Enhancing the technological maturity of robot swarms

Darko Bozhinoski
IRIDIA
IRIDIA, Université Libre de Bruxelles
darko.bozhinoski@ulb.be
0000-0002-6853-0310

Mauro Birattari
IRIDIA
IRIDIA, Université Libre de Bruxelles
mbiro@ulb.ac.be
0000-0003-3309-2194

***Abstract*—The field of swarm robotics has seen significant growth in recent years, with potential applications in a variety of areas. This paper delves into the current research challenges in swarm robotics from a software engineering perspective. The paper presents three key research directions that will pave the way toward creating industry-adoptable robot swarms.**

***Index Terms*—Keywords: robot swarms, automatic design methods, digital twins, mission and system specification, design models, runtime models.**

I. Introduction

Swarm robotics is a field of study that focuses on the use of large groups of robots to accomplish tasks that are beyond the capabilities of a single robot [2], [8], [23]. In swarm robotics, the behavior of the group is determined by the interactions between the individual robots and their environment. These robots work together in a self-organized and distributed manner, with no central leader or external infrastructure. The use of a robot swarm is motivated by its potential to be reliable, scalable, and flexible, making it a suitable approach for operating autonomous robots in challenging environments where individual robot failure or loss is likely, supporting infrastructures are unavailable, and communication is limited [10].

Although a number of robot swarms have been developed [15], there is currently a lack of a reliable engineering approach for designing control software for these systems. To increase the level of technological readiness for applications so that robot swarms can be successfully used in real-world environments, swarm robotics needs to mature as an engineering discipline. Currently, most swarm designers use a trial-and-error method, testing and improving individual-level behaviors until the desired collective behavior is achieved [13]. This approach is more akin to craftsmanship than engineering, as the quality of the result depends on the experience and intuition of the designer.

Designing robot swarms is challenging because of their self-organizing and distributed nature. One of the main difficulties in swarm robotics is the relationship between individual robots and the swarm, known as the individual/swarm dichotomy. While it is possible to specify the mission that the swarm should accomplish, the swarm itself cannot be preprogrammed. Instead, the designer must create the behaviors of the individual robots, which will then interact with each other and the environment to produce the desired collective behavior of the swarm. Finding a reliable and satisfactory way to bridge the gap between the requirements at the swarm level and the individual-level behaviors of the robots remains an open issue in the field.

For robot swarms to be widely adopted for practical use rather than just being a subject of research in laboratories, it is necessary for researchers in software and systems engineering to put forth significant effort in developing methodologies and tools to help designers create reliable and efficient swarms. This will enable faster adoption of robot swarms in industrial settings.

In this vision paper, we aim to pave the way toward creating technologically mature robot swarms. We identify the key challenges from the software and systems engineering perspective that hinder the process of adoption of robot swarms outside of laboratory settings.

II. Challenges and Research Directions

C1. Inconsistent system and mission specification

The main focus in the field of robot swarms is developing algorithms and control systems that can effectively coordinate the actions of a large group of robots in order to achieve a common goal. This involves designing communication protocols that allow the robots to share information and coordinate their actions, as well as developing strategies for self-organization and adaptability to changing conditions. Little research has been devoted to approaches that consider swarm robotics as a systems engineering discipline. In most of the robot swarm studies, the missions in which the swarm is evaluated are poorly documented, and the specifications of the swarm itself are vaguely described. The inconsistency between different swarm studies makes it extremely difficult for researchers to reproduce the results [5]. It is also not clear if the obtained results are transferable to different missions and robot types.

To the best of our knowledge, requirements specification for swarm robotics has not been properly addressed as a research question. The highest level of abstraction that has been extensively discussed in swarm robotics is the development process. A work in this direction is Buzz, a scripting language for programming heterogeneous robot swarms [1], [21]. The

only work where a textual specification language has been developed for a class of swarm missions is one of our previous studies [5]. It is clear that further work in this direction is strongly needed for faster adoption of robot swarms in real settings.

