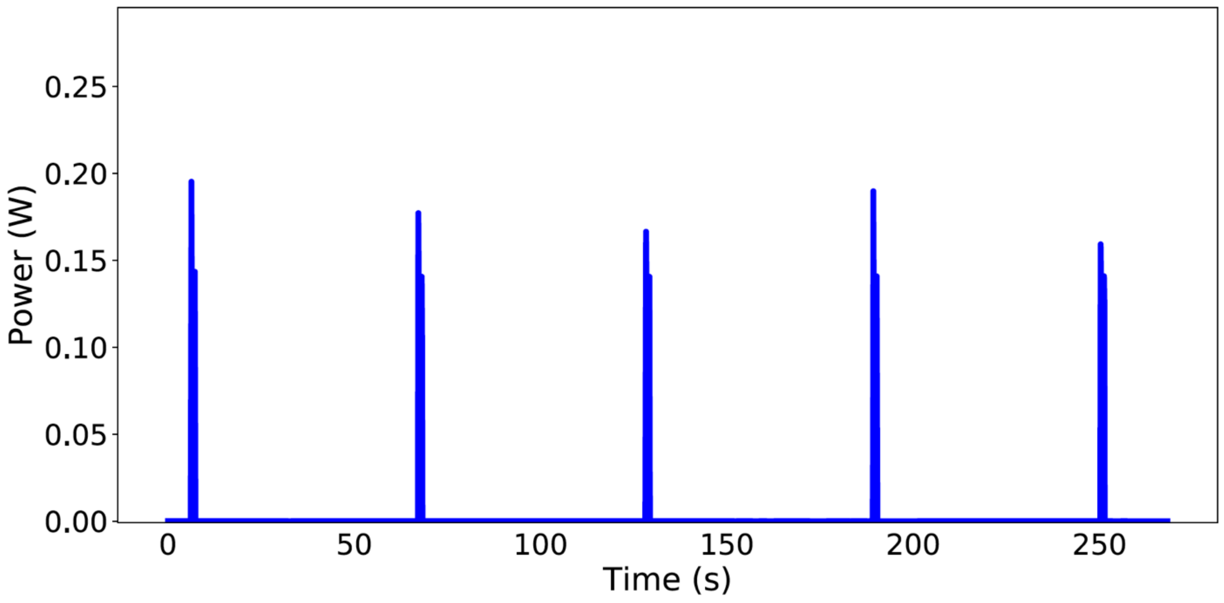


Figure 8: Energy Management Module in sleep mode and performing an ADC measurement every minute to track the energy harvesting amount.

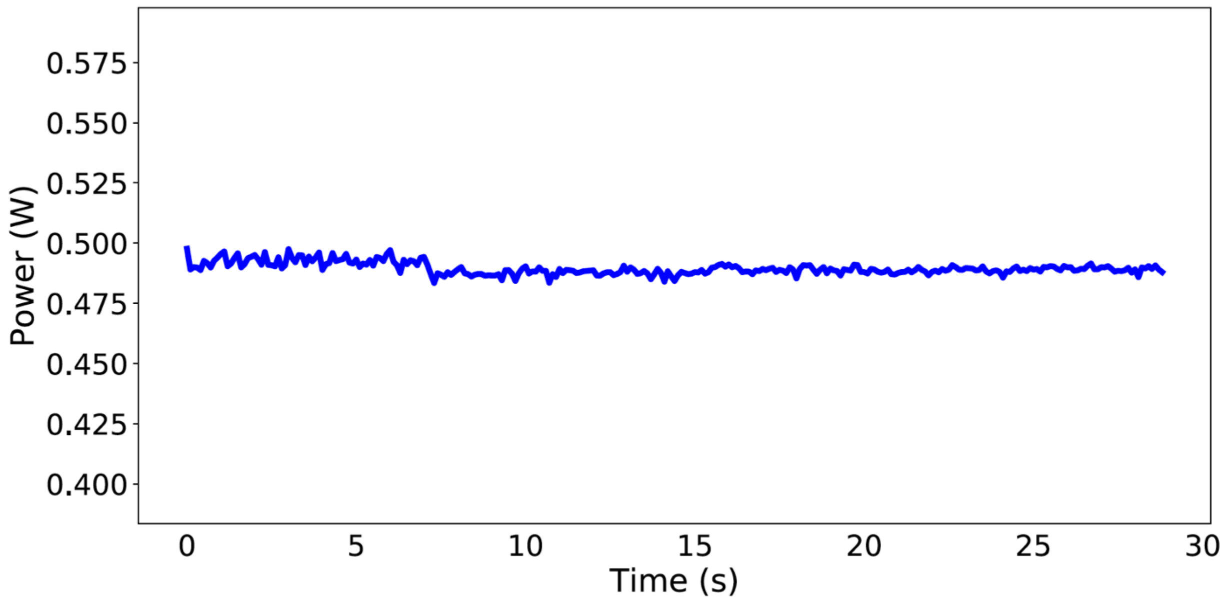


Figure 9: Energy Management Module power consumption during active mode.

Startup Time from Off Mode to Standstill Mode

Start-up time is the time from sending the command from the scheduler until the robot is in standstill mode and ready to do the tasks. Optimizing startup time is crucial for robots that spend extended periods in idle mode to minimize energy consumption. Figure 10 displays the density distribution of startup time and startup energy consumption. Although the peak probability density centres around a startup time of approximately 60 seconds, we focus on the worst-case scenario with a measured T_r^{start} of 102 seconds and E_r^{start} of 683 J for TB. Choosing the worst-case scenario enhances the reliability of idle mode energy management strategies.

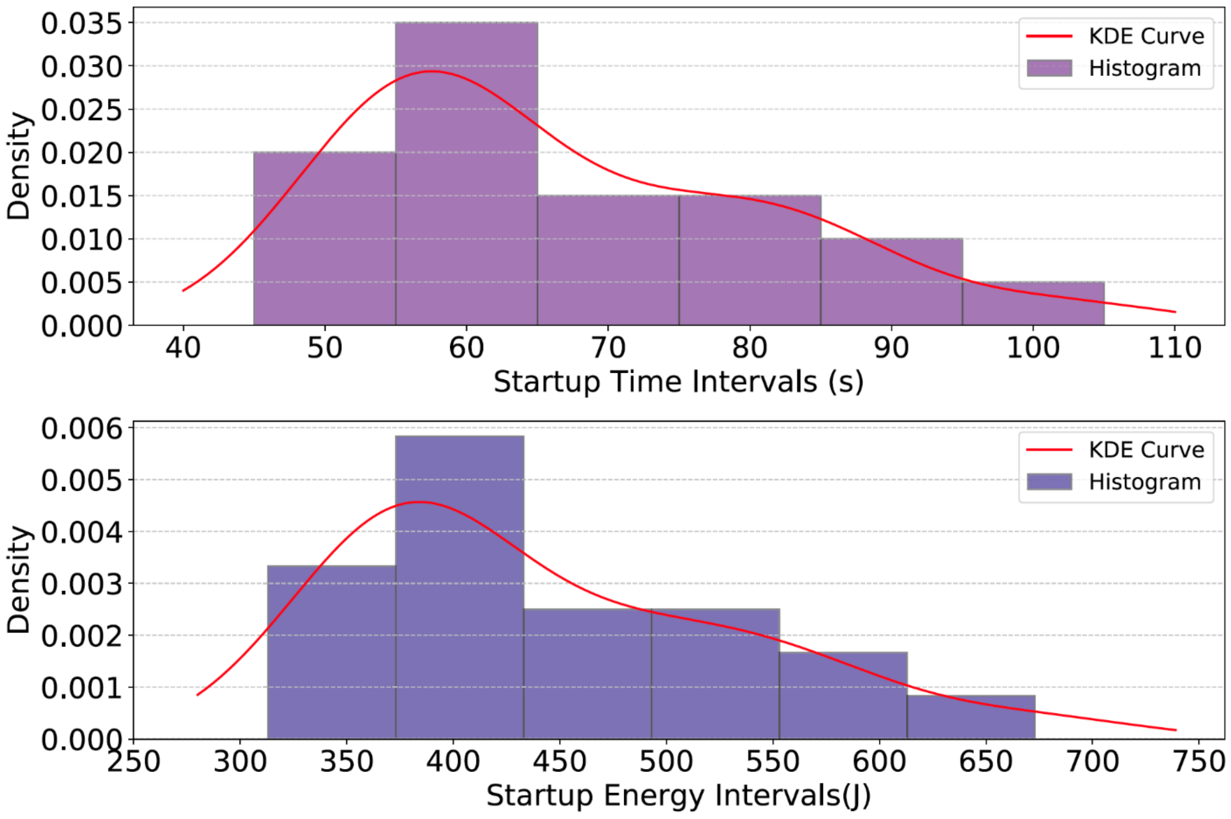


Figure 10: TurtleBot3 start-up time measurement. This figure shows the distribution of the TB start-up time and energy consumption during 20 tests.

