PROBLEMY MECHATRONIKI
UZBROJENIE, LOTNICTWO, INŻYNIERIA BEZPIECZEŃSTWA

ISSN 2081-5891

12, 3 (45), 2021, 53-70

PROBLEMS OF MECHATRONICS
ARMAMENT, AVIATION, SAFETY ENGINEERING

Swarm Behaviour Optimisation Methods Based on an Original Algorithm

Krzysztof FALKOWSKI*, Michał DUDA

Military University of Technology,
Faculty of Mechatronics, Armament and Aerospace
2 Sylwestra Kaliskiego Str., 00-908 Warsaw, Poland
Corresponding author's e-mail address and ORCID:
krzysztof.falkowski@wat.edu.pl; https://orcid.org/0000-0002-0279-3791

Received: March 16, 2021; Revised: April 14, 2021; Accepted: May 18, 2021 /
Published: September 30, 2021

DOI 10.5604/01.3001.0015.2428

Abstract. This article presents an authorial swarm algorithm that performs coverage tasks using the Sweep Coverage method. The presented solution assumes stochastic movement of the objects in the swarm which allows them to be simple ones. Our goal was to find an optimal number of objects in the swarm. The main evaluated factors are time and energy consumption. Changing input data allowed us to designate different cases and to examine the influence of varying parameters of a single boid on a whole swarm behaviour.
Keywords: swarms, swarm algorithm, sweep coverage, coverage task, optimisation

1. INTRODUCTION

Swarm Systems are more commonly used in practical applications, in which high complexity requires optimal solutions. Therefore scientists are searching for new Swarm Algorithms which would allow performing the task more and more efficiently. Such solutions are inspired by nature where herds of animals can move in order, although they consist of dozens of individual organisms. Insects, such as ants or bees, are the second group of biological exemplars. Their swarm organization and control are phenomenal compared to the simplicity of a single component. The resultant output of the whole swarm depends on local interactions between system components and could not be achieved by a single object acting alone [1, 2].

Simulations of Swarms are used in virtual areas such as animations or artificial intelligence. The first computer model of moving animal herd was proposed by Reynolds in 1987 and it was called the Boid Algorithm. Reynolds modelled the behaviour of moving bird flock or fish shoal by relatively simple interactions between single simulated objects called boids [3]. The base of this algorithm were three simple rules [4]:

- separation – each object moves to try to avoid creating local concentrations, so it keeps its distance from all of its neighbours;
- alignment – each object tries to follow its neighbours which allow the swarm to move in the same direction at the same speed;
- cohesion – each object moves towards a local centre of mass.

A research area of high interest are coverage tasks. Whether by calculating the best configuration of highly numerous swarms over the area or planning the best path for a smaller group, the goal is to cover the biggest possible fraction of the task area. This is used in a wide span of practical applications from military monitoring [5] to floor cleaning [6]. Two basic types of coverage are Blanket coverage and Sweep coverage [7].

Blanket Coverage goal is to provide a static arrangement of boids over the area. It provides good results even assuming a random distribution of the swarm [8] and as it is shown by Wang [9] this method can be significantly improved when combined with a suitable algorithm.

Although being a very accurate method, Blanket Coverage requires a swarm count sufficient to cover the whole area with boids in a certain relationship. This might be a serious drawback as the bigger the research area the more drones are required.

Sweep Coverage is a method in which elements move over the research area and the goal is to pass over the most possible points of it. It can be achieved with organized algorithms such as these presented by Miao [10] or by foreplanning the path either for a single object[6] or for a whole swarm [11].

Sweep Coverage uses fewer boids than Blanket Coverage to perform the task but this comes with other requirements. To foreplan the path, data over the whole research area needs to be known or it should be possible to share and store such information between boids [5]. Either this or the sheer complexity of the movement algorithm requires greater computing power or more advanced objects.

In our project, we combined simple boids with a Sweep Coverage method. We assumed that boids move randomly and might only stop when they come too close to another boid (collision avoidance) or they return if they cross the area boundary. The shape of the research area is the only information given over it.

