UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Paulo Alfredo Frota Rezeck

HeRo: An Open Platform for Robotics Research and Education

Belo Horizonte
2019

Paulo Alfredo Frota Rezeck

HeRo: An Open Platform for Robotics Research and Education

Final Version

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Master in Computer Science.

Advisor: Luiz Chaimowicz

Belo Horizonte
2019

Rezeck, Paulo Alfredo Frota

R467h HeRo [recurso eletrônico]: an open platform for robotics research and education / Paulo Alfredo Frota Rezeck — 2019.
1 recurso online (97 f. il, color.)

Orientador: Luiz Chaimowicz.
Dissertação (mestrado) - Universidade Federal de Minas Gerais, Departamento de Ciência da Computação, Instituto de Ciências Exatas
Referências: f. 91-97.

1. Computação – Teses. 2. Robótica – Teses. 3. Robôs móveis – Teses. 4. Robôs – Design – Teses. 5. Robôs Autônomos. I. Chaimowicz, Luiz. II. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Computação. III.Título.

CDU 519.6*82 (043)

Ficha Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa CRB 6/1510 Universidade Federal de Minas Gerais - ICEx

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

HERO: AN OPEN PLATFORM FOR ROBOTICS RESEARCH AND EDUCATION

PAULO ALFREDO FROTA REZECK

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

[illegible]

PROF. LUIZ CHAIMOWICZ - Orientador
Departamento de Ciência da Computação - UFMG

[illegible]

PROF. OMAR PARANAIBA VILELA NETO
Departamento de Ciência da Computação - UFMG

[illegible]

PROF. GUILHERME AUGUSTO SILVA PEREIRA
Departamento de Engenharia Mecânica e Aeroespacial - West Virginia University

Belo Horizonte, 7 de Fevereiro de 2019.

I humbly dedicate this thesis to the advancement of robotics, encapsulating earnest efforts aimed at fostering greater accessibility within this domain. It is with profound reverence and utmost sincerity that I offer this work as an open-source solution to inspire and invigorate the sphere of education and research in robotics, particularly in the context of Brazil, with the unwavering aspiration that it may propel the nation towards greater strides of knowledge and innovation in this promising field.

Acknowledgments

First and foremost, I express my deepest appreciation to my beloved wife, Caroline, whose unwavering love, support, and understanding have been a constant source of motivation and strength throughout this journey. To my entire family, with special mention to my father, mother, brother, and sister, I am eternally indebted for their boundless love, unwavering encouragement and support, which have been the bedrock of my accomplishments. I am also indebted to my parents-in-law for their remarkable friendship and warmth, as they have graciously embraced and supported me with open arms.

A debt of gratitude is due to Professor Luiz Chaimowicz, whose expert guidance, mentorship, and friendship have been pivotal in shaping my academic growth and success. His support and provision of essential resources have been indispensable in my pursuit of knowledge and expertise in this field. I also express heartfelt thanks to Héctor Azpúrua and Maurício Ferrari for their support, both technically and personally, and for their invaluable friendship, which has enriched my journey in countless ways. I also extend my sincere appreciation to the colleagues at VeRLab, notably Elerson Rubens, Rodrigo Chaves, Bruna Frade, Alan Neves, Felipe Cadar, Michel Melo, Washington Ramos, Lucas Carvalho, and many others, for their collaborative spirit and shared knowledge, which have been instrumental in shaping the ideas presented in this work.

A special acknowledgment is due to the esteemed members of this thesis committee, Professor Guilherme A. S. Pereira and Professor Omar P. V. Neto, whose suggestions and insightful feedback have significantly enriched the quality of this thesis.

Lastly, I sincerely express gratitude to the Instituto Tecnológico Vale (ITV), Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for their financial support, which has played a crucial role in enabling the realization of this research.

"*Simplicity is the ultimate sophistication.*"

(Leonardo da Vinci)

Resumo

Esta dissertação apresenta o design e desenvolvimento de uma nova plataforma robótica de baixo custo especificamente projetada para experimentos com enxames de robôs. Embora os robôs pequenos possam apresentar limitações inerentes em locomoção, sensoriamento e comunicação, grupos destes robôs simples podem superar restrições individuais e alcançar tarefas complexas, além das capacidades de um único robô. Dessa forma, a plataforma proposta segue o conceito de robótica de enxame, em que os robôs são otimizados para diferentes atividades coletivas. Apesar de vários robôs terem sido propostos, muitos destes são inadequados para enxames devido ao alto custo das plataformas e às logísticas de aquisição. Nesta dissertação, realizamos um levantamento dos robôs de enxame existentes, considerando sua adequação para uso em experimentos. Com base em nossa avaliação, a plataforma proposta se destaca devido à seu custo, facilidade de montagem e integração com o ROS (Robot Operating System), garantindo facilidade de programação. No projeto, priorizamos alta modularidade, eficiência de custos usando componentes disponíveis comercialmente e alta capacidades de processamento e sensoriamento. Experimentos demonstraram sua robustez, apresentando controle de movimento estável, localização precisa, sensoriamento de longo alcance e longa autonomia de energia. Com o ROS, a plataforma provou ser confiável e escalável em termos de programação e comunicação. Em conclusão, a plataforma proposta oferece uma solução viável e adequada para experimentação em enxames, devido ao seu custo reduzido, design amigável ao usuário e tamanho compacto. Trabalhos futuros devem aprimorar os filtros internos para localização e o processo de montagem.

Palavras-chave: Robótica Móvel, Design Robótico, Robótica de Enxame, Robô Autônomo

Abstract

This thesis presents the design and development of a novel low-cost robotic platform specifically tailored for swarm experimentation. While small robots may have inherent limitations in locomotion, sensing, and communication, leveraging groups of these simple robots can overcome individual constraints and achieve complex tasks beyond the capabilities of a single robot. Our platform embraces the concept of swarm robotics, wherein robots are optimized for generic collective activities. Although various swarm robots have been proposed, many of these are unsuitable, mainly due to the high cost of the platforms and the logistics for acquiring them. In this thesis, we surveyed existing swarm robot platforms, considering their suitability for experimentation. Based on our evaluation, our proposed platform stands out due to its affordability, ease of assembly, and seamless integration with ROS (Robot Operating System), ensuring convenient programmability. Throughout the design process, we prioritized high modularity, cost-effectiveness using commercially available components, and reliable processing power and sensing capabilities. Experiments were conducted on motion control, localization, distance sensing, power consumption, and communication to assess the robot's capabilities. The results demonstrated its robustness, presenting fast and stable motion control, accurate localization, long-range distance sensing, and long-term power autonomy. With ROS, the platform proved reliable and scalable in terms of programming and communication. In conclusion, our swarm platform offers a viable and suitable solution for swarm experimentation, owing to its reduced cost, user-friendly design, and compact size. Future work will focus on improving the internal filters for localization and the assembly process of the platform.

Keywords: Mobile Robotics, Robot Design, Swarm Robotics, Autonomous Robot

List of Figures

List of Tables

Lists of Acronyms

ADC Analog-to-Digital Converter.

CAPES Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior.

CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico.

DHCP Dynamic Host Configuration Protocol.

DIY Do It Yourself.

FOTA Firmware Over-The-Air.

I2C Inter-Integrated Circuit.

IDE Integrated Development Environment.

IMU Inertial Motion Unit.

IR Infrared Radiation.

ITV Instituto Tecnológico Vale.

LiDAR Light Detection and Ranging.

PCB Printed Circuit Board.

PID Proportional Integral Derivative.

PWM Pulse-Width Modulation.

RF Radio Frequency.

ROS Robot Operating System.

SLAM	Simultaneous Localization and Mapping.
UART	Universal Asynchronous Receiver/Transmitter.
URDF	Unified Robot Description Format.
UWB	Ultra Wideband.

Contents

Chapter 1

Introduction

The current state of electrical component miniaturization, coupled with the increasing efficiency in hardware and software, allows the development of smaller and compact robotic systems. The convenience of using these small, simple, yet capable robots has gathered the research community's attention toward practical applications using swarm robotics.

In this thesis, we take advantage of the recent technological advancements using mass-produced components that are smaller, affordable, and long-term available to design a novel robot tailored for swarm applications. The robot is called HeRo and stands as an entirely open, cost-effective, effortlessly assembled platform. The design and assembly procedures aligned with contemporary trends, including the Maker Movement and Do It Yourself (DIY), promote the replication of the robot using additive manufacturing and readily available components. Furthermore, the robot boasts seamless integration with the ROS, a preeminent robotic framework nowadays.

1.1 Contextualization

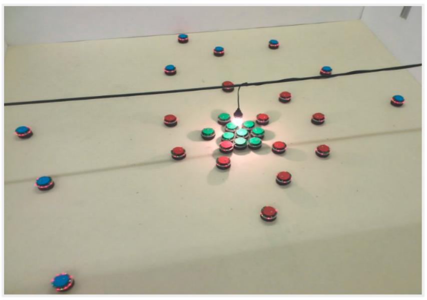
Small robots exhibit inherent limitations in terms of locomotion, sensing, and communication. While individually less powerful than their larger and more advanced counterparts, the aggregation of these simple robots can yield a multitude of advantages, facilitating the execution of intricate tasks that surpass the capabilities of a solitary unit.

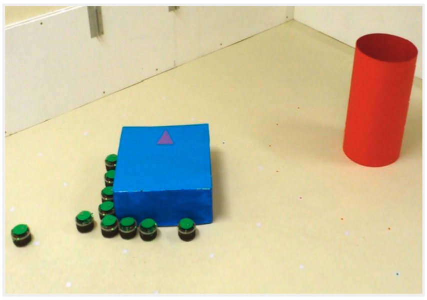
For instance, the collaborative efforts of a cluster of these robots engaged in expansive area exploration or the transportation of bulky loads might prove more effective and dependable compared to relying solely on a singular, more prominent robot.

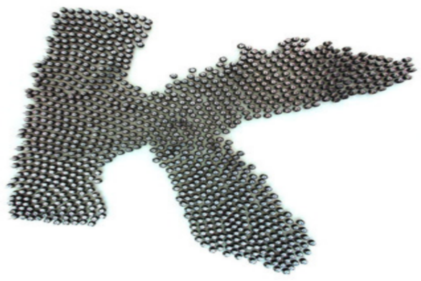
Swarm robotics constitutes a relatively new research field wherein intricate collective behaviors arise from the synergistic interaction of numerous robots, each endowed with autonomous functionality. The concept of swarm robotics originates from studying and modeling the impressive coordination abilities of biological systems such as ant or bee colonies (Bonabeau et al., 1999). What distinguishes this field is the elegance inherent in the simplicity of individuals' units, enclosing restricted localization, sensing, and communication capabilities, while inherently possessing attributes such as robustness, scalability, and versatility (Yogeswaran and Ponnambalam, 2010).

For instance, the utilization of a swarm of simple robots over more potent counterparts may bring numerous advantages, such as reduced costs, heightened reliability, augmented fault tolerance, distributed sensing, improved workload distribution, and massive task parallelization. In particular, groups of simpler robots working as a swarm have been the subject of study in the pursuit of various collective behaviors aimed at resolving tasks such as aggregation (Dorigo et al., 2004), segregation (Santos and Chaimowicz, 2014), pattern formation (Hsieh et al., 2008), self-assembly and morphogenesis (Baele et al., 2009), object clustering, assembly, and construction (Nitschke et al., 2012), collective search and exploration (Howard et al., 2006), collective motion (Marcolino et al., 2017; Inácio et al., 2018), collective transportation (Rubenstein et al., 2013), self-deployment (Couceiro et al., 2012), and foraging (Campo et al., 2010), among many others. Figure 1.1 presents examples of categories of swarm robots effectively addressing some of these tasks.

The potential of robotic swarms is increasingly aligning with a wide range of real-world applications, carrying significant societal and economic implications. The requirement for distributed and decentralized processing, relying only on local information, brings distinct practical benefits that set it apart from other robotic paradigms, affording scalability, resilience, and adaptability. This inherent attribute may further accentuate the use of swarm robots across a spectrum of tangible scenarios, including search and rescue


(a) Aggregation (Chen et al., 2012).


(b) Transportation (Chen et al., 2013).


(c) Self-organization (Rubenstein et al., 2014b).

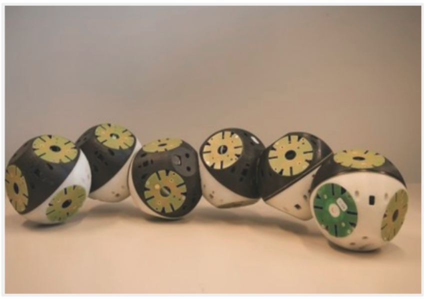

(d) Self-assembly (Sproewitz et al., 2010).

Figure 1.1: Swarm robots producing different collective behaviors.

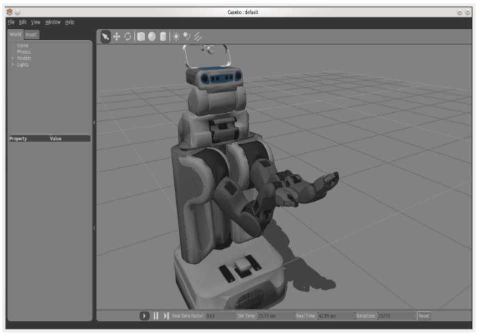
missions in hazardous or inaccessible locales, mitigation of oil spills in aquatic domains, transportation of weighty objects, environmental monitoring, and surveillance, among other noteworthy instances.

Nonetheless, the integration of robot swarms into operational contexts remains an aspiration not yet fully realized. Nowadays, progress primarily resides within the stage of experimentation, often confined to simulation and proof-of-concept showcases within laboratory settings or similarly controlled testing environments. This current situation can be attributed, in part, to the ongoing challenge of designing robots capable of translating prevailing theoretical frameworks into pragmatic solutions for real-world problems. Notably, some of the principal obstacles encompass tasks of achieving high-fidelity perception and communication, and optimal power utilization at a small form factor scale.

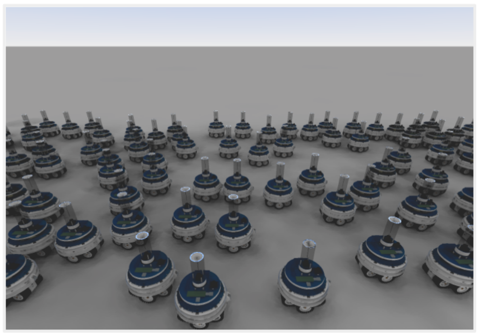
In conjunction with the technological limitations inherent in current swarm robots, researchers have encountered constraints arising from two crucial factors in numerous instances. The first factor resides in the relatively elevated costs associated with individual robots, both in terms of initial outlay and the temporal investments requisite for their development and assembly, thereby engendering challenges in securing funding and facilitating research endeavors. This consideration is of major significance in the context of swarm robotics, wherein a significant number of robots are requisite, necessitating a maintenance management policy and the use of a scalable framework to facilitate the deployment of robots. The second factor pertains to the requisite small dimensions of the robots, allowing for the experimentation of sizable collectives within limited spaces such as desktops or small rooms. However, this drive for compactness can potentially reduce robots' capabilities and restrict the range of algorithms for testing. Consequently, to circumvent the inherent trade-offs involving robot size, capabilities, and cost, a significant portion of experiments involving substantial robot groups are performed exclusively in simulation environments.

1.2 Motivation

The robot simulation stage constitutes a valuable instrument that facilitates the design and assessment of novel robotic systems. Over the years, numerous simulation frameworks have arisen, enabling researchers to replicate diverse robotic systems, ranging from sophisticated robots to simpler counterparts that engender intricate swarm behaviors, exemplified in Figure 1.2. Furthermore, robot simulation furnishes a wide range of options for tackling complex problems, affording researchers the opportunity to explore, visualize, and analyze robotic systems, even in the absence of real-world counterparts.



(a) Gazebo simulation a PR2 robot.



(b) ARGoS simulating swarmanoids robots.

Figure 1.2: Simulation of different robotic systems.

Preeminent among the advantages inherent in robot simulations over real experimentation is its reproducibility, that is, its ability to perform the same behavior when applied to the same conditions (Christiano et al., 2016). Beyond reproducibility, the employment of simulations offers other important benefits, enumerated as follows:

- **Affordability**: the necessity of acquiring costly robotic platforms for the validation of theoretical concepts is obviated, rendering simulations a cost-effective alternative;
- **Safety**: simulations offer a secure alternative for conducting experiments, devoid of potential risks to individuals or the physical integrity of the robot itself;
- **Robustness**: concerns about sensor malfunctions are alleviated, as simulations prevent the need to contend with hardware-related failures;
- **Temporal Manipulation**: simulation environments afford the flexibility to manipulate the passage of time, facilitating accelerated or decelerated experimentation;
- **Convenience**: the need for a pre-experiment setup, inclusive of addressing battery-related constraints, is obviated, enhancing the overall ease of experimentation.

Despite the evident advantages conferred by simulation, it is important to acknowledge that simulations do not always encapsulate the full spectrum of complexities and outcomes encountered in real-world experimentation. Even in instances where more precise simulators are employed to mirror real-world dynamics more faithfully, such fidelity can entail substantial computational demands, resulting in a scenario where the computation of seconds of simulation requires days, thereby posing challenges to the feasibility of ordinary simulations. Consequently, several significant aspects intrinsic to robotics physical dynamics may not be possible to model within simulation environments. Foremost among the limitations associated with simulation deployment are:

- **Under-Modeling**: many of pertinent physical phenomena remain unaccounted for within simulation models;

- **Parametric Mismatch**: even if the underlying physical equations are accurately represented, the estimation of appropriate parameters, such as inertia and friction, is of paramount importance;

- **Deformable Objects Complexity**: simulation of deformable entities, encompassing flexible bodies and interactions with fluids, poses considerable challenges.

Modern robotics research frequently leverages simulation as an initial developmental phase for methodology formulation and experimentation, followed by validation using real robots. However, within the field of swarm robotics, this approach is restricted due to the exigency of managing numerous robots, a potentially cost-intensive endeavor.

Early efforts have proposed diverse robotic platforms attempting to bridge the gap between simulation and real-world experimentation. However, a substantial proportion of these robots prove impractical, mainly attributed to the elevated costs associated with the platforms and the logistics for acquiring them.

1.3 Objectives

Given the cost-effective nature that characterizes research endeavors into robotic swarms, the objective of this thesis is to engineer a robotic platform tailored for swarm robotics, structured upon the following requisites:

- **Cost-Effectiveness**: robots should adhere to stringent affordability criteria, since most swarm configurations typically involve an extensive number of robots, ranging from dozens to hundreds;

- **Compact Proficiency**: robots should be compact in dimensions while concurrently incorporating sensing capacities, enabling dynamic environmental interactions. Furthermore, they ought to demonstrate extended battery lifespans, thereby ensuring the sustained operation necessary for the emergence of collective behavior;

- **Fault Resilience**: robots should exhibit robust fault tolerance attributes, bolstering their reliability within a swarm context;
- **Scalable Versatility**: robots should evince an inherent capability to execute diverse tasks as the number of robots expands in number. A case in point is the requisite scalable communication capabilities poised to accommodate a large number of robots;
- **Accessible Reproducibility**: robots should be featured by the uncomplicated assembly, even for people with fundamental electronics and mechanics skills;
- **Seamless Programmability**: robots should be characterized by facile programmability and seamless integration with contemporary robotic frameworks, such as ROS.

The nature of this thesis lies in reconciling all these conditions within a single design, which may pose a primary challenge. The design choices concerning a specific requisite, such as size, produce additional constraints that resonate across other domains, including sensing capabilities and power capabilities. Consequently, the design process requires an integrative approach that adeptly accommodates these manifold constraints, culminating in pragmatic design solutions amenable to multipurpose applications.

1.4 Contributions

Within the field of swarm robotics, this thesis has contributed through the design of a novel robotic platform engineered for swarm applications. The principal contributions within this work include:

- The design of a small robotic platform distinguished by its compelling affordability, priced at a mere 18 USD, and its streamlined assembly process through the integration of off-the-shelf components. This platform stands as an entirely open-source

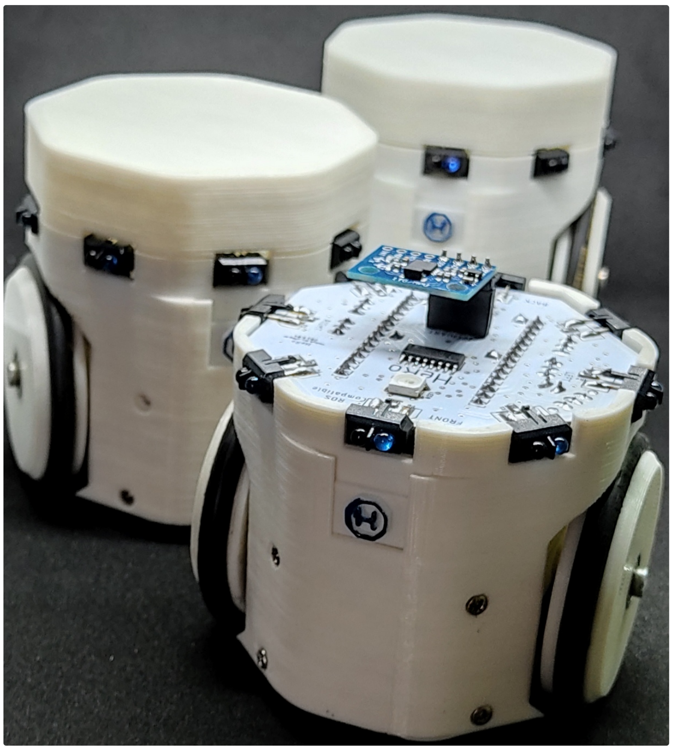


Figure 1.3: Design of the proposed open swarm robotic platform. The body of the robots was designed for and fabricated using additive manufacturing, while the electronic and mechanical parts were readily available components.

solution that is seamlessly integrated with ROS, the most used robotic framework available today. The design further aligns with emerging paradigms like the Maker Movement and DIY, thereby affording the potential for its replication. An illustrative depiction of this robotic platform is featured in Figure 1.3.