TurtleBot3 Energy Model

To estimate the coefficients of the energy consumption model in Eq.3, the robot is tested under two different conditions of no-load and a 5 kg payload. The wheels' speed data is obtained from the odometry node in ROS, and the actual power consumption of the robot is measured using an additional power meter sensor. Additionally, σ_2 and σ_3 can be mathematically combined as $\widehat{\sigma_2} = \sigma_2 + \frac{\sigma_3}{m}$. Using the linear regression method, the coefficients of Eq.3 are estimated ($\sigma_0 = -0.0381, \sigma_1 = 0.4736, \widehat{\sigma_2} = 0.4543$). The results show a mean absolute percentage error of 3.5% with R^2_{score} = 87.3. The model performance validation is shown in **Figure 11**.

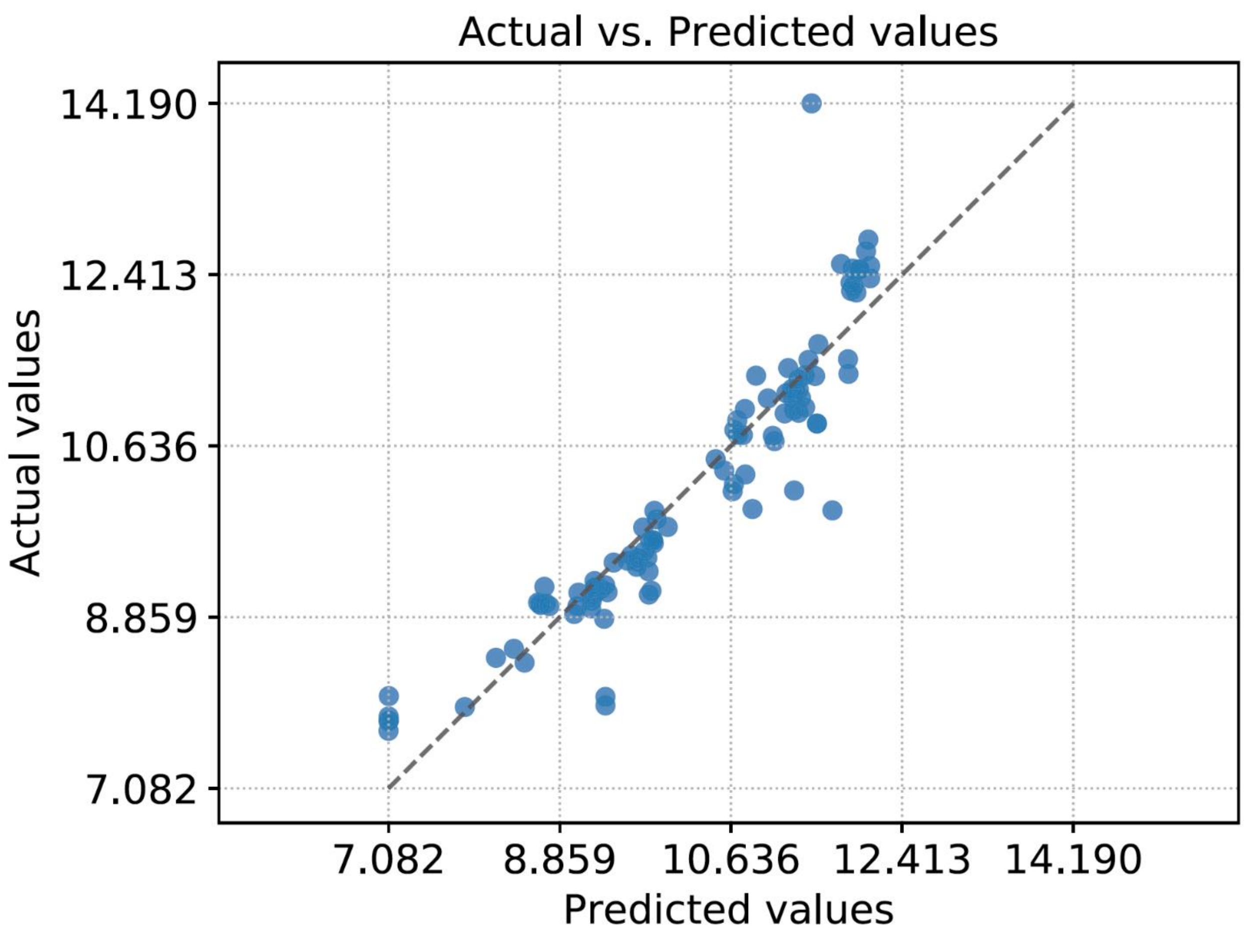


Fig. 11: Power consumption estimation validation with the test data under two different payloads (no-load and 5 kg payload) shows the bounded error for the performed estimation.

Idle Period Energy Consumption Experiment

This experiment compares the energy consumption of the robot during idle mode with and without the implementation of the EMM's idle management method. The scheduler assigns two delivery tasks t1 and t2 to the robot r. The idle time between these two tasks is $T_{r,j}^{idle}$ = 300 (s), the scheduler manages the idle mode based on Algorithm 2. The results show that by using the EMM, the robot consumes 1205 J of energy. Without the EMM, the consumption is 2250 J, which indicates a 54% reduction in energy consumption by using the EMM.

Figure 12 demonstrates that the proposed idle management method significantly reduces energy consumption during idle mode. This improvement highlights the effectiveness of the EMM in optimizing energy usage and extending the operational lifespan of the robot battery.

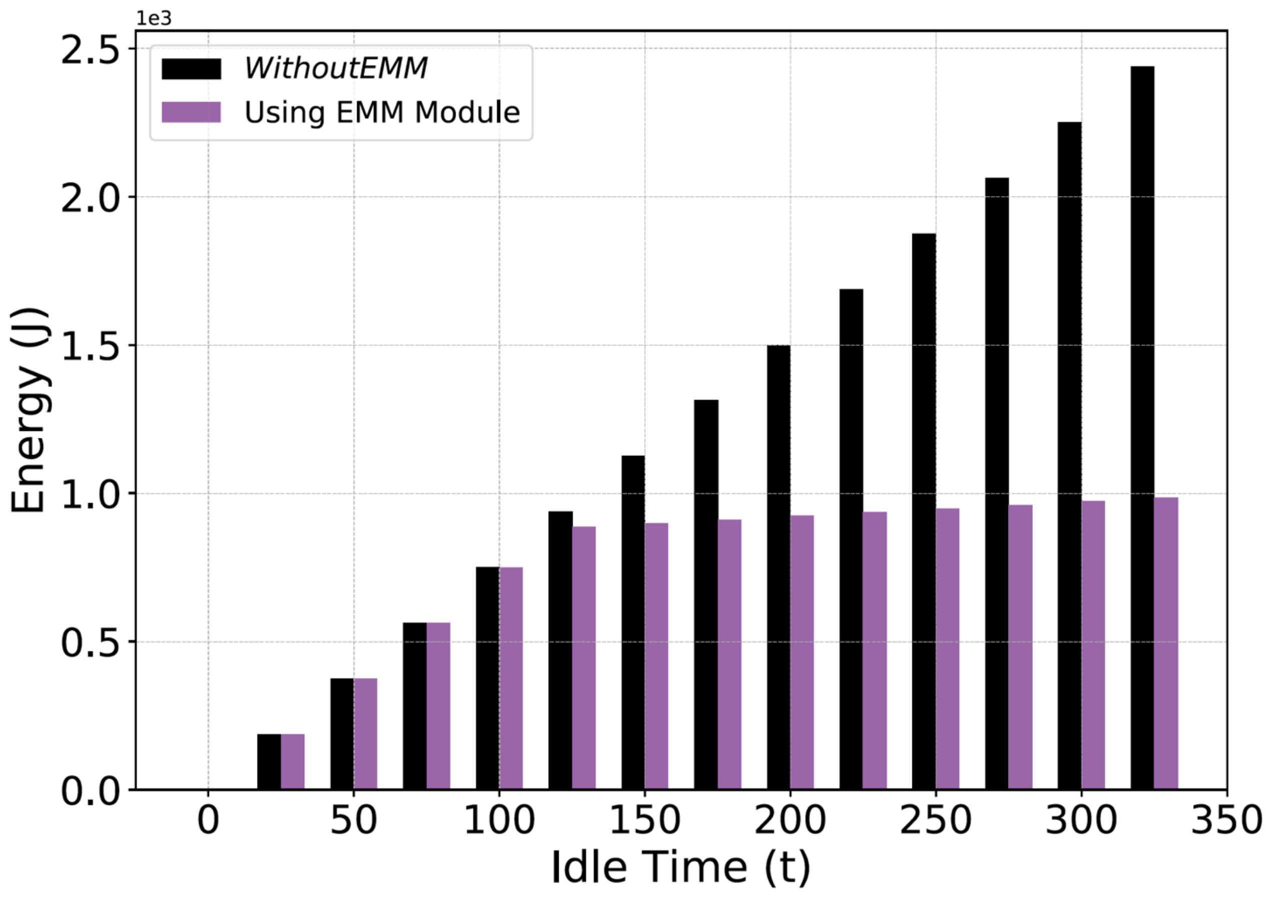


Figure 12: Energy consumption of the TurtleBot3 robot in idle state using EMM and without EMM. For every 10 seconds, the idle time exceeds the start-up time, the EMM can reduce energy consumption by an average of 2.6% compared to the method without using the EMM.

