Swarm Robotics and Quantum Computing in Simulated Environments

Alexandre Thomas[1,†], Maria Mannone[2,*,†], Valeria Seidita[2] and Antonio Chella[2,3]

[1]Faculté d'Ingénieur, Université de Bourgogne, Dijon, France
[2]Department of Engineering, University of Palermo, Italy
[3]ICAR, National Research Council (CNR), Palermo, Italy

Abstract

Inspired by nature, robotic swarms can be dramatically helpful in search-and-rescue missions after natural or human-caused disasters, especially when the environmental conditions make dangerous the direct intervention of human rescuers. In this research, we focus on a simulated scene for a target-reaching mission, simulated on Webots, jointly with a recent quantum algorithm modeling pairwise interactions in a swarm. The platform Webots is known for its portability and free accessibility. Quantum computing offers new potentialities regarding efficiency and computational time, but its application to the robotic domain is a largely unexplored field. Here, we propose a connection between the IBM quantum simulators and the Webots platform, perform a comparison between a purely random procedure and the results obtained with the quantum circuit, and discuss possible future developments of the research.

Keywords

swarm robotics, quantum computing, search and rescue, simulation

1. Introduction

The observation of nature inspires artistic creation, theoretical thinking, and practical applications. It is the case of robotics developments modeled upon the structure of natural swarms. In nature and in its robotic imitations, a swarm is constituted by a set of multiple entities, simple in their design and skills, able to achieve a complex task through their mutual interaction, information-exchange, and collaboration. Similarly to natural swarms, a robotic swarm is robust (losing a unit does not affect the whole [18, 30, 28, 2]) and scalable (invariance of behavior upon the change of swarm size). Swarm robotics is widely used in search-and-rescue contexts where the intervention of human rescuers is dangerous or made impossible because of environmental conditions. Swarm robotics, including flying swarms [33], can thus be crucial in disaster management [1], and exploiting their self-organizational properties [7]. We will use a search and rescue scenario in this paper for validating the proposed approach.

Recently, quantum computing, a branch of computer science derived by the basic laws of quantum mechanics [32, 15], has been applied to artificial intelligence [35, 21], robotics [12, 5, 10, 22] and swarm robotics [20, 36, 3, 8]. There are two main reasons: enhancement of

10th Italian Workshop on Artificial Artificial Intelligence and Robotics (AIRO 2023)
**Corresponding author.
‡These authors contributed equally.
alexandre.thomas.scolaire@gmail.com (A. Thomas); mariacaterina.mannone@unipa.it (M. Mannone); valeria.seidita@unipa.it (V. Seidita); antonio.chella@unipa.it (A. Chella)

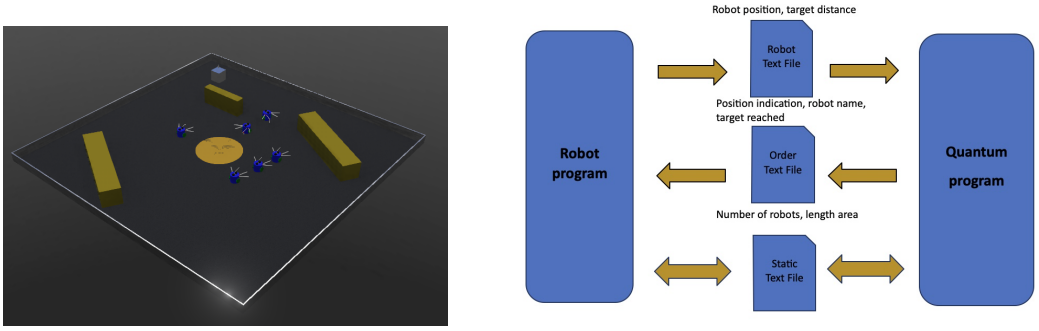
the efficiency of robotic algorithms and test of the translation from classic codes to quantum ones. In fact, quantum computing is yet at its beginning, opening a new chapter of computer science. Thus, the translation itself from classical codes to quantum ones is a topic of theoretical research and computational test.