Although a number of approaches have been proposed in the software and systems engineering literature for designing and testing robotic systems [9], they have not been investigated in the context of swarm robotics. Most of these approaches focus on decoupling the different phases of the robot life-cycle, which appears to be inappropriate in swarm robotics. Indeed, they model the system to be realized at a level of abstraction that is too high and neglects the complex robot-robot and robot-environment interactions that characterize the operation of a robot swarm. For example, these approaches assume that it is possible to establish a mapping between the high-level collective goals of the swarm and low-level individual behaviors of the robots comprised therein [6], [7], [11]. Unfortunately, the swarm robotics praxis indicates that making such a mapping explicit is not generally possible [4], [12].

R1. A systematic approach to designing robot swarms Software engineers should provide support with domain-specific languages, tools, methodologies, and frameworks for swarm designers to be able to design swarm missions. We argue that precise formal abstractions are needed to specify the environment and the elements of which the system consists.

R1.1. Reference model specification. In swarm robotics, the notion of the reference model has been implicitly used without a clear formal definition of what exactly it represents. As it can be seen in a collection of methods [13], [19] reference models capture basic characteristics about the individual robot that are strongly related to a set of predefined behaviors such as the speed of the robot wheels, proximity sensor, ground sensor readings, etc. However, the reference model fails to capture properties that: (i) show the repetitiveness of the mission execution; (ii) the transferability to another mission. For example, it is not clear if a robot swarm that has performed an aggregation task where robots need to aggregate in a specific region without using an external guiding system can be reused in a different mission or if it can successfully repeat the predicted performance on the same mission. To design and develop robot swarms that will reliably perform complex missions, there is a strong need to model additional aspects of the swarm related to its evolution, such as the deterioration of the hardware elements (e.g., sensors, and actuators), battery levels, the state of the robots after completing the mission certain number of times, etc. There is a strong demand for formalisms, tools, and languages that will precisely define structures, design the architecture, create test cases, and evaluate robot swarm capabilities through the whole swarm lifecycle. One possible way to move forward is through the definition of the concept of a skill that has been already proposed in general robotics, however, it has not been discussed in the field of swarm robotics, yet.

R1.2. Mission model specification. In most of the swarm literature, designers specify the goal that should be achieved without providing details about the context. A crucial aspect in modeling the swarm mission is the context in which the swarm operates and the objective function that should be optimized. Software engineering has played an important role in identifying patterns, and classes of missions for general single robots. In [20], [20], authors analyze specification patterns for robotic missions. Specification patterns are a set of guidelines that are used to specify the requirements and constraints of a robotic mission. These patterns have been used to clearly and accurately define the goals, tasks, and actions that a robot is expected to perform during the mission. There is currently a lack of research in the field of swarm robotics that examines patterns in swarm missions. This gap in knowledge motivates the need to develop a new research direction focused on designing a classification system for objective functions that can be easily modified and reused by swarm designers. Objective functions are an essential component of swarm mission definitions, as they shape the design of the system and how it will operate. However, in many of the current studies in swarm robotics, objective functions are not explicitly defined or discussed, but rather are incorporated into the algorithms being developed. The proposed research aims to address this issue by creating a classification schema with objective functions as building blocks that can be easily used and adapted by swarm designers.

Challenge 2. Uncertainty in robot swarm performance: Swarms robotics deals with several sources of uncertainty such as operating in an unstructured or dynamic environment, uncertainty related to robot communication due to limited range, bandwidth, and delays, uncertainty subject to various sources of noise (e.g., measurement noise, communication noise, and hardware failures, etc.). Moreover, sensors may not always provide accurate readings, or there may be interference or errors in the communication signals between the robots. There is a crucial need to consider new approaches to the different phases of the robot swarm development process: designing, deploying, and testing robot swarms. We need to be able to tackle each of these different sources of uncertainty in a systematic way.