Robot Recovery Scenario

The robot recovery process involves recharging the robot using ambient energy sources when it is in a low-energy state. We compare the recovery time and energy required for the robot using indoor and outdoor energy harvesting.

The experiment assumes a 20×14 meter rectangular warehouse environment with one charging station at the coordinates of L_p = (−10, 7) (meter) with the reference point (0,0) at the middle of the map (cf., Figure 10). The position of the robot is L_r = (10, −7). Based on the robot model (cf., Eq. 3), the E_{t_p} and T_{t_p} needed for the robot to navigate from its current position to the nearest charging point are 1255 J and 145.3 s, considering the obstacles and path planning method using the Gazebo simulation environment. The $T_r^{startup}$ for the TB is considered as 102 seconds and $E_r^{startup}$ = 683J (i.e., the worst-case from **Figure 13**). Also, the harvested power for outdoor and indoor is considered as $P_{r\ outdoor}^{harvest}$=4.38 W and $P_{r\ indoor}^{harvest}$=0.1 W based on the real experiments of the used solar panel under the sun and under the indoor light.

The current battery of the robot is assumed E_{r_1} = 150 J. The $T_{outdoor}^{recovery}$ = 645.2 seconds (10.75 minutes) and $T_{indoor}^{recovery}$ = 17667 seconds (4.9 hours). The results show the effectiveness of this method in recovering the robot, especially for outdoor scenarios.

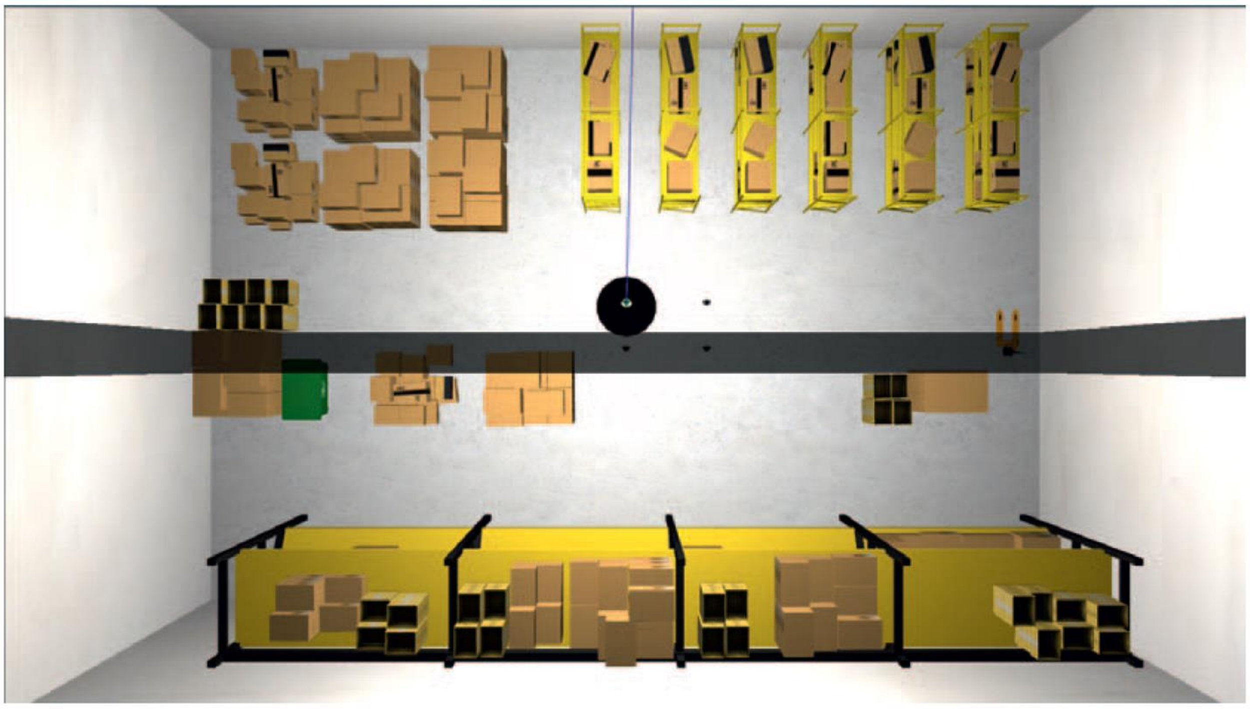


Figure 13: Gazebo simulation environment map.

5.4 Conclusion and Future Work

This paper introduces an innovative Energy Management Module (EMM) that utilizes a Bluetooth Low Energy (BLE) connection to enhance the control and efficiency of ambiently powered robotic swarms. The EMM facilitates control over multiple modes—active, standstill, off, dead battery, and malfunction—enabling a more energy-aware and flexible scheduler. Key benefits include improved energy management during idle periods, precise wake-up timing, and the capability to revive dead batteries using a low-power connection. Furthermore, the EMM enhances robot reliability by performing hard resets during malfunctions. Predicting recovery times allows the scheduler and monitoring unit to adjust robot control based on reliability and energy optimization objectives, thereby ensuring efficient task scheduling and robot availability.

Integrating the EMM with a scheduler optimizes energy consumption during idle periods based on the duration of the idle time. In a case study, we found that for every 10 seconds the idle time exceeds the start-up time, our method can reduce energy consumption by an average of 2.6% compared to the method without using the EMM. Additionally, experiments demonstrate the EMM's efficacy in managing dead battery robot recovery using energy harvesting. Results indicate worst-case recovery times of 10.7 minutes for outdoor scenarios and 4.9 hours for indoor environments.

6. Teaching and Tutorials

The CapBots were developed during the course of the OpenSwarm project, however, considerable progress has been made on increasing their Technology Readiness Level (TRL) to Level 6 – Demonstration in a Relevant Environment.

Specifically, KU Leuven has produced 50 CapBots, some of which are shown in Figure 14 below, which have been used in masters level education.

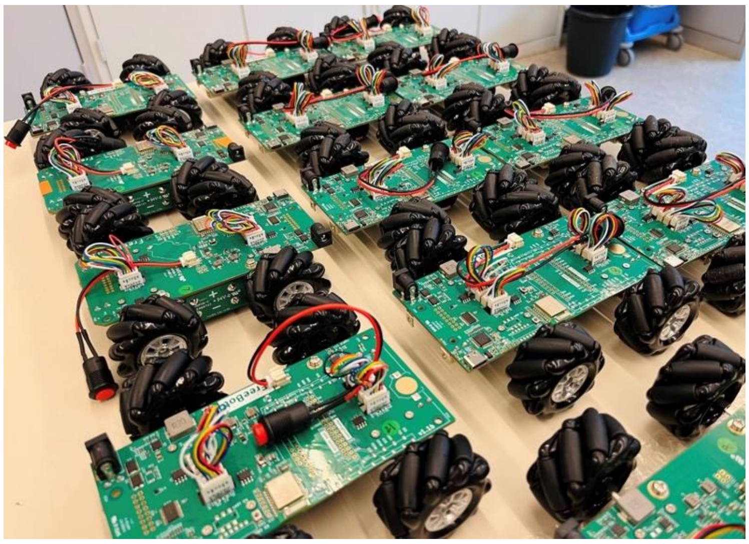


Figure 14 – Newly Manufactured CapBots being prepared for teaching

Specifically, the CapBot was used to teach:

- **Industrial Internet Infrastructure**, a masters-level Computer Science course at KU Leuven worth 5 ECTS points with 64 students. The CapBot is used to teach students about the end-to-end Internet of Things (IoT) toolchain, including: wireless networking, middleware and cloud support for cyber physical systems (BLE, MQTT, security and data visualization). (https://onderwijsaanbod.kuleuven.be/syllabi/e/H04I0AE.htm#activetab=doelstellingen_idp2778240).
- **Software for Real Time Control**, a masters-level Electrical Engineering course at KU Leuven worth 3 ECTS points with 73 students. The CapBot is an ideal platform to each real-time operating systems using the combination of the embedded Zephyr OS and time-sensitive peripherals like motor control. Students were expected to complete an embedded programming exercise using the CapBot, which is provided in Appendix 2. (https://onderwijsaanbod.kuleuven.be/syllabi/e/H09J9AE.htm#activetab=doelstellingen_idp36144).

A CapBot tutorial has been submitted to ICRA 2025, the premier robotics conference. This is provided in Appendix 3.

7. Conclusion and Future Work

This deliverable has summarized the outcomes of Task 4.1, which focused on realizing flexible charge storage and energy aware scheduling for swarm robots. This has, so far, led to two major scientific contributions; the CapBot battery-free robot [1] and the Energy Management Module for ambiently-powered swarms [2].

Outcome	Type	Key Performance Data
CapBot [1]	Conference paper. Open-source hardware and software reference design.	>97% duty cycle (operational time vs charge time), 24 minutes autonomy, fast trophallaxis in the field.
EMM [2]	Conference paper and reference implementation.	Reduces energy consumption and effectively eliminates dead-battery incidents.

Considered in sum, CapBot [1] and EMM [20] work in concert to tackle two of the biggest problems facing swarm robotics. CapBot reduces cost and pollution due to the replacement of swarm robot batteries, while drastically reducing down-time due to recharging.