- Beyond its pertinence to the field of swarm robotics, this versatile robotic platform extends the potential to serve as a promising educational tool, poised to exert a substantial influence owing to its broad appeal and extensive applicability. The intersection of affordability and flexibility intrinsic to small and simple robots make them perfect as a pedagogical resource, particularly for instructing robotics subjects spanning mobile robotics, robot control, embedded computing, signal processing, introductory programming, wireless sensor networks, and even fundamental programming techniques.

1.5 Publications

The research efforts encapsulated within this thesis have culminated in important peer-reviewed publications listed in the following:

- *Paulo Rezeck, Héctor Azpúrua and Luiz Chaimowicz* **HeRo: An open platform for robotics research and education**. 2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR), Curitiba, Brazil, 2017, pp. 1-6, https://doi.org/10.1109/SBR-LARS-R.2017.8215317. [Qualis-CC B1]
- *Paulo Rezeck, Héctor Azpúrua, Maurício Corrêa and Luiz Chaimowicz* **HeRo 2.0: a low-cost robot for swarm robotics research**. Autonomous Robot (2023). https://doi.org/10.1007/s10514-023-10100-0. [Qualis-CC A2]

1.6 Organization

The remainder of this thesis is structured as follows. A review of the literature on small robot platforms and systems is delineated in Chapter 2. The mechanical and electrical design, as well as the software and communication architecture, are presented in Chapter 3 and Chapter 4, respectively. The robot's performance with respect to a set of metrics is evaluated experimentally in Chapter 5. In Chapter 6, we present the use of HeRo in some swarm applications. Finally, Chapter 7 brings the conclusions and directions for future work.

Chapter 2

Related Work

Over the years, researchers have delineated crucial attributes subjected to the design of robot swarms (Olaronke et al., 2020). In addition to aligning the design of these platforms with the overarching requisites of swarm applications, a paramount point is the attributes of footprint and cost, as these variables are pivotal in facilitating the scalability of swarm systems. Another point of equal significance is the integration of a scalable communication architecture, promoting seamless robot-robot interactions and the prospect of reprogramming through a remote server, ensuring that the addition of other robots does not restrict the robustness of the swarm's communication framework.

Other elementary capabilities a robot must have are a way of sensing and locomotion. Among the cost-effective options for sensing, IR range sensors emerge as a commonplace choice, enabling proximity estimation between the robot and environmental objects. Alternative sensing modalities include sonar arrays and camera sensors; however, these can be more prominent, cost-intensive, and typically necessitate data processing, which could strain embedded processors and increase power consumption.

Complementing sensing, locomotion is an imperative ability for robots to navigate from one locale to another. Constrained by the dimensions of the robot, electric motors, and wheels emerge as pragmatic actuation mechanisms. The prevailing paradigm adopts two actuated wheels to design differential-drive robots outstanding for their facile and effective control, especially when contrasted with intricate models like stick-slip omnidirectional drive, reliant on three vibration sticks for propulsion (Klingner et al., 2014). Another pivotal facet of a swarm robot is the capacity to estimate its position relative to a reference frame. A conventional technique within robotics is the computation of odometry, often achieved by attaching encoders to the motors for the estimation of wheel positions, subsequently enabling the computation of the distance traversed by the wheels.

Aligned with these requisites, a wide range of small and relatively simple robots have been proposed as a swarm platform. Most of these platforms stand as open-source solutions or incorporate components available through open-source channels, while a subset remains proprietary or confined to commercial outlets. Within this chapter, our discourse undertakes a comprehensive survey of the most salient and pertinent platforms tailored for swarm experimentation, expounding upon the pros and cons associated with each. We divide them according to their locomotion mechanisms and restrict this comparison to only small robots (less than 100 mm), which are generally more suitable for swarm robotics. A summary encapsulating this comparative assessment is presented within Table 2.1.

2.1 Vibration-based platforms

In recent advances, researchers have designed robots that employ vibration-based motion mechanisms. Generally, such mechanisms can be seamlessly integrated into the robot; however, they necessitate additional intricacies in the design of the robot's motion control algorithms. This mechanism also requires a smooth and even experimentation surface and may move relatively slowly. Furthermore, the absence of a convenient form of odometry renders precise long-distance movement challenging or impractical if such information is necessary.

A notable example of vibration-based robot that has garnered significant attention within the swarm robotics community is the **Kilobot** (Rubenstein et al., 2014a), originating from *Harvard University* - USA. This open-source platform, with individual components amounting to a mere 14 USD, has also been commercialized at a price point of 100 USD. Equipped with an ATmega328 (8-bit at 16 MHz) microcontroller, the Kilobot has an ambient light sensor on its top surface and an IR sensor on its bottom,

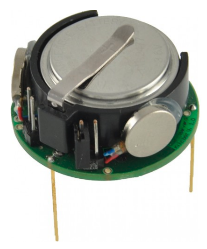
Table 2.1: Comparison of popular swarm robotics platforms.

Robot	**Cost (USD)**	**Size (mm)**	**Motion/Speed (mm/s)**	**Communication**	**Autonomy (h)**	**Builtin Sensors**	**Open Source**	**ROS Enabled**	**FW Programming**
Kilobot (Rubenstein et al., 2014a)	100/14*	33	vibration, 10	IR	3 – 24	proximity, light	✓	-	OTA
Droplets (Klingner et al., 2014)	100	44	vibration, 10	IR	∞	distance, light, bearing	✓	-	wired
Alice (Caprari and Siegwart, 2003)	N/A	22	wheel, 400	IR/RF	1 – 10	distance, light	-	-	wired
AMiR (Arvin et al., 2009)	85*	75	wheel, 100	IR	2	distance, light, bearing	-	-	wired
E-puck (Mondada et al., 2009)	975	70	wheel, 130	Wi-Fi/Bluetooth	1 – 3	distance, light, camera, mic, imu	-	✓	OTA
Jasmine (Kernbach, 2011)	120*	30	wheel, 300	IR	1 – 2	distance, light, bearing	✓	-	wired
GRITSBots (Pickem et al., 2015)	50*	30	wheel, 250	RF	1 – 10	distance, bearing, imu	✓	-	OTA
Zooids (Le Goc et al., 2016)	50*	26	wheel, 440	RF	1 – 2	touch sensor	✓	-	wired
MicroMVP (Yu et al., 2017)	90*	80	wheel, 250	Zigbee	1 – 2	-	✓	✓	wired
Cellulo (Özgür et al., 2017)	140*	75	wheel, 180	Bluetooth	2	touch, visual odometry	-	-	OTA
Colias IV (Hu et al., 2018)	100*	40	wheel, 350	Bluetooth	1 – 3	distance, light, camera, mic, imu	✓	-	wired
Mona (Arvin et al., 2018a)	120	65	wheel, 150	RF	∞	distance, light, encoder	✓	✓	wired
HeRo	**18***	**73**	**wheel, 250**	**Wi-Fi**	**3 – 9**	**distance, light, encoder, imu**	✓	✓	**OTA**

*parts only

used for both proximity readings and communication. Operating on a unique locomotion principle reliant on two vibration motors, this design choice reduces both cost and size. However, this condition also places an upper limit on the robot's top speed, capping it at 10 mm/s. The integration of an overhead controller device facilitates IR communication with all robots, enabling remote control and wireless firmware updates. Despite its relatively higher commercial cost and limited sensing capabilities, research endeavors have successfully carried out experiments involving up to 1000 robots (Slavkov et al., 2018), showcasing Kilobot's potential as an enticing platform for swarm applications. Refer to Figure 2.1a for an illustration of the Kilobot robot.

The **Droplet** (Klingner et al., 2014; Farrow et al., 2014), developed at the *Correll Robotics Lab* at the *University of Colorado Boulder*, USA, constitutes another compelling instance within the vibration-based small robot design. Slightly larger than the Kilobot (44 mm), the Droplet features enhancements in both its locomotion and sensing mechanisms. Featuring six IR sensors for proximity and orientation estimation, and inter-robot communication, the Droplet also employs three vibration motors to enable omnidirectional control — a pragmatic feature, given its modest speed (10 mm/s). The adoption of an Xmega128a3u (16-bit at 32 MHz) microcontroller indicates an improvement over Kilobot, endowing it with capabilities for control, data processing, and generalized computation. Augmenting its functionality, the Droplet is primed for sustained experimentation through an energetically charged floor bedecked with alternating positive charge and ground stripes. Beyond powering, this bedecked enables data transmission, facilitating swarm programming directly via the floor interface. While the commercial cost is similar to that of the Kilobot (100 USD), the Droplet necessitates a powered floor mechanism to sustain its operational power. Figure 2.1b shows the physical encapsulation of the Droplet, showcasing its 3D-printed plastic shell.


(a)

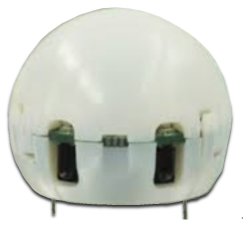

(b)

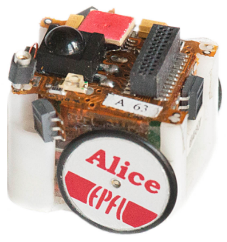
Figure 2.1: An illustration of **Kilobot** (a) and **Droplet** (b) robots. They represent two of the latest and more compact platforms for swarm robotics.

2.2 Wheel-based Platforms

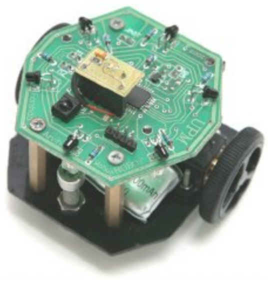
While vibration-based locomotion requires minimal mechanical complexity for actuating the robot, its efficacy diminishes when it comes to precision over long distances due to inherent nonlinearity and propensity for excessive slippage. Conversely, wheel-based systems offer enhanced practicality in terms of control and efficiency, as the torque generated by the motor acts directly and roughly linearly on the wheel. Presented below are some wheel-based robotic platforms.

Alice (Caprari and Siegwart, 2003), developed for swarm applications at the *Autonomous Systems Lab at École Polytechnique Fédérale de Lausanne* (EPFL), Switzerland, stands as one of the early miniature robots in this category. Based on a two-wheeled differential drive configuration, Alice boasts a lightweight plastic chassis with a PCB on top. Occupying a small footprint of 22 mm, it uses a pair of high-efficiency swatch motors for locomotion, enabling speeds of up to 400 mm/s. A low-power PIC16F877 (8-bits at 4 MHz) microcontroller controls the robot and executes diverse applications. Designed for the use of various sensory modules, the robot houses 4 IR sensors positioned around its perimeter to facilitate obstacle detection and short-range robot-to-robot communication. An upper-mounted IR receiver enables external command reception, while a Radio Frequency (RF) module facilitates remote communication. Additionally, Alice supports varied expansion modules, for instance, a gripper module and a linear camera. The initial version of Alice uses two watch batteries as its power source. Subsequently, these batteries were replaced by solar panels and lithium batteries to enhance autonomy. This configuration facilitated continuous charging, enabling the robot to operate for around ten hours typically. However, the authors noted that the light source utilized for solar panels was incompatible with the robot's IR proximity sensors, thereby curtailing the spectrum of applications for this robot. Refer to Figure 2.2a for a depiction of the latest iteration of the Alice robot.

AMiR (Arvin et al., 2009), an open-source robot originating from the *University Putra*, Malaysia, offers a budget-friendly solution with an assembly cost of approximately 85 USD and a compact 75 mm footprint. The robot is actuated by two-wheeled differential drive mechanics that use two micro DC internal gear motors to reach speeds of up to 100 mm/s. An ATmega168 (8-bits 8 MHz) microcontroller handles communication, motion control, perception, and user-defined tasks. The robot carries 6 IR sensors enabling proximity and bearing estimation and also short-range robot-to-robot communication. It uses a 3.7 VDC 200 mAh lithium battery allowing it to operate for up to 2 hours. While programming the robots requires them to be connected by cable to a computer, simulated models in Player and Stage facilitated the development and test of user tasks. The AMiR's adoption as a swarm solution is highlighted by various researchers and robotics educators (Arvin et al., 2011, 2014). See Figure 2.2b for its visual representation.



(a)

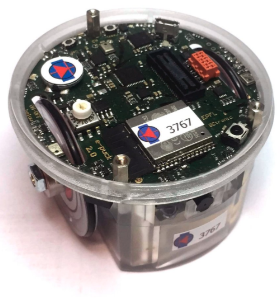


(b)

Figure 2.2: An illustration of **Alice** (a) and **AMiR** (b) robots. They represent two of the early robots for swarm experimentation

The **E-puck** (Mondada et al., 2009) stands as a notable achievement among compact commercial robots, initially tailored for educational purposes and subsequently embraced for research in swarm robotics. Using a two-wheeled differential drive mechanism, the E-puck carries a price tag of approximately 975 USD. With a modest 70 mm footprint, the robot employs two planetary-geared step motors for actuation, enabling velocities of up to 100 mm/s. The latest iteration of the E-puck integrates an STM32F4 (32-bits at 180 MHz) microcontroller, while an Espressif ESP32 serves as the Wi-Fi/Bluetooth module. The robot boasts versatility when equipped with diverse built-in sensors, encompassing microphone arrays, proximity sensors, a 640 × 480 pixel camera, and an inertial motion unit. Programming the E-puck is facilitated through a serial cable or Bluetooth interface, with Wi-Fi handling communication with a server or setting mesh network. The platform's extendability is evident through sensor augmentation possibilities, including omnidirectional camera and bearing modules, alongside potential processing module integration using Raspberry Pi. Powering its operations is a 3.7 VDC 1200 mAh lithium battery, affording operational times of up to 3 hours. Bolstered by an expanding user community, the E-puck's software, documentation, and discussion forums thrive, favoring its integration with diverse simulators and robotics frameworks, such as Gazebo and ROS. Notwithstanding its merits, the commercial version's cost renders the E-puck inaccessible for large-scale swarm implementations. Figure 2.3a shows the recent e-puck robot.

Jasmine (Kernbach, 2011), developed at the *University of Stuttgart*, Germany, stands as another widely adopted two-wheeled differential drive small robot. With a price tag of around 120 USD for its parts, Jasmine presents an undersized footprint of just 30 mm. Its motion is promoted by two micro DC internal gear motors, allowing the wheels to reach a maximum speed of 300 mm/s. The third version of the robot is equipped with an ATmega168 (8-bits at 20 MHz) microcontroller and uses 6 IR sensors for proximity and bearing estimation, light measurements, and communication with other robots. Positioned atop the robot are LEDs that serve a dual purpose, offering status monitoring and a means for debugging. The platform also encourages customization through various tailored boards that can expand its capabilities, enhancing sensing and connectivity, among other attributes. On the energy front, the latest version of Jasmine relies on a 3.7 VDC 250 mAh lithium battery, affording operational times of up to 2 hours. Notably, the robots exhibit autonomous recharging capabilities as they engage with a pair of metal contacts (power and ground) affixed to a wall. This convenient mechanism enables the robot to detect its battery's need for recharge, prompting autonomous navigation to a recharging dock without the need for human intervention. Figure 2.3b depicts the Jasmine robot with an upper-extension board that enables global localization capabilities.


(a)

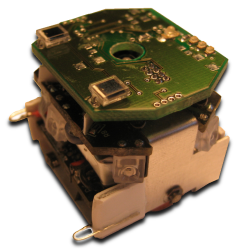

(b)

Figure 2.3: An illustration showcasing the **E-puck** (a) and **Jasmine** (b) robots is presented. These robots stand as prominent choices extensively employed in swarm research efforts.

GRITSBot (Pickem et al., 2015) is a compact robot developed at the *Georgia Institute of Technology*, USA. GRITSBot plays an essential role within the *Robotarium* project, an initiative that strives to democratize multi-agent experiments by providing the research community with accessible testbed resources (Pickem et al., 2017; Wilson et al., 2020). Structurally, the GRITSBot exhibits a small footprint (30 mm), with component costs amounting to a mere 50 USD. The robot is ingeniously structured into three modular layers, each accommodating five integral functional components. The motor layer governs the control of two stepper motors for motion and odometry estimation. The mainboard boasts an Atmega328 microcontroller, operating at 8-bit and 16 MHz, alongside a wireless communication module, a battery charging circuit, and a power supply. A Nordic nRF24L01 microchip, adept at low-power communication and operating at 2.4GHz, enables robot-to-robot interaction, over-the-air firmware updates, and remote server-based control. The adoption of these low-power transceivers stems from their efficient power consumption, which contrasts with Wi-Fi. However, their trade-off is a lower data rate capped at 2 Mbit/s. Furthermore, their operation on the same frequency as Wi-Fi makes them susceptible to interferences in indoor environments. The sensor layer integrates six IR distance sensors, an accelerometer, and a gyroscope. A 3.7 VDC 400 mAh LiPo battery empowers the robot, granting it extensive operational autonomy of up to five hours. It is noteworthy that the robots are further endowed with autonomous movement capabilities to a power source, facilitating the automated recharge of batteries. This ingenious robot design is visually represented in Figure 2.4a.

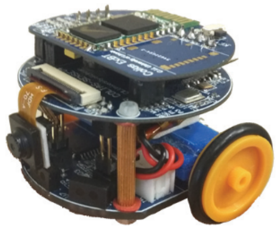
Zooid (Le Goc et al., 2016), a design born out of a collaborative effort between the Shape Lab at *Stanford University* (USA) and the Aviz team at *Inria* (France), represents a remarkable open-platform small robot tailored for swarm applications. Fitting an impressive footprint of just 26 mm, this open-source robot is available at an approximate cost of 50 USD. Despite its diminutive size, Zooid incorporates an intricately designed non-collinear motor configuration, a strategic choice that contributes to its compact form and agility, echoing the performance of robots with colinear motors. At its computational core lies an STM32F4 (32-bit at 48 MHz) microcontroller, managing logic computations while establishing seamless wireless communication with a master computer via an nrf24L01 2.4 GHz radio chip. Distinguished by its tactile sensors, Zooid boasts touch sensors that align with the tactile demands of swarm applications. Notably, the robot's capacity for localization is facilitated by a distinctive projector-based tracking system. This sophisticated system projects sequences of gray-coded patterns onto a flat surface, permitting Zooid's photodiodes to unravel these patterns into precise position and orientation determinations. A salient advantage emerges in the form of a projector-based tracking system's real-time stability, as it circumvents latency typically associated with networked communication for local feedback control, ensuring robust and steadfast position control. This innovation, however, comes at an estimated cost of 700 USD, an investment that is reflected in its noteworthy precision, comparable to that of overhead-camera localization systems. The visual representation of the Zooid robot is featured in Figure 2.4b.

(a) (b)

Figure 2.4: An illustrative display featuring the **GRITSBot** (a) and **Zooids** (b) robots, two of the most miniature wheeled-based swarm robots.

Colias introduces a compelling alternative to AMiR, custom-designed for swarm robotic applications and originating from the *University of Lincoln*, UK. Costing around 100 USD in parts, Colias employs only PCB boards as its chassis, resulting in a compact footprint of 40 mm. The design of Colias emphasizes modularity through extension boards, enabling distinct features and functions to operate independently. Powered by an ATmega168 (8-bit at 8 MHz) microcontroller, the mainboard undertakes motor control and data processing. Additionally, the mainboard houses IR sensors dedicated to proximity measurements and obstacle detection. The locomotion system leverages two differential-driven wheels capable of achieving a maximum speed of 350 mm/s. The recent iteration, Colias IV (Hu et al., 2018), introduces several advancements. Augmented with a powerful ARM Cortex M4 microcontroller operating at 180MHz, this version integrates two digital microphones, a 9-axis motion sensor, and a compact VGA camera for visual tasks. Expanding its communication capabilities, a Bluetooth extension module empowers Colias IV to establish a connection with remote host devices like laptops or smartphones. This functionality enables the reception of motion commands and the transmission of sensor data. In addition to its hardware prowess, Colias comes equipped with a suite of fundamental software libraries for sensor data interpretation and motion control. However, programming still necessitates a physical connection with the user computer, which could potentially limit the scalability of experimentation. Figure 2.5a showcases the latest design iteration of the Colias robot.

MicroMVP (Yu et al., 2017) is a compact robot developed at MIT, USA, featuring an open-source design with a chassis crafted through 3D printing technology. Boasting an inherently straightforward and effortless assembly process, MicroMVP's compact footprint measures just 80 mm. Costing approximately 90 USD to construct, this robot is built around an ATmega32U4 (8-bit at 16 MHz) microcontroller outfitted with integrated xBee radio support. The robot's locomotion is facilitated by a pair of DC-geared motors that actuate the wheels. Adhering to simplicity, MicroMVP employs off-the-shelf components exclusively and does not provide any form of sensing, thereby constraining its potential as a versatile swarm robot. However, this platform utilizes an overhead camera alongside fiducial markers affixed atop the robot to enable the localization of MicroMVP robots, consequently serving as a mechanism for closed-loop control or emulating sensors. It is worth highlighting that MicroMVP employs relatively pricier components, culminating in an estimated assembly cost of 90 USD. Figure 2.5b visually encapsulates the fully assembled MicroMVP robot.


(a)

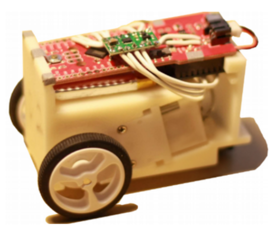

(b)

Figure 2.5: An illustrative display featuring the **Colias IV** (a) and **MicroMVP** (b) robots, two recent alternatives for swarm robotics.