C2.1 Uncertainty emerging from the design of robot swarms. As already mentioned above, robot swarms have been mostly designed manually: an individual-level behavior is iteratively improved and tested until a desired collective behavior is obtained. The quality of the resulting solution strongly depends on the experience and intuition of the designer. To avoid, or at least reduce, the uncertainty induced by the crucial role of the human designer, automatic and semi-automatic design approaches have been proposed [3], [14], [22], [24]. In semi-automatic design, a human designer plays an active role in guiding an optimization process, while in fully automatic design, the optimization process does not require or allow any human intervention. The most promising approaches that have been proposed so far for the automatic design of robot swarms are in the area of design by optimization [16]. In design by optimization, the design problem is re-

formulated into an optimization problem: an optimization algorithm searches a space of candidate solutions to maximize an objective function. In this context, a candidate solution is an instance of control software and the objective function is a mission-dependent metric that measures the performance of the swarm on the given mission [4]. The semi-automatic design process for a swarm utilizes an optimization algorithm while incorporating human input to iteratively improve the control software for the desired mission. The designer uses their intuition and experience to evaluate the behavior produced by the optimization process and make modifications to elements such as control software architecture, sensor reading pre-filtering, models, optimization algorithm parameters, and the objective function being optimized until they are satisfied with the final result or cannot improve further. In the robot swarm literature, it is not common for researchers to provide information on the role of the designer or the number of iterations needed to reach the final outcome (human-in-the-loop). The design method is manually tailored to the specific mission, making it heavily dependent on human input and the results are often hard to reproduce. Hence, human-in-the-loop is a significant challenge in the design of robot swarms as the expertise of the swarm designer has a strong influence on their development.

C2.2. Uncertainty emerging from simulation. In the swarm robotics literature, swarm designers distinguish between two major classes of automatic design methods depending on whether the design phase happens before or after the deployment of the software on the physical robots, and that is [13]: offline and online. In offline design methods, the design process happens in a separate simulation phase before the robot swarm is deployed. This design phase involves evaluating a large number of different instances of control software. Online design methods involve designing the robot swarm while it is already deployed in its operational environment. The majority of research has been directed toward creating offline automatic design methods.

While offline automatic design methods contribute towards the reduction of uncertainty in the robot swarm performance, they still suffer from the reality gap. The reality gap in robot swarms refers to the difficulty in accurately simulating or emulating the behavior of large numbers of robots in the real world due to unreliable hardware and the complexity of the physical interactions between the robots. Due to the reality gap, control software generated in simulation suffers a performance drop when deployed on physical robots. The reality gap arises because it is often infeasible to design robot swarms in the same conditions in which they will ultimately operate. This is due to the difficulty of creating accurate and detailed simulations that fully capture the dynamics of the real world. The reality gap is a significant challenge in the field of swarm robotics, as it makes it difficult to accurately predict the performance of swarm algorithms deploying them on real robots.

C2.3 Uncertainty emerging from the evaluation process There are only a few instances where new automatic design methods and concepts are compared to previously introduced ones. At present, there is a lack of established, consistent practices for evaluating and comparing automatic design methods through empirical assessment. In [17], authors propose an initial work in the direction of comparing automatic design methods. Many other studies that have introduced automatic design methods are specific to a particular application and cannot be generalized to other domains. They often focus on answering scientific questions that are not directly related to automatic design, such as the plausibility of biological models or the justification of animal behaviors in evolutionary terms. The experimental protocol, the definition of benchmark missions, and the evaluation criteria are crucial elements in establishing robot swarms as an engineering discipline.

R2.1. Digital twins for robot swarms. As swarm robotics establishes as an engineering science, we believe that the concept of "digital twin" will play an important role in designing reliable robot swarms. Digital twins are virtual representations of physical objects or systems, and they can be used to simulate and optimize the performance of robot swarms [18]. There are three important parts in the digital twin of an object: (i) a model of the object, (ii) an evolving set of data relating to the object in real settings, and (iii) a means of dynamically updating or adjusting the model in accordance with the data. In most cases of the development of robot swarms, a straightforward development process is followed where designers iterate over a desired behavior in simulation, after which they deploy that behavior on a real swarm. The concept of a digital twin allows the swarm designers to continue monitoring and controlling, virtual testing, predictive maintenance, and lifetime estimation of the robot swarm, even after deployment. An individually tailored model can be used for: (i) maintenance scheduling: by using the individual robot twin to update parameters related to known possible faults and thus identifying problems and fixing those issues before they become catastrophic, (ii) lifetime prediction: including the ability to revise robot swarm lifetime estimate in service. (iii) performance assurance, to check that any measured deviations from the swarm specification do not compromise performance to an unacceptable degree.