Key publications:

- Mengyao Liu, Fan Yang, Sam Michiels, Tom Van Eyck, Danny Hughes, Said Alvarado-Marin, Filip Maksimovic, and Thomas Watteyne. 2024. Demo Abstract: **CapBot, a Battery-Free Swarm Robotics Platform**. In Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems (SenSys '23). Association for Computing Machinery, New York, NY, USA, 488–489.
- Mohmmadsadegh Mokhtari, Parham Haji Ali Mohammadi,, Dragan Subotic, Ritesh Kumar Singh, Bram Vanderborght, Maarten Weyn, and Jeroen Famaey. **Low-power Energy Management Module for Ambiently Powered Robotic Swarms**, under submission to Robotics and Automation Letters (IEEE RA-L).

Next Steps:

- **CapBot:** Investigating the unification of the CapBot with INRIA's DotBot for the 1000 node OpenSwarm testbed. Scaling the CapBot up in size to enable basic warehouse duties such as moving pallets. Scaling the CapBot down in size to match the form factor of the popular KiloBot platform [6] (which has now been discontinued), to enables large scale swarm experiments in small spaces.

- **EMM:** Awaiting acceptance at RA-L or resubmission to another robotics journal. Extending the CapBot design to support ambient energy harvesting such as solar, so that it can be used to support and test the EMM technlogy. Reworking the EMM implementation to operate within the resource constraints of the CapBot's nRF52840 processor, which is significantly more restricted than the current Linux-based platform.

Bibliography

[1] F. Brown, “Recyclable Phone Batteries Are Now A Reality,” August 2023, Accessed: May 7, 2024. [Online]. Available: https://happyeconews.com/recyclable-phone-batteries-are-now-a-reality/

[2] J. A. Escalera, M. J. Doyle, F. Mondada, and R. Groß, Evo-Bots: A Simple, Stochastic Approach to Self-assembling Artificial Organisms. Cham: Springer International Publishing, 2018, pp. 373–385. [Online]. Available: https://doi.org/10.1007/978-3-319-73008-0 26

[3] H. Schioler and T. D. Ngo, “Trophallaxis in robotic swarms - beyond energy autonomy,” in 2008 10th International Conference on Control, Automation, Robotics and Vision, 2008, pp. 1526–1533.

[4] M. Liu, F. Yang, S. Michiels, T. Van Eyck, D. Hughes, S. Alvarado-Marin, F. Maksimovic, and T. Watteyne, “Demo abstract: Freebot, a battery-free swarm robotics platform,” in Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems, ser. SenSys '23. New York, NY, USA: Association for Computing Machinery, 2024, p. 488–489. [Online]. Available: https://doi.org/10.1145/3625687.3628401

[5] M. U. Farooq, A. Eizad, and H.-K. Bae, “Power solutions for autonomous mobile robots: A survey,” Robotics and Autonomous Systems, vol. 159, p. 104285, 2023. [Online]. Available:

https: //www.sciencedirect.com/science/article/pii/S0921889022001749

[6] M.Rubenstein,C.Ahler,andR.Nagpal,“Kilobot:Alowcostscalable robot system for collective behaviors,” in 2012 IEEE International Conference on Robotics and Automation, 2012, pp. 3293–3298.

[7] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, “The e- puck, a robot designed for education in

engineering," in Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, 2009, pp. 59–65.

[8] F. Riedo, M. Chevalier, S. Magnenat, and F. Mondada, "Thymio ii, a robot that grows wiser with children," in 2013 IEEE Workshop on Advanced Robotics and its Social Impacts, 2013, pp. 187–193.

[9] M. Resnick, J. Maloney, A. Monroy-Herna ́ndez, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai, "Scratch: programming for all," Commun. ACM, vol. 52, no. 11, p. 60–67, nov 2009. [Online]. Available: https://doi.org/10.1145/1592761.1592779

[10] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The robotarium: A remotely accessible swarm robotics research testbed," in 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1699–1706.

[11] M. Bonani, V. Longchamp, S. Magnenat, P. Re tornaz, D. Burnier, G. Roulet, F. Vaussard, H. Bleuler, and F. Mondada, "The marxbot, a miniature mobile robot opening new perspectives for the collective- robotic research," in 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, pp. 4187–4193.

[12] F.Arvin, S.Watson,A.Turgut,J.Espinosa,T.Krajn ík,andB.Lennox, "Perpetual robot swarm: Long-term autonomy of mobile robots using on-the-fly inductive charging," in J Intell Robot Syst, no. 92, 2018, p. 395–412.

[13] G. Liang, Y. Tu, L. Zong, J. Chen, and T. L. Lam, "Energy sharing mechanism for a freeform robotic system - freebot," in 2022 Inter- national Conference on Robotics and Automation (ICRA), 2022, pp. 4232–4238.

[14] D. P. Muffoletto, C. Mandris, S. Olabisi, K. M. Burke, J. L. Zirnheld, H. L. Moore, and H. Singh, "Design and analysis of a smart power management system for ultracapacitor-powered robotic platform," in 2010 IEEE International Power Modulator and High Voltage Confer- ence, 2010, pp. 643–646.

[15] K. Johnson, Z. Englhardt, V. Arroyos, D. Yin, S. Patel, and V. Iyer, MilliMobile: An Autonomous Battery-free Wireless Microrobot. New York, NY, USA: Association for

Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3570361.3613304

[16] F. Belhachemi, S. Rael, and B. Davat, "A physical based model of power electric double-layer supercapacitors," in Conference Record of the 2000 IEEE Industry Applications Conference. Thirty-Fifth IAS Annual Meeting and World Conference on Industrial Applications of Electrical Energy (Cat. No.00CH37129), vol. 5, 2000, pp. 3069–3076 vol.5.

[17] S. Moayedi, F. Cingo ̇z, and A. Davoudi, "Accelerated simulation of high-fidelity models of supercapacitors using waveform relaxation techniques," IEEE Transactions on Power Electronics, vol. 28, no. 11, pp. 4903–4909, 2013.

[18] A. Abadie, S. Alvarado-Marin, F. Maksimovic, M. Vucinic, and T. Watteyne, "Robotap: Over-the-air programming of robotic swarms," in 2024 IEEE Workshop on Crystal-Free/-Less Radio and System-Based Research for IoT (CrystalFreeIoT). Los Alamitos, CA, USA: IEEE Computer Society, may 2024, pp. 38–43. [Online]. Available: https://doi.ieeecomputersociety.org/10. 1109/CrystalFreeIoT62484.2024.00012

[19] F. Yang, A. S. Thangarajan, S. Michiels, W. Joosen, and D. Hughes, "Morphy: Software defined charge storage for the iot," in Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems, ser. SenSys '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 248–260. [Online]. Available: https://doi.org/10.1145/3485730.3485947

[20] Mohmmadsadegh Mokhtari, Parham Haji Ali Mohammadi,, Dragan Subotic, Ritesh Kumar Singh, Bram Vanderborght, Maarten Weyn, and Jeroen Famaey. **Low-power Energy Management Module for Ambiently Powered Robotic Swarms**, under submission to Robotics and Automation Letters (IEEE RA-L).

[21] Mike Hayes, Giorgos Fagas, Julie Donnelly, Raphaël Salot, Guillaume Savelli, Peter Spies, Gerd vom Boegel, Mario Konijnenburg, David Stenzel, Aldo Romani, Claudio Gerbaldi, Francesco Cottone and Alex Weddell, **Enables – European Infrastructure Powering Internet of Things**, available online at: https://www.enables-project.eu/wp-content/uploads/2021/02/EnABLES_ResearchInfrastructure_PositionPaper.pdf

[22] Tom Verstraten, Md Sazzad Hosen, Maitane Berecibar, and Bram Vanderborght, Selecting suitable battery technologies for untethered robot, Energies, vol. 16, no. 13, p. 4904, 2023.

[23] Mohsen Soori, Behrooz Arezoo, and Roza Dastres, Optimization of energy consumption in industrial robots, a review Cognitive Robotics, vol. 3, pp. 142–157, 2023.

[24] Mike Wesselhöft, Johannes Hinckeldeyn, and Jochen Kreutzfeldt, Controlling fleets of autonomous mobile robots with reinforcement learning: A brief survey, Robotics, vol. 11, no. 5, p. 85, 2022.

[25] Raphael Rustici Garcia, André Carvalho Bittencourt, and Emilia Villani, Relevant factors for the energy consumption of industrial robots, J Braz. Soc. Mech. Sci. Eng., vol. 40, p. 464, 2018.

[26] Omer Melih Gul, Energy harvesting and task-aware multi-robot task allocation in robotic wireless sensor networks, Sensors, vol. 23, no. 6, p.3284, 2023.

[27] Zixiang Li, Qiuhua Tang, and LiPing Zhang, Minimizing energy consumption and cycle time in two-sided robotic assembly line systems using restarted simulated annealing algorithm, Journal of Cleaner Production, vol. 135,pp. 508–522, 2016.

[28] Paryanto, Matthias Brossog, Martin Bornschlegl, and Jörg Franke, Reducing the energy consumption of industrial robots in manufacturing systems, The International Journal of Advanced Manufacturing Technology, vol. 78, no. 5, pp. 1315–1328, 2015.

[29] J. Mukund Nilakantan, George Q. Huang, S.G. Ponnambalam, An investigation on minimizing cycle time and total energy consumption in robotic assembly line systems, Journal of Cleaner Production, vol. 90, pp. 311–325, 2015.

