SETSCI conference proceedings
SETSCI Conference Proceedings
15 , 13-17, 2023

Cognitive Models and Artificial Intelligence Conference

26-28 October, 2023, Ankara, Türkiye

https://doi.org/10.36287/setsci.6.1.009

BMYZ 2023

Development of a Simulator for UAV Swarm Operation Planning using Evolutionary Algorithms

Metehan Aydın, Gazi Erkan Bostancı, Mehmet Serdar Güzel and Koray Açıcı

Department of Computer Engineering/Engineering Faculty, Ankara University, Ankara, Turkey
Department of Artificial Intelligence and Data Engineering/Engineering Faculty, Ankara University, Ankara, Turkey
eng.metehan.aydin@gmail.com, ebostanci@ankara.edu.tr, mguzel@ankara.edu.tr, kacici@ankara.edu.tr

Abstract – In latest years, UAVs are widely used in many areas. Their advantages attracts researches all around the world. When UAVs are wanted to be used effectively, swarm systems offer better and more advantageous solutions. Using a UAV swarm has some advantages over a single agent UAV but operation planning must be considered carefully. There are lots of operation planning algorithms in literature. If these algorithms are evaluated for their effectiveness, evolutionary algorithms comes to the fore. Operation planning algorithms and UAV swarms are generally tested in simulation environments. There are lots of simulation environments and tools. They can handle simulations in many areas ranging from aviation to automobile. However, they do not offer any features dedicated for UAV swarms. In this work, a simulation software is designed for UAV swarms. The designed simulation software can plan an operation for an UAV swarm using evolutionary algorithms and simulate them. In this paper, the designed software and its architecture is explained in details.

Keywords – UAV swarm, simulation, operation planning, evolutionary algorithms, genetic algorithms

I. Introduction

An unmanned aerial vehicle (UAV) is a vehicle which can fly without a pilot or a crew member on it. UAVs are being widely used in these days. The main reasons behind their growing usages are the advantages [1] they offer. The main advantages of UAVs are safety and their prices. UAVs are very capable vehicles. However, there are some cases where they are wanted to be more capable. In such cases, UAV swarms are used.

Swarm systems is a field in multi-agent systems and deal with the coordination of multiple agents [2]. Its purpose is that instead of using a complex agent, coordinating relatively simple agents to complete a task. The behaviors of the social insects are taken to model the coordination of the agents in the swarm and these behaviors give some abilities to swarm systems. [3]. These abilities are robustness, flexibility and scalability. Thanks to robustness, swarm systems can complete a task despite of failures on an environment or agents. Swarm systems can generate effective solutions for a range of problems with flexibility. Scalability gives the ability of increasing or decreasing number of agents in a swarm. When an UAV swarm is formed, these abilities can be beneficial and make UAVs more operational [4]. However, in order to complete an operation successfully, operation planning must be considered carefully [5]. When the latest developments in the literature are examined, it can be seen that evolutionary computational based operation planning systems are widely used for their efficiency and flexibility advantages.

Evolutionary computation solves problems by mimicking evolution steps in nature [6] [7]. Mimicking a natural process gives some abilities [8][9] to evolutionary computation. These abilities are as follows.

- Flexibility: Evolutionary computation algorithms can be applied to different kinds of problems.
- Dynamic: Evolutionary computation algorithms can be used for dynamic problems.
- Distributed: Evolutionary computation algorithms can be operated without being dependent on a central authority or decision maker.
- Autonomous: Evolutionary computation algorithms can work without human interaction.

An evolutionary computation algorithm generally consists of four steps [10] [11]. These steps are initialization, selection, genetic operation and termination. Genetic operation is composed of two steps; crossover and mutation. The order of these steps is shown in Figure 1.

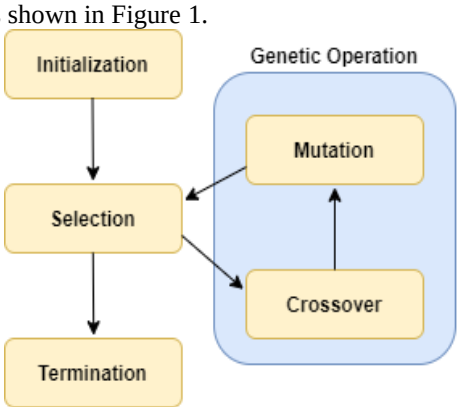

Fig. 1 Order of steps in an evolutionary computation algorithm

The algorithm starts with initialization step. In initialization, a solution set called population is created. Each solution in population is called chromosome. Then, the algorithm passes to selection step. In selection step, there is a function called fitness. The function evaluates the population and chooses the best chromosomes. Then, crossover step is started. In this step, two chromosomes are combined into one chromosome. After crossover step, the algorithm passes to mutation step. In mutation step, some chromosome values are altered. Then, the algorithm passes to selection state again. The cycle for

selection, mutation and crossover is repeated until a desired solution is obtained [12] [13].