R2.2. SwarmOps. With the introduction of digital twins in the field of robot swarms, there will be a possibility for opening SwarmOps as a new research direction. SwarmOps will involve applying DevOps principles to the development and operation of robot swarms. It would involve using automation and continuous integration/continuous deployment (CI/CD) practices to streamline the process of building, testing, and deploying software updates to the robots in the swarm. Its goal is to improve the time-to-market and overall performance of the robot swarms through agility and automation. By using SwarmOps to continuously deliver updates, the quality and resilience of robot swarms will be improved. To achieve this, building new frameworks for designing and implementing digital twins will play a crucial role in advancing the technological readiness for robot swarms.

Challenge 3. Design vs. runtime models. In offline design

methods, the design process happens in a simulation before the robot swarm is deployed. Once the offline phase is completed, the robot swarm is deployed in its operational environment. Using software engineering terminology, the output of this phase is design models that can be deployed on the real robots. Online methods attempt to create a runtime model that is more suited to the specific task at hand than those produced using offline methods. Using software engineering terminology, the output of this online phase is a set of runtime models that can be deployed on real robots. The features of the runtime models are likely constrained to a reduced set of mission and system features because the runtime models are used and updated by the individual robots themselves while they are operational, which means that the computational resources and response time are also limiting factors. There have been some works that propose online methods but this direction has not been investigated properly.

R3. The interplay between design and runtime models We believe that the interplay between design and runtime is a promising research direction that has not been explored in the area of swarm robotics. While complex design models can be obtained through an offline optimization method, online methods are constrained to create runtime models that should be updated in a distributed way. It is crucial for system and software engineers to understand which mission and system aspects can be managed through a design model and which should be managed through the collection of runtime models of the individual robots. In this direction, it is important to consider the complexity of the design versus the complexity of the runtime model employed directly on the real robots, the granularity of the design and runtime models, the accuracy of the runtime models, the robustness of the models, etc. We consider that software and system engineers will play an important role in transferring the principles they have been exploring in designing distributed self-adaptive systems which will pave the way toward the design of more efficient and resilient robot swarms.

III. Summary

For robot swarms to be widely adopted for practical use it is necessary for researchers in software and systems engineering to put forth significant effort in developing methodologies and tools to help swarm designers create reliable and efficient swarms. In this vision paper, we discussed three main challenges that must be addressed before deploying robot swarms in industrial settings. Moreover, we outlined research directions in which software engineers can provide meaningful insights in addressing the aforementioned challenges.

ACKNOWLEDGMENT

This work is supported by the Belgian Fonds de la Recherche Scientifique – FNRS [grant number 40001145].

References

[1] G. Beltrame, E. Merlo, J. Panerati, and C. Pinciroli, "Engineering safety in swarm robotics," in *Proceedings of the 1st International Workshop on Robotics Software Engineering*, 2018, pp. 36–39.

[2] G. Beni, "From swarm intelligence to swarm robotics," in *International Workshop on Swarm Robotics*. Springer, 2004, pp. 1–9.

[3] M. Birattari, A. Ligot, and K. Hasselmann, "Disentangling automatic and semi-automatic approaches to the optimization-based design of control software for robot swarms," *Nature Machine Intelligence*, vol. 2, no. 9, pp. 494–499, 2020.

[4] D. Bozhinoski and M. Birattari, "Designing control software for robot swarms: Software engineering for the development of automatic design methods," in *ACM/IEEE 1st International Workshop on Robotics Software Engineering, RoSE*. New York: ACM, 2018, pp. 33–35.