The purpose of this article is to discuss the method of finding optimal, energy-efficient solutions in a swarm algorithm based on an authorial project. Assuming that every boid in the swarm consumes a certain amount of energy in every unit of time, the goal is to designate a perfect number of objects in the group, which allows performing the task in the shortest possible time, consuming the least energy. The article also describes the impact of changing test variables on the result.

2. PROJECT OBJECTIVES AND ASSUMPTIONS

Swarm Algorithm solutions are used in many research areas such as robotics, telecommunications, transport, etc. One such field is military application [12]. Swarm of UAVs could search a certain area for mines or other dangerous objects. In civil-military applications, a group of drones could search for victims and perform search and rescue missions.

This work is an introduction to the authorial solution of virtual, discrete-time Swarm Algorithm used for searching a certain area, providing collision avoidance feature that would guarantee no impacts between objects, regardless of their number. To specify the task, the following assumptions have been set:

- the swarm aims to search a certain area in the shortest possible time and consuming the least possible energy. According to this, a solution would be a compromise between the amount of energy consumed and the time needed to finish the task,
- the way that searching is performed cannot be an orderly way. A single boid is a simple object and its movement is of stochastic nature. It moves in random directions and it could only control its velocity,
- simulation is executed in two dimensions and the target area is plain,
- single object knows the position of each boid and the distance to them. It can also react if it moves too close to another object.

In this article all distances are presented without a unit as at this stage the algorithm does not represent any physical system. Also, velocity is given in this virtual distance per iteration.

3. SWARM ALGORITHM DESCRIPTION

Taking Boids Algorithm as an example, research has been made to find inspiration in nature. Studies of different swarming insects showed that the behaviour of flies running on the table matches the assumptions posted above. Each fly moves independently and their movement seems to be random, yet they can avoid collisions. In a situation when two insects would hit each other, one of them simply stops letting the other move further [13].

The developed algorithm allows simulating a research area (Fig. 3.1.a), which is a circular plain of a given radius. The centre of the circle is also an origin of the coordinate system. Furthermore, the research area is divided into control points with the certain resolution $\Delta\mathbf{a}$. The only purpose of these points is to allow determining what part of the area has been searched. The movement of a single boid is independent of them.

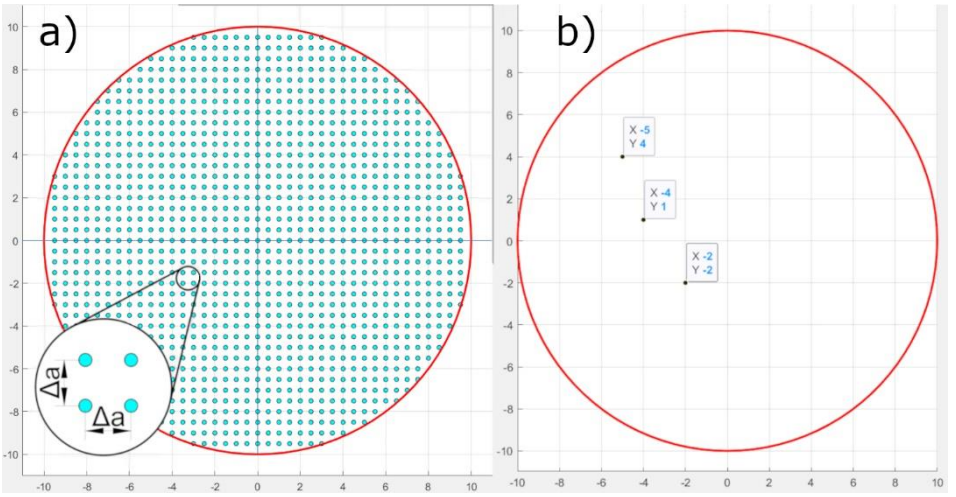

Fig. 3.1.a) Exemplary research area (Radius R=10) divided into control points with the resolution Δa=0.5, b) Set up of three exemplar boids with their starting coordinates.