When a swarm system is established, it is hard to realize its algorithms in real life [14]. Because, there are many agents in a swarm system. Establishing a swarm then testing its algorithms in real life is a challenging task. For these reasons, swarm system researchers most of the time work with simulators. They simulate a swarm system and its algorithms in a simulation environment.

Swarm systems are generally simulated in tools [15] like MATLAB [16] or SCADE Suit [17]. These kind of tools are good for simulation. They have very advanced simulation capabilities. But, they are developed for multi-purpose usages. Almost anything can be simulated using these kind of tools. However; in the case of simulating a swarm system, they do not offer any dedicated features or abilities. Thus, it is tedious to set up a swarm system simulation using these kind of platforms. There are also some tools which are dedicated for swarm system simulation but they are not for UAV swarms. Besides them, most of the simulation tools are depended on a single operating system. This situation restricts the area of use for the tools. In addition, the situation also makes it difficult to install these tools.

In this work, a simulation tool dedicated for UAV swarms are designed. The purpose of the tool is planning and simulating swarm operations. The designed tool can work in different operating system. The supported operation systems are Linux, Windows and MacOS. In this paper, we explain how the simulation tool is designed in details. The rest of the paper is structured as follows. In section 2, materials and method of the work is addressed. Section 3 presents some experiments and their results and the conclusion of the work is given in Section 4.

II. Materials and Method

The software is developed using C++ Programming Language and Object Oriented Programming paradigm. Abilities of the software are as follows.

- Operation planning of a single UAV.
- Operation planning of a UAV swarm.
- Adding a new UAV model.
- Defining a UAV swarm that has homogenous architecture.
- Defining a UAV swarm that has heterogeneous architecture.
- Adding a new threat type.
- Defining an operation that has one type of threat.
- Defining an operation that has multiple types of threats.
- Adding a map to simulation.
- Applying different kind of evolutionary computation algorithms for operation planning.

The developed software has graphical user interface for easy use. Users can set up simulation environments, swarm systems and operations using the graphical interface. Qt Framework [18] is used in order implement the graphical interface and some of the abilities mentioned above.

Qt Framework is an application development framework that offers developers to design cross-platform applications with graphical user interfaces. Qt has lots of libraries for C++ programming languages ranging from 2D rendering to networking and more. These features allow developers to create robust applications for various areas.

The architecture of the developed software has four main infrastructures. These infrastructures are as follows.

- Graphical user interface
- Software manager
- Object creation infrastructure
- Operation planning infrastructure
- Simulation infrastructure

Figure 2 shows relations of these infrastructures.

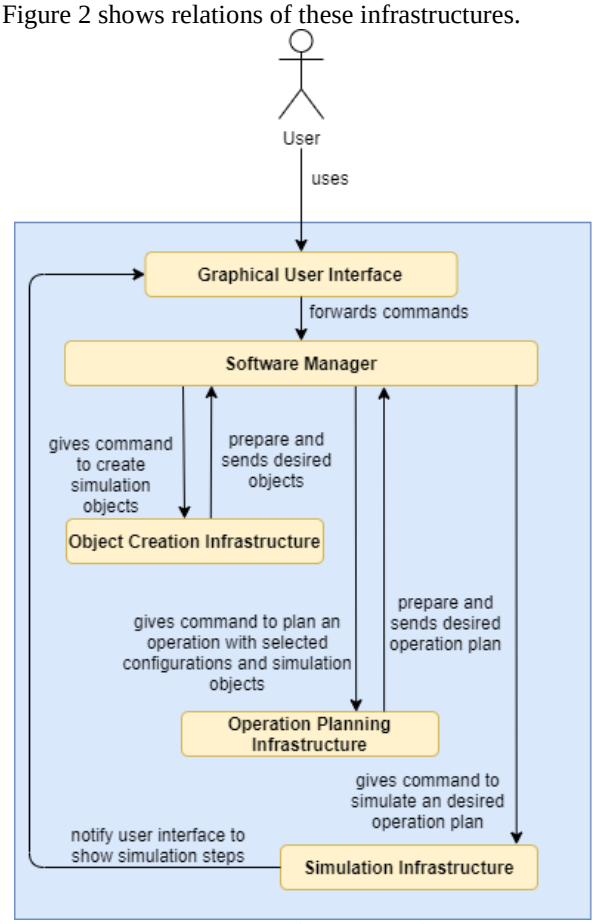

Fig. 2 Relationships among the infrastructures of designed simulation software

As it can be seen in the figure, a user can manage all his processes via graphical user interface. The graphical user interface forwards all the commands to software manager. Users can control the software using buttons on the user interface.