In our recent research, we exploited quantum computing to model the pairwise interactions between the robots of a swarm in a search and rescue mission [24, 25]. In particular, we quantized the information about position of a robot in the arena, and success in individual target searching, describing them via a quantum formalism as state superposition. The pairwise interaction is modeled via a quantum logic gate, which takes into account the information from a robot, compares it with the information of another robot, and suggest a possible new position in space to be reached. For computational reasons, we then considered, as inputs of the logic gate, only the information coming from the robot that is the most successful one in the first step of the random exploration of the arena. The quantum gate is called cyclically, until the swarm reaches the target. Thus, we modeled the behavior of single elements of the swarm and then observe the emergent behavior of the global swarm as a unique and single entity pursuing a specific goal. In the context of swarm robotics, it is common to use simulation tools. Having hundreds or thousands of small robots operating in a real environment can be very time and resource consuming for a research lab. The same is true for quantum computing. From a technological point of view, there are several problems that need to be solved if simulation is to be properly used in swarm modeling with quantum computing. The simplest problem is interoperability. Most robotics applications use simulation tools that do not yet incorporate quantum approaches. They allow us to design and implement different types of robotic applications that can launch and solve missions, but the real limitation is to consider the probability of a particular behavior. We are working to provide a framework for modeling robot applications, where the behavior of the robots is not deterministic, but depends, for example, on the environment or on the interactions between the robots. The number of interactions needed to reach the target depends on the number of robots because the larger the number of units, the more likely that one of them is in proximity of the target, and thus, the less communications are needed. We claim that in these cases we can exploit the power of quantum approaches, but at the same time we need tools for quantum simulation.

Here, we consider an ideal terrestrial scenario simulated on Webots, proposing a real-time interaction with IBM quantum simulators (Section 2), presenting the results of our experiments in Section 3. We discuss advantages, limits, and possible developments of our approach (Section 4). Our contribution is a real-time, fully working code development and connection of algorithms to: *(i)* visualize the motion of quantum-driven robots; *(ii)* use the robotic sensory information on position and (perceived) target proximity as inputs for a quantum circuit; *(iii)* use the output of the quantum circuit as input for the subsequent steps of the robotic simulation.

2. Webots and Quantum Computing

We coded simple terrestrial robots, with two wheels, distance sensors, and GPS sensors. The scenario is a squared arena, with a variable number of obstacle, and the target characterized by a yellow circle, see Figure 1 (a).



(a) initial setup

(b) final setup

Figure 1: (a) Scenario of the Webots simulation; (b) Structure of the communication between the quantum code and Webots. One file gives the robot information, and another one gives it the order. A file acts as a common variable shared between the robot code and the quantum method.

Here, we join the codes from [25], accessing IBM quantum simulators, with the Webots platform for robotic simulation, through a file-based interaction, testing limits and advantages of this approach. First, to interact with the quantum gate circuit, we normalize the coordinates between 0 and 1, avoiding negative values. We made this choice to work with non-negative probability amplitudes. Then, we perform some tests with a forced reshuffle of positions, and with the quantum-gate loop. We also added another function, concerning position accuracy and control. In this scenario, the robots receive the coordinates of the points to reach in the next steps of the simulation, and send back their current position. That is, the robots communicate between them, telling each other their position and the points where they have to go. Random movement is activated when, at a certain time point, the reward of all robots is low, and a step of space exploration seems fundamental. In the simplified scheme of [25], another simplification has been made: instead of calling the quantum circuit for each pairwise interaction, only the robot with the highest reward feeds its information inside a quantum circuit. The output of the circuit will provide the most recommended point to be reached.

States, initialization, and communication protocol need to be set to ensure the coordination between the code for each single robot and the quantum circuit. To establish such a communication, we use a text file to represent a dialogue between the two programs. The text file acts as a log of the communication and stores the data about the actual position of the robots and their reward, and the position where the robot needs to go. The overall process is summarized in Figure 1 (b). We need to indicate which program sends information, or if the information received are processed.