Any number of virtual objects can be subsequently be added to the research area. They are represented by the points with XY coordinates which specify their position on the surface (Fig 3.1.b) to the origin. When boids are set up, the random heading angle $\boldsymbol{\alpha_1 i}$ from 0 to 360 degrees is assigned to each of them. A different angle is drawn for each boid separately.

A simulation is divided into iterations. In each iteration, every object takes a step at an assigned angle and with a given velocity. Before making a move the angle of each boid is changed by a different, random value $\boldsymbol{\Delta\alpha_{ij}}$.

The number is drawn according to a normal distribution between -90 to 90 degrees. Resultantly the greater the angle change the less possible it is to appear. The probability of changing the angle by 90 degrees is close to zero. The movement of the object was shown in Fig. 3.2.

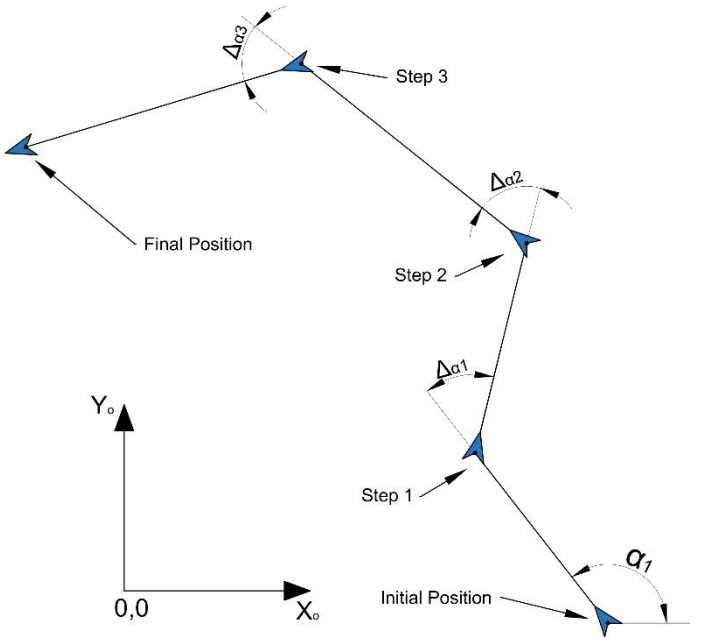

Fig 3.2. Movement of an object

Coordinates of the next position are calculated using the current location and the assigned angle updated by a random change. The position is calculated using the equations below:

$$X_{i+1} = X_i + V * \cos(\alpha_i + \Delta\alpha_{i+1}) \quad (1)$$

$$Y_{i+1} = Y_i + V * \sin(\alpha_i + \Delta\alpha_{i+1}) \quad (2)$$

- X, Y are the position coordinates of a boid,
- V is the velocity of a boid,
- α is the heading angle,
- $\Delta\alpha$ is the change of the heading angle,
- i, i+1 are the current and next iteration.

After randomising the angle change and calculation of new coordinates, two conditions are being checked before taking a step:

1) if the object goes outside the research area - in this case, the new angle change is drawn. Operation is repeated until the final position fits within the area,

2) if the boid steps closer than the safe distance to another object – in this case, that step is not taken. The boid skips this iteration and it tries to move again at the next one. This was shown in Fig. 3.3.

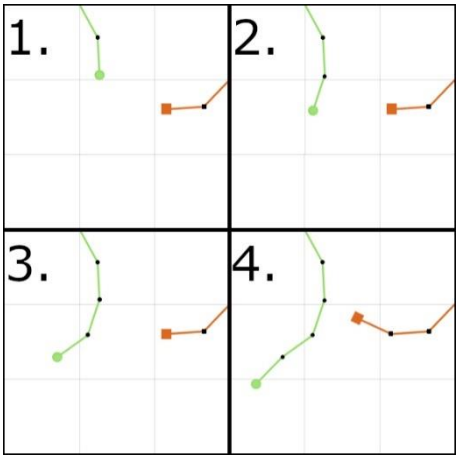

Fig. 3.3. Interaction between two boids shown in four subsequent steps. The rectangular, orange boid skips its movement until the circular, green one passes