Software manager controls the infrastructures and provides collaboration among them. It handles all the functionality the software has and contains all the software data including UAVs, threats and maps. All buttons in graphical user interface triggers a process in software manager. These processes manage desired commands to help preparing a simulation environment and simulate an operation. Thanks to software manager, modifications to the tool can be done easily and new features can be added without any changes to existing codes. Thus, software manager makes the tool flexible and robust.

Objects creation infrastructure gets commands from simulation manager. It is responsible for creating UAV, threat and map instances. When user wants to create these objects, he orders them via graphical user interface. Graphical user interface notifies object creation infrastructure via software

manager. Classes are defined for representing UAVs, threats and map. These classes implements prototype design pattern. Object creation infrastructure uses prototype design pattern embedded in each classes to create objects. When the objects are created, they are sent to software manager to store. Object creation infrastructure can create all kind of objects. In case of adding creating new types of objects, its implementation can be added easily.

When an operation wanted to be planned, operation planning infrastructure is executed. Its purpose is planning desired operations using desired algorithms. It takes an operation definition as input from software manager and create an operation plan as output to software manager. An operation definition includes properties of UAV swarm, properties of threats, properties of operation and the map. Properties of UAV swarm defines there are how many agents in the swarm and models of each agents. Operation planning infrastructure is designed to plan operation for any size of swarm. So, number of agents in the swarm can be 1 or N. Properties of threats addresses how many threats are there in the operation and what is the type of each threats. Operation planning infrastructure can handle any number and any types of threats. There can be 0 or N number of threats in the operation. Properties of an operation describes operation location and kind of operation. Operations can be located anywhere in the map. Operation kinds can be surveillance and reconnaissance. The map is set using graphical user interface and stored in software manager. In case of planning an operation, a map is also needed. Operation planning infrastructure is designed to handle any maps. Operation definition is processed to create an operation plan. The process is managed by the algorithm applied. Thanks to modular design of operation planning infrastructure, any kind of operation planning algorithm can be implemented or any kind of 3rd party libraries can be added. Operation planning infrastructure can also be moved to another software. In this work, an evolutionary computation algorithm is implemented in the tool. The algorithm outputs a planned operation. The planned operation is taken by operation planning infrastructure and sent to software manager.

Simulation infrastructure manages simulations. When user starts or stop simulation via graphical user interface, the commands are forwarded to simulation manager. Simulation manger executes simulation infrastructure. There is an interface named SimItem in the software. All the objects which can be simulated must implement SımItem interface. The interface provides simulation capabilities. Simulation infrastructure uses SımItem interface to simulate objects. The behaviours of objects are defined in functions provided by the interface. The objects are changed during simulation with respect to these functions. Simulation is executed step by step. In each step, simulated objects are changed and changes are reflected to screen by notifying graphical user interface. Thanks to simulation infrastructure's modular design, new simulation capabilities can be added or existing ones can be modified easily.

The infrastructures of the software provides all the functionalities for the software. Their designs allow new functionalities to be added with ease. Following figure has a use case diagram which shows the functionalities of the software.

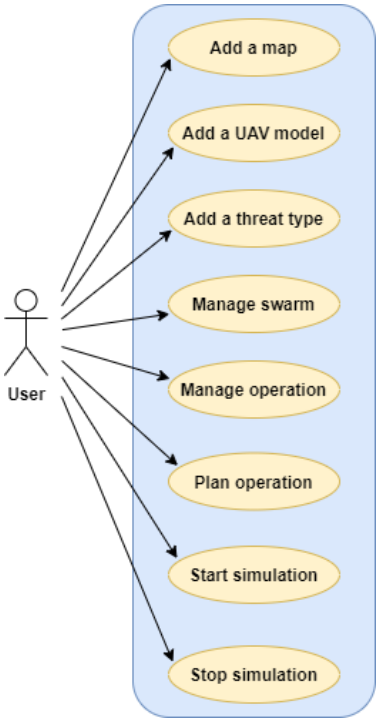

Fig. 3 Functionalities of the software

When a user wants to add a map, there is a class whose name is Map. A map is represented with Map class. An instance of Map class is created via object creation infrastructure. The instance is used for representing the selected map in the software.