Let us now analyze the details of the proposed approach. The *Robot program*, implemented for N robots, creates for each robot the file *ship_swarm(N)_data.txt*, to write the actual position of the assigned robot. The *Quantum program* creates the file *statics_variables.txt*, containing the size of the arena and the number of robots available in Webots. These files are not modified during the simulation. *Robot program* starts only if the files *statics_variables.txt* are present, and sends to each of them the length of the arena. If N robots send the same length, then an N-line will be present in the text file. The number of lines indicate the number of robots in

the simulation. The quantum program acknowledges the modification of *statics_variables.txt*, and gets the number of robots jointly with the length of the area. Then, it creates N files called *position_indicator_0.txt*. Finally, the program in Webots reads the positions written in *position_indicator_0.txt*, and rewrites the robots' new positions on *ship_swarm(N)_data.txt*. Thus, these files are updated at each step of the simulation, and constitute the real bridge between the application on Webots and the quantum code in Python. The whole process is repeated until the robots reach the target with the chosen precision.

The first program needs to set up the random shuffling without communication, looping until all robots are close to the target. The target is defined as the main objective of the swarm; we build a variable defining its minimum range. Multiple obstacles are included, to test obstacle avoidance. Obstacles and target are set on the arena by the user, while the robots are spread randomly. At each iteration of the loop, the program computes the new position of each robot and finds out if it is near to the target, or too close to an obstacle. If the latter is true, the robot reshuffles its position again, until it finds a correct spot. Thus, the robot keeps moving, unless it already reached the target.

In Pseudocode 2, we initially perform the class initialization and choose the number of robots, obstacles, and position of the target. A loop ensures that all robots near to the target do not collide with the obstacles. Here we take the position of the closest robot from the target as a reference to implement the quantum circuit, giving us two coordinates. These coordinates are then communicated to the other robots, to help them reach the target more efficiently. The swarm behavior emerges from the local, pairwise interactions modeled via the quantum circuit, and leads the swarm reach the target. To avoid superpositions between the robots, they are spread on a small circle around the outcome of the quantum circuit. In Section 3, we present the results obtained with the forced, random reshuffle (Pseudocode 1) and with the quantum circuit (Pseudocode 2, Table 2).

Algorithm 1 With the forced reshuffle

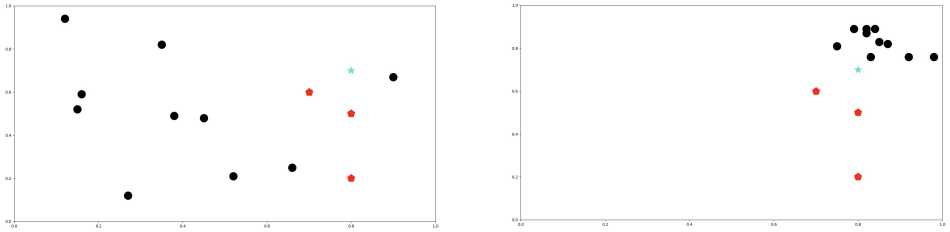
Inputs: Class Target (T: Target)
Class Obstacle (number of obstacles to define, O_1, O_2, O_3...)
Class Robotx (name: R_1, R_2, R_3...)
Initialize instance Target (position, name)
Initialize multiples instances Obstacle (position, Name)
Initialize multiples instances Robot (Name, position, reward)
while all robots are not near the target **do**
    for each robot created **do**
        if the robot is not near the target **then**
            while the robot is near an obstacle **do**
                Shuffle the position randomly
            end while
            Write the new parameters in a text file attach to the robot
        end if
    end for
    Increment the iteration
end while
Display the area and the number of iterations