[30]

Ondřej Benedikt, Baran Alikoç, Přemysl Šůcha, Sergej Čelikovský, Zdeněk Hanzálek, A

polynomial time scheduling approach to minimise idle energy consumption: An application to an industrial furnace, Journal, October 2019.

[31] Jose Batista Abikarram, Katie McConky, and Ruben Proano, **Energy cost minimization for unrelated parallel machine scheduling under real time and demand charge pricing**, Journal of Cleaner Production, vol. 208, pp. 232–242, 2019.

Glossary

EMM – Energy Management Module: an energy monitoring and management approach from IMEC (the second major technical contribution of this deliverable).

LiSO2 – Lithium Thionyl Chloride: a long-life dense chemistry for primary (i.e. non-rechargeable) batteries.

LiFePO4 - lithium-iron-phosphate: a long-life dense chemistry for secondary (i.e. rechargeable) batteries.

Duty Cycle: the proportion of operational-time to down-time, representing how much of a device's life is spent doing useful work.

Trophallaxis: in-the-field transfer of energy between agents such as insects feeding each other in the natural world, or robots recharging each other in the swarm.

CapBot - a battery-free swarm robotics platform: from KU Leuven (the first major technical contribution of this deliverable).

CortexM: a 32 bit microcontroller design from ARM that is designed for low-power devices.

BLE – Bluetooth Low Energy: a low-power radio in the 2.4 gigahertz radio band.

IEEE 802.15.4: a low-power radio technology in the 2.4 gigahertz radio band.

Mecanum wheels: an omnidirectional wheel design that allows for 360 degree motion.

Zephyr OS: an open-source operating system for embedded devices.

Universal Asynchronous Receiver/Transmitter (UART): a simple one-to-one interconnect that is commonly used to connect discrete computational units.

Message Queue Telemetry Transport (MQTT): an application-level publish-subscribe network standard for disseminating data between sources and sinks.

The TICK (Telegraf, Influx, Capacitor, Kronograf) stack: an open-source cloud based software suite for gathering, storing and analysing sensor data.

Ambiently Powered Swarm Robots (APSRs) are a subset of swarm robots that use power from their environment to support their execution.

Micro Controller Units (MCUs) is a miniature CPU with integrated peripherals, which forms the core of an embedded devices.

Idle Period Management Algorithm (IPMA) a component of the energy management module which determines sleep times for ambiently powered swarm robots.

The Internet of Things (IoT) a collection of – typically embedded and low-power – devices which connect everyday things to the Internet.

Gazebo simulation environment: a common simulation environment for robotics.

Turtle Bot (TB): a common Linux-class swarm robotics device.

Analog-to-Digital Converters (ADC) measure a continuous voltage in the real-world and convert this to a digital approximation.

ESP32 – a popular WiFi microcontroller based upon a Tensilica Xtensa LX6.

.

Appendix 1 – CapBot Assignment for Industrial Internet Infrastructure

Industrial Internet Infrastructure (III), Group Assignment, 2024

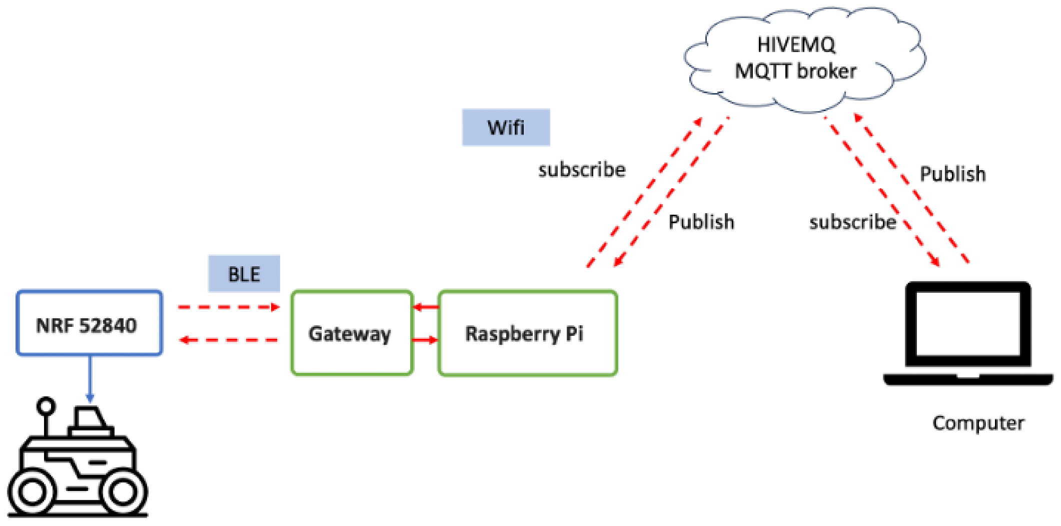
Introduction

You will be in a group of 4 to 5 students, designing an end-to-end IoT system from the sensor/actuator to the cloud. The system should be well designed, secure, reliable and performant according to design principles that you have learned about in the III lectures and prior classes.

Requirements

The system consists of a sensor/actuator, two gateways and a server. The sensor/actuator sends sensing information to the server, while the server sends commands to the sensor/actuator. A user can use the server to control the sensor/actuator and also see all the sensed information via a web front end offered by the server. We provide a wheeled robot as the sensor/actuator, an nRF52840[1] as the Bluetooth Low Energy (BLE) gateway and a raspberry pi zero as the Internet gateway.

[1] https://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.0.pdf



Sensor/Actuator

The **sensor/actuator** is a wheeled robot. An nRF52840 is the controller of the robot. The APIs to move the wheels are provided by us. You need to write a program on the robot which contains the following three major functional blocks:

a. Securely establish a BLE connection with the **BLE gateway**
b. Receive commands from the **BLE gateway**, and call APIs to move the robot according to the commands
c. Read the voltage of the super capacitor and the RPM of the four motors every 200ms, and then send the average voltage and RPM of the last 5 readings (1s) to the **BLE gateway** in a single message.

You are free to define the format of messages between the robot and the BLE gateway yourself. Each message should contain your group number, relevant security data (if any) and the sensed data described above.

BLE Gateway

The **BLE gateway** is a nRF52840 DK board. You should connect the gateway to the **sensor/actuator** by BLE, and to the **Internet gateway** (raspberry pi zero) by UART. You should write a synchronized multi-thread program to:

a. Establish a secured BLE connection with the **sensor/actuator**
b. Establish a UART connection with the **Internet Gateway**
c. Receive messages from the **sensor/actuator** and send them to the **Internet Gateway**
d. Receive messages from the **Internet gateway** and send them to the **sensor/actuator**

Internet Gateway

The **Internet gateway** is a Raspberry pi zero in this system. The gateway is connected to the **BLE Gateway** (nRF52840) by UART, and connected to the Internet by WiFi. You need to write a synchronized multi-thread program on the raspberry pi zero to:

a. Run an MQTT client to publish/subscribe messages
b. Publish the voltage and the RPM information to the MQTT broker
c. Subscribe and receive the moving command MQTT messages and send the commands to the robot

The MQTT topics should be "III2024/your_group_number/sense" and "III2024/your_group_number/control".

Server

The **server** runs on a laptop provided by your group. The server runs:

a. A user-friendly tool to control the robot
b. A correctly configured instance of the TICK stack, wherein:
    i. Telegraf is used to collect sensing data from the robot over MQTT.
    ii. InfluxDB is used to store that data in an appropriate format.
    iii. Chronograph/Grafana/InfluxDB is used to effectively visualize the data.
    iv. The kapacitor/InfluxDB is used to generate alerts on appropriate channels whenever the charge available on the super-capacitor drops by an additional 10% (i.e. 90%, 80%, 70%, etc.).

You should apply security methods to ensure end-to-end security across your system. You can either set up a MQTT broker (e.g. Mosquitto) on your computer or use a public MQTT broker on the internet. The super capacitor voltage range is 0.9V-2.8V, so we consider 2.8V as 100% and 0.9V as 0%.

Required Deliverables

- By 23:59 on Sunday May 19th (submit on Github classroom):
  - A report of up to 3 pages that describes the design of your system.
  - A well-organized zip file containing all of your source code.
- Evaluation on May 21st:
  - A presentation of 5 minutes introducing your solution.
  - A demonstration and walk-through of your system in operation.

For questions regarding the project, you can send an email to any of the instructors:

- yinze.li@kuleuven.be (**coordinator** and cloud/middleware help)
- bingwu.fang@kuleuven.be (cloud/middleware help)
- brendan.mackenzie@kuleuven.be (embedded help)
- lowie.deferme@kuleuven.be (embedded help)

Score distribution

The total score is 20 points. You will be given partial grades if you partially achieve the functionalities for each criteria.