In each iteration, an area of a given radius is scanned around every boid. Every control point within that distance is marked out. An example of this operation is shown in Fig 3.4. where two subsequent steps are shown - blue rectangular markers represent unchecked points and red, circular are the scanned ones. The biggest green circle shows the area scanned around the boid. Simulation controls what part of these points was already checked and stops if it exceeds the assumed fraction. The number of iterations needed to complete the task is then saved.

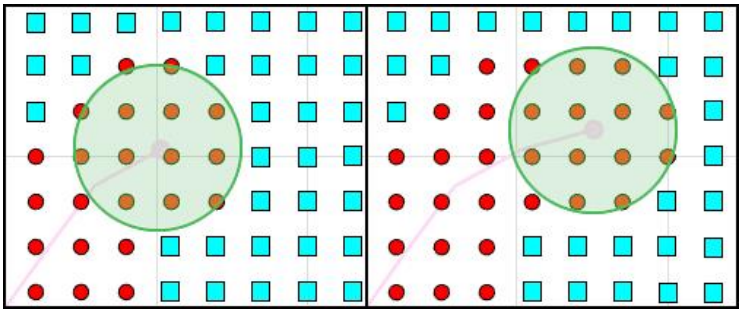

Fig. 3.4. Scanning of surrounding control points

4. TESTS

After developing the algorithm series of tests were taken. The first goal was to evaluate the number of iterations needed to search the testing area depending on the number of boids used for simulation. Furthermore, the number of emergency stops, caused by safe distance violation was counted. By knowing the values above, one can calculate the energy required to perform the task and energy-effectiveness of the swarm.

To designate energy requirement, an assumption was made that in every iteration, every boid in simulation consumes a certain amount of Energy. For this consideration, this energy was measured in virtual units that had no physical equivalent. The total amount of energy, required to finish the task, was calculated by multiplying iterations by the number of boids in the attempt.

$$E_t = Number\ of\ boids\ \cdot Iterations \tag{3}$$

According to the assumption above, the boid consumes the energy even if it is not moving. That case counts towards total energy but it does not contribute to goal achievement. This means, that stopping is energy-inefficient. The used total energy was compared to the total number of collision avoidance stops, which allowed us to see what fraction of moves was prohibited. Energy-effectiveness shows what part of the total energy was consumed for actual steps.

$$\varphi_E = 100\% - \frac{S_E}{Et} \tag{4}$$

where S_E is the number of safe distance violation stops.

To evaluate the best compromise between time and energy-effectiveness, the additional rate μ_{TE} has been introduced. This value equals the number of iterations divided by energy-effectiveness. This ratio shows what number of boids is the best when the most important is to perform the task in the shortest possible time for assumed the lowest energy consumption. The lower is this ratio, the better compromise it represents.

$$\mu_{TE} = \frac{I_{AVG}}{\varphi_E} \tag{5}$$

where I_{AVG} is the average number of iterations required to perform the task.

Due to randomness of movement, the results of single tests had a very a large dispersion. Therefore, a series of 10 tests were performed for each group of boids to draw the average results. Furthermore, for low numerous swarms, an examination of 100% of the research area was disproportionally time-consuming, so the target percentage of the scanned area was set to 90%.

At this stage of the project, bodis were only virtual objects, so they did not have a physical form. It means that dimensions or movement constants were not known. In this case, for testing an algorithm, all parameters were set up as a dependent value to the radius R of the research area. Three variables were specifying the behaviour of a single boid and a swarm as a whole:

- velocity of a single object (distance moved in one iteration):

$$V = \psi \cdot R \tag{6}$$

- safe distance:

$$d_{safe} = \vartheta \cdot R \tag{7}$$

- radius of an area scanned during one step:

$$r_{scan} = \xi \cdot R \tag{8}$$

For each of them, there was assigned the separate coefficient ψ, ϑ, and ξ. This allowed performing reference tests, which designated an impact of each of above variables on the results. The control points interval was set as the constant $\Delta a = 0.025R$. This parameter only impacts the accuracy, not the results themselves.