Cellulo (Özgür et al., 2017) stands as a pioneering tactile small robot platform, originating from *École Polytechnique Fédérale de Lausanne* (EPFL), Switzerland. Distinguished by its fusion of autonomous capabilities with haptic-enabled multi-user tactile interaction, Cellulo serves as a versatile research tool in fields encompassing rehabilitation, gaming, and human-computer interaction. Characterized by its compact form factor and cost-effectiveness, the robot chassis consists of 3D printed components, encapsulating its essence within a diminutive 75 mm footprint. The construction of the Cellulo robot entails an approximate expenditure of 120 USD. The current iteration of the Cellulo robot integrates a myriad of features, including a self-localization system founded on an activity sheet (covering the arena surface) and a downward-facing camera, six capacitive touch buttons, Bluetooth communication, a PIC32MZ microcontroller boasting 32-bit processing at 200 MHz and holonomic motion. This motion configuration employs an omnidirectional ball drive mechanism (Özgür et al., 2016), empowering the robot to initiate motion in any conceivable direction, thereby transcending the limitations of differential drive motion commonly found in swarm-like robots. Significantly augmenting its utility, the localization system grants the ability to ascertain the global pose of multiple robots, even in the presence of challenges such as kidnapping and occlusions stemming from user interaction.

However, the system does exhibit certain limitations; its deployment and storage procedures are intricate, and it remains susceptible to external contaminants, exemplified by dust and wheel marks on the activity sheet. Moreover, as the robot's holonomic motion relies upon an omnidirectional ball drive, the accumulation of rubber shards within the mechanism over time may impact its performance. Figure 2.6a visually encapsulates the essence of the Cellulo robot, a synthesis of innovation and practicality.

Mona (Arvin et al., 2018a) emerges as a versatile open-source robot, conceived as a customized evolution of the Colias platform. Built at the *University of Manchester*, UK, Mona serves as a platform for investigating the feasibility of the Perpetual Robotic Swarm concept (Arvin et al., 2018b). It features a small footprint with dimensions of 65 mm and is equipped with a low-cost ATmega328 (8-bit at 16 MHz) microcontroller. The robot's drive mechanism is composed of two wheels, enabling it to reach a maximum speed of 150 mm/s. Designed with modularity in mind, Mona accommodates additional modules, facilitating expansions like wireless communication, vision, and computation capabilities. Mona's unique attribute lies in its inductive charging approach, complemented by features such as a RF transceiver and battery-level monitoring module. This pack of attributes empowers Mona with large-scale, enduring autonomy for robotics research. Exhibiting compatibility with various standard programming environments, Mona finds dual utility in education and research at the *University of Manchester*. Born from collaboration with a commercial partner, the robot is affordably priced at 120 USD per unit, making it accessible to those delving into the field of swarm robotics. Mona remains grounded in the spirit of open-source philosophy, encompassing both hardware and software aspects of its design. Figure 2.6b showcases the latest iteration of the Mona robot, encapsulating its innovation and potential.

(a)

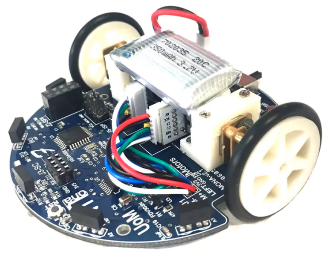

(b)

Figure 2.6: An illustrative showcase highlighting the **Cellulo** (a) and **Mona** (b) robots, representing pioneering achievements in tactile interaction and perpetual autonomy.

2.3 Design Choices

The comparison presented in Table 2.1 highlights various design choices that are prevalent across multiple platforms. Common features include the use of wheel-based locomotion and the integration of distance sensors within the robots' sensor arrays. In contrast, certain advanced attributes are found in only a select few designs, like Wi-Fi communication and compatibility with ROS, potentially due to recent advancements making these technologies more accessible. In the case of HeRo, we have aimed to blend established and widespread solutions with innovative enhancements, resulting in robots that are both versatile and dependable. The subsequent section provides an overview of HeRo's key characteristics, which will be elaborated upon in the subsequent chapters.

2.4 Proposed platform: HeRo

In this thesis, we introduce the project and implementation of a novel small robot tailored for swarm robotics applications. This robot boasts remarkable attributes, most notably its ultra-low cost, amounting to a mere 18 USD for its constituent parts. Furthermore, the robot is easy to assemble and is seamlessly integrated with ROS allowing easy programming. Illustrated in Figure 2.7 are a group of HeRo robots.

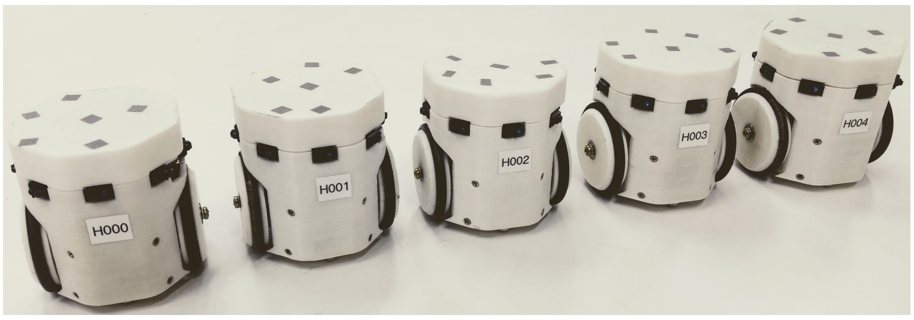


Figure 2.7: An illustration of a group of HeRo robots.

The presented version represents a significant advancement from its initial version (Rezeck et al., 2017). A summary of the characteristics of all HeRo versions is described in Table 2.2.

Table 2.2: Characteristics of the different HeRo versions.

	HeRo v0.1	HeRo v1.0	**HeRo v2.0**
Board	Arduino Nano	ESPressif ESP8266 - ESP12	ESPressif ESP8266 - ESP12
MCU	Atmel Atmega328 8-bit @ 16 MHz	Tensilica LX106 32-bit @ 80/160 MHz	Tensilica LX106 32-bit @ 80/160 MHz
Communication	RF nrf24l01 2.4 Ghz	Wi-Fi 802.11bgn	Wi-Fi 802.11bgn
Actuation	Servo Motors	Servo Motors	Servo Motors
Footprint	10 cm	8 cm	7.3 cm
Sensors	None	3 x IR sensors	8 x IR sensors, encoders and IMU
Battery	3.7 VDC 1000 mAh Li-Po	3.7 VDC 1000 mAh Li-Po	3.7 VDC 1800 mAh Li-Po
Cost*	9 USD	14 USD	18 USD

* parts only

In this enhanced iteration, the main microcontroller is an Espressif ESP8266 (32-bit at 160 MHz), responsible for executing motor control tasks, as well as acquiring and processing data from various onboard sensors. Notably, the microcontroller boasts a built-in Wi-Fi module, fostering resilient and dependable communication among the robots via TCP/IP protocols. The robot's movement mechanism relies on two differential-driven wheels, attaining a maximum velocity of 250 mm/s. The mainboard configuration incorporates an array of 8 IR sensors, enhancing the platform's sensing capabilities for light intensity and distance measurements, an essential component for obstacle detection. Additionally, an Inertial Motion Unit (IMU) contributes to refined odometry and general-use sensing, while the inclusion of rotary encoders significantly aids in localization and precise motion control.

A distinguishing characteristic of the mainboard lies in its modular design, which serves as a hallmark feature facilitating the effortless integration of a wide array of components. These encompass versatile additions such as cameras, communication radios, and displays, enhancing the platform's adaptability and functionality. This modular adaptability ensures that the platform can be tailored to specific research and application needs. To streamline programming, the platform supports Firmware Over-The-Air (FOTA) updates via a Wi-Fi interface. This capability empowers users to upload their custom codes to multiple robots remotely, simplifying the experimental process. Moreover, HeRo is engineered for compatibility with ROS, a widely used robotics framework. This compatibility is achieved through a TCP/IP connection that enables communication between HeRo and a remote computer executing ROS.

Given the imperative of power autonomy for extended experimentation, HeRo is equipped with a potent Li-Po battery, ensuring up to 10 hours of operation. The platform distinguishes itself through a careful balance of affordability and capacity, ease of assembly, and its seamless compatibility with ROS. These contributions collectively position HeRo as a distinctive and valuable asset in the evolving landscape of swarm robotics research and application. Subsequently, we delve into a comprehensive exposition of the mechanical and electrical design intricacies of the robot.

Chapter 3

Mechanical and Electrical Design

This chapter focuses on the mechanical and electrical design of our swarm robot. All decisions were driven by a pragmatic approach that prioritized the utilization of readily available components for straightforward production and assembly, all while maintaining a minimal cost without compromising processing power and sensing capabilities. Thus, we proceed to outline the most suitable architecture for HeRo, reached through an evaluation of various microcontroller boards, wireless technologies, sensors, actuators, and additive manufacturing model designs.

3.1 Mechanical Design

One of the primary steps in mobile robot development involves modeling its mechanical model. This process encompasses the definition of the robot's kinematic model, its method of actuation, and its underlying structural design.

3.1.1 Kinematic Model

After a thorough literature review, it became evident that the majority of robots designed for swarm robotics adhere to the differential-driven model. Essentially, a differential wheeled robot is characterized by two independently powered wheels positioned on opposite sides of its body. Directional changes are achieved by adjusting the relative speeds of these wheels, thereby eliminating the need for an extra steering motor. Figure 3.1 provides a visual depiction of the robot's differential drive mechanism in action.

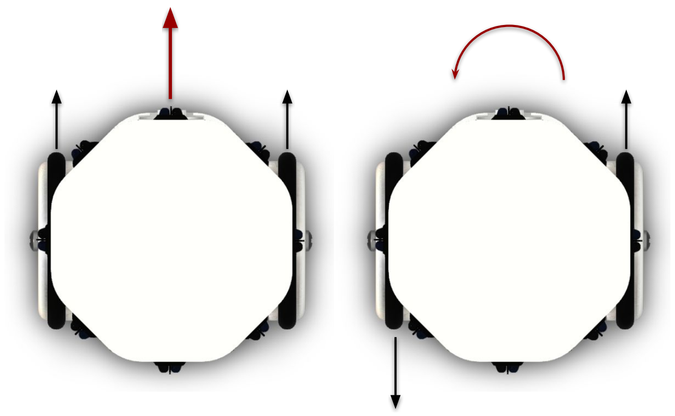


Figure 3.1: An illustration of a differential-driven robot. Two independently powered wheels on opposite sides of the robot's body provide maneuverability and speed control without an extra steering motor.

The choice to adopt this model was driven by its suitability for creating a compact, cost-effective robot that maintains excellent maneuverability and speed through a straightforward actuation mechanism.

3.1.2 Actuators and Encoders

The choice of actuators plays a crucial role in the dynamics of a differential robot. One common and cost-effective method to actuate the wheels involves the utilization of geared DC motors. These motors not only facilitate the robot's movement but also allow the integration of encoders to calculate odometry information, which is essential for tasks like localization and closed-loop motion control.

However, the incorporation of geared DC motors and encoders can notably elevate the overall cost of the robot. To align with our goal of creating a cost-effective solution, we opted for small continuous servo motors as the driving force for the robot's wheels.

These continuous servo motors are similar to geared DC motors, equipped with an *H-bridge* component for motor speed control. The cost-effectiveness and compact nature of commercially available servo motors render them an appealing solution for our robot.

The continuous *SG90 servo* motors exhibit satisfactory precision in speed control, bolstered by an integrated microchip that regulates motor speed and direction using Pulse-Width Modulation (PWM) signals. Additionally, these motors deliver a substantial torque of 1.8 kgf/cm, allowing us to employ a 50 mm diameter wheel to achieve a peak linear speed of 250 mm/s with a torque of 0.3 kgf/cm, all while maintaining optimal traction.

Instead of directly attaching the wheel to the motor shaft, we decided to design a mechanism where the wheel attaches to the robot chassis and a gear mechanism with ratio (1:1) transfers torque from the motor to the wheel. This configuration not only mitigates backlash and wheel misalignment issues that could impact encoder readings but also ensures a smoother transfer of power.

As the chosen continuous servo motors lack built-in encoders, we took advantage of the presence of larger wheels to devise a mechanical transmission system (1:6) connecting the wheel to a mechanical rotary encoder. In line with our criteria of affordability, availability, and compactness, we settled on *Kailh* rotary encoders. These encoders are commonly employed in mouse devices to quantify scroll button movement, registering 48 steps per cycle. By introducing a wheel-encoder transmission (1:288), we bolster the wheel's position measurement accuracy to 1.25° degrees of resolution. This enhancement translates to the robot detecting a wheel step of 0.54 mm during movement. Figure 3.2 visually depicts the motor-wheel and wheel-encoder transmission system, where the motor, rotary encoder, and wheel shaft remain fixed to the robot chassis while the other components remain dynamic. A video demonstration showcasing this motion transmission mechanism in action can be accessed at Youtube[1].

[1]Motion transmission mechanism: https://youtu.be/II_Zh2doAx4.

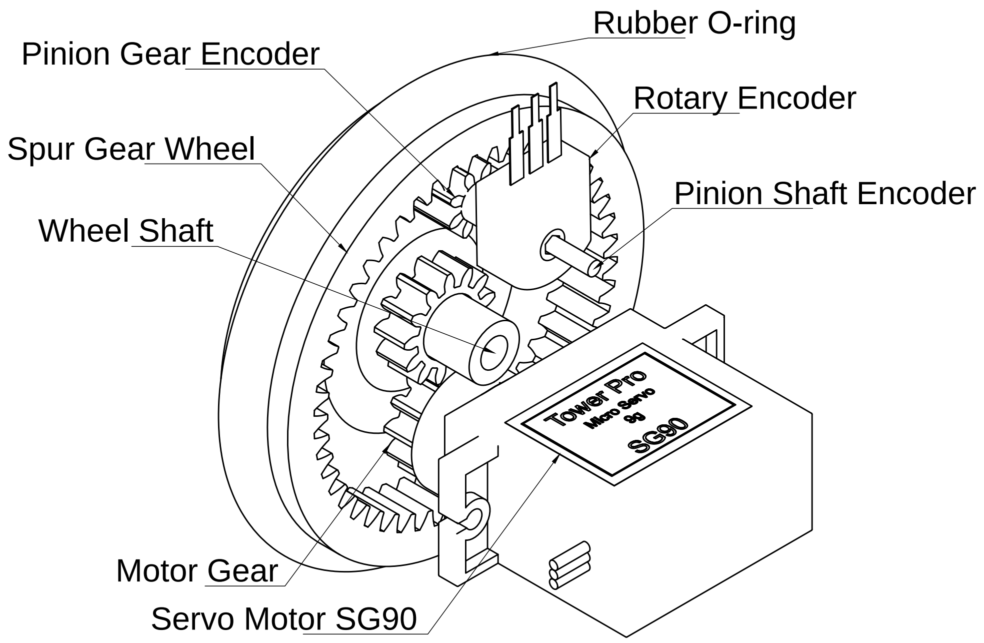


Figure 3.2: Torque transmission mechanism from the motor to wheel and wheel to the rotary encoder.

3.1.3 Structural Design

After defining the actuation mechanisms, the next step involves the design of the robot's chassis. Our approach emphasizes modularity and 3D printability to facilitate assembly and accommodate potential extensions. This enables easy fabrication through standard 3D printers. The overall structure of the robot encompasses four key components: the motor and board chassis, cover, and the e-Hat module.

The **motor chassis** serves as a basis for both motors and the wheel shafts to which the wheels are affixed. Given the robot's use of two actuated wheels, it balances on two points of contact with the ground. To ensure optimal balance and alignment, we incorporate two adjustable caster wheels. These caster wheels are affixed to the motor chassis and offer the ability to finely calibrate the robot's balance. Atop the motor chassis, the **board chassis** accommodates the encoders, battery, and the processing board.

Considering future expansions, we strategically utilize the inherent modularity of the chassis to introduce the innovative concept of the **e-Hat**. This component is seamlessly integrated onto the top section of the robot, effectively serving as an expandable shield that considerably widens the spectrum of both sensing and acting capabilities the robot can harness. As an illustrative example, we have successfully engineered an e-Hat variant equipped with an IMU sensor. However, the potential is not confined to this single implementation – various other modules can be seamlessly integrated into the e-Hat configuration. Possibilities include integrating cameras for visual perception, incorporating Light Detection and Ranging (LiDAR) devices for enhanced distance sensing, integrating actuators for manipulation, or even introducing Ultra Wideband (UWB) transceivers for global indoor localization – a testament to the flexibility and potential extensibility of the robot's architecture. This modular design empowers researchers and developers to readily tailor the robot's functionalities to the specific requirements of various applications.

To complete the design, we introduce a **cover** component. This part serves a dual purpose: preventing the accumulation of dust within the robot's internals while safeguarding the main processing board and gears. Beyond protection, the cover contributes to the robot's visual aesthetics. Figure 3.3 provides an expanded view of the robot's design, while Table 3.1 offers some key specifications. An interactive CAD visualization is accessible through the A360 platform[2].

Table 3.1: General specifications of the robot.

Specification	**Value**
Size	$0.068 \times 0.073 \times 0.076$ $(L \times W \times H)$ m
Weight	0.156 Kg
Moment of Inertia	$I_{xx} = I_{yy} = 1.27e^{-4}$ and $I_{zz} = 1.04e^{-4}$ Kgm$^{\mathbf{2}}$
Wheels Distance	0.0631 m
Wheel Diameter	0.0492 m
Linear Speed	0.25 m/s

[2]Robot Design CAD: https://a360.co/3lWHiv0.

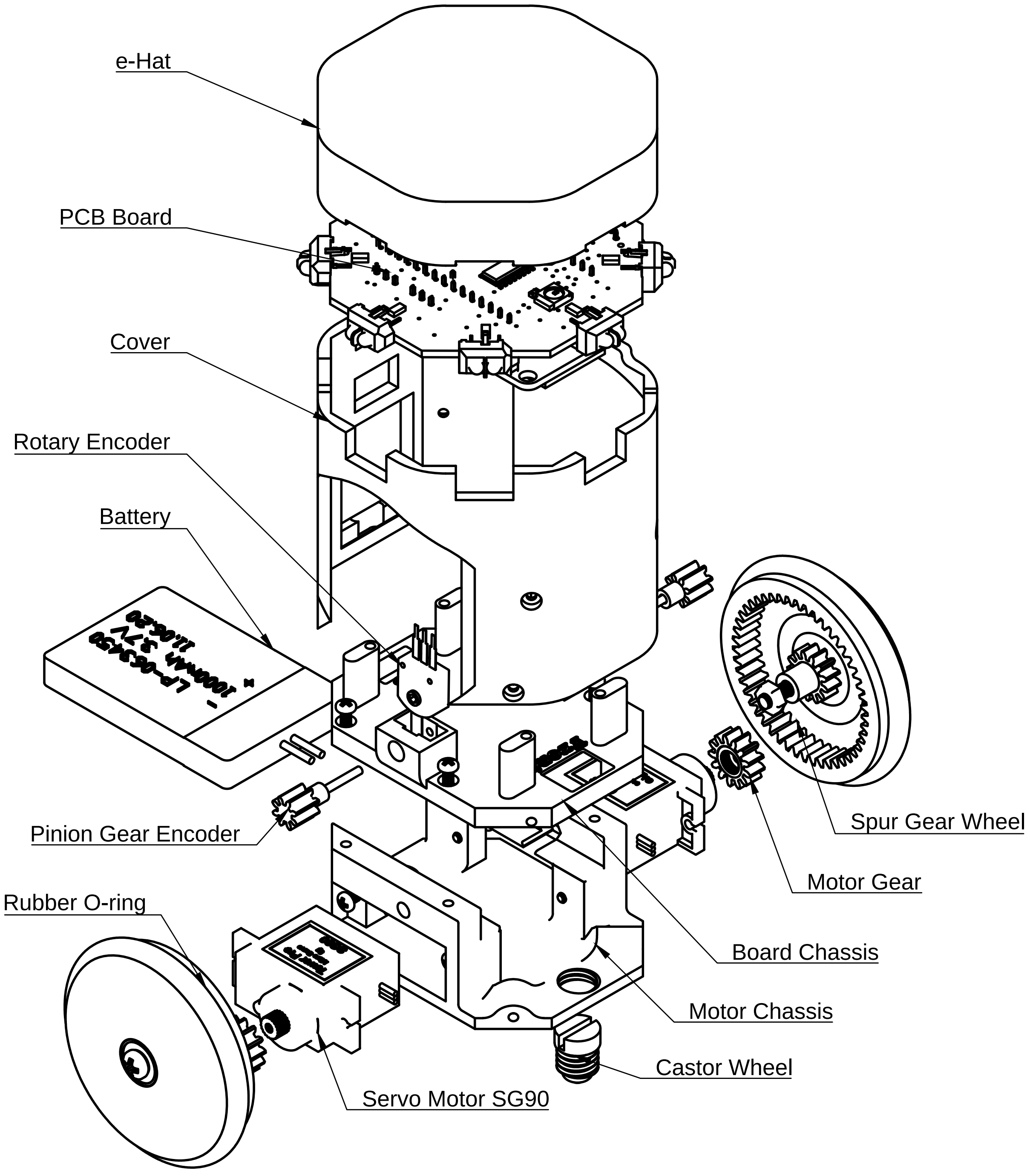


Figure 3.3: An expanded view of the robot's components and body parts.

3.2 Electrical Design

In addition to detailing the robot's mechanical design, we also delve into its electrical design. This process encompasses defining and integrating electronic components within the robot, including the processing unit, sensors, and power management system. Figure 3.4 shows an overview of the electrical components of the robot.