Algorithm 2 With the quantum circuit

Inputs: Class Target (T: Target)
Class Obstacle (number of obstacles to define, O_1, O_2, O_3...)
Class Robotx (name: R_1, R_2, R_3...)
Initialize instance Target (position, name)
Initialize multiples instances Obstacle (position, Name)
Initialize multiples instances Robot (Name, position, reward)
while all robots are not near the target **do**
    Calculate the outcome by taking the closest robot from the target
    for each robot created **do**
        if the robot is not near the target OR not the closest one **then**
            while the robot is near an obstacle **do**
                Shuffle the position around the outcome
            end while
            Write the new parameters in a text file attach to the robot
        end if
    end for
    Increment the iteration
end while
Display the area and the number of iterations

3. Results

In this Section, we present the results of our tests, where we maintained fixed some parameters, changing other ones. We compare the results obtained with a purely random procedure (Table 1), based on forced steps of reshuffle (Pseudocode 1), with the ones obtained using the quantum circuit (Pseudocode 2, Table 2). Figure 2 shows an example of the visualization in Python, derived from [25], with progressive approaching of the swarm centroid to the target. 32 iterations were needed to achieve the final output in Figure 2. Considering the number of iterations required by the two methods, we can deduct which is the best strategy to find the target.



(a) initial setup (b) final setup

Figure 2: Initial and final configurations of one of the run of the code. The red pentagons are the obstacles, the turquoise star is the target, and the black dots are the robots. The goal of the robots is finding the target. As the setup (a), the robots are initially scattered through the arena. Then, they collect information from their sensory devices and exchange messages between them. These information enter the logic gate, whose result is sent to the robots in terms of new positions to be reached for the further step of exploration. In (b), the swarm found the target avoiding the obstacles, and thus the mission is successful.

Testing the forced reshuffle. We test the consistency of the forced-reshuffle method and the needed number of iterations to let the swarm converge on the target, according to fixed and variable parameters: number of robots, obstacles, reward threshold. Table 1 presents the results of the first three experiments, referring to the Pseudocode 1. In the first experiment, only the number of robots changes, while the other parameters are maintained fixed. We notice that the more robots we add, the more iterations are needed to reach the target. In the second experiment, we change the number of obstacles and their position. In the third experiment, only the minimum reward changes.

From the reshuffle strategy, we find that setting a too-high value as the minimum reward may let the robot be closest as possible to the target, but it requires a higher number of iterations to achieve this kind of precision. Thus, we deduce that the forced reshuffle is mainly chaotic, due to the inconsistence of the placement. The number of iterations can be low if we have several robots.

Table 1
Results of the forced reshuffle (Algorithm 1)

	0 obstacles, min. reward 0.8				
	No. robots	mean iter.	st. dev.	min iter.	max iter.
exp. 1	5	16.85	6.08	5	29
	10	20.9	8.18	8	39
	20	27.4	7.78	16	43
	5 robots, min. reward 0.8				
	No. obstacles	mean iter.	st. dev.	min iter.	max iter.
exp. 2	5	13.1	7.21	5	30
	10	17.75	17.14	3	62
	20	8.85	6.633	3	28
	3 obstacles, 5 robots				
	min reward	mean iter.	st. dev.	min iter.	max iter.
exp. 3	0.9	77.55	78.52	13	299
	0.8	19.8	10.75	5	47
	0.5	2.9	1.21	1	6

Table 2
Results from the quantum circuit (Algorithm 2)

	0 obstacles, min. reward 0.8				
	No. robots	mean iter.	st. dev.	min iter.	max iter.
exp. 1	5	10.71	9.01	1	33
	10	8.9	8.18	1	30
	20	13.62	8.96	2	32
	5 robots, min. reward 0.8				
	No. obstacles	mean iter.	st. dev.	min iter.	max iter.
exp. 2	5	6.05	5.72	1	21
	5	6.05	5.72	1	21
	10	11.29	9.48	1	33
	20	15.57	15.63	1	51
	3 obstacles, 5 robots				
	min reward	mean iter.	st. dev.	min iter.	max iter.
exp. 3	0.85	28.14	21.62	4	94
	0.8	7.86	6.62	1	26
	0.5	1.14	0.48	1	3

Testing the quantum circuit. Let us now describe the results obtained with the quantum circuit (Pseudocode 2). Rather than a forced reshuffle, the program here predicts the location of the target using the quantum circuit, forcing the robots to be scattered in a small neighbor of the circuit's outcome. The results of our three experiments are presented in Table 2.