- Embedded (8 points):
  - BLE secure connection
  - Sensor reading
  - Drive robot motors
  - UART with pi
- Middleware & cloud (8 points):
  - Robot controller
  - MQTT connection (python, telegraf)
  - InfluxDB databse
  - Visualization
  - Alerting
- Presentaion: 2
- Report: 2

Appendix 2 – CapBot Assignment for Software for Real Time Control

Software for Real-Time Control (SRC), Group Assignment, 2024

Introduction

There are two ways to complete the assessment for Software for Real-Time Control:

- **Option 1**: Assessment by examination in the standard assessment period. If you prefer this option, you do not need to complete the assignment.
- **Option 2**: Assessment by assignment during the semester. In this case, you must complete the assignment as outlined below. **If you begin the assignment, you must complete it!**

You will be in a group of 4 to 5 students, designing real-time software for the Zephyr OS running on the CapBot research platform. The software you develop should build upon the concepts that you have learned in the lectures. It will be assessed based upon: (a.) a demonstration of the software and questions on the code, (b.) a 10 minute presentation on the approach and (c.) a 2-page written report on the assignment, format to be specified by April 21st.

Preperation

Before you arrive at **the first lab session** on wednesday, please install VS Code, the Nordic nRF toolchain and the SDK. (Make sure to follow the v2.x.x tab)

Assignment

Your group should select one of the four assignments outlined below. Multiple groups may complete the same topic.

3.1 MicroROS CapBot Port

This project will port the embedded version of the Robot Operating System (MicroROS) to the CapBot platform (i.e. the nRF52840). Full details of MicroROS can be found here: https://micro.ros.org/.

It is expected that MicroROS will run on top of the CapBot's existing Zephyr OS (https://www.zephyrproject.org/). Alternative architectures, such as MicroROS on bare-metal are possible, but not recommended.

The MircoROS port does not need to be complete. In fact, it may cover only a subset of the core MicroROS concepts. This should be agreed with the lab supervisors by the end of Lab 2. Your work will be marked against this agreement.

Once the MicroROS port is complete, the existing network and motor control libraries should be ported back to the platform and used to demonstrate the assignment.

There is lots of supporting material online to help you with this port. MicroROS has already been ported to the Zephyr OS and the Cortex-M4 processor. See https://micro.ros.org/docs/overview/hardware/ for more details.

3.2 MultiRobot Control with Bluetooth

This project will replace CapBot's current Bluetooth Low Energy (BLE) wireless control library, which uses the Nordic Semiconductor UART/Serial Port Emulation over BLE. This library is limited as it supports only one-to-one robot control, whereas we intend to control many CapBots at the same time in the future (https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk5.v13.1.0%2Fble_sdk_app_nus_eval.html).

You will develop a new BLE network library for the CapBot, which allows a single manager to control many robots. However, you will only demonstrate its operation using 1 manager and 2 robots. Its scalability will be inferred from the design.

In building your solution, you are free to build upon any of the existing BLE profiles from Nordic, or you can craft a ground up solution by sending raw BLE packets. https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk5.v15.0.0%2Fexamples_ble.html.

Building a correct, yet inefficient protocol will be straightforward, however, building a high-performance network protocol is complex. You should strive to maximize performance (throughput, latency, jitter and reliability), while minimizing energy consumption and network load. Your network should also scale to support up to 100 robots in the same location.

3.3 Precision Motor Control

This project will replace the current CapBot motor control libraries, which offer rather rudimentary direction and speed control with libraries that offer the greatest precision possible given the available actuators (motors) and sensors (rotary encoders).

Specifically, your motor control drivers should enable 360 degree motion using the four Mecanum wheels of the CapBot and at a precise user-requested speed measured in kmph, while logging the energy consumption of this locomotion using capacitor voltage. In order to demonstrate these advanced motor controllers, you may need to extend the existing robot control software.

As the core mission of this assignment is conceptually simple, a higher standard of evaluation will be expected. You should produce scientific quality evaluation of the accuracy of your approach running on multiple CapBots. This evaluation should include: speed, energy consumption and control accuracy. The design of your experimental methodology is a core part of the assignment, and should be agreed with the lab supervisors by the end of Lab 2.

3.4 Energy Benchmarking of Software

This project will exploit the fact that the energy stored in a capacitor can easily be measured by reading its voltage to implement statistically robust benchmarking of the energy consumption of host software.

Specifically, you will extend the Zephyr OS running on the CapBot (https://www.zephyrproject.org/) so that the energy consumed by each task can be measured by the OS. In some cases, the energy consumed by a software task may be below the resolution of the nRF52840 Analog to Digital Converter (ADC). In this case, many task executions may be necessary to achieve accurate results.

You will evaluate the accuracy of your approach using a precision lab multimeter together with General Purpose Input/Output (GPIO) to signal the start and end of the measured tasks. You should agree a plan for your implementation and evaluation with the lab supervisors by the end of Lab 2.

This assignment will be primarily judged on two factors: (1) a scientifically sound assessment of the fundamental accuracy that this approach can provide given the components used on the CapBot and (2) achieving the best possible accuracy within the previously identified constraints. This may necessitate calibration for manufacturing differences in components.

Required Deliverables

- By the end of Lab 2: agree a plan for your assignment with the lab supervisors.
- By 23:59 on Friday May 17th (submission procedure to be announced):
    - A report of up to 2 pages that describes the design of your system.
    - A well-organized zip file containing all of your source code.
- By appointment in the week following submission (scheduling to follow):
    - A presentation of max. 10 minutes introducing your solution.
    - A demonstration of max. 5 minutes showcasing your solution.

For questions regarding the project, you can send an email to any of the instructors:

- mengyao.liu@kuleuven.be (**coordinator** and cloud/middleware help)
- brendan.mackenzie@kuleuven.be (embedded help)
- lowie.deferme@kuleuven.be (embedded help)

Appendix 3 – CapBot Tutorial Proposal for ICRA 2025

Tutorial: Hands-on with a Battery-Free Robot Swarm

Danny Hughes[1] (primary contact), Mengyao Liu[1], Lowie Deferme[1], Tom Van Eyck[1], Sam Michiels[1], Said Alvarado-Marin[2], Filip Maksimovic[2], Thomas Watteyne[2], Genki Miyauchi[3], Jessica Jayakumar[3], Mohamed S. Talamali[3], Roderich Groß[3,4]

I. Short Description of the Event

Title: Hands-on with a Battery-Free Robot Swarm

Workshop Link:
https://distrinet.pages.gitlab.kuleuven.be/taskforces/nes/freebot/icra-2025-tutorial

Type and Duration: Half-day tutorial

Short Summary for ICRA Website: Swarm robotics uses large collections of robots that work together in a cooperative yet decentralized manner to achieve their objectives, often drawing inspiration from biological systems such as insect colonies. However, power remains a critical problem for the swarm. Conventional batteries are slow to recharge, have limited lifespans and contain toxic chemicals which require special disposal techniques. This tutorial will explore a different future, by giving attendees hands-on experience with a novel battery-free swarm robotics platform that tackles these problems. CapBot combines supercapacitor charge storage with low power design to deliver a unique feature set, including: sub-cm accurate localization, full recharge in 16 seconds, 51 min of autonomous operation at 0.74 km/h and Bluetooth networking. Each CapBot is also capable of performing rapid trophallaxis [8], where robots donate charge to each other in the field. Attendees will learn to program CapBots in C or Python, using standards-based tools to connect their robots to cloud based visualization and control software. Together, they will then use the system that they have built to perform an energy aware rescue mission using the CapBot.

II. Detailed Description

In this section we describe the proposal tutorial in terms of its goals in Section II.A, the CapBot platform in Section II.B, supporting software in Section II.C and target attendee profile in Section II.D.

A. Goals of the Tutorial

The goal of this tutorial is to provide attendees of the ICRA conference with a valuable hands-on experience with a novel suite of swarm robotics technologies, including:

- Showing the potential of battery free robotics in general and for swarm robotics in particular [7].
- Demonstrating a novel and highly accurate localization technique [2].
- Showing practical applications of trophallaxis (robot-to-robot charge transfer).
- Providing a meaningful programming experience with a C-based programming track (on the robot) and Python programming track (on the server).
- Showcasing how a standards-based IoT stack can be used to easily connect large numbers of embedded devices to cloud-based intelligence.