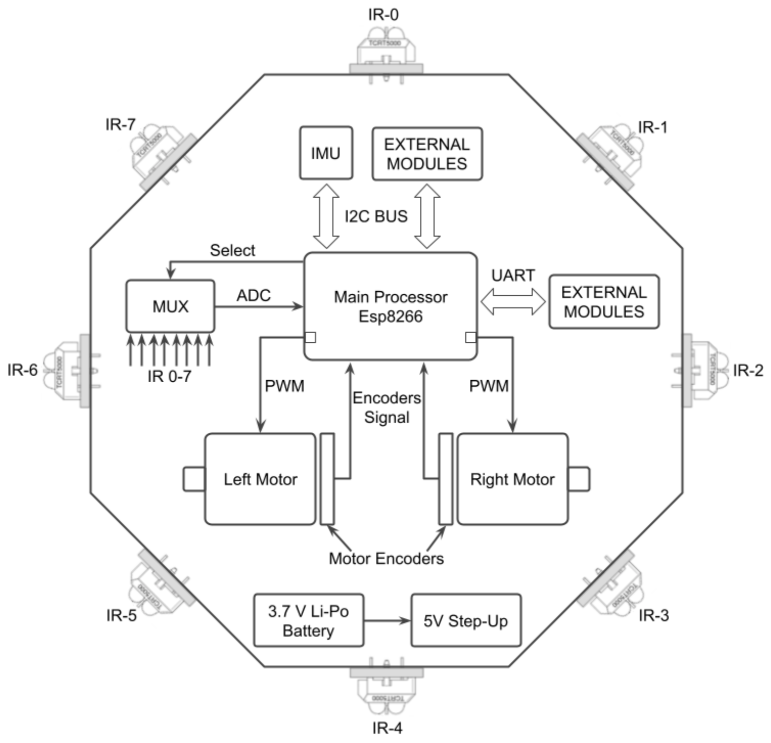


Figure 3.4: An overview of robot electrical components.

3.2.1 Microcontroller

One of the major decisions in the robot's electrical design is the careful selection of an appropriate microcontroller. This component plays a vital role in determining the robot's computational capabilities, as well as the scope of components that can be integrated into the system.

Among the array of microcontroller options, the ATMega series has garnered substantial attention for its application in the development of small, cost-effective robots. These microcontrollers are characterized by their affordability, efficiency, ease of programming, and widespread popularity within Maker communities. However, the limited processing power of this series poses challenges when attempting to incorporate numerous components within the robot. Recent alternatives like the STM32 family and Raspberry Pi Pico boards have emerged, boasting 32-bit ARM Cortex processors that offer substantial power. While these microcontrollers exhibit impressive computational performance within compact dimensions, they are also accompanied by higher price points.

After considering many alternatives, we have chosen the Espressif ESP8266 as the main processing unit for our robot. This microcontroller stands out for its remarkable affordability, coupled with a robust processing capacity of 32-bit at 160 MHz, and plenty of 4 MB of memory. Additionally, the built-in Wi-Fi microchip equips the robot with swift IEEE 802.11 connectivity, complete with a full TCP/IP stack. This feature empowers the robots to establish seamless communication amongst themselves or with remote computers, leveraging robust and scalable protocols. Moreover, the ESP8266 is renowned for its efficiency, programmer-friendly nature, and widespread adoption within Maker communities, fostering a conducive environment for the development of custom modules tailored to the robot's needs. Table 3.2 compares this microcontroller and one of the most popular and cheaper ATMega microcontroller.

Table 3.2: Comparison of evaluated microcontrollers for the robot.

	ESPressif ESP8266	Arduino Nano
MCU	**Tensilica LX106 32-bit @ 160 MHz**	Atmel AtMega328 8-bit @ 16 MHz
RAM	**128 KB**	2 KB
Flash	**4 MB**	32 KB
Wi-Fi	**802.11bgn**	None
GPIO	**17**	14
ADC	**1 (10 bits)**	8 (10 bits)
PWM	**12**	6
BUS	**SPI/I2C/I2S/UART**	SPI/I2C/UART
Price*	**2.50 USD**	2.30 USD

* approximated cost

3.2.2 Sensing

An important prerequisite for small robots deployed in swarm experiments is the ability to measure distances to nearby robots and obstacles. In the case of HeRo, we opted for IR sensors owing to their compact size and cost-effectiveness. To achieve this, we strategically positioned eight IR transmitters and receivers (TCRT5000) around the robot's circumference at $45°$ intervals. This selection stems from its affordability, reasonable resolution, and range.

While the TCRT5000 sensor is not typically employed for longer distances (100 mm), and its documented maximum range[3] is limited to 20 mm, we devised a technique to extend its range without significant compromise its accuracy. By implementing a technique called Pulsed Over-Current Driving LED (Lin and Chen, 2011), we could boost the detection range to up to 200 mm without substantial accuracy loss. In essence, by shortly applying a voltage to the poles of the IR emitter LED, its resistance drops due to low conductor temperature, allowing a higher current flow (up to 3 A for $t < 25$ μs). This results in intensified emission of IR light. The LED's resistance then gradually increases and stabilizes, leading to reduced light intensity (60 mA max) when the LED is continuously activated. Our setup generates pulses lasting 100 μs at a 0.2% duty cycle. This arrangement, with short overcurrent pulse durations and ample cooling-off blanking intervals, facilitates the safe operation of even the most inexpensive and ubiquitous IR LEDs at extreme currents. Although this technique might marginally reduce the IR LED lifespan, it is expected that the LED would still operate over a year.

To control the activation of IR LEDs, a MOSFET component is employed. Given the limited number of Analog-to-Digital Converter (ADC) pins on the microcontroller – featuring just one pin with 10-bit resolution – an 8-channel analog multiplexer is incorporated to enable the microcontroller to read data from all eight IR phototransistors. This setup ensures distance measurements, mitigating the impact of environmental light interference by exclusively utilizing the IR receivers.

[3]TCRT5000: www.vishay.com/docs/83760/tcrt5000.pdf

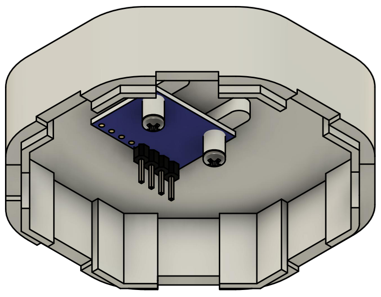
As previously mentioned, the robot encompasses two sets of rotary encoders, which are coupled to the wheels through a transmission mechanism. A rotary encoder is a sensor that generates digital signals in response to motion, thereby providing information about position, velocity, and direction. Leveraging the intrinsic functionality of a typical mouse device wheel as a precise encoder – a cost-effective component costing less than 0.10 USD – we have seamlessly incorporated it as a robotic sensor. This component features a conductive disc and three contacts that produce two quadrature square wave signals as the encoder shaft rotates, enabling the counting of 48 pulses per shaft revolution and enabling the identification of the direction of rotation.

In addition to the encoders, the robot houses two *WS2812b* RGBA LED indicators, which are harnessed for status monitoring and debugging purposes. These addressable LEDs incorporate an integrated circuit that enables communication via a one-wire interface, following a daisy chain topology. This interface allows control of multiple LEDs in series via a single digital pin. The brightness and color of each LED can be precisely manipulated, affording the capability to produce intricate and unique visual effects to communicate various statuses with simplicity.

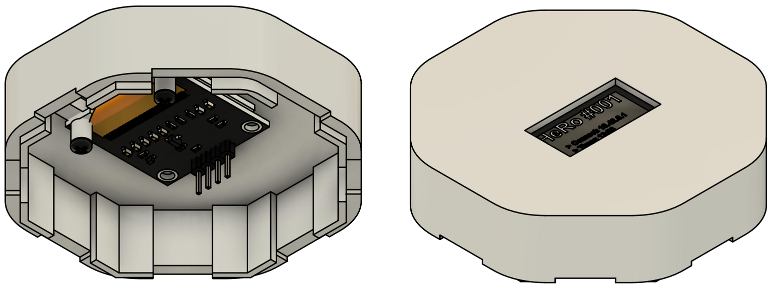
3.2.3 E-Hat

In addition to its built-in capabilities, the robot's functional scope can be expanded through the utilization of e-Hats. These modular add-ons serve as versatile shields, empowering users to tailor specific modules to suit their distinct application requirements. Mechanically, an e-Hat module attaches to a dedicated 4-pin bus located on the robot's top surface, which can be configured to interface through either Inter-Integrated Circuit (I2C) or Universal Asynchronous Receiver/Transmitter (UART) protocols. Alongside communication, this bus also offers a reliable power source, delivering a regulated 5 VDC supply (max 800 mA) to power the module.

In this thesis, we have developed two illustrative e-Hat modules for the purpose of demonstration and experimentation (refer to Figure 3.5). The first module encompasses an IMU e-Hat, equipped with an *MPU6050* sensor housing both a gyroscope and accelerometer. This component's data can be integrated into velocity and position calculations to address potential odometry inaccuracies, such as those arising from wheel slippage. The second module takes the form of a display e-Hat, capable of serving as either a user interface or as a component of a location system reliant on a camera and fiducial markers.



(a) IMU e-Hat.



(b) Display e-Hat.

Figure 3.5: Illustration showcasing two e-Hat versions developed for the HeRo platform.

3.3 Power Supply

Ensuring sufficient power autonomy is crucial for conducting extensive experiments. To achieve this, the HeRo platform is equipped with a 3.7 VDC 1800 mAh Li-Po battery, chosen for its balance between capacity and size. Managing the voltage supply, an *MT3608* DC-DC step-up module regulates the battery's voltage to a steady

5 VDC, suitable for powering the robot's components. This setup enables the robot to sustain up to 3 hours of continuous operation, considering the concurrent use of all on-board components. The motors receive power directly from the step-up module to ensure minimal voltage drop and maintain consistent speed. Moreover, a *TP4056* module is employed to facilitate recharging the battery through a USB cable, enhancing the platform's practicality and ease of maintenance.

3.4 Assembly

Streamlining the assembly process is a primacy for the HeRo platform, given that many of its components are readily available off-the-shelf items. To simplify the setup, we consolidated these components onto a single PCB board, carefully designing it to facilitate easy assembly, even for those less experienced. For added convenience, users can also opt to have the PCB board assembled by various specialized PCB manufacturers, which now offer their services at remarkably reasonable costs. The front and back views of the PCB's design are illustrated in Figure 3.6. A comprehensive tutorial detailing the robot assembly procedure is available on the project's website[4].

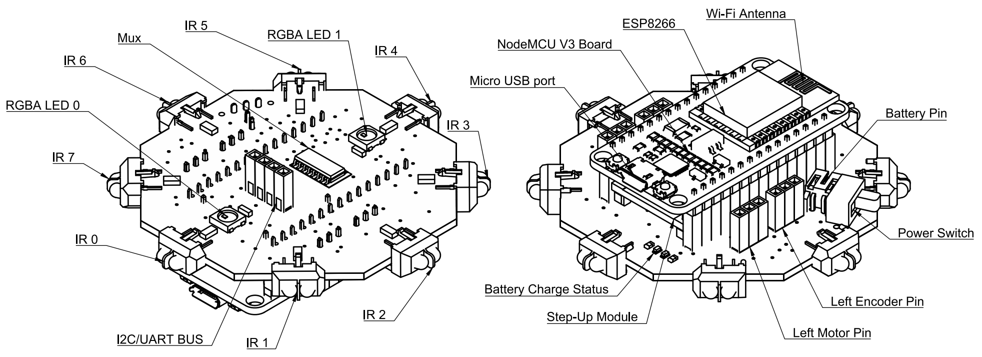


Figure 3.6: Overview of HeRo's PCB board design.

[4]Tutorial: https://verlab.github.io/hero_common.

3.5 Part Costs

With the mechanical and electrical components of the robot and its assembly process established, it becomes possible to estimate its overall cost. A summary of the component costs used in HeRo can be found in Table 3.3. It is important to note that these prices are based on retail purchasing from standard part distributors on the Internet. If parts were to be acquired in bulk directly from manufacturers, we expect that these costs would be significantly reduced. Finally, after completing the robot assemblies, the most suitable result is depicted in Figure 3.7.

Table 3.3: Parts cost per robot unit.

Parts	**Quantity**	**Cost (USD)**
Servo Motors SG-90	2	2.06
Mouse Encoder 48 PPR	2	0.10
ESP8266 Nodemcu	1	2.50
Rubber O Ring 38mm	2	0.10
IR TCRT5000	8	0.68
LED RGB WS2812b	2	0.51
IMU MPU6050	1	0.85
LI-PO Battery 3.7 VDC 1800 mAh	1	5.85
PCB board and Components	1	4.30
3D Printer Parts (PLA) and Fastening	1	1.50
Total		**18.72 USD**

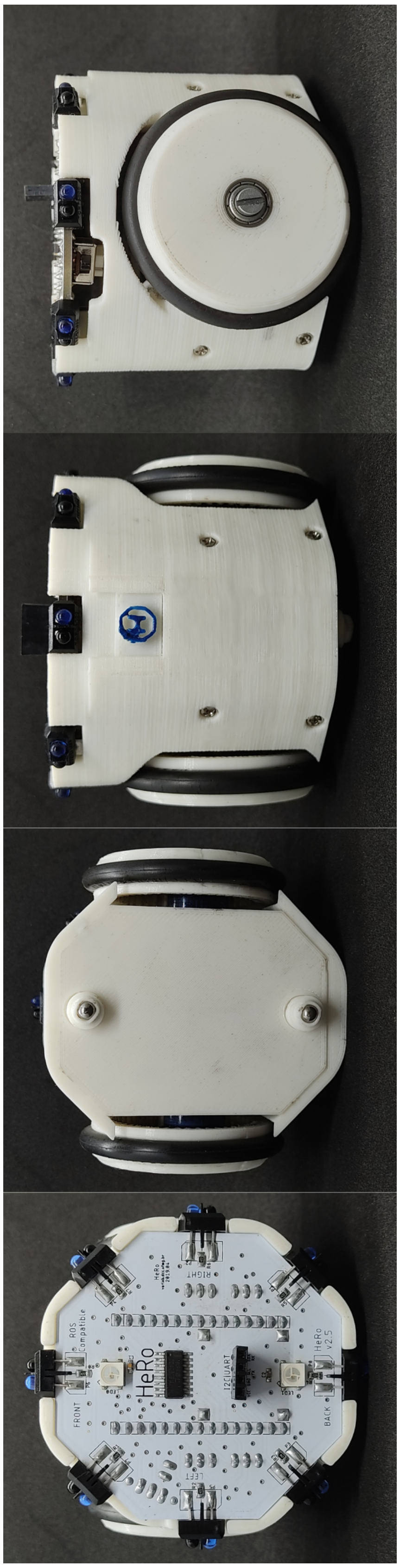


Figure 3.7: Top, bottom, front, and left views of the HeRo swarm platform.

Chapter 4

Software and Communication Architecture

Alongside the mechanical and electrical components of the robot, we delve into a computational framework that enhances its applicability in swarm applications. This chapter details a comprehensive software and communication architecture designed to facilitate the programming of multiple robots. Additionally, we present a simulated environment that proves helpful during the preliminary stages of application development.

4.1 Software Architecture

When dealing with robot swarms, the challenge of efficiently programming multiple robots quickly comes to the forefront. In the context of experimentation, a widely adopted approach involves employing a master-slave architecture. In this setup, the robots (slaves) communicate remotely with a computer (master) that executes the user's application. This method offers efficiency by obviating the need to frequently flash firmware for application changes. However, while convenient, this approach may not faithfully simulate deployment-level effects, such as local communication glitches, constrained processing capabilities, and other potential challenges. Consequently, employing strategies like FOTA becomes pivotal, enabling users to wirelessly load their applications onto the robot and run them directly.

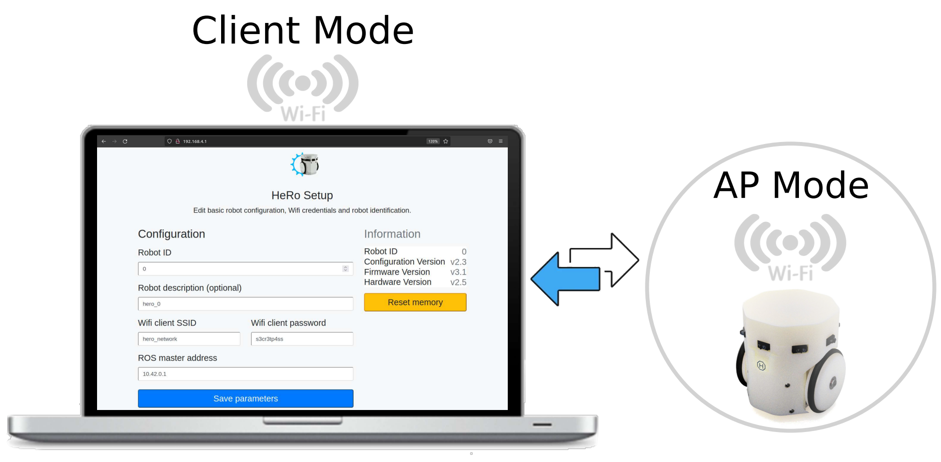
In this thesis, we propose a versatile architecture that accommodates both the aforementioned programming approaches. Our architecture encompasses a firmware compatible with ROS and integrates FOTA functionality. In the ensuing sections, we expound upon the intricacies of this software architecture.

4.1.1 Firmware

The firmware stands as a fundamental component of the robot, serving as the unit that computes procedures like motor control and sensor data processing. For HeRo, we opted to construct the firmware using the Arduino Integrated Development Environment (IDE). This platform, renowned for its user-friendliness and wide usage in Maker communities, renders the firmware more accessible for comprehension and modification. Moreover, it provides diverse libraries that facilitate manipulating microcontroller ports, driving actuators and sensors, and handling TCP sockets.

At its core, the firmware is built upon the rosserial framework, a pivotal choice that renders the robots compatible with ROS. Rosserial encompasses an assortment of tools, including a protocol that wraps conventional ROS serialized messages and multiplexes multiple topics and services over network sockets. This abstraction of communication intricacies engenders a concise and efficient implementation. In practice, users need only configure some communication parameters to enable robot connectivity via TCP/IP networking to a ROS-equipped remote computer. To streamline this setup procedure, avoiding the need for recurrent firmware reprogramming, we presented a remote configuration mode accessible via a web interface (see Figure 4.1a).

Initiating this interface requires only turning on the robot in configuration mode. This mode creates an access point for user connection via computer or smartphone. Accessed through a web browser, the robot's webpage enables the user to customize parameters such as its name, access point credentials, ROSMaster IP address, and port. Once the configuration is established, the robot autonomously connects to the ROS server, allowing the user to exploit its functionalities via topics and services (as depicted in Figure 4.1b). This streamlined configuration process can be executed within minutes and retains its settings even after the robot is powered off.



HeRo Setup

Edit basic robot configuration, Wifi credentials and robot identification.

Configuration

Robot ID: 0

Robot description (optional): hero_0

Wifi client SSID: hero_network

Wifi client password: s3cr3tp4ss

ROS master address: 10.42.0.1

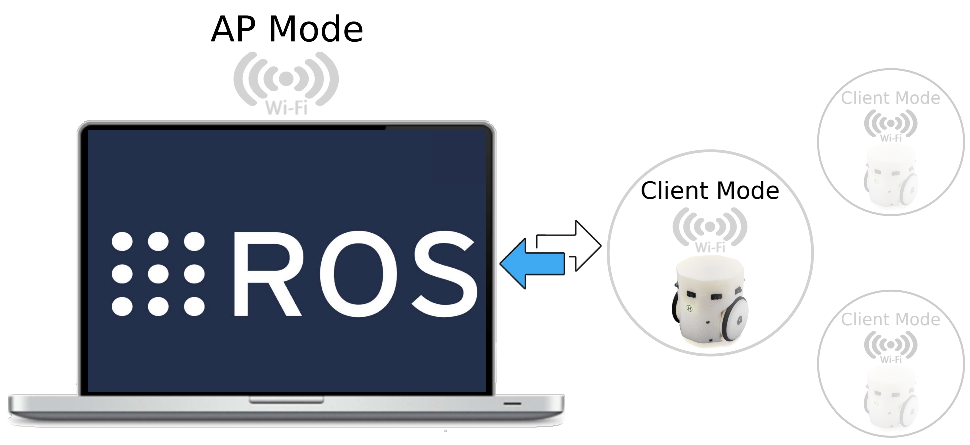
Save parameters

Information

Robot ID	0
Configuration Version	v2.3
Firmware Version	v3.1
Hardware Version	v2.5

Reset memory

(a) Remote configuration mode.



(b) ROS communication mode.

Figure 4.1: Robot firmware modes: (a) remote configuration and (b) ROS communication mode. The first one helps the user configure the robots to connect to a server running ROS without requiring reprogramming the robot. After properly setting up the robot, it connects automatically with ROS server allowing the user to send and receive commands through ROS topics and services.

Beyond providing the master-slave communication architecture, the firmware comprises basic modules for computing kinematic control, odometry, and sensor data. In the subsequent sections, we delve into the details of these firmware modules.

Sensors

The robot is equipped with an array of sensors in its most basic configuration, excluding the e-Hat module. These sensors include eight IR transceivers strategically positioned around the robot and two quadrature mechanical rotary encoders. This ensemble of sensors enables the robot to perceive and interact with its environment effectively.