The main tests were carried out for $\psi, \vartheta, \xi = 0.05$ and groups consisting of 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 25, 50 and 100 boids. In the remaining tests, one of the ψ, ϑ, ξ factors was changed sequentially in a given range to designate the influence of the change on results. As these tests were only comparative ones, they were carried out for the groups of 1, 5, 10, 50, 100 boids. This allowed seeing the character of the change.

5. ANALYSIS OF RESULTS

The results (Tab.5.1 and Fig.5.1) show that the number of iterations needed to perform the task, decrease exponentially with an increasing number of boids in the test. In the groups of 1 to 10 boids, additional objects in the swarm cause a significant reduction of steps required to finish the task.

Further increase in the size of the group still causes a decrease in the length of the tests but differences are lesser. This is a result of increasing number of safe distance violation situations. The more objects in the research area, the greater chance they would cause a collision, and in this case they do not move.

Table. 5.1 Results

Number of boids in the test	1	2	3	4	5	6	7	8	9	10	15	25	50	100
Avarage number of iterations I_{avg}	804.6	402.1	292.6	203.8	187.5	138.4	117.6	113.4	97.8	84.4	58.5	36.2	25.2	19.3
Standard Deviation σ	175	73.1	96.2	31.7	71.3	25.5	29.2	24.7	19.4	18.8	9.4	5.5	3.4	3.5
Avarage number of Collision Avoidance Stops S_E	0	5.1	5.1	11.2	13.1	18.4	17.5	31.7	40.1	44.1	78.7	142.4	394.2	1329.1
Energy-Effectiveness φ_E	100%	99%	99%	99%	99%	98%	98%	97%	95%	95%	91%	84%	69%	31%
Time to Energy-Effectiveness Ratio μ_{TE}	804.6	404.6	294.2	206.5	190.1	141.5	120.1	117.5	102.5	89.1	64.2	42.9	36.6	62.2

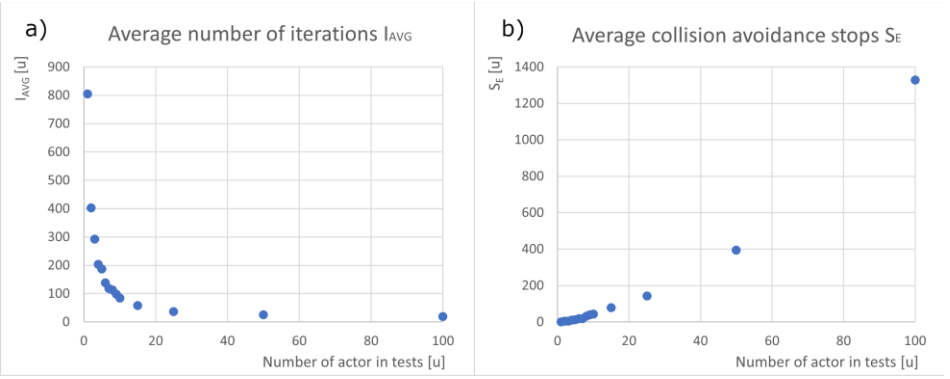

Fig.5.1. Dependence of Average number of iterations (a) and Average number of collision avoidance stops (b) on a number of boids in the group

Due to this, the energy-effectiveness continuously decreases with increase in size of the group as shown in Fig. 5.2.a. For the biggest group, even though the task was performed relatively quickly, the effectiveness was significantly smaller. For a single boid group, the effectiveness is always 100% as there is no other object to collide but the average time of the test is incomparably higher. The shortest time, combined with the energy-effectiveness of over 90%, shows for a group of 15 boids. As shown in Fig.5.2.b, the best μ_{TE} ratio is for a swarm of 50 boids and for rising beyond that point. This concludes that there can be found a minimum of this rate. Depending on what requirements are set for the task, either for the minimal energy-effectiveness or the most viable combination of time and energy consumption, there can be found the best number of boids to perform the task.