[5] D. Bozhinoski and M. Birattari, "Towards an integrated automatic design process for robot swarms," *Open Research Europe*, vol. 1, p. 112, 2021.

[6] D. Bozhinoski, D. Di Ruscio, I. Malavolta, P. Pelliccione, and M. Tivoli, "Flyaq: Enabling non-expert users to specify and generate missions of autonomous multicopters," in *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. San Diego, CA, USA: IEEE, 2015, pp. 801–806.

[7] D. Bozhinoski, D. Garlan, I. Malavolta, and P. Pelliccione, "Managing safety and mission completion via collective run-time adaptation," *Journal of Systems Architecture*, vol. 95, pp. 19–35, 2019.

[8] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[9] D. Brugali and E. Prassler, "Software engineering for robotics," *IEEE Robotics & Automation Magazine*, vol. 16, no. 1, pp. 9–15, 2009.

[10] T. De Wolf and T. Holvoet, "Design patterns for decentralised coordination in self-organising emergent systems," in *International Workshop on Engineering Self-Organising Applications*. Springer, 2006, pp. 28–49.

[11] D. Di Ruscio, I. Malavolta, and P. Pelliccione, "A family of domain-specific languages for specifying civilian missions of multi-robot systems," in *First Workshop on Model-Driven Robot Software Engineering-MORSE*, York, UK, 2014, pp. 13–26.

[12] M. Dorigo, M. Birattari, and M. Brambilla, "Swarm robotics," *Scholarpedia*, vol. 9, no. 1, p. 1463, 2014.

[13] G. Francesca and M. Birattari, "Automatic design of robot swarms: achievements and challenges," *Frontiers in Robotics and AI*, vol. 3, p. 29, 2016.

[14] G. Francesca, M. Brambilla, A. Brutschy, V. Trianni, and M. Birattari, "AutoMoDe: a novel approach to the automatic design of control software for robot swarms," *SI*, vol. 8, no. 2, pp. 89–112, 2014.

[15] L. Garattoni and M. Birattari, "Swarm robotics," in *Wiley Encyclopedia of Electrical and Electronics Engineering*, J. Webster, Ed. Hoboken NJ: John Wiley & Sons, 2016.

[16] H. H. Hoos, "Programming by optimization," *Communications of the ACM*, vol. 55, no. 2, pp. 70–80, 2012.

[17] A. Ligot, A. Cotorruelo, E. Garone, and M. Birattari, "Toward an empirical practice in offline fully automatic design of robot swarms," *IEEE transactions on evolutionary computation*, vol. 26, no. 6, pp. 1236–1245, 2022.

[18] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346–361, 2021.

[19] F. J. Mendiburu, D. G. Ramos, M. R. Morais, A. M. Lima, and M. Birattari, "Automode-mate: Automatic off-line design of spatially-organizing behaviors for robot swarms," *Swarm and Evolutionary Computation*, vol. 74, p. 101118, 2022.

[20] C. Menghi, C. Tsigkanos, P. Pelliccione, C. Ghezzi, and T. Berger, "Specification patterns for robotic missions," *IEEE Transactions on Software Engineering*, vol. 47, no. 10, pp. 2208–2224, 2019.

[21] C. Pinciroli and G. Beltrame, "Buzz: An extensible programming language for heterogeneous swarm robotics," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3794–3800.

[22] M. Quinn, L. Smith, G. Mayley, and P. Husbands, "Evolving controllers for a homogeneous system of physical robots: structured cooperation with minimal sensors," *PTRSA*, vol. 361, pp. 2321–43, 11 2003.

[23] E. Şahin, "Swarm robotics: From sources of inspiration to domains of application," in *International workshop on swarm robotics*. Springer, 2004, pp. 10–20.

[24] V. Trianni, *Evolutionary swarm robotics: evolving self-organising behaviours in groups of autonomous robots*. Springer, 2008, vol. 108.