In the first experiment we change the number of robots, maintaining fixed the other parameters. Contrary to what happened for the forced loop, adding more robots here does not increase the number of required iterations. Thus, we deduce that the scalability of the method is better while using the quantum circuit. In the second experiment, we only change the number of obstacles and their position. In this case, adding obstacles makes the target search less easy. The circuit outcome is precise only when one of the robots is able to come very close to the target. In the third experiment, we only change the minimum reward. Because the precision of 0.9 would require a great number of iterations and a larger time to perform, we choose instead 0.85. As for the third experiment of the forced loop, asking for a too-high minimum reward requires a higher number of iterations.

We notice that the quantum circuit outcome does not get us directly to the exact location of the target; instead, it seems to help find it, letting the robot reach the target. After some tests, the method seems to work greatly for a larger swarm.

Accuracy and trajectories. Finally, we considered accuracy and control. The simulations in Webots can create a semi-realistic environment by managing congestion, collision, and motion. While moving from a point to another one, each robot needs to take into account physical constrains such as the rotational inertia and translational motion. Also, to make the simulation more real, each robot needs to acknowledge the new position, get aligned to the trajectory, and go to the spot the more precisely as possible. The robot tries to turn and converge to the required position, but it can be only approximately on the suggested spot, due to its own inertia and imperfection of the arena. Concerning the implementation, we can either simulate the key movement along the trajectory, or simulate the whole trajectory.

4. Discussion and Conclusions

We started from a recently-proposed quantum approach for pairwise interactions in a robotic swarm in a search-and-rescue mission. We rewrote the code as a Python file, and connected it to Webots robotic simulator, to acquire information on reward directly from the sensors of the robots, coded in C. To test the advantage of this approach, we performed a comparison with a strategy based on forced position reshuffle steps. We found that the application of the quantum procedure leads to precise results with a minor number of iterations, confirming the advantage given by quantum computing in this framework. In the current model, it is not possible to dynamically change the target, e.g., with a moving object. However, this enhancement can be introduced in further versions of the code.

If the "most updated" robot of the swarm (that is, the robot that is more precisely following the directions obtained through the logic gate, or that shows a higher proximity to the target) occurs in a failure, e.g., the battery has run out, to a certain extent, the system is supposed to spontaneously correct itself. In fact, let us imagine that the robots reach the position suggested by the gate, which had however been influenced by the wrong indications of a faulty robot. If the new position is not closer to the target, the other robots will figure this out according to their new sensory measurements. Then, the decision-making cycle would start again, with the information from the (new) successful robot as the new input. However, further mechanisms of improvement and automatic correction in case of faulty robots could be developed.

As drawback, the connection between the quantum method and the Webots platform is still quite complex, needing multi-tasking to perform without error. Next research will lead to a centralization inside the same file of both quantum and robotic-simulation parts, shortening the overall code. A better communication map between the two programs can also be established. Next research may also lead to a specific communication program, considering its complexity management and relevance for the simulation structure.

The definition of new algorithms exploring and exploiting the efficiency and novelty of quantum computing can lead to future new robotic applications in search-and-rescue missions. These applications would not only enhance technology, from energy-saving issues to quantum advancements, but, in the framework of disaster and rescues, can potentially help save lives.

Availability of materials

The original quantum computing codes in Jupyter, and former Webots simulations (by M.M.), can be accessed at https://github.com/medusamedusa/10_little_ants. The new Webots and Python codes (by A.T.) can be accessed at https://github.com/AlexandreThomasKL/Webot_repertory/tree/main.