The eight IR sensors offer a panoramic view of the surroundings, facilitating obstacle detection and distance estimation. These sensors are connected to a 10-bit ADC port on the microcontroller via an 8-channel analog multiplexer. This setup not only allows for estimating distances to obstacles but also provides ambient light measurements once one can control the IR emitter. Formally, obstacle detection and distance estimation rely on fundamental principles of electromagnetic radiation and its reflection. Therefore, the sensor's output, denoted as $s(d, \gamma)$, can be modeled as described by Benet et al. (2002):

$$s(d, \gamma) = \frac{\alpha}{d^2} \cos(\gamma) + \beta, \tag{4.1}$$

where $s(d, \gamma)$ is the sensor's output value, d stands for the distance to the object, and γ denotes the angle of incidence with the surface. The model variable α comprises multiple parameters, including the reflectivity coefficient, the output power of the emitted IR light, and the sensitivity of the sensor, all of which are empirically estimated. On the other hand, β is the offset that accounts for the ambient light effect. Regular measurements are taken to calibrate β after performing the calculations defined in Equation 4.1.

As mentioned, HeRo is equipped with two quadrature encoders, one attached to each wheel. Quadrature encoders are commonly employed to measure the speed and direction of a rotating shaft. These encoder channels are connected to the microcontroller's interrupt pins. With each pulse, an interrupt routine is triggered within the microcontroller, incrementing an independent counter variable that helps estimate the distance traveled by each wheel. The frequency of the pulses is measured to estimate the velocity of each wheel. The output from the encoders serves as input for closed-loop motion control and localization algorithms, enabling the robot to navigate in the environment.

Motion Control

In our previous chapter, we introduced the robot as a two-wheel differential-drive mobile robot comprising two servo motors, each equipped with a quadrature encoder. The robot operates under non-holonomic constraints, meaning it cannot move directly along its wheel axis relative to its body reference. Instead, it achieves direction changes by adjusting its wheels' relative instantaneous speed, eliminating the need for an additional steering motor. An effective method for controlling the robot's motion in a 2D space involves managing its linear and angular speed.

To facilitate this control approach, we can employ classical kinematics modeling for a differential-drive mobile robot (Siegwart et al., 2011). This modeling allows us to calculate the robot's velocity in its own reference frame or in the inertial frame, as illustrated in Figure 4.2. Formally, the instantaneous velocity, as expressed in both the robot body frame and the inertial frame, can be defined as follows:

$$\mathbf{v}^R(t) = \begin{bmatrix} v_x(t) \\ 0 \\ \omega(t) \end{bmatrix}, \tag{4.2}$$

$$\mathbf{v}^I(t) = \begin{bmatrix} v_x(t)\cos(\omega(t)) \\ v_x(t)\sin(\omega(t)) \\ \omega(t) \end{bmatrix}. \tag{4.3}$$

Despite the fact that one may control the robot's velocity in any reference frame, in this thesis, we find controlling it concerning the robot frame more convenient as it can simplify the control problem and make it easier to achieve the desired motion. From now on, we will describe how we control the instantaneous velocity of the robot in its

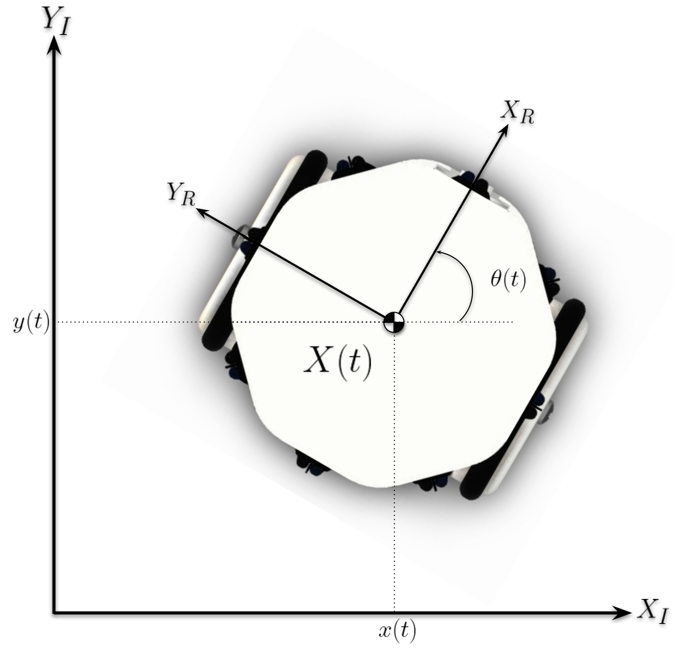


Figure 4.2: Robot represented in an inertial reference frame.

own frame. That is, we want to control the linear speed $v_x(t)$ along the X_R-axis and its angular speed $\omega(t)$ around the Z_R-axis for $\mathbf{v}^R(t)$.

By assuming such velocities as inputs, we must map them into wheel velocities so that we can control the robot. The problem of mapping the relationship between robot velocity and wheel speed is called inverse kinematics. That is, given the linear speed $v_x(t)$ and angular speed $\omega(t)$, we can compute the desired left speed $v_l(t)$ and right speed $v_r(t)$, to produce the specific motion of the robot (see Figure 4.3). The equation below describes the inverse kinematic model concerning the robot reference frame:

$$\begin{bmatrix} v_l(t) \\ v_r(t) \end{bmatrix} = \begin{bmatrix} \frac{2v_x(t)-l\omega(t)}{2} \\ \frac{2v_x(t)+l\omega(t)}{2} \end{bmatrix}, \tag{4.4}$$

where $v_l(t)$ and $v_r(t)$ are the tangential speeds of the left and right wheels; $v_x(t)$ and $\omega(t)$ are the linear and angular speeds of the robot in its own reference frame; and l is the distance between the left and right wheels.

After computing the desired tangential speed on each wheel, we need to control the motors so that they maintain these speeds. The Proportional Integral Derivative (PID) controller is the most common control algorithm used for this application. It can correct the present error through proportional action, eliminate steady state offsets

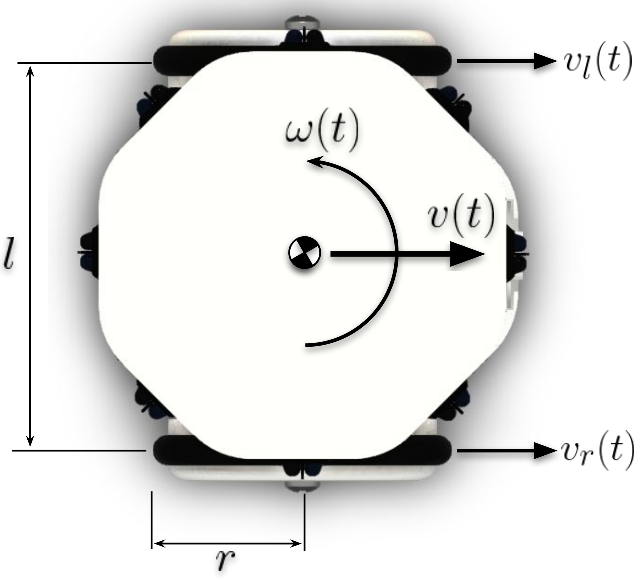


Figure 4.3: Robot local reference frame.

through integral action, and better estimate future trends through a derivative action. The mathematical model of a PID is defined by

$$u(t) = K_p e(t) + K_i \int_0^t e(t)\,dt + K_d \frac{de(t)}{dt}, \tag{4.5}$$

where $u(t)$ is the control signal to each motor, that is, PWM signals; dt is the control loop interval time; $e(t)$ is the error regarding the desired and current tangential speeds of each wheel; and K_p, K_i and K_d, all non-negative, denote the coefficients for the proportional, integral, and derivative terms, respectively.

To estimate the error $e(t)$, we subtract the desired tangential speed from the current tangential speed estimated by the encoders. That is, we count how far each wheel has turned and compute the rate for a loop interval. Formally, the current tangential speed for both wheels is computed by

$$\bar{v}_l(t) = \frac{\Delta s_l}{dt}, \tag{4.6}$$

$$\bar{v}_r(t) = \frac{\Delta s_r}{dt}, \tag{4.7}$$

where Δs_l and Δs_r are the distance each wheel has traveled for a time interval dt, respectively.

Moreover, we use a simple Kalman filter to reduce the noise of the reading and improve the quality of both estimated speeds. Formally, we compute the following process for each wheel measurement:

$$K = \frac{\sigma_e^-}{\sigma_e^- + \sigma_m}, \tag{4.8}$$

$$\hat{v} = \hat{v}^- + K(\bar{v} - \hat{v}^-), \tag{4.9}$$

$$\sigma_e = (1 - K)\sigma_e^- + |\hat{v}^- - \hat{v}|q, \tag{4.10}$$

where σ_e is the estimation uncertainty adjusted by the filter; σ_m is the measurement uncertainty, that is, how much we expect the estimated speed can vary; K is called Kalman gain; $\bar{v}$ is the current measured speeds, *i.e.*, $\bar{v}_l(t)$ and $\bar{v}_r(t)$; $\hat{v}$ is the filtered speed; q is the process variance, that is, how fast the measurement moves; and finally, the superscript $(^-)$ indicates previous values of a variable.

To conclude, we can summarize the robot velocity control with a block diagram, which is illustrated in Figure 4.4.

Localization

Odometry stands as the most used method for determining the position of a mobile robot relative to an inertial reference frame, as depicted in Figure 4.2. In practical applications, odometry offers readily available real-time positioning information between periodic absolute position measurements. Various types of sensors are commonly used for odometry, and this thesis focuses on utilizing encoders placed on each wheel to track their rotation. By monitoring the wheel rotations, we can estimate how far the robot has moved forward and its current position.

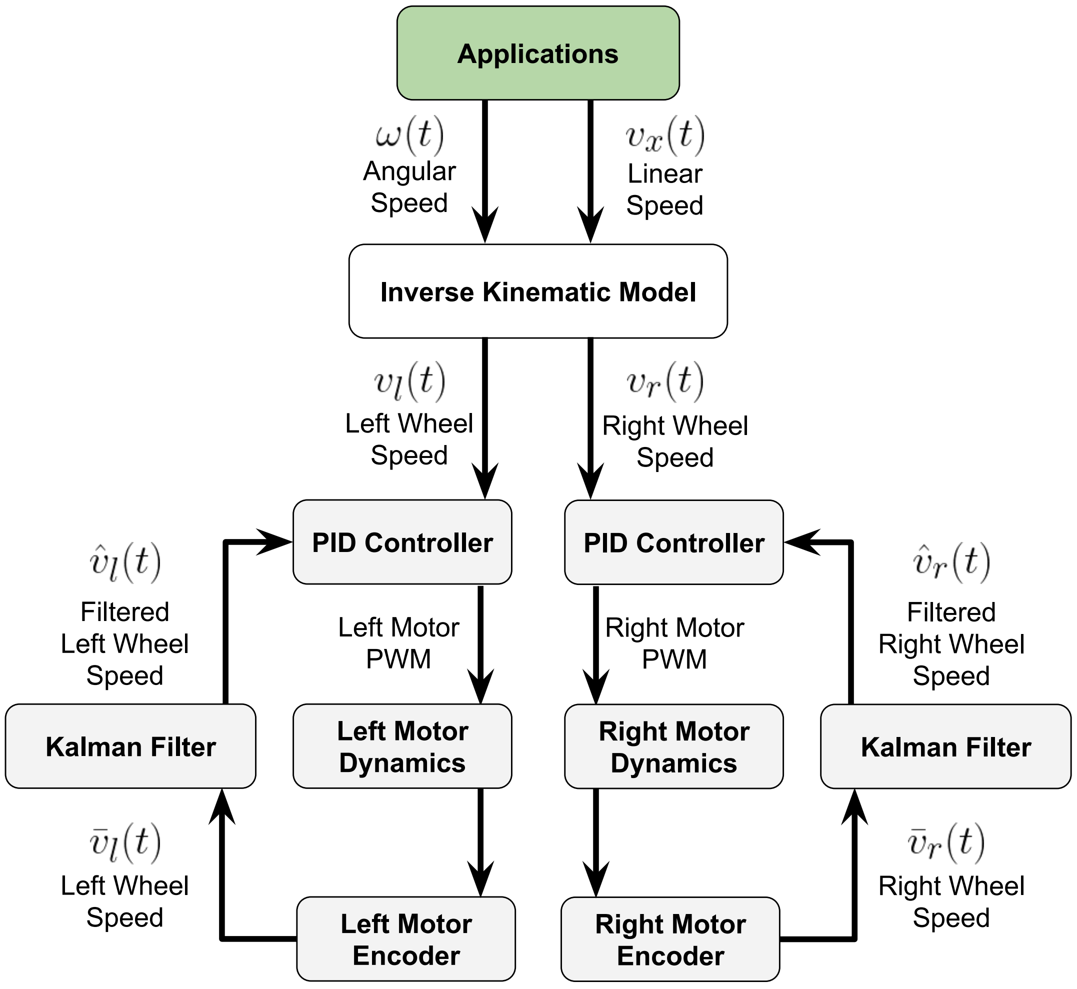


Figure 4.4: Diagram illustrating the velocity control of a differential robot.

The distance traveled by the robot can be calculated as the average of the distances each wheel has turned, as described in Equation 4.11. On the other hand, the robot's heading can be estimated (assuming minimal wheel slip) from the difference in these displacements relative to the distance between the wheels, as shown in Equation 4.12.

$$\Delta L = \frac{r(\Delta s_r + \Delta s_l)}{2}, \tag{4.11}$$

$$\Delta \theta = \frac{r(\Delta s_r - \Delta s_l)}{l}, \tag{4.12}$$

where Δs_r and Δs_l represent how much each encoder has turned in the loop time interval; r is the wheel radius; and l represents the distance between the wheels of the robot, as shown in Figure 4.3.

Once we have computed how far the robot has traveled and turned, we can integrate this information to estimate its current pose regarding the inertial reference frame. Considering the pose of the robot at time t in a plane is given by the state vector

$$\mathbf{X}(t) = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix}, \tag{4.13}$$

the pose of the robot after a loop time interval dt is given by

$$\mathbf{X}(t+dt) = \mathbf{X}(t) + \begin{bmatrix} \frac{\Delta L}{\Delta\theta}(\sin(\theta(t)+\Delta\theta) - \sin(\Delta\theta) \\ \frac{\Delta L}{\Delta\theta}(\cos(\theta(t)+\Delta\theta) - \cos(\Delta\theta) \\ \Delta\theta \end{bmatrix}, \tag{4.14}$$

where $\Delta\theta$ is the variation of the robot orientation in Z_I-axis in a time interval, that is, $\Delta\theta = \theta(t+dt) - \theta(t)$.

Note that if the robot moves in a straight line, the change in angle $\Delta\theta$ is zero, and the odometry model (4.14) becomes undefined since $\frac{\Delta L}{\Delta\theta}$ is undefined. In this case, a different model should be used to compute the odometry, such as using the change in distance ΔL. Thus, in order to approach this special case, we test this condition, and if it occurs, the following model is computed for the odometry,

$$\mathbf{X}(t+dt) = \mathbf{X}(t) + \begin{bmatrix} \Delta L\cos(\theta(t)) \\ \Delta L\sin(\theta(t)) \\ \theta(t) \end{bmatrix}. \tag{4.15}$$

4.2 Communication Architecture

To establish seamless communication between workstations and robots, we have implemented HeRo as a ROS-compatible robot connected through TCP/IP.

The ROS (Quigley et al., 2009) is an open-source meta-operating system tailored for robotics applications. It offers a wide array of services akin to a typical operating system, including hardware abstraction, low-level device control, common functionality implementation, inter-process message-passing, and package management. Moreover, ROS provides a suite of tools and libraries for acquiring, building, writing, and executing code across diverse platforms.

Communication in this framework follows a publish-and-subscribe model, where topics, consisting of predefined message structures, facilitate interaction among multiple nodes (processes) in the network. These topics, including odometry, are open to access by any node within the network, ensuring seamless scalability for both publishers and subscribers. This approach enables robots to communicate with one another in a well-defined and flexible manner.

However, it's worth noting that many swarm robots, including HeRo, face constraints regarding computational resources, making it challenging to run a full-fledged native ROS instance due to their limited processing capabilities. To address this limitation and enable the integration of ROS functionalities into less powerful microcontrollers without the necessity for a complete ROS installation, we have implemented the communication module using the rosserial protocol. This protocol has demonstrated reliability and scalability, proving its effectiveness in swarm systems (West et al., 2018). Rosserial[1] serves as the conduit for encapsulating standard ROS serialized messages and multiplexing multiple topics and services over a network socket.

In essence, the rosserial nodes play a pivotal role in the communication process. They transform data from the standard structured XMLRPC protocol, which is conventionally managed by TCP within ROS, into serialized data that can be transmitted to the microcontroller. Moreover, these nodes perform the reverse operation by deserializing data received from the microcontroller, and reassembling it into the appropriate message structures, thus ensuring seamless integration with the standard ROS network.

While the robot remains compatible with ROS 1 using the rosserial framework,

[1]Rosserial: `http://wiki.ros.org/rosserial`.

achieving full compatibility with ROS 2 has proven more challenging. One obstacle is the absence of a ported rosserial framework for ROS 2, and microROS[2], a ROS 2 alternative to rosserial, lacks support for the microcontroller utilized by the robot (ESP8266). To provide an alternative means for ROS 2 users to interface with the robot, we offer a containerized environment using Docker[3]. This allows ROS 2 users to utilize packages like ROSBridge to bridge the gap between ROS 2 and the ROS 1 package. Figure 4.5 presents an overview of the communication architecture.

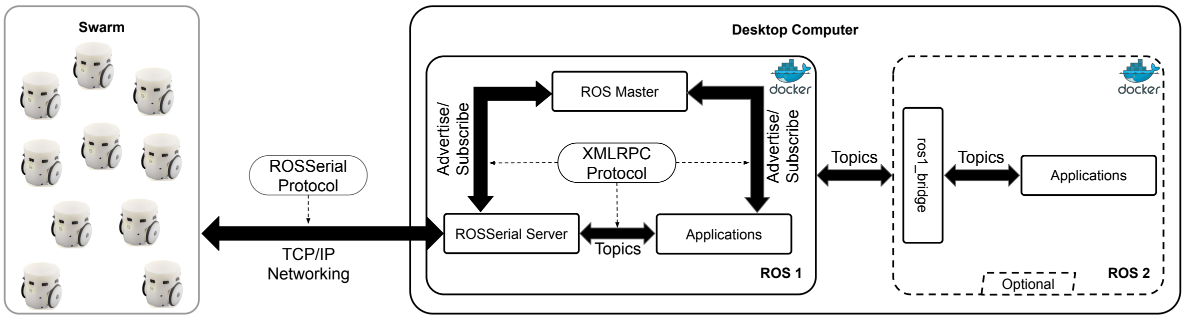


Figure 4.5: An overview of the communication process. The robot's microcontroller acts as a bridge to the sensors and actuators and then rosserial acts as another bridge from the microcontroller to ROS. ROS 2 users can optionally instantiate a ROS 1 bridge interface and interact with the robots. The system infrastructure is organized in Docker containers, which promotes its installation and use.

In theory, network bandwidth imposes limits on the number of connected robots, with more robots necessitating additional connections and consuming network capacity. However, we have not observed any significant overhead when communicating with multiple robots, even when using a consumer-grade wireless network router. Typically, a network can handle 254 devices, but network techniques such as subnets allow for expanding this limit as needed. A comprehensive study demonstrating the reliability and scalability of this protocol for swarm robots is available in West et al. (2018).

[2]microROS: https://micro.ros.org.
[3]Docker: https://www.docker.com.

4.3 Simulation and Visualization

Simulations are indispensable in robotics research, serving as valuable tools for the swift and efficient testing of novel concepts, strategies, and algorithms. Additionally, effective visualization tools are crucial during experiments to provide enhanced monitoring and observation of robot execution. In alignment with this, we have developed a simulation model for HeRo, designed to seamlessly integrate with Gazebo and RViz.

4.3.1 Gazebo Simulator

For simulating our robots within the ROS environment, we opted for Gazebo, a highly integrated choice for ROS users. Gazebo (Koenig and Howard, 2004) is a versatile multi-robot simulator designed for both indoor and outdoor environments. It excels at simulating a multitude of robots, sensors, and objects in intricate three-dimensional spaces. In the ROS-Gazebo integration, the 3D model of a robot or its components is represented through XML files, referred to as Unified Robot Description Format (URDF). These URDF files detail the robot's structures, encompassing its parts, joints, dimensions, texture, and other pertinent attributes.

Once the robot's description is encapsulated in the URDF file, creating a simulated model is a straightforward process. Typically, this involves using the built-in plugins provided by Gazebo. However, this approach can become inefficient when simulating multiple robots and may demand significant computational resources. To address this challenge and optimize the computational load, we have devised a compact plugin that consolidates all of the robot's functionalities. This optimization allows us to maximize processing efficiency for each simulated robot without straining Gazebo's physics engine. Figure 4.6 shows multiple instances of HeRo being simulated within the Gazebo environment.