A new UAV model is added to software with entering a model name and a color. Color is used to represent a UAV in the map. There is a class in the software whose name is UAV. The class inherits SimItem and represents an UAV. When the user enters a UAV model and a color, an UAV class instance is created via object creation infrastructure and stored in software manager.

Type, color, width and height must be entered when a new threat type is wanted to be added. Type field stores type of a threat. Color field is for representing a threat in the map. Width and height fields define size of a threat. A threat is represented in the software with a class whose name is Threat. Threat class also inherits with SimItem class. When the user enters the fields, a Threat class instance is created via object creation infrastructure and stored in software manager.

Manage swarm functionality is provided to determine number of UAVs for each UAV model and number of threats for each threat types. When the user enters desired number of threats and UAVs, object creation infrastructure creates the instances and sends them to software manager to store.

In the software, operation properties are managed via a class whose name is Operation. Operation class represents an operation. There is an Operation class instance which is created default. Properties of an operation are type, width and height. Type property indicates type of the operation. There are two types of operation in the software. These are surveillance and reconnaissance. One of them can be selected. Width and height properties define the size of the operation. When user wants to manage an operation, he modifies the instance.

Plan operation functionality is triggered to plan an operation. Software manager creates an operation definition and sends it to operation planning infrastructure. Operation

planning infrastructure plans the operation and sends it to again software manager to store. The stored operation can be simulated any time.

In order to manage simulation functionalities, commands are forwarded to software manager. Software manager controls simulation manager to start or stop the simulation.

Figure 4 shows a planned operation.

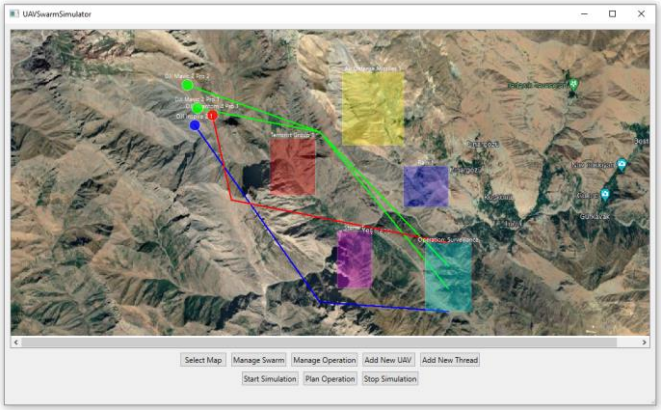

Fig. 4 A planned operation

III. RESULTS

Two experiments are done to prove that our software design can work effectively with different algorithms. The algorithms used are NSGA-2 [19] and MOEA/D [20].

Experiments are executed for two agents and two objectives. The objectives are shortest path and safest path. Thus, the operations are planned for the path which is safest and shortest. Fitness functions of algorithms are same and as follows.

Let D be distance between source and destination points.

$$D = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \quad (1)$$

Let P be penalty value.

$$P = \begin{cases} 0, & if\ the\ path\ doesn't\ have\ a\ threat \\ 100, & if\ the\ path\ has\ a\ threat \end{cases} \quad (2)$$

Let F be fitness function.

$$F = D + P \quad (3)$$

Both of the algorithms are run with following configurations. Bit-flip mutation is selected as mutation method. Binary random sampling is used as sampling method. Two-point crossover is chosen for crossover method. Figure 5 shows test case for the experiments.

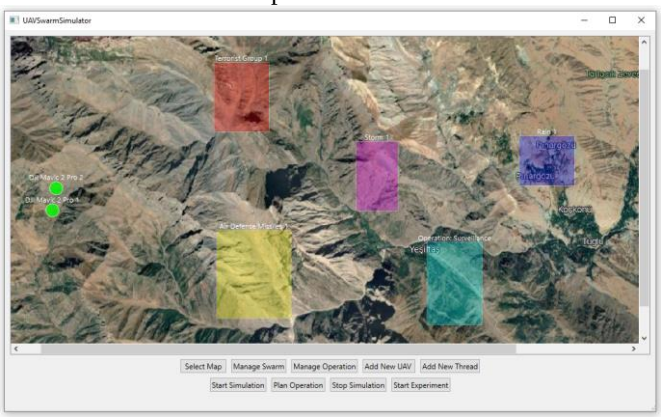

Fig. 5 Test case fort he experiments

First, MOEA/D algorithm is tested. The test is completed in 7430ms. Figure 6 shows best and average fitness/generation graph for first and second agents.