References

[1] Abraham, L., Biju, S., Biju, F., Jose, J., Kalantri, R., Rajguru, S.: Swarm Robotics in Disaster Management. International Conference on Innovative Sustainable Computational Technologies (CISCT) 1-5 (2019)

[2] Alkilabi, M., Narayan, A. & Tuci, E. Cooperative object transport with a swarm of e-puck robots: robustness and scalability of evolved collective strategies. *Swarm Intelligence*. **11** pp. 185-209 (2017)

[3] Atchade-Adelomou, P., Alonso-Linaje, P., Albo-Canals, J. & Casado-Fauli, D. qRobot: A Quantum Computing Approach in Mobile Robot Order Picking and Batching Problem Solver Optimization. *Algorithms*. **14** (2021), https://www.mdpi.com/1999-4893/14/7/194

[4] Bell, J. On the Einstein Podolsky Rosen Paradox. *Physics*. **1**, 195-200 (1964), https://journals.aps.org/ppf/pdf/10.1103/PhysicsPhysiqueFizika.1.195

[5] Benioff, P. Quantum robots and environments. *Physical Review A*. **58** pp. 893 (1998)

[6] Berman, S., Lindsey, Q., Sakar, M., Kumar, V. & Pratt, S. Experimental Study and Modeling of Group Retrieval in Ants as an Approach to Collective Transport in Swarm Robotic Systems. *Proceedings Of The IEEE*. **99**, 1470-1481 (2011)

[7] Busnel, Y., Caillouet, C., Coudert, D.: Self-Organized Disaster Management System by Distributed Deployment of Connected UAVs. 2019 International Conference on Information and Communication Technologies for Disaster Management (ICT-DM), Paris, France (2019), 1-8

[8] Chella, A., Gaglio, S., Pilato, G., Vella, F. & Zammuto, S. A Quantum Planner for Robot Motion. *Mathematics*. **10** pp. 2475 (2022)

[9] F. De Rango, N. Palmieri, X.S. Yang *et al.*, "Swarm robotics in wireless distributed protocol design for coordinating robots involved in cooperative tasks," *Springer Soft Computing*, vol. 7, no. 4, pp. 4251–4266, 2018, doi: 10.1007/s00500-017-2819-9.

[10] Dong, D., Chen, C., Li, H. & Tarn, T. Quantum Reinforcement Learning. *IEEE Transactions On Systems Man And Cybernetics Part B (Cybernetics)*. **38** (2008), https://fr.art1lib.org/book/18461932/6b4359

[11] Dong, X. & Sitti, M. Controlling two-dimensional collective formation and cooperative behavior of magnetic microrobot swarms. *The International Journal Of Robotic Research*. **39**, eabe4385 (2020), https://journals.sagepub.com/doi/full/10.1177/0278364920903107

[12] Dong, D., Chen, C., Zhang, C. & Chen, C. Quantum robot: structure, algorithms and applications. *Robotica*. **4** pp. 513-521 (2006)

[13] P. Fazio, M. Mehic and M. Voznak, "On the Relationship Between Speed and Mobility