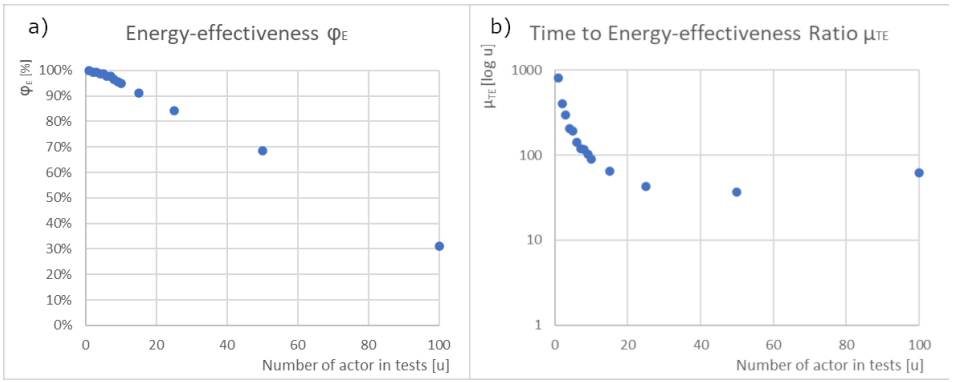

Fig.5.2. Dependence of Energy-effectiveness (a) and Time to Energy-effectiveness Ratio (b) on the number of boids in the group.

Furthermore, as shown in Table 5.1, the largest standard deviation appears for the sample with one boid and it is decreasing with the increasing size of the group. The reason for this is that for smaller groups, and especially for single boid tests, the time to complete the task depends on a random distribution of the path. The fewer boids in the group, the better chance that the path follows an already searched area. In this case, the number of steps increases when the fraction of scanned area stays the same. For greater groups probability that any of the objects moves through the not scanned area is much greater.

After the main tests had been carried out, the secondary trials were performed to evaluate an impact on the results caused by changing the following variables: the velocity V, the safe distance $d_{safe,}$ and the radius of scanned area r_{scan}. Series of tests were carried out in which subsequently one of the coefficients ψ, ϑ, ξ was changed while others were constant. For every parameter, there were performed two additional series of tests to see the influence of lowering and increasing this variable.

At this stage, the algorithm did not check the path between step points of boids. There could appear situations in which the segment between these points crossed through the safe zone of another object. Due to this, a velocity of a single object could not be more than twice a Safe Distance, as this would allow one object to move directly over the other, straight through the center of the safe zone.

Firstly, varying of a Safe Distance has been checked for the following values of the coefficient ψ = [0.1; 0.05; 0.033]. The results were shown in Fig.5.3. and Table 5.2. The bigger safe distance, the more iterations the trials lasted and the differences are greater for more numerous groups. It also causes more Safe Distance Violations' Stops and what follows a drop of energy-effectiveness which for groups of 50 and 100 boids is below zero.

This means that in these cases boids mostly did not move as bigger safe zones made it almost impossible to take a safe step.