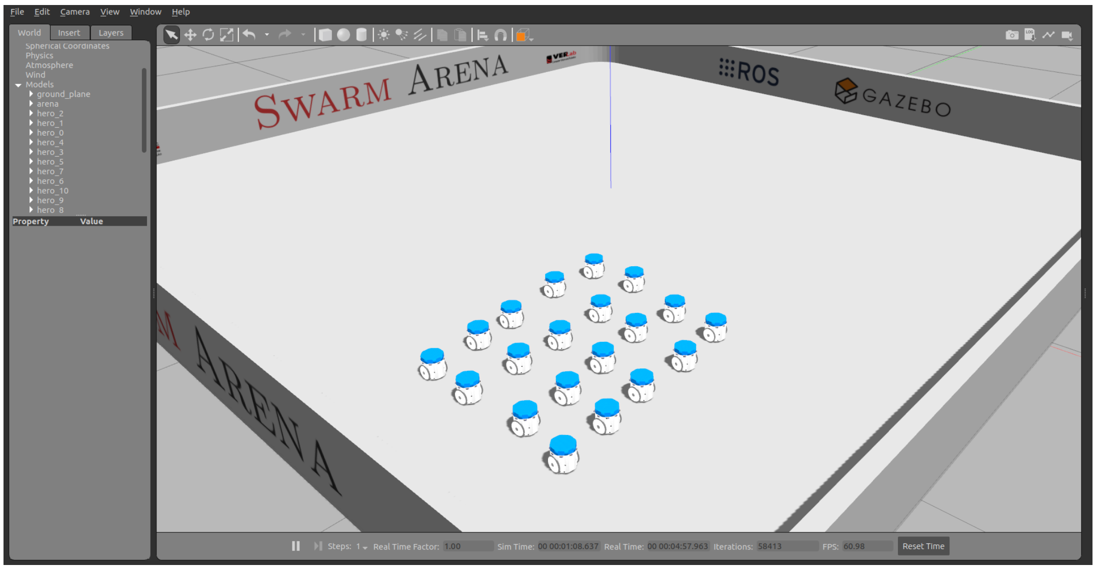
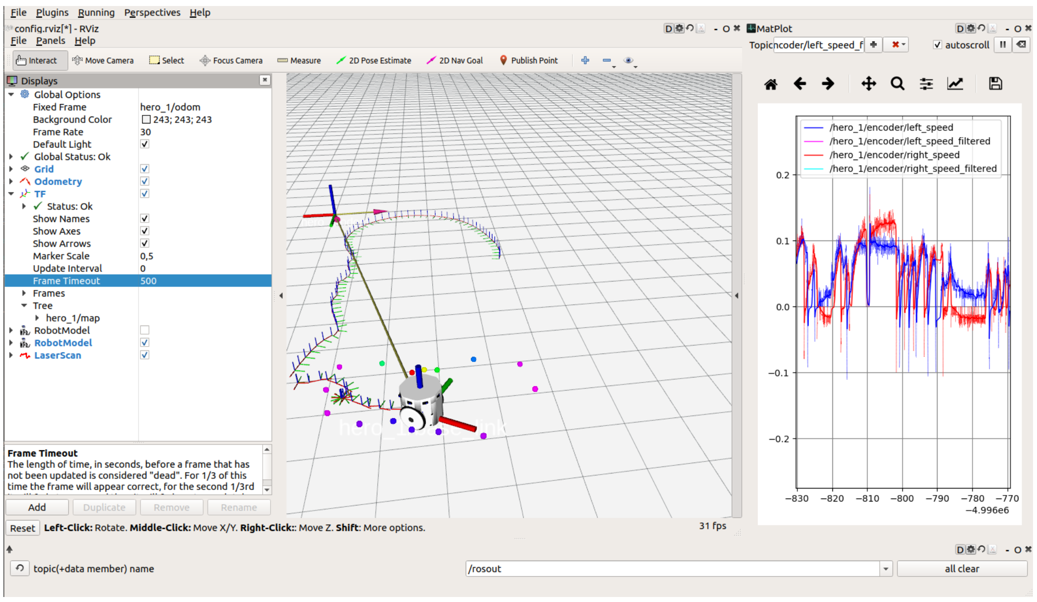


Figure 4.6: Multiple instances of HeRo being simulated in the Gazebo simulator.

4.3.2 Robot Visualization Tool

In addition to simulation, having a robust robot visualization tool is crucial for monitoring the state of sensors and actuators during experiments. Within ROS, we leverage the RViz visualization tool for this purpose. RViz offers comprehensive 3D visualization of the robot by loading the URDF file and can project sensor data obtained through ROS topics like odometry, laser scans, and IMU readings using various plugins. It is important to note that RViz is a visualization tool and not a simulator. Therefore, the robot visualized in RViz can represent either a real robot or a simulated one, depending on the source of the information.

Figure 4.7 provides an example of visualizing a real robot in RViz. In the image, you can observe the 3D model of the robot superimposed onto a colored axis, indicating the robot's pose concerning an initial frame (the colored axis in the background of the scene). The sequence of small axes illustrates the temporal trajectory of the robot, computed through odometry data. The colored spheres surrounding the robot move closer or further from the robot, representing distance sensor readings. On the right side of the image, you can track the linear velocity of each of the robot's wheels. RViz serves as a powerful tool for fully monitoring and analyzing a robot's behavior during experiments.



Global Options	
Fixed Frame	hero_1/odom
Background Color	243; 243; 243
Frame Rate	30
Default Light	✓
✓ Global Status: Ok	
Grid	✓
Odometry	✓
TF	✓
✓ Status: Ok	
Show Names	✓
Show Axes	✓
Show Arrows	✓
Marker Scale	0,5
Update Interval	0
Frame Timeout	500
Frames	
Tree	
hero_1/map	
RobotModel	
RobotModel	✓
LaserScan	✓

Frame Timeout
The length of time, in seconds, before a frame that has not been updated is considered "dead". For 1/3 of this time the frame will appear correct, for the second 1/3rd

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift:** More options.

topic(+data member) name

Figure 4.7: RViz showing a single HeRo robot. RViz is a 3D visualization tool for ROS, allowing control and observation of the current state of the robot.

4.4 Programming

Our communication architecture provides two distinct programming modes for the robot, each with its own set of advantages and considerations.

In the first mode, we have the flexibility to program and execute applications on a centralized server, which then communicates with and controls each robot in a decentralized manner. In this setup, every algorithm operates within its dedicated process on the server and establishes Wi-Fi connections with their respective robots. This approach offers remarkable convenience and scalability, making it particularly advantageous in the initial stages of experimentation involving multiple robots. Leveraging the ROS framework enhances this mode, granting us access to a diverse array of tools and the flexibility to work with various programming languages, typically Python and C++.

However, there are scenarios where running algorithms remotely may not align with the requirements of robot swarm applications. In such cases, it becomes essential for the algorithm to execute directly on each robot, facilitating simultaneous programming for multiple units. In this programming mode, we harness FOTA technology to upload firmware to multiple robots via Wi-Fi. This process entails using the Arduino IDE to develop and compile the application and subsequently transmitting the binary code to the robots through the command line. While this approach offers convenience in terms of simultaneous programming, it does have certain limitations in the availability of high-level tools. Additionally, it necessitates using a programming language compatible with the microcontroller, typically C/C++. The choice between these programming modes depends on the specific requirements and constraints of the robotic application at hand.

Chapter 5

Performance Evaluation

This chapter encompasses a series of experiments designed to assess the capabilities of our robot as an adept swarm robot. Our evaluation begins by analyzing motion control and comparing the robot's odometry with that of the *E-puck*, a widely recognized commercial swarm robot. Furthermore, we delve into the performance and scalability of communication aspects when utilizing ROS, conducting a thorough analysis. Additionally, we investigate the robot's energy consumption across various application demands.

5.1 Motion Control Analysis

In this experiment, we conducted an evaluation of the robot's velocity control system, which aims to ensure that the robot achieves a desired velocity within its own reference frame by regulating the speeds of its wheels. As previously outlined, the wheel speed control relies on a PID controller. Feedback information is derived from the current wheel speed, estimated through encoder readings and filtered using the Kalman filter. The experiment utilized empirically determined parameters for both methods: $K_p = 1200$, $K_i = 2300$, and $K_d = 0.1$ for the PID controller, along with sensor noise of 0.02 m/s (measurement uncertainty) and a process variance of 0.2 for the Kalman filter.

To assess the controller's performance, we examined its response time and residual ripple. The experiment involved initiating the robot from a stationary position and setting a desired linear speed of $v_x = 0.0$ m/s and angular speed of $\omega = 3.17$ rad/s, causing the robot to perform a stationary turn for 4.5 seconds before coming to a stop. Figure 5.1 illustrates the results of this performance analysis.

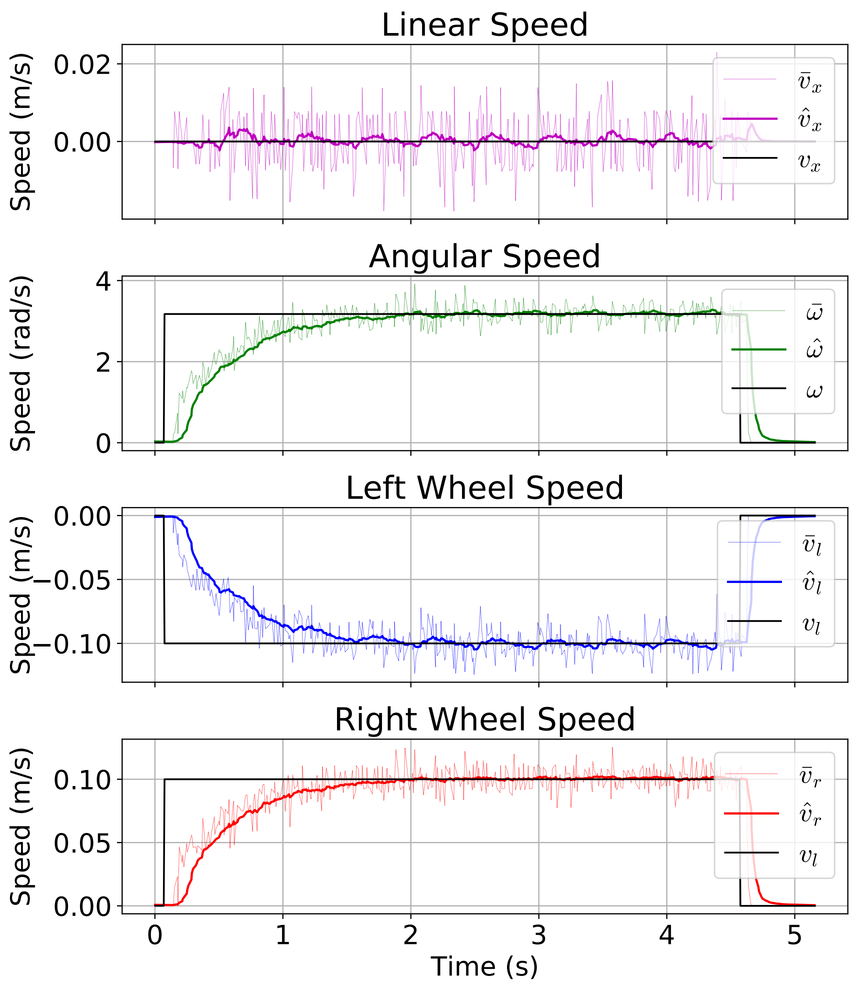


Figure 5.1: Analysis of the robot velocity control showing the accuracy in reaching the desired linear v_x and angular ω speeds concerning the robot reference frame. To reach such motion, the robot computes the desired speeds on each wheel, v_l, and v_r, and then uses a PID controller to control the motors. The current wheel speeds are computed from the encoder's readings, $\bar{v}_l$ and $\bar{v}_r$, and filtered, $\hat{v}_l$ and $\hat{v}_r$, to reduce noise.

As expected, both the left and right wheels exhibit similar response times, reaching the desired tangential speeds ($v_l = -0.1$ m/s and $v_r = 0.1$ m/s) in approximately 1.5 seconds. We deliberately opted for a more conservative controller without overshoot to prevent abrupt movements that could hinder the robot's controllability. Following the achievement of the desired speeds for both wheels, we measured a mean absolute error of 0.33 ± 2.3 mm/s for the left wheel and 0.10 ± 1.4 mm/s for the right wheel. These results are noteworthy, especially considering the utilization of low-cost components in the robot. Moreover, by individually controlling the speed of each wheel, the robot successfully reached the desired linear and angular speeds, exhibiting a mean absolute error of 0.22 ± 1.0 mm/s for linear speed and 0.00358 ± 0.0530 rad/s for angular speed. These results demonstrate the effectiveness of the robot's motion control system.

5.2 Localization Analysis

In this experiment, we evaluated our robot's odometry and compared the results with those obtained by the E-puck (Mondada et al., 2009). To facilitate a comprehensive analysis of the capabilities of both robots, we implemented the same odometry model and employed an identical experimental setup.

This comparison is particularly intriguing because the E-puck utilizes relatively expensive stepper motors, whereas our robot is equipped with inexpensive servo motors. The E-puck computes its odometry by counting the steps commanded to each motor, achieving a maximum resolution of 1024 steps per wheel revolution – exceeding the resolution provided by our encoders (288 steps per revolution). However, the E-puck lacks feedback when it comes to motor skipping steps, which can potentially lead to the generation of false-positive counts.

To assess the accuracy of pose estimation for both robots, we employed the OptiTrack tracking system[1] as a ground truth reference. The trajectory executed by both robots comprised a rectangular shape measuring 1.3×1.1 meters, delineated by four corner points. Each robot was tasked with sequentially navigating to these four points until completing three loops. Both robots covered equal distances while maintaining consistent velocities to ensure the reliability of the comparison.

Additionally, we enhanced one of the HeRo robots with an IMU e-Hat, combining inertial sensors to improve orientation estimation and, consequently, odometry. This IMU

[1]OptiTrack: http://optitrack.com/.

module incorporates a gyroscope and an accelerometer, featuring a built-in MPU (motion processing unit) that combines data from both sensors to generate orientation estimates. In this particular case, while there was still a certain degree of orientation estimation drift present in the IMU data, it was notably smaller than the drift observed in the odometry data. Consequently, we substituted the orientation from odometry with that provided by the IMU. Figure 5.2 displays the trajectories executed by (a) an E-puck, (b) a HeRo without the e-Hat module, and (c) a HeRo equipped with the IMU e-Hat incorporating a gyroscope and accelerometer. A video of this experiment can be viewed on YouTube[2].

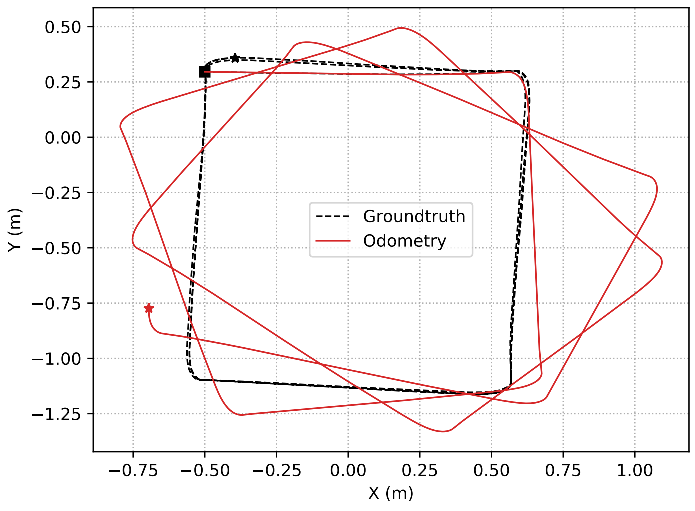
As evident from the results, HeRo's odometry is comparable to that of the E-puck. Considering that the E-puck is widely recognized as one of the most robust and frequently used robots for swarm experimentation, this suggests that our robot, HeRo, presents an attractive solution. Notably, the components used in HeRo are highly cost-effective when contrasted with those of the E-puck. Furthermore, the inclusion of the module with inertial sensors significantly improved the robot's orientation estimation, rendering localization more robust and expanding its potential applications.

5.3 Distance Sensor

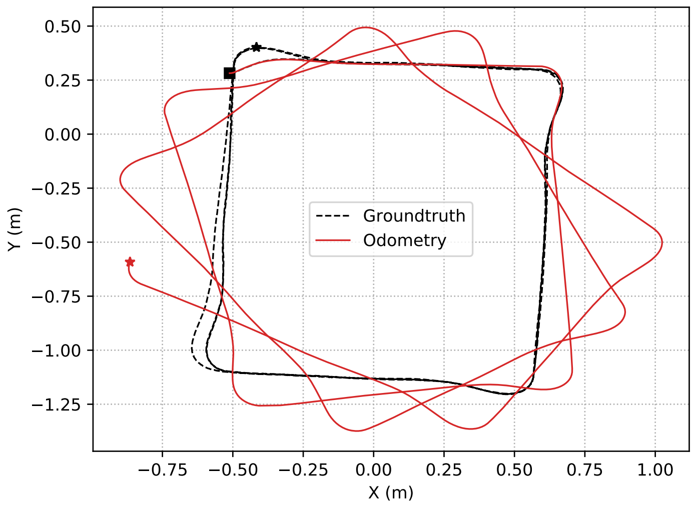
In this experiment, our objective is to evaluate the performance of the IR sensor concerning its capability to estimate distances to white obstacles. However, before embarking on the assessment of the sensor's performance, it is crucial to first characterize the sensor by establishing a relationship between its signal outputs and the corresponding distance measurements.

To achieve this, we initiated the characterization process by obtaining sensor readings using a 10-bit ADC input. These readings were acquired at various object distances, spanning from a close proximity of 0 cm to a maximum distance of 40 cm, with intervals

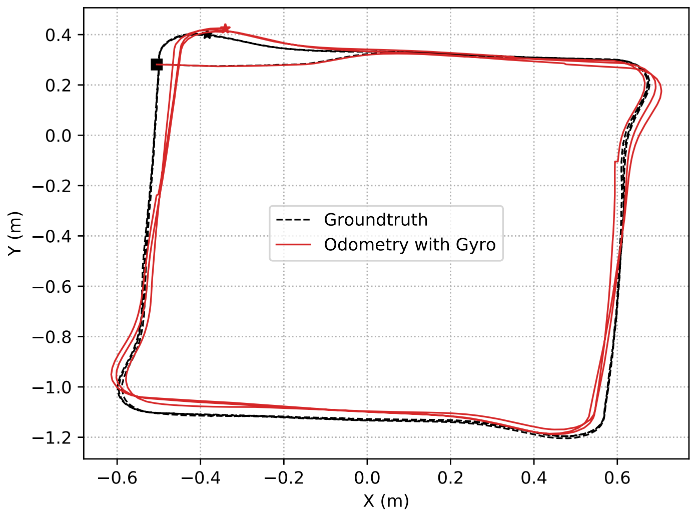
[2]Odometry Comparison: `https://youtu.be/9s6Fg20u0pc`.



(a) E-puck odometry.



(b) HeRo odometry.



(c) Enhanced HeRo odometry with IMU.

Figure 5.2: Comparison of trajectories executed by (a) E-puck, (b) HeRo, and (c) HeRo equipped with e-Hat and inertial sensors.

of one centimeter between each measurement point. An important aspect of this characterization process involved mitigating the influence of ambient light interferences. To accomplish this, we conducted two sets of readings for each measurement point: the first set with the IR emitter deactivated, and the second set with the IR emitter activated. The disparity between these two sets of readings yielded a more robust measurement of the effective light intensity reflected by the obstacle.

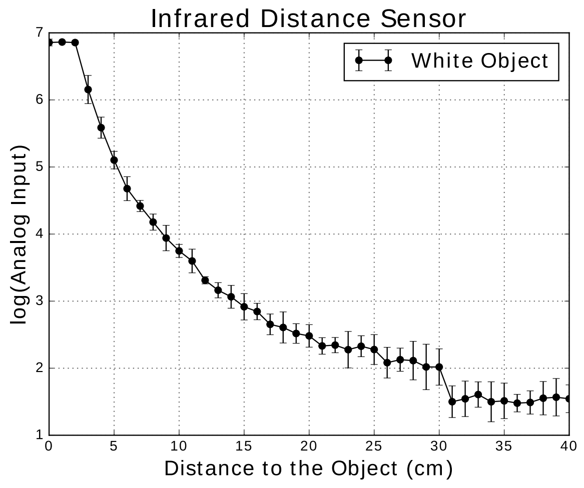
Figure 5.3a provides a graphical representation of these measurements, with the y-axis displayed on a logarithmic scale for clarity. It becomes evident from the results that the IR sensor is capable of detecting objects within a range of up to 30 centimeters. However, to enhance the accuracy of distance estimation, we made the deliberate choice to limit the effective range for our measurements to a maximum distance of 20 centimeters.

Subsequent to the collection of these characterization measurements, we proceeded to perform the calibration of the distance sensor by applying Equation 4.1. The calibrated distance estimates, which represent the sensor's ability to convert digital signals into precise distance measurements, are presented in Figure 5.3b.

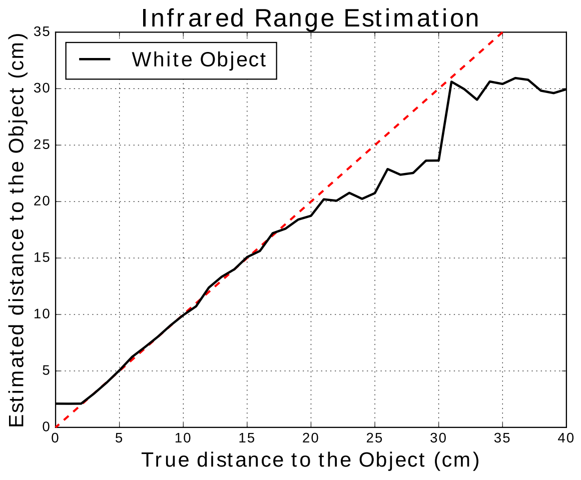
5.4 Communication

Effective communication mechanisms are paramount for swarm robots, especially when dealing with a large number of robots. In this experiment, we focus on evaluating the scalability of communication with regard to bandwidth, which represents the maximum data throughput a communication channel can handle. We specifically assess this aspect in the context of programming the robots using ROS. To gauge the network's capacity to support numerous robots, we calculate the total bandwidth consumed by a single robot and determine the maximum bandwidth that the network can accommodate.

Table 5.1 provides an overview of the measured bandwidth for each communication topic (communication channel) between the robot and a server running ROS. It is



(a)



(b)

Figure 5.3: Analysis of the infrared distance sensor. (a) Shows the readings obtained from a single IR sensor as a function of the distance to the white target; and (b) shows the estimated distance after calibrating the sensor for a maximum range of 20 cm.

important to note that we assume these topics publish or subscribe to messages at predefined frequencies, aligning with the default processing rate of HeRo. To obtain these measurements, we utilized the *rostopic tool*, which offers insights into the packet size of an individual message. This calculation factors in an overhead of 20 bytes for the TCP packet (relevant for the Wi-Fi data connection) and an additional 8 bytes for *rosserial* serialization. The message data size varies depending on the specific topic type. Furthermore, the *rostopic tool* provides the actual bandwidth for each topic, which allows us to determine the total bandwidth consumption by a single robot, amounting to 44 KBps.