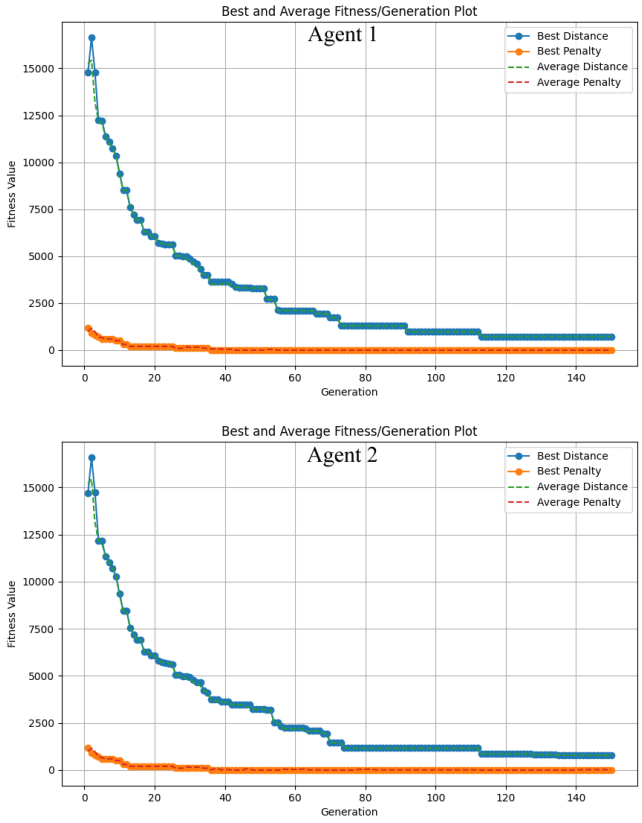

Fig. 6 Fitness plots for agents using MOEA/D

Then, NSGA-2 algorithm is tested. The test takes 13955ms. Figure 7 shows best and average fitness/generation graph for first and second agents.

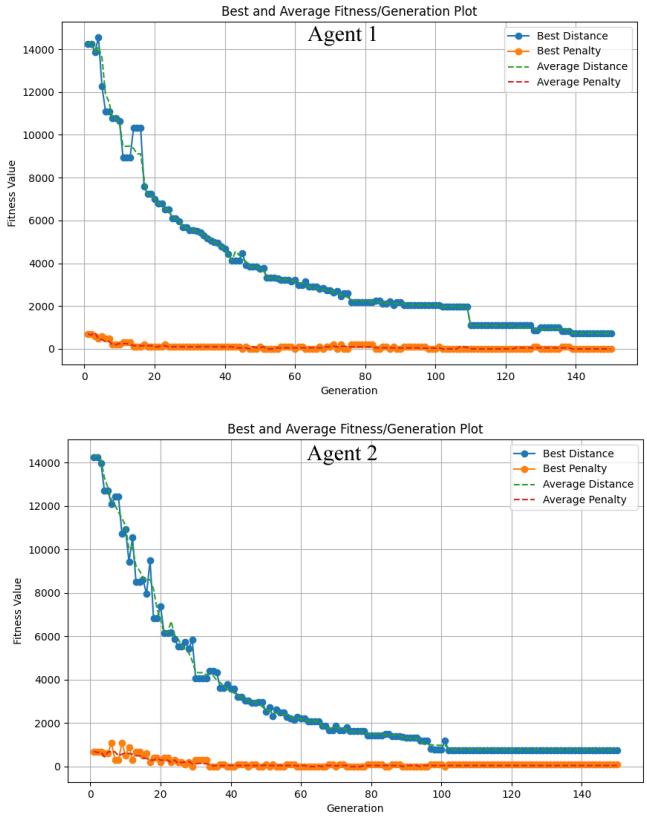

Fig. 7 Fitness plots for agents using NSGA-2

As it can be seen in plots, when both of the algorithms were executed, their solutions initially were not effective. The path distances were too long and their penalty values were too high. However, as the algorithms proceeds, distances are getting shorter and penalty values are getting lower. Both of the algorithms give optimal solutions at the end of the execution. Whereas NSGA-2 gives the optimum result after 100^{th} generations, MOED gives the optimum result after 140^{th} generations. Although MOEA/D completes the task earlier than NSGA-2, NSGA-2 reaches the optimal solutions quicker.