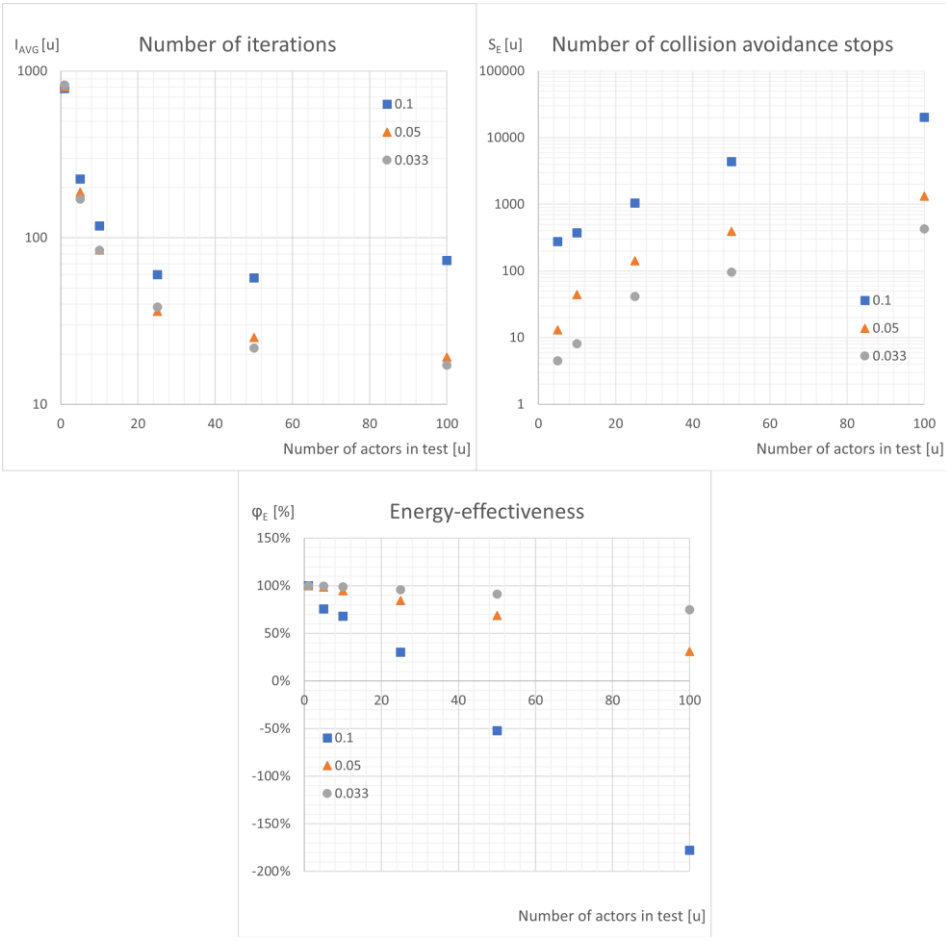

Fig.5.3. Influence of varying a safe distance on the results

Table 5.2. Influence of varying ψ on the results

Safe Distance $d_{safe} = \psi \cdot R$						
Average number of iterations						
ψ	**1**	**5**	**10**	**25**	**50**	**100**
0.1	785.6	224.6	117.8	60	57.4	73.2
0.05	805	188	84	36	25	19
0.033	828	171	84	38	22	17
Collision avoidance stops						
ψ	**1**	**5**	**10**	**25**	**50**	**100**
0.1	0	275	375.1	1044.4	4366	20320.7
0.05	0	13	44	142	394	1329
0.033	0	5	8	42	97	431
Energy-Effectiveness						
ψ	**1**	**5**	**10**	**25**	**50**	**100**
0.1	100%	75.5%	68.2%	30.4%	-52.1%	-177.6%
0.05	100%	98.6%	94.8%	84.2%	68.7%	31.0%
0.033	100%	99.5%	99.0%	95.7%	91.1%	75.0%

The second verification was carried out for a velocity and varying the coefficient y as follows ϑ = [0.025; 0.05; 0.067]. As it is shown in Fig.5.4 and Table 5.3, the higher velocity, the better performance and inversely, lower velocity causes worse task conduction. The changes for smaller groups are more significant than in the Safe Distance case and contrary the changes are lesser for more numerous groups.