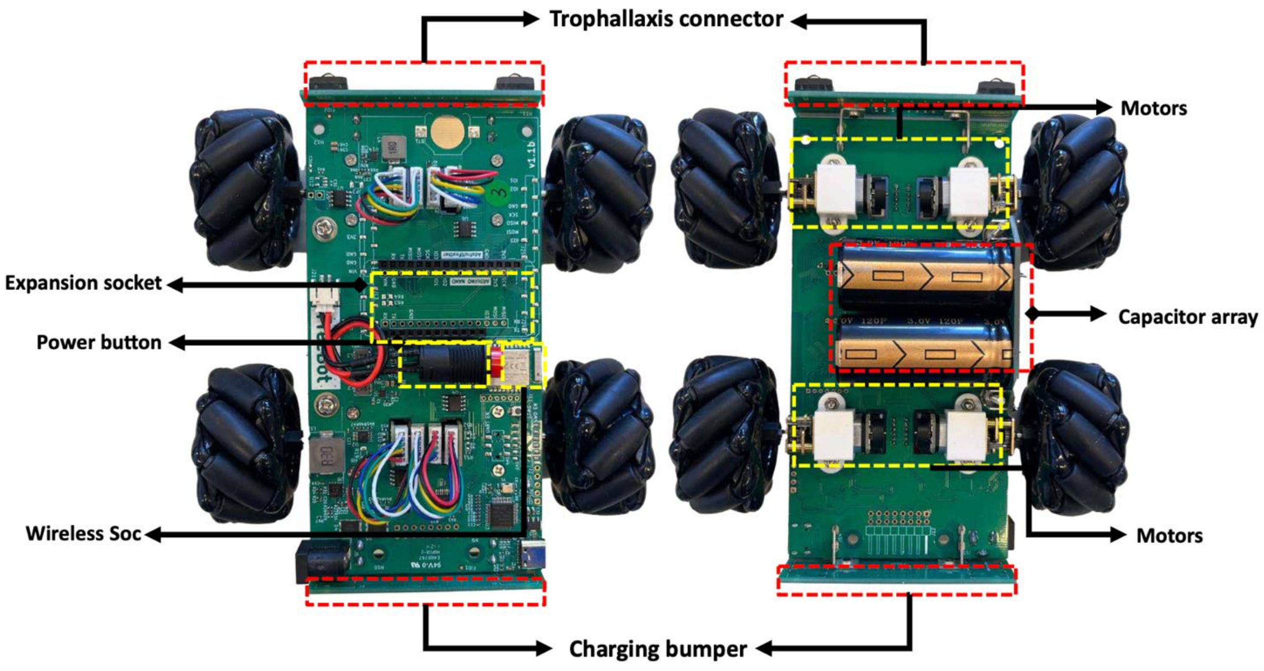


Figure 1: The CapBot as seen from above (right) and below (left).

B. The CapBot Platform

The CapBot, as shown in Figure 1, was developed in the context of the EU OpenSwarm project (https://openswarm.eu/) to tackle the problem of providing a large-scale swarm testbed (we target a 1000 node deployment). In this context, conventional batteries pose a significant problem due to their slow recharging, toxic chemistries and limited charging cycles. At the end of their relatively short lives (on average 3.5 years), special techniques are required to safely dispose of these batteries which are recycled at relatively low rates [1]. To tackle the above-mentioned problems we have developed a novel and fully open source swarm robot called CapBot which offers a unique performance profile:

- *Rapid charging:* CapBots recharge from a drive-in charger in under 16 seconds and offer 51 minutes of autonomy at top speed (0.74 km/h), enabling testbeds or tutorials to operate indefinitely at a high operational duty cycle (>99% uptime), while driving down pollution and maintenance effort due to battery replacement.
- *Trophallaxis:* Like many creatures in the animal kingdom, CapBots can 'feed' each other, by transferring energy rapidly in the field, as shown in Figure 2.

- *Wireless networking* using Bluetooth Low Energy (BLE) enables broadcast communication over the swarm for gathering robot telemetry and controlling actuators.
- *Lighthouse localization* [2] enables robots to localize themselves within the swarm using a combination of cameras and server-side mapping.

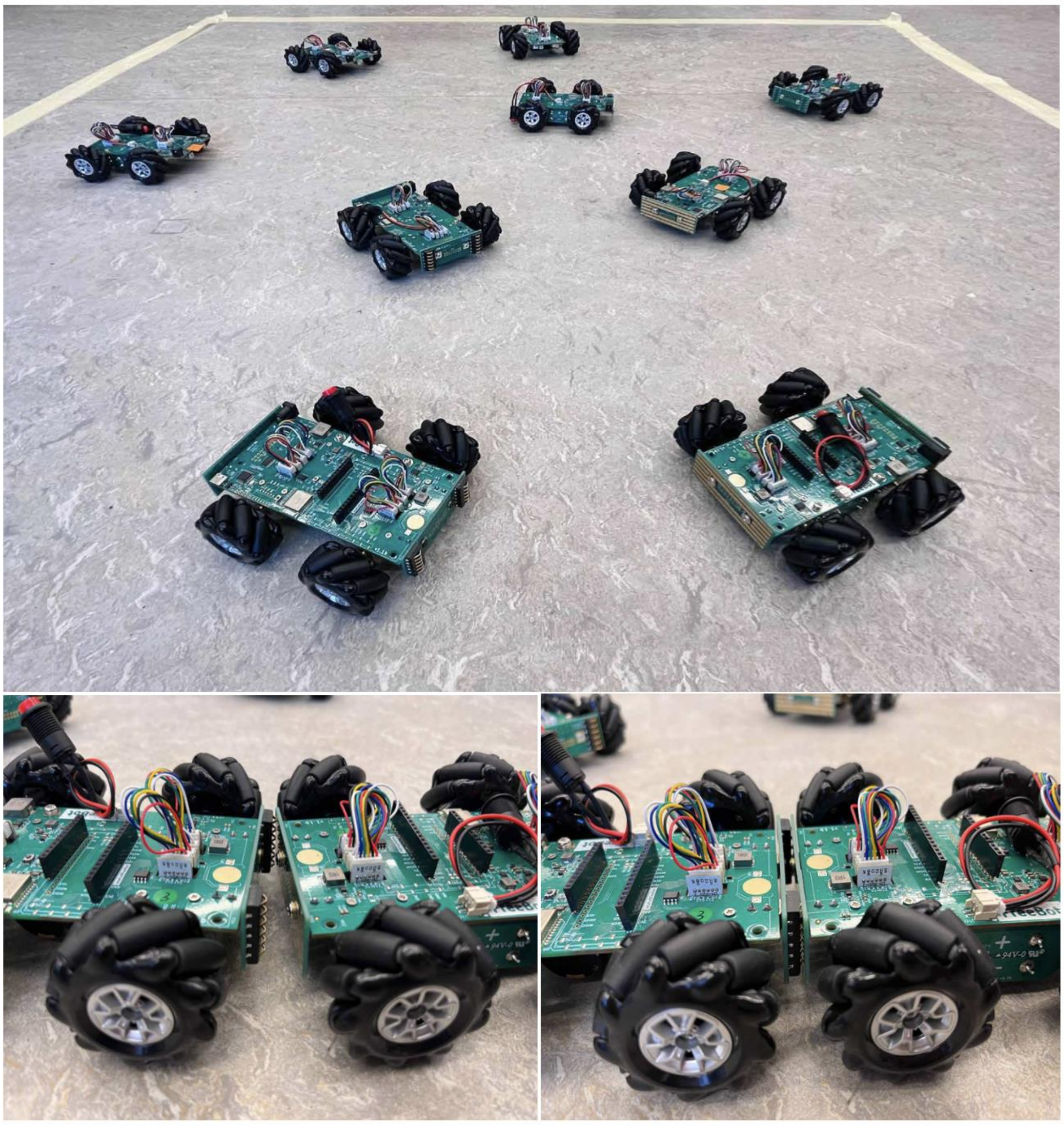


Figure 2: A group of CapBots (top), two of which perform trophallaxis to transfer charge (bottom).

Hardware and software for the CapBot is available under an open-source license at: https://github.com/openswarm-eu/ICRA2025_BatteryFreeRobot. The device has a low bill-of-materials at under $50 (US). We note that a paper describing the CapBot has also been submitted to the technical paper track of ICRA 2025. A video showing the platform in operation is available here: https://youtu.be/GnikfWc2suw.

A. *Languages and Supporting Software Stack*

The tutorial will be executed by pairs of attendees, who will work in two complementary and interacting tracks:

- *Embedded track:* targets attendees with basic low level programming skills in C or C++. This track will gain experience of controlling the CapBot by writing code for the Zephyr OS [3] running on the robot's onboard nRF52840[1] microcontroller and control code for the robots BLE radio [4].
- *Mainstream track:* targets attendees with basic Python programming skills on mainstream platforms (i.e. laptops and servers). This track will gain experience of gathering robot telemetry using the MQTT middleware [5] and visualizing this data in a control/monitoring console using the TICK stack.

It is our hope that by pairing attendees with different skills, we will promote interaction between different areas of the robotics community. Table 1 summarizes the technologies used and their purpose.

Table 1: Supporting Software Stack

Robot Firmware	Zephyr OS, a popular real-time OS for low-end embedded devices [3].
Wireless Networking	Bluetooth Low Energy (BLE): a ubiquitous low-power wireless network [4].
Middleware	Message Queue Telemetry Transport (MQTT), a popular publish-subscribe middleware [5].

Server Software	The TICK Stack (Telegraph, Influx, Capacitor, Kronograf) [6], an open-source database, visualization and alerting system.

While the complete software stack may appear complex, this complexity will be mitigated by the provision of template solutions, which will be iteratively released to attendees. This enables attendees to do creative work, while ensuring that they do not fall behind as solutions to each stage of the tutorial will be provided along a fixed timeline.

The tutorial builds on two years of experience using the CapBot in teaching medium-sized classes of 50 masters students at KU Leuven on courses including Software for Embedded Systems, Industrial Internet Infrastructure and Software for Realtime Computing. Student feedback has been very positive in all cases.

III. Relevance to ICRA 2025

In our view this tutorial captures an interesting element of capture the *present and the future of robotics*, moving beyond inefficient and polluting battery-based power sources towards more environmentally friendly energy storage methods, as well as showcasing novel approaches to charging robots and sharing energy, such as trophallaxis [8].