IV. CONCLUSION

UAVs are used in many areas. Increasing popularities of UAVs result for new researches. According to latest development in literature, UAV swarms can be a better way to use UAVs effectively. Using a swarm can be effective if operation planning algorithm is well designed. According to latest developments, evolutionary algorithms can be a good choice for UAV swarms.

Swarm researches test their algorithms mostly in simulators. There are lots of simulation tools but most of them offers multi-purpose solutions. In this work, a simulation tool is designed which can work with evolutionary algorithms. Architecture of the tool is presented and two experiments are made. Each of the experiments is made with same parameters. MOEA/D and NSGA-2 algorithms are selected for operation planning algorithm. Results of the experiments show that the developed tool can work with different algorithms effectively.

REFERENCES

[1] Baballe, M. A., Bello, M. I., Alkali, A. U., Abdulkadir, Z., Muhammad, A. S., & Muhammad, F. (2022). The Unmanned Aerial Vehicle (UAV): Its Impact and Challenges. Journal homepage: https://gjrpublication. com/gjrecs, 2(03).

[2] Iñaki Navarro and Fernando Matía, “An Introduction to Swarm Robotics,” ISRN Robotics, vol. 2013, Article ID 608164, 10 pages, 2013.

[3] E. Şahin, “Swarm robotics: from sources of inspiration to domains of application,” in Swarm Robotics Workshop: State-of-the-Art Survey, E Şahin and W. Spears, Eds., Lecture Notes in Computer Science, no. 3342, pp. 10–20, Berlin, Germany, 2005.

[4] Cheraghi, Ahmad Reza, Sahdia Shahzad, and Kalman Graffi. "Past, present, and future of swarm robotics." Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys) Volume 3. Springer International Publishing, 2022.

[5] Sampedro, C., Bavle, H., Sanchez-Lopez, J. L., Fernández, R. A. S., Rodríguez-Ramos, A., Molina, M., & Campoy, P. (2016, June). A flexible and dynamic mission planning architecture for uav swarm coordination. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 355-363). IEEE.

[6] Whitley, Darrell. "An overview of evolutionary algorithms: practical issues and common pitfalls." Information and software technology 43.14 (2001): 817-831.

[7] Spears, William M., et al. "An overview of evolutionary computation." European conference on machine learning. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993.

[8] Fogel, David B. "The Advantages of Evolutionary Computation." Bcec (1997): 1-11.

[9] Back, Thomas, Ulrich Hammel, and H-P. Schwefel. "Evolutionary computation: Comments on the history and current state." IEEE transactions on Evolutionary Computation 1.1 (1997): 3-17.

[10] Mitchell, Melanie, and Charles E. Taylor. "Evolutionary computation: an overview." Annual Review of Ecology and Systematics 30.1 (1999): 593-616.

[11] Pena-Reyes, Carlos Andrés, and Moshe Sipper. "Evolutionary computation in medicine: an overview." Artificial Intelligence in Medicine 19.1 (2000): 1-23.

[12] Jones, Gareth. "Genetic and evolutionary algorithms." Encyclopedia of Computational Chemistry 2.1127-1136 (1998): 40.

[13] Tian, Ye, et al. "Evolutionary large-scale multi-objective optimization: A survey." *ACM Computing Surveys (CSUR)* 54.8 (2021): 1-34.

[14] Soria, Enrica, Fabrizio Schiano, and Dario Floreano. "SwarmLab: A MATLAB drone swarm simulator." 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020.

[15] Calderón-Arce, Cindy, Juan Carlos Brenes-Torres, and Rebeca Solis-Ortega. "Swarm robotics: Simulators, platforms and applications review." Computation 10.6 (2022): 80.

[16] (2023) MATLAB website [Online]. Available: https://www.mathworks.com/products/matlab.html

[17] (2023) SCADE Suit website [Online]. Available: https://www.ansys.com/products/embedded-software/ansys-scade-suite

[18] (2023) Qt Framework website [Online]. Available: https://www.qt.io/

[19] Deb, Kalyanmoy, et al. "A fast and elitist multiobjective genetic algorithm: NSGA-II." IEEE transactions on evolutionary computation 6.2 (2002): 182-197

[20] Zhang, Qingfu, and Hui Li. "MOEA/D: A multiobjective evolutionary algorithm based on decomposition." IEEE Transactions on evolutionary computation 11.6 (2007): 712-731.