Table 5.3. Influence of varying ϑ on the results

Velocity $V = \vartheta \cdot R$						
Average number of iterations						
ϑ	**1**	**5**	**10**	**25**	**50**	**100**
0.067	557.1	130.1	57.4	28.5	17	13.7
0.05	805	188	84	36	25	19
0.025	2289	500	230	99	66	28
Collision avoidance stops						
ϑ	**1**	**5**	**10**	**25**	**50**	**100**
0.067	0	17.1	15.9	78.2	178.5	879
0.05	0	13	44	142	394	1329
0.025	0	286	230	688	1395	5012
Energy-Effectiveness						
ψ	**1**	**5**	**10**	**25**	**50**	**100**
0.067	100%	97.4%	97.2%	89.0%	79.0%	35.8%
0.05	100%	98.6%	94.8%	84.2%	68.7%	31.0%
0.025	100%	88.6%	90.0%	72.1%	57.6%	13.4%

The third checked variable was the radius of the area scanned during every step. Tests were performed for the factor ξ = [0.05; 0.1; 0.2]. The results can be seen in Table 5.4 and in Fig.5.5. As it is shown in the results, the time to finish a trial is inversely proportional to the radius of scanning.

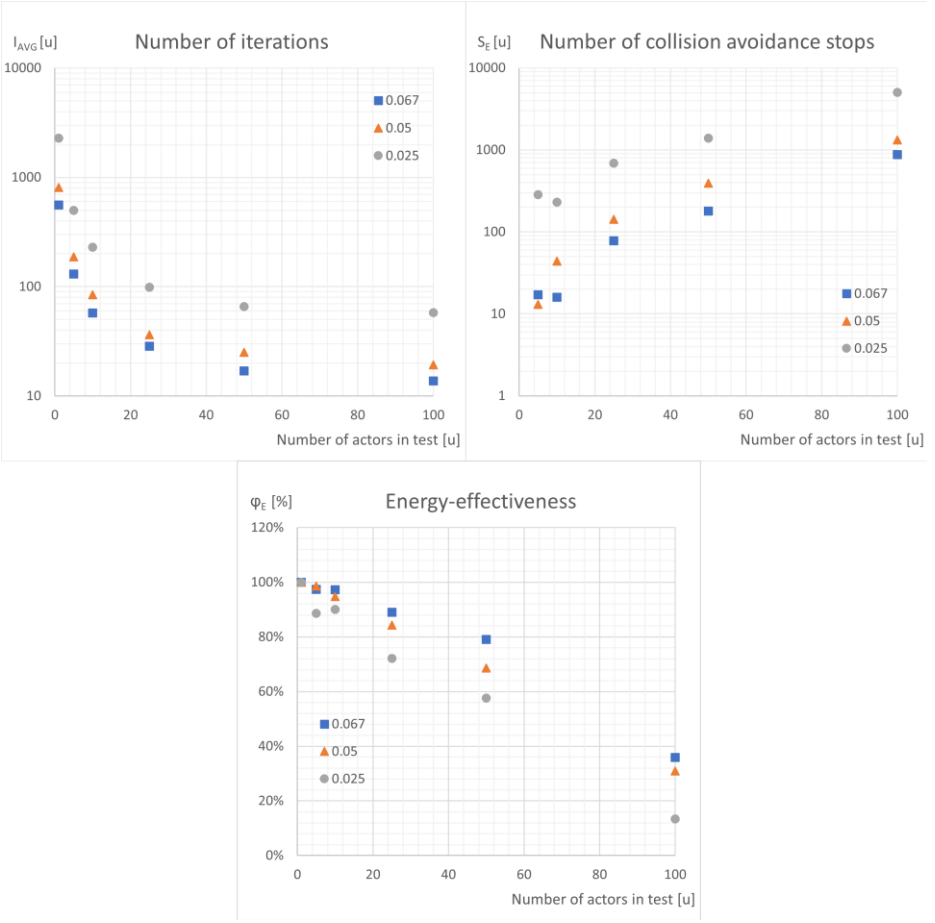

Fig.5.4. Influence of varying a single boid's velocity on the results

Table 5.4. Influence of varying ξ on the results

Radius of area scanned $r_{scan} = \xi \cdot R$						
Average number of iterations						
ξ	**1**	**5**	**10**	**25**	**50**	**100**
0.05	1700.8	342.3	160.8	71.6	40	30.1
0.1	805	53.	84	36	25	19
0.2	514	101	46	23	16	11
Collision avoidance stops