Considering that the Wi-Fi module employed in our robot is designed to handle

Table 5.1: Maximum amount of data that can travel through a ROS topic. These topics are operating at different frequencies, set as the default rate of HeRo processing.

ROS Topics	Frequency (Hz)	Packets Size (KB)	Bandwidth (KBps)
/imu	30	0.320	8.60
/laser	20	0.130	3.20
/odom	30	0.730	18.55
/encoder	30	0.100	3.15
/led	2	0.016	0.321
/cmd_vel	20	0.048	0.967
/tf	30	0.068	8.542
Total			**43.33**

at least 1 MBps according to its datasheet[3], we are currently utilizing a mere 4.2% of its maximum capacity. Moreover, the robot connects to a consumer-grade wireless network router in infrastructure mode, which supplies a maximum bandwidth of 150 Mbps (or 18 MBps). Given that one HeRo robot consumes only 44 KBps for communication, we can theoretically **support approximately** 420 **robots** within this network. It's important to note that typical Dynamic Host Configuration Protocol (DHCP) configurations may not be sufficient to address all these IP addresses. However, alternative approaches, such as subnetworking or the use of multiple routers, can be implemented to circumvent this limitation.

5.5 Power Consumption

Power autonomy is crucial when assessing the robot's operational capacity. In this experiment, we delve into the power consumption of the robot's various components to determine its power autonomy. By measuring the current (in mA) consumed by the robot in three typical scenarios, we gain insights into its energy requirements.

More specifically, the three situations under investigation are as follows:

[3]Datasheet: www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf.

1. **Active Sensors and Communication:** In this scenario, we examine the impact of the communication frequency on the robot's power consumption. To this end, we measure the current for three different communication frequency rates: 5 Hz, 20 Hz, and 40 Hz. Remarkably, the results indicate that these frequency variations have a minimal influence on power consumption.

2. **Indicator LEDs Turned On:** While maintaining a communication frequency of 20 Hz, we activate the two indicator LEDs and adjust their white light intensity from half to full. Notably, we observe a significant effect on power consumption, with the LEDs at full intensity drawing nearly 50 mA.

3. **Motors Active:** In this configuration, with a communication frequency rate of 20 Hz, we power up the two motors. To analyze the impact of the robot's velocity on power consumption, we measure the current for two different velocities. As anticipated, the motors emerge as the most power-hungry components, and their velocity directly influences power consumption.

Table 5.2 shows the consumption (in mA) of the robot for these combinations. For a complete assessment of power consumption, we consider a typical usage scenario encompassing the following parameters:

- Sensors and communication operate at 20 Hz.
- Indicator LEDs emit light at half intensity.
- Motors function at a speed of 10 cm/s.

Under these conditions, the robot's power consumption averages 550 mA with a slight deviation of 25 mA. Equipped with a 3.7 VDC 1800 mAh Li-Po battery, the robot is estimated to have a minimum power autonomy of 3 hours and a maximum of 9 hours. This scenario may provide crucial insights into the robot's endurance and suitability for various applications.

Table 5.2: Power consumption of HeRo considering a voltage of 3.7 VDC. Typical power consumption is the average of 30 samples.

Mode	**Min**	**Typical (mA)**	**Max**
Sensing & Communication at 5 Hz	152	161 ±9	180
Sensing & Communication at 20 Hz[1]	153	175 ±16	205
Sensing & Communication at 40 Hz	170	183 ±9	205
Sensing & Communication[1] & LEDs (50%)[2]	187	205 ±14	245
Sensing & Communication[1] & LEDs (100%)	225	247 ±2	292
Sensing & Communication[1] & Motors (10 cm/s)[3]	455	512 ±30	584
Sensing & Communication[1] & Motors (25 cm/s)	613	660 ±25	717
Typical Use[123]	396	550 ±47	628

[1] Typical frequency rate used for sensing and communication.
[2] Common brightness used in the LEDs indicators (White color).
[3] Common velocity performed by the robot during the experiments.

Chapter 6

Applications

This chapter extends the scope of performance evaluation and delves into practical demonstrations of the robot's versatility. Through a series of diverse applications, we showcase the robot's capacity to tackle a spectrum of tasks and challenges across swarm robotics.

6.1 Mapping

In this experiment, we show the robot's mapping capabilities. More specifically, we explore its capacity to undertake mapping tasks using a single real robot configured to communicate with a remote computer running ROS. We employ the Gmapping[1] package, a laser-based Simultaneous Localization and Mapping (SLAM) algorithm, which is a standard ROS package for mapping environments.

To evaluate the robot's mapping capacity, we construct an environment comprised of white cardboard surfaces, simulating a corridor within a 1.20×1.20 meter area. The resulting occupancy map generated by the robot is depicted in Figure 6.1. The axes sequence (x-red, y-green, z-blue) overlaid on the map illustrates the robot's temporal pose computed from its odometry data. At the bottom of the figure, a top-view image provides insight into the environment setup and the ground truth trajectory executed by the robot. A video showcasing the execution of this experiment is accessible on Youtube[2].

[1]GMapping: http://wiki.ros.org/gmapping.
[2]Mapping Performance: https://youtu.be/_RWCCI8BI1s.

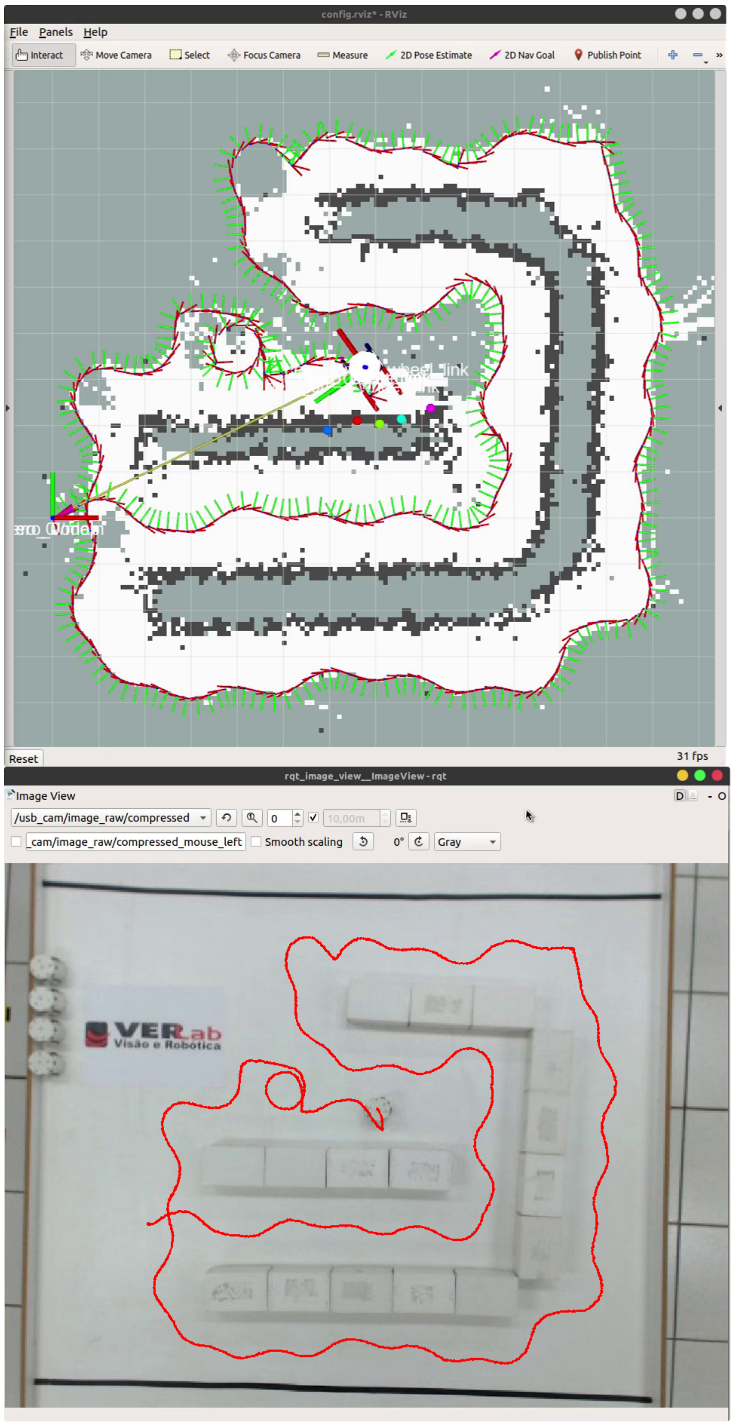


Figure 6.1: An occupancy map produced by the robot using only eight IR sensors and odometry. At the top, it is the RViz visualization tool showing the map computed by the robot. The sequence of axes (x-red, y-green) shows the trajectory computed by the odometry. At the bottom is a top-view image showing the environment and the ground truth trajectory performed by the robot.

Although mapping tasks typically demand sophisticated sensors like LiDAR and more precise localization methods, our results demonstrate that the robot, equipped with eight IR sensors and wheel odometry, can successfully perform mapping tasks. While the excellence of mapping essentially relies on the SLAM algorithm, this experiment underscores the robot's capability to execute mapping tasks effectively, rendering it a compelling choice for cooperative mapping research.

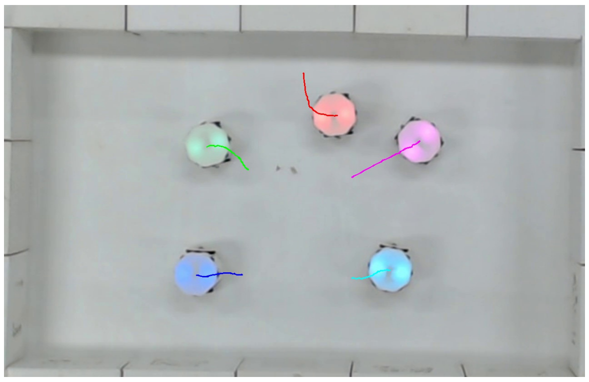
6.2 Decentralized coverage

In this experiment, we showcase the efforts of five robots engaged in a coverage task within a compact, confined environment measuring 0.8×1.20 meters. Different from the previous experiment, which necessitated more substantial computational processing, this one executes the coverage algorithm directly within the robot's firmware. This approach eliminates the requirement for the robots to communicate with a server running ROS.

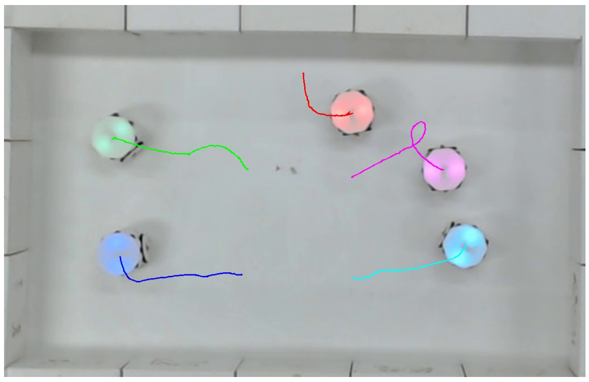
The coverage method entails the robots autonomously navigating the environment in a randomized manner while adeptly avoiding obstacles and collisions with other robots. Particularly, this is accomplished solely through local sensory input. Figure 6.2 provides a visual sequence of images captured from an overhead camera perspective. Each of the five robots emits a distinct color, facilitating their identification and tracking. The lines overlaid on the images illustrate the trajectory of each robot. A video of this experiment is accessible on Youtube[3].

The chosen environment for this experiment is relatively compact, taking into consideration the number of robots involved. This decision was made to assess the robots' maneuverability and their ability to manage potential interference among their IR sensors. Consequently, the successful navigation of the robots within the confined space, while effectively avoiding collisions over extended durations, underscores the feasibility of

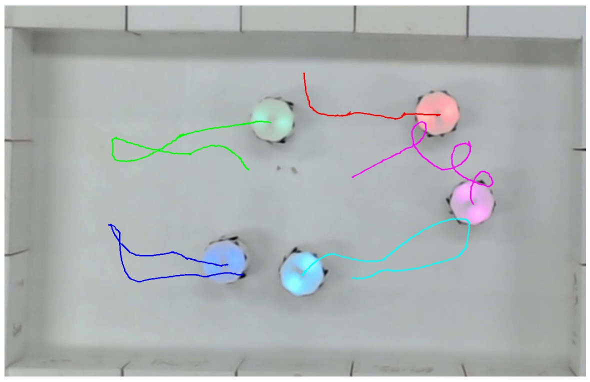
[3]Decentralized Coverage: `https://youtu.be/KmQXBcXKBtE`



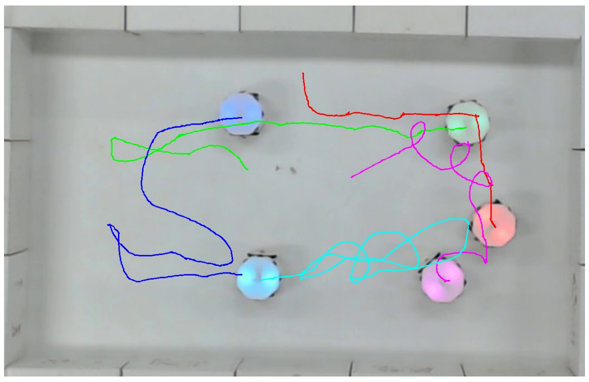
(a) $t = 2$ s.



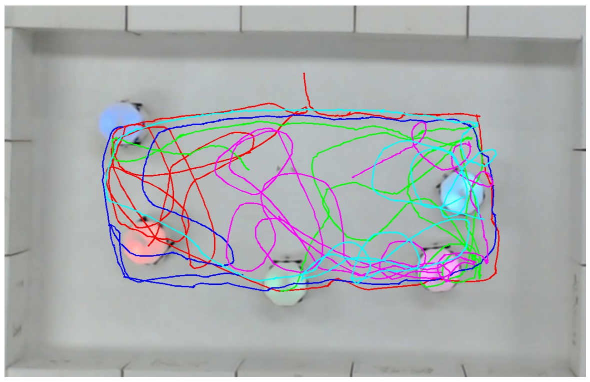
(b) $t = 4$ s.



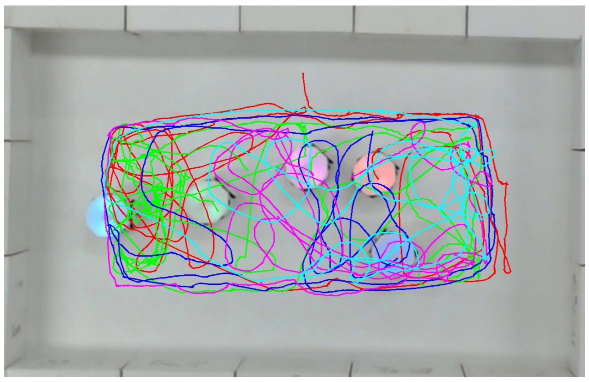
(c) $t = 10$ s.



(d) $t = 20$ s.



(e) $t = 80$ s.



(f) $t = 120$ s.

Figure 6.2: Snapshots of an experiment showing five HeRo robots performing decentralized coverage.

operating multiple robots in close proximity within a restricted area.

6.3 Flocking Behavior

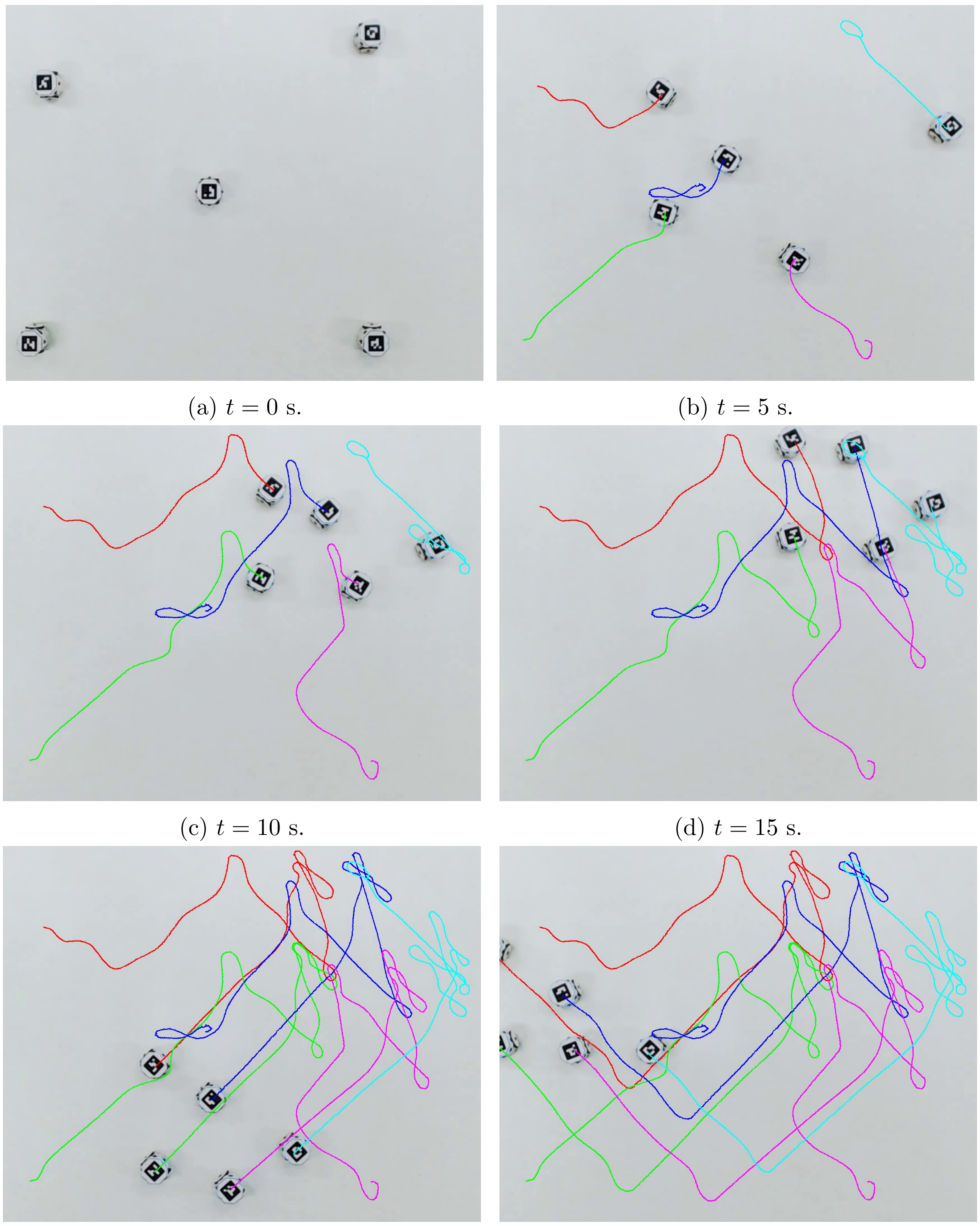
In this experiment, we showcase the robot's capability to engage in flocking behavior. We deployed five robots initially arranged in a random distribution within the environment. The implemented flocking algorithm for this experiment is decentralized, relying solely on information pertaining to the relative positions and velocities of neighboring robots (for more in-depth details, refer to (Rezeck et al., 2021b)).

Since our robots lack onboard sensors that can estimate such information, we employed a remote server running ROS to emulate such sensor functionality. To achieve this, we employed an overhead camera coupled with the Apriltag tracker algorithm (Wang and Olson, 2016; Malyuta, 2018). This combination enabled us to pinpoint the locations of the robots within the environment, subsequently computing their relative positions and velocities. This information was then provided to the flocking algorithm.

Figure 6.3 presents a sequence of images captured by the overhead camera, depicting the initial configuration of the robots. Over the course of the experiment, the robots autonomously organized themselves into a cohesive group, effectively navigating the environment. A video illustrating this experiment is accessible on Youtube[4].

This experiment served a dual purpose. First, it assessed the robot's motion control responsiveness when interacting with other robots. As each robot may require specific calibration parameters for its PID controller, the flocking task necessitates a synchronized and aligned motion among all agents. Misconfigured motion control parameters could lead to incorrect group navigation. Thus, through this experiment, we ascertained that our robots are properly calibrated and proficient in executing the flocking task, resulting in coordinated and cohesive group navigation.

[4]Flocking Behavior: `https://youtu.be/u7iioSKtHU8`.



(a) $t = 0$ s. (b) $t = 5$ s.

(c) $t = 10$ s. (d) $t = 15$ s.

(e) $t = 25$ s. (f) $t = 30$ s.

Figure 6.3: Snapshots of an experiment showing five HeRo robots performing flocking behavior.

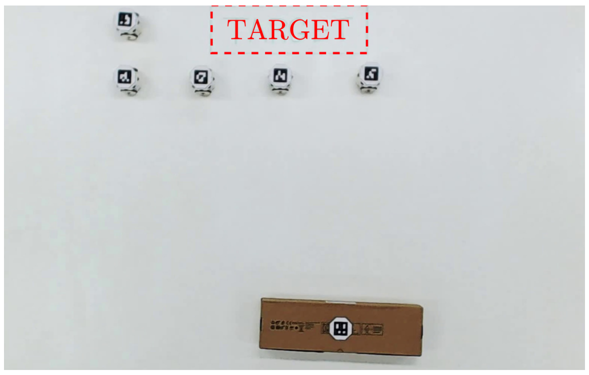
6.4 Cooperative Transportation

Finally, we conducted experiments evaluating the robot's performance in a cooperative transportation task. This particular task involves the swarm of robots collectively pushing an object toward a specified goal location. The strategy employed for these experiments is elaborated in Rezeck et al. (2021a). Notably, this strategy does not necessitate a priori knowledge of the object's shape or location; rather, it relies solely on awareness of the object's target location. Consequently, the robots can navigate the environment, form groups, and upon detecting the object, strategically maneuver around it to identify contact points for effectively pushing it toward the goal.