Sampling Frequency in Dynamic Urban Networks," in IEEE Systems Journal, vol. 17, no. 1, pp. 696-707, March 2023, doi: 10.1109/JSYST.2022.3186640.
[14] P. Fazio, M. Mehic, M. Voznak, "Effects of sampling frequency on node mobility prediction in dynamic networks: A spectral view," In Digital Communications and Networks, 2022, https://doi.org/10.1016/j.dcan.2022.05.008.
[15] Feynman, R., Gottlieb, M. & Pfeiffer, R. Quantum Behavior. *The Feynman Lectures On Physics*. (1965)
[16] Gómez, S., Díaz-Guilera, A., Gómez-Gardeñes, J., Pérez-Vicente, C., Moreno, Y. & Arenas, A. Diffusion Dynamics on Multiplex Networks. *Physical Review Letters*. **110**, 028701 (2013,1)
[17] Granell, C., Gómez, S. & Arenas, A. Competing spreading processes on multiplex networks: Awareness and epidemics. *Physical Review E*. **90**, 012808 (2014,7)
[18] Hamann, H. Swarm Robotics: A Formal Approach. (Springer,2018)
[19] Ivancevic, V. Entangled swarm intelligence: Quantum computation for swarm robotics. *Mathematics In Engineering, Science And Aerospace*. **7**, 441-451 (2016)
[20] Koukam, A., Abbas-Turki, A., Hilaire, V. & Ruichek, Y. Towards a Quantum Modeling Approach to Reactive Agents. *2021 IEEE International Conference On Quantum Computing And Engineering (QCE)*. (2021)
[21] Kwak, Y., Yun, W., Jung, S., Kim, J. & Kim, J. Introduction to Quantum Reinforcement Learning: Theory and PennyLane-based Implementation. *International Conference On Information And Communication Technology Convergence (ICTC)*. (2021)
[22] Lamata, L., Quadrelli, M., Silva, C., Kumar, P., Kanter, G., Ghazinejad, M. & Khoshnoud, F. Quantum Mechatronics. *Electronics*. **10** pp. 2483 (2021)
[23] Li, Z., Liu, W., Li-E, G. & Li, L. Path Planning Method for AUV Docking Based on Adaptive Quantum-Behaved Particle Swarm Optimization. *IEEE Access Multidisciplinary*. **7** pp. 78665-78674 (2019)
[24] Mannone, M., Seidita, V., Chella, A.: Categories, Quantum Computing, and Swarm Robotics: A Case Study. Mathematics, **3**(372), 2022, https://doi.org/10.3390/math10030372
[25] Mannone, M., Seidita, V., Chella, A.: Modeling and Designing a Robotic Swarm: a Quantum Computing Approach. Swarm and Evolutionary Computation, **79**(101297), 2023, https://doi.org/10.1016/j.swevo.2023.101297
[26] Mannone, M., Seidita, V., Chella, A., Giacometti, A., Fazio, P.: Energy and SNR-Aware Robotic Swarm Coordination for Aquatic Cleaning Operations. In: 97th IEEE Vehicular Technology Conference (VTC) 2023, in press, Florence, Italy
[27] Oung & D'Andrea, R. The distributed flight array. *Mechatronics*. **21** pp. 908-917 (2011)
[28] Rubenstein, M., Ahler, C., Hoff, N., Cabrera, A. & Nagpal, R. kilobot: a low cost robot with scalable operations designed for collective behavior. *Robotic Autonomous Systems*. **62** pp. 966-975 (2014)
[29] Schmickl & Crailsheim, K. Collective Perception in a Robot Swarm. (Springer,2007)
[30] Schranz, M., Umlauft, M., Sende, M. & Elmenreich, W. Swarm Robotic Behaviors and Current Applications. *Frontiers In Robotics And AI*. **7** (2020)
[31] Solé-Ribalta, A., Gómez, S. & Arenas, A. Congestion Induced by the Structure of Multiplex Networks. *Physical Review Letters*. **116**, 108701 (2016,3)
[32] Stolze, J. & Suter, D. Quantum Computing: A Short Course from Theory to Experiment. (Wiley,2004)

[33] Terzi, M., Anastasiou, A., Kolios, P., Panayiotou, C., Theocharides, T.: SWIFTERS: A Multi-UAV Platform for Disaster Management. 2019 International Conference on Information and Communication Technologies for Disaster Management (ICT-DM), Paris, France (2019), 1-7

[34] M. Tropea, N. Palmieri, F. De Rango, "Modeling the coordination of a multiple robots using nature inspired approaches," In Proceedings of *Communications in Computer and Information Science (CCIS)*, 2020, pp. 124–133, doi: 10.1007/978-3-030-45016-8_13.

[35] Wichert, A. Principles of Quantum Artificial Intelligence. (World Scientific,2020)

[36] Zhu, K. & Jiang, M. Quantum Artificial Fish Swarm Algorithm. *Proceedings Of The 8th World Congress On Intelligent Control And Automation.* (2010)