The tutorial material is state of the art in the sense that the CapBot was accepted for demonstration at SenSys 2023. a leading IoT conference [7] and is under submission to ICRA as a full technical paper. Similarly the lighthouse localization technique that we employ was accepted for demonstration at ICRA 2024 [2] and won the best demonstration award. Likewise, the supporting software technologies used in this proposal (i.e. Zephyr [3], BLE [4], MQTT [5] and TICK [6]) represent a 'best of breed' in their respective areas.

We believe that a tutorial showcasing battery free robots with precision localization that can perform rapid trophallaxis [8] represents an original tutorial that will expand the diversity and originality of workshop and tutorial content at ICRA 2025.

IV. Format of the Workshop and Tentative Schedule

The tutorial is comprised of a mix of *presentations* from experts on the underlying technologies, and interactive activities in the form of programming sessions and collaborative swarm activities. We aim to support a maximum of 50 attendees. A tentative schedule for the tutorial is shown in Table 2 below.

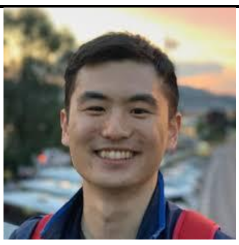
Table 2: Tentative Tutorial Schedule

	Embedded Track	**Mainstream Track**
00:00 – 00:15	Welcome and Introductions by Danny Hughes	
00:15 – 00:30	Introducing the CapBot by Mengyao Liu	
00:30 – 01:00	Wireless Motor Control (Bluetooth)	Visualizing CapBot Telemetry (TICK)
01:00 – 01:15	Introduction to Lighthouse localization by Thomas Watteyne	
01:15 – 02:00	Trophallaxis Control (Zephyr)	Connect location to locomotion (MQTT)
02:00 – 03:00	Putting it all together: integrating the embedded and mainstream (i.e. server or laptop) code.	
03:00 – 03:15	Mission briefing: save the stranded robot! by Genki Miyauchi	
03:15 – 04:15	All teams collaborate to rescue a discharged and stranded robot using trophallaxis.	
04:15 – 04:30	Participant feedback and concluding thoughts. by Danny Hughes	

Table 2 shows the proposed schedule for the tutorial, which lasts 4 hours 30 minutes. We aim for maximum interaction, with only 1 hour and 15 minutes allocated to presentations, compared to 3 hours and 15 minutes for programming and robot missions. We have developed a rewarding track for both mainstream (i.e. laptop/server) programmers who can work in python and embedded developers who can work in C.

As programmers with these skillsets tend to have different backgrounds, we expect that this will promote additional interaction between attendees. Table 4 lists the workshop speakers and their backgrounds.

Table 3: Confirmed Speakers

	Prof. Danny Hughes, head of Networked Embedded Software (NES) taskforce of DistriNet, KU Leuven (Belgium): https://distrinet.cs.kuleuven.be/people/DannyHughes
	Mengyao Liu, 3rd Year PhD Student at DistriNet, KU Leuven (Belgium) and designer of the CapBot platform: https://distrinet1.cs.kuleuven.be/people/MengyaoLiu
	Dr. Thomas Watteyne, Research Scientist & Innovator with the AIO Team at INRIA Paris (France): https://twatteyne.wordpress.com/
	Dr. Genki Miyauchi, Postdoc with the Natural Robotics Lab at the University of Sheffield (UK): https://www.sheffield.ac.uk/naturalrobotics/people/genki-miyauchi

As can be seen from Table 3, the confirmed speakers of the tutorial span a range of research areas (Embedded Systems, Internet of Things and Robotics), seniority levels (from young PhD researchers to experienced Professors) and demographics, hailing from five different countries and four different academic institutions. We hope that this will enable us to connect to attendees from a similarly diverse range of backgrounds.

The event is intended as an in-person hands-on event, though presentations will be recorded along with a highlight reel of the robot missions (subject to attendee agreement). As described previously, this tutorial builds on two years of expertise delivering lab assignments using the CapBot platform at KU Leuven for large groups of students. Attendees need only basic programming experience using either C or Python to successfully complete the tutorial.

V. Plan to Solicit Participation

The target audience for this tutorial are robotics practitioners and researchers with basic programming skills. We will reach out to this audience through a range of media:

– *Email lists* such as robotics-worldwide, ACM SigMobile and seworld.
– *Social media platforms* such as LinkedIn, X and Threads, including the creation of a dedicated advertisement video on YouTube.
– *Flyers and posters* at related events the organizers will attend between now and the conference
– *Leveraging existing academic networks* through the partners portfolio of projects and collaborators.

Based upon our past expertise of delivering this type of hands-on technical tutorial we expect a minimum attendance of 25 and can support up to a maximum of 50.

VI. Plan to Encourage Interaction

We have taken great care to craft an inherently interactive tutorial schedule, which we hope will be a memorable event for attendees. By working on a shared problem in pairs that combine different skillsets, we hope to break down barriers between attendees. This team interaction builds throughout the schedule to the group-wide collaborative 'robot rescue' exercise, which is educational, while at the same time aiming to retain fun and game-like elements.

In addition to the core team activities, we will begin the event with an interactive introduction and 'icebreaker'. After this, the organizers will lead by example, circulating throughout the tutorial, connecting with the attendees, helping them to build their code and answering all questions.

The tutorial will conclude with a feedback session and an invitation for the attendees to feed into the CapBot design and stay in touch with organizers to begin building a community around battery free swarm robotics.

VII. Dissemination Plan

As a programming-focused tutorial, we do not expect to directly disseminate attendee submissions following the event. Nevertheless, to maximize the impact of the event and promote future events on the same theme, we will record all presentations for publication on YouTube and other social media networks, along with a highlight reel of the attendee experience and especially the results of the collaborative 'robot rescue' mission. All materials will, naturally, also be hosted on the tutorial website, which will be linked to the main www.openawarm.eu domain.

VIII. Diversity and Inclusion

We aim to promote diversity of attendees in terms of race, gender, age and socioeconomic backgrounds. This begins with our speaker lineup which has diverse demographics and seniority levels, with speakers from several different countries and speaking a variety of languages. During the tutorial, we will do everything within our power to foster a universally welcoming event, using inclusive language in all communications, and remaining appropriately sensitive to cultural differences. We will pay particular attention to diversity and inclusion in the post-event feedback.

Considering technical background, we aim to be as inclusive as possible. While we do require basic programming expertise, we have created two rewarding and educational paths through the tutorial for both high-level Python developers and low-level C developers. In terms of socio-economic background, low-cost robotics platforms such as CapBot, which is open source and has a bill-of-materials of under $50 (US), can be a significant enabler for expanding access to robotics research and education.

IX. Compliance

The organizers will be present at the workshop and that the workshop will comply with the RAS Guidelines. The tutorial organizers have read and wil abide by the IEEE RAS Code of Conduct.

X. Required and Provided Equipment

Required Equipment: a standard computer projector and screen together with sufficinent extension cords to provide a power socket within a reasonable distance of the attendees. Approximately 1m² of floor or table space is required per team of two attendees to adequately test the robots.

Provided Equipment: We wil provide one CapBot per attendee up to a maximium of 50 attendees, including a reaonable pool of spares (+10%) along with all required chargers, programmers and test equipment. In addition a micro-server will be used to host an instace of the TICK stack [6] and an MQTT broker [5] to support the tutorial.

Acknowledgment

This document is issued within the frame and for the purpose of the OpenSwarm project. This project has received funding from the European Union's Horizon Europe Framework under Grant 101093046. Views and opinions expressed are however those of the author(s) only and the European Commission is not responsible for any use that may be made of the information it contains.

References

[1] F. Brown, *Recyclable Phone Batteries Are Now A Reality*, August 2023, Accessed: September 25, 2024. [Online]. Available: https://happyeconews.com/recyclable-phone-batteries-are-now-a-reality/.

[2] S. Alvarado-Marin, C. Huidobro-Marin, M. Balbi, T. Savić, Thomas Watteyne, et al.. *Lighthouse Localization of Miniature Wireless Robots*. IEEE Robotics and Automation Letters, In press, pp.1-8. 10.1109/LRA.2024.3405345. hal-04589717

[3] The Zephyr Project Accessed: September 25, 2024. [Online]. Available: https://zephyrproject.org/

[4] Bluetooth Low Energy standard, Accessed: September 25, 2024. [Online]. Available: https://www.bluetooth.com/learn-about-bluetooth/tech-overview/

[5] Message Queue Telemetry Transport (MQTT), Accessed: September 25, 2024. [Online]. Available: https://mqtt.org/

[6] The TICK Stack, Accessed: September 25, 2024. [Online]. Available: https://www.influxdata.com/blog/introduction-to-influxdatas-influxdb-and-tick-stack/

[7] M. Liu, F. Yang, S. Michiels, T. Van Eyck, D. Hughes, S. Alvarado-Marin, F. Maksimovic, and T. Watteyne. 2024. Demo Abstract: FreeBot, a Battery-Free Swarm

Robotics Platform. In Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems (SenSys '23). Association for Computing Machinery, New York, NY, USA, 488–489. https://doi.org/10.1145/3625687.3628401

[8] H. Schioler and T. D. Ngo, "Trophallaxis in robotic swarms - beyond energy autonomy," in 2008 10th International Conference on Control, Automation, Robotics and Vision, 2008, pp. 1526–1533.