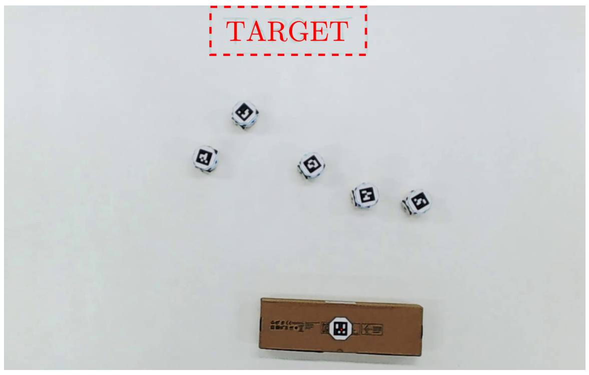
This decentralized approach obviates the need for global information about the swarm or the object, instead relying on each robot's ability to estimate relative positions and velocities of its neighbors while distinguishing between the object and other obstacles. To emulate the necessary sensors for this task, we once again utilized an overhead camera location system. Figure 6.4 provides a sequence of snapshots captured by the overhead camera, illustrating the robots' actions. Throughout the sequence, the robots dynamically congregate and coordinate their efforts to transport the object to its specified goal. A video showcasing this experiment is accessible on Youtube[5].

These experiments confirmed that our robots can effectively manage their motion while interacting with objects, demonstrating an appropriate level of traction to initiate and sustain cooperative transportation tasks.

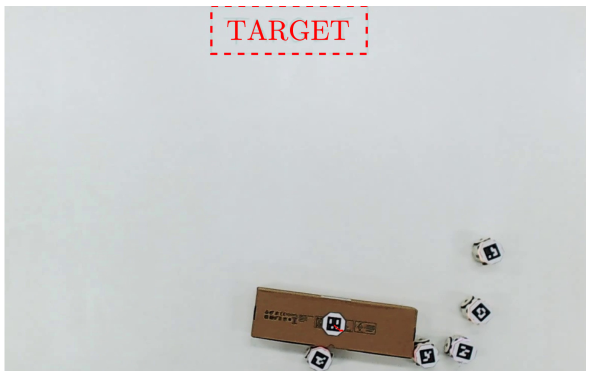
[5]Cooperative Transport: `https://youtu.be/hAS7FKYkKWQ`.



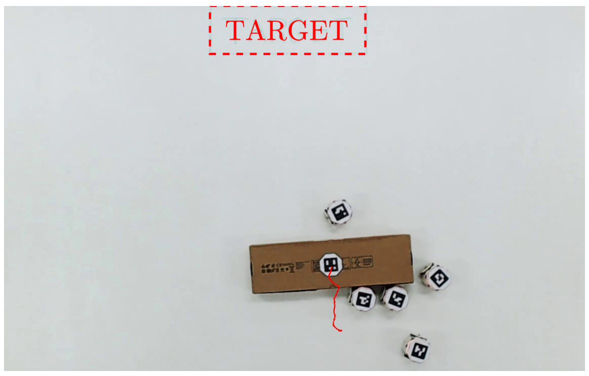
(a) $t = 0$ s.



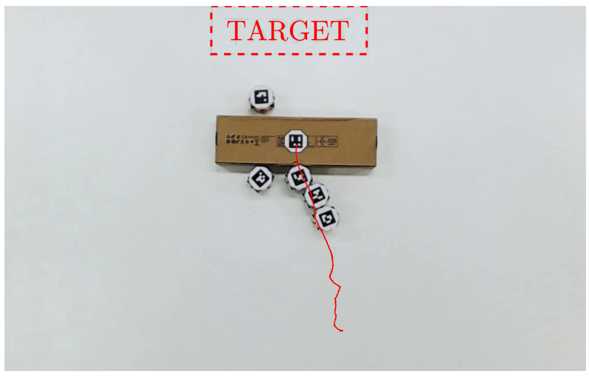
(b) $t = 3$ s.



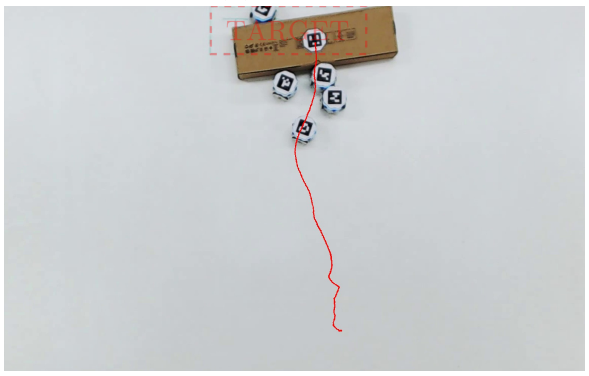
(c) $t = 10$ s.



(d) $t = 15$ s.



(e) $t = 20$ s.



(f) $t = 25$ s.

Figure 6.4: Snapshots of an experiment showing five real HeRo robots transporting an object toward its goal location.

Chapter 7

Conclusion and Future Work

This thesis introduced HeRo, an innovative open-source swarm robotic platform that addresses the demand for cost-effective, adaptable, and scalable solutions within the context of swarm robotics. The development of HeRo involved meticulous design considerations, harnessing readily accessible components and additive manufacturing processes to create a robot that boasts exceptional performance and practicality.

7.1 Conclusion

Despite its diminutive physical footprint, HeRo packs an impressive point in terms of sensing and networking capabilities. Engineered with versatility in mind, it boasts built-in WiFi communication and the convenience of over-the-air firmware upgrades. This adaptability serves as a catalyst for accelerated development, providing researchers with the agility to explore novel features and enhancements swiftly. What truly sets HeRo apart is its seamless integration with the ROS, a ubiquitous platform for robotic applications. This compatibility not only unlocks doors but also swings them wide open for developers, streamlining the creation and deployment of innovative functionalities. Whether one is delving into swarm robotics research or developing educational material, HeRo's capabilities prove to be a significant asset.

To further expedite experimentation and development, HeRo provides a fullyfledged simulation environment that seamlessly integrates with Gazebo, a popular and powerful simulation engine. This environment serves as an invaluable tool for continuous testing and rapid prototyping. Researchers and enthusiasts may alike benefit from the ability to explore HeRo's potential without the limitations imposed by physical robots. It enables them to fine-tune algorithms, validate control strategies, and simulate large-scale swarm scenarios. The flexibility offered by this simulation environment accelerates the development cycle, leading to quicker iterations and more robust robot behaviors.

HeRo's performance undergoes careful evaluation in a series of experiments, covering diverse aspects such as sensor accuracy, odometry precision, power autonomy, swarm communication, and control effectiveness. Remarkably, HeRo consistently demonstrates capabilities that either meet or surpass those of comparable commercial platforms, all while maintaining an affordable price point. These results underscore HeRo's well-rounded suitability for a wide array of swarm applications and emphasize its potential as an educational tool, making it accessible to both researchers and students.

Furthermore, the practicality of HeRo is demonstrated through a range of swarm applications, including mapping, decentralized coverage, flocking behavior, and cooperative transportation tasks. These applications showcase HeRo's adaptability, versatility, and effectiveness when addressing diverse challenges. In doing so, we reaffirmed HeRo's status as a highly promising and accessible platform for swarm robotics research, extending its appeal to a broad audience of robotics enthusiasts and professionals.

Finally, HeRo represents a substantial stride forward within the context of swarm robotics, encapsulating the core principles of affordability, adaptability, and extensibility. Its resilient performance, coupled with seamless integration into established tools and frameworks, places it as an interesting asset for researchers, educators, and robotics enthusiasts alike. In an ever-evolving technological landscape, HeRo serves as a resounding testament to the boundless opportunities within swarm robotics, promising transformative influences across a multitude of domains. This versatile platform not only advances the field but also opens doors for innovative applications, paving the way for a future where swarm robotics can make profound impacts across diverse sectors.

7.2 Limitations

While the robot has demonstrated commendable performance, several considerations deserve attention regarding its utilization and maintenance. In the following, we list and discuss some aspects to provide a comprehensive overview of its limitations.

- **Reproduction/Assembly**: Although the robot has a simple mechanical design, the use of additive manufacturing technologies, such as conventional 3D printers, does not always allow a proper fit of the parts. Then, it requires manual adjustments or finishing during assembly, demanding time and effort. This process is essential for the robot's transmission mechanisms, impacting wheel movement and encoder readings if left unattended.

- **Robot calibration**: The robot is designed to use affordable parts and components that have been available for several years. As expected, low-cost components also impact robot performance, requiring the user to calibrate the IR sensors and motors occasionally. As each component has different characteristics, the calibration process is required for each of the eight IR sensors and the two servo motors.

- **Wireless recharge**: Another point is the lack of convenience in charging the battery of each robot by plugging in a cable. Wireless charging modules have recently become commonplace, but they still come at a high cost. Although the idea of having an automatic recharge system is interesting, due to the cost and size, we decided to wait and deal with the manual recharge of the robots.

- **Mechanical wear**: Finally, another point impacted by the use of low-cost components is their durability. Although the rotary encoder is an interesting solution, some low-cost models have a short lifespan for our application, requiring replacement after months of use. In addition to the encoder, we also have to check the gear mechanism since we use ABS/PLA material that wears out with use and storage.

7.3 Future Work

The development and enhancement of the HeRo platform are ongoing research aimed at expanding its capabilities and keeping it at the forefront of swarm robotics technology. Several areas of focus for future work have been identified to further improve HeRo's performance and versatility. In the following, we highlight some of the most natural evolutions that can be performed in the short-mid term:

- One avenue of development is the creation of additional e-Hats, augmenting the platform's capabilities. These new expansions will provide users with a broader range of options for tailoring HeRo to specific applications and research needs.
- Improvements in internal filters for localization are also on the horizon. Enhancing the precision and accuracy of HeRo's localization system will contribute to better overall performance and reliability in various tasks.
- The assembly process is another point for refinement. Streamlining the assembly procedures will make it easier for users to construct and customize their HeRo robots, reducing barriers to entry and encouraging broader adoption of the platform.
- Localization will continue to evolve, with investigations into alternatives such as UWB or other indirect wireless methods. Exploring new localization methods can lead to more robust and adaptable swarm robotics applications.
- A crucial step in keeping HeRo's software stack up-to-date with the latest robotics advancements is the full migration to ROS 2. This transition will ensure that HeRo remains compatible with modern software frameworks, facilitating continued development and integration with cutting-edge tools and libraries.

In summary, the future of HeRo involves a continuous commitment to improvement and expansion. As the platform evolves, it will empower researchers, educators, and enthusiasts to push the boundaries of swarm robotics, opening doors to innovative applications and breakthroughs in the field.

Bibliography

Farshad Arvin, Khairulmizam Samsudin, Abdul Rahman Ramli, et al. Development of a miniature robot for swarm robotic application. *International Journal of Computer and Electrical Engineering*, 1(4):436–442, 2009.

Farshad Arvin, Khairulmizam Samsudin, Abdul Rahman Ramli, and Masoud Bekravi. Imitation of honeybee aggregation with collective behavior of swarm robots. *International Journal of Computational Intelligence Systems*, 4(4):739–748, 2011.

Farshad Arvin, Ali Emre Turgut, Nicola Bellotto, and Shigang Yue. Comparison of different cue-based swarm aggregation strategies. In *International Conference in Swarm Intelligence*, pages 1–8. Springer, 2014.

Farshad Arvin, Jose Espinosa, Benjamin Bird, Andrew West, Simon Watson, and Barry Lennox. Mona: an affordable open-source mobile robot for education and research. *Journal of Intelligent & Robotic Systems*, pages 1–15, 2018a.

Farshad Arvin, Simon Watson, Ali Emre Turgut, Jose Espinosa, Tomáš Krajník, and Barry Lennox. Perpetual robot swarm: long-term autonomy of mobile robots using on-the-fly inductive charging. *Journal of Intelligent & Robotic Systems*, 92(3-4):395–412, 2018b.

Guy Baele, Nicolas Bredeche, Evert Haasdijk, Steven Maere, Nico Michiels, Yves Van de Peer, Thomas Schmickl, Christopher Schwarzer, and Ronald Thenius. Open-ended on-board evolutionary robotics for robot swarms. In *2009 IEEE Congress on Evolutionary Computation (CEC 2009)*, pages 1123–1130. IEEE, 2009.

Gines Benet, Francisco Blanes, José E Simó, and Pascual Pérez. Using infrared sensors for distance measurement in mobile robots. *Robotics and autonomous systems*, 40(4): 255–266, 2002.

Eric Bonabeau, Directeur de Recherches Du Fnrs Marco, Marco Dorigo, Guy Théraulaz, Guy Theraulaz, et al. *Swarm intelligence: from natural to artificial systems*. Number 1. Oxford university press, 1999.

Alexandre Campo, Álvaro Gutiérrez, Shervin Nouyan, Carlo Pinciroli, Valentin Longchamp, Simon Garnier, and Marco Dorigo. Artificial pheromone for path selection by a foraging swarm of robots. *Biological cybernetics*, 103(5):339–352, 2010.

Gilles Caprari and Roland Siegwart. Design and control of the mobile micro robot alice. In *Proceedings of the 2nd International Symposium on Autonomous Minirobots for Research and Edutainment, AMiRE 2003: 18-20 February 2003, Brisbane, Australia*, pages 23–32. CITI, 2003.

Jianing Chen, Melvin Gauci, Michael J Price, and Roderich Groß. Segregation in swarms of e-puck robots based on the brazil nut effect. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 163–170. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

Jianing Chen, Melvin Gauci, and Roderich Groß. A strategy for transporting tall objects with a swarm of miniature mobile robots. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 863–869. IEEE, 2013.

Paul Christiano, Zain Shah, Igor Mordatch, Jonas Schneider, Trevor Blackwell, Joshua Tobin, Pieter Abbeel, and Wojciech Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint arXiv:1610.03518*, 2016.

Micael S Couceiro, Carlos M Figueiredo, David Portugal, Rui P Rocha, and Nuno MF Ferreira. Initial deployment of a robotic team-a hierarchical approach under communication constraints verified on low-cost platforms. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 4614–4619. IEEE, 2012.

Marco Dorigo, Vito Trianni, Erol Şahin, Roderich Groß, Thomas H Labella, Gianluca Baldassarre, Stefano Nolfi, Jean-Louis Deneubourg, Francesco Mondada, Dario Floreano, et al. Evolving self-organizing behaviors for a swarm-bot. *Autonomous Robots*, 17 (2-3):223–245, 2004.

Nicholas Farrow, John Klingner, Dustin Reishus, and Nikolaus Correll. Miniature six-channel range and bearing system: algorithm, analysis and experimental validation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6180–6185. IEEE, 2014.

Andrew Howard, Lynne E Parker, and Gaurav S Sukhatme. Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. *The International Journal of Robotics Research*, 25(5-6):431–447, 2006.

M Ani Hsieh, Vijay Kumar, and Luiz Chaimowicz. Decentralized controllers for shape generation with robotic swarms. *Robotica*, 26(5):691–701, 2008.

Cheng Hu, Qinbing Fu, and Shigang Yue. Colias iv: The affordable micro robot platform with bio-inspired vision. In *Annual Conference Towards Autonomous Robotic Systems*, pages 197–208. Springer, 2018.

Fabrício R Inácio, Douglas G Macharet, and Luiz Chaimowicz. United we move: Decentralized segregated robotic swarm navigation. In *Distributed Autonomous Robotic Systems*, pages 313–326. Springer, 2018.

Serge Kernbach. Swarmrobot. org-open-hardware microrobotic project for large-scale artificial swarms. *arXiv preprint arXiv:1110.5762*, 2011.

John Klingner, Anshul Kanakia, Nicholas Farrow, Dustin Reishus, and Nikolaus Correll. A stick-slip omnidirectional powertrain for low-cost swarm robotics: Mechanism, calibration, and control. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 846–851. IEEE, 2014.

Nathan P Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IROS*, volume 4, pages 2149–2154. Citeseer, 2004.

Mathieu Le Goc, Lawrence H Kim, Ali Parsaei, Jean-Daniel Fekete, Pierre Dragicevic, and Sean Follmer. Zooids: Building blocks for swarm user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 97–109. ACM, 2016.

Ming-Shian Lin and Chern-Lin Chen. An led driver with pulse current driving technique. *IEEE transactions on power electronics*, 27(11):4594–4601, 2011.

Danylo Malyuta. Guidance, navigation, control and mission logic for quadrotor full-cycle autonomy. Master's thesis, ETH Zurich, 2018.

Leandro Soriano Marcolino, Yuri Tavares dos Passos, Álvaro Antônio Fonseca de Souza, Andersoney dos Santos Rodrigues, and Luiz Chaimowicz. Avoiding target congestion on the navigation of robotic swarms. *Autonomous Robots*, 41(6):1297–1320, 2017.

Francesco Mondada, Michael Bonani, Xavier Raemy, James Pugh, Christopher Cianci, Adam Klaptocz, Stephane Magnenat, Jean-Christophe Zufferey, Dario Floreano, and Alcherio Martinoli. The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, volume 1, pages 59–65. IPCB: Instituto Politécnico de Castelo Branco, 2009.

Geoff S Nitschke, Martijn C Schut, and AE Eiben. Evolving behavioral specialization in robot teams to solve a collective construction task. *Swarm and Evolutionary Computation*, 2:25–38, 2012.

Iroju Olaronke, Ikono Rhoda, Ishaya Gambo, OA Ojerinde, and Olaleke Janet. A systematic review of swarm robots. 2020.

Ayberk Özgür, Wafa Johal, and Pierre Dillenbourg. Permanent magnet-assisted omnidirectional ball drive. In *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, number CONF. IEEE, 2016.

Ayberk Özgür, Séverin Lemaignan, Wafa Johal, Maria Beltran, Manon Briod, Léa Pereyre, Francesco Mondada, and Pierre Dillenbourg. Cellulo: Versatile handheld robots for education. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 119–127. ACM, 2017.

Daniel Pickem, Myron Lee, and Magnus Egerstedt. The gritsbot in its natural habitat-a multi-robot testbed. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 4062–4067. IEEE, 2015.

Daniel Pickem, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron, and Magnus Egerstedt. The robotarium: A remotely accessible swarm robotics research testbed. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1699–1706. IEEE, 2017.

Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

Paulo Rezeck, Renato M. Assunção, and Luiz Chaimowicz. Cooperative object transportation using gibbs random fields. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9131–9138, 2021a. doi: 10.1109/IROS51168.2021.9635928.

Paulo Rezeck, Renato M. Assunção, and Luiz Chaimowicz. Flocking-segregative swarming behaviors using gibbs random fields. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8757–8763, 2021b. doi: 10.1109/ICRA48506.2021.9561412.

Paulo AF Rezeck, Hector Azpurua, and Luiz Chaimowicz. Hero: An open platform for robotics research and education. In *Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR), 2017 Latin American*, pages 1–6. IEEE, 2017.

Michael Rubenstein, Adrian Cabrera, Justin Werfel, Golnaz Habibi, James McLurkin, and Radhika Nagpal. Collective transport of complex objects by simple robots: theory and experiments. In *Proceedings of the 2013 international conference on Autonomous* *agents and multi-agent systems*, pages 47–54. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

Michael Rubenstein, Christian Ahler, Nick Hoff, Adrian Cabrera, and Radhika Nagpal. Kilobot: A low cost robot with scalable operations designed for collective behaviors. *Robotics and Autonomous Systems*, 62(7):966–975, 2014a.

Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014b.

Vinicius Graciano Santos and Luiz Chaimowicz. Cohesion and segregation in swarm navigation. *Robotica*, 32(2):209–223, 2014.

Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots.* MIT press, 2011.

Ivica Slavkov, Daniel Carrillo-Zapata, Noemi Carranza, Xavier Diego, Fredrik Jansson, J Kaandorp, Sabine Hauert, and James Sharpe. Morphogenesis in robot swarms. *Science Robotics*, 3(25), 2018.

Alexander Sproewitz, Philippe Laprade, Stéphane Bonardi, Mikaël Mayer, Rico Moeckel, Pierre-André Mudry, and Auke Ijspeert. Roombots-towards decentralized reconfiguration with self-reconfiguring modular robotic metamodules. In *Proceedings of IEEE IROS2010*, number EPFL-CONF-149384, pages 1126–1132. Ieee Service Center, 445 Hoes Lane, Po Box 1331, Piscataway, Nj 08855-1331 Usa, 2010.

John Wang and Edwin Olson. Apriltag 2: Efficient and robust fiducial detection. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4193–4198. IEEE, 2016.

Andrew West, Farshad Arvin, Horatio Martin, Simon Watson, and Barry Lennox. Ros integration for miniature mobile robots. *Towards Autonomous Robotic Systems (TAROS)*, 2018.

Sean Wilson, Paul Glotfelter, Li Wang, Siddharth Mayya, Gennaro Notomista, Mark Mote, and Magnus Egerstedt. The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems. *IEEE Control Systems Magazine*, 40(1):26–44, 2020.

M Yogeswaran and SG Ponnambalam. Swarm robotics: An extensive research review. In *Advanced Knowledge Application in Practice*. InTech, 2010.

Jingjin Yu, Shuai D Han, Wei N Tang, and Daniela Rus. A portable, 3d-printing enabled multi-vehicle platform for robotics research and education. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1475–1480. IEEE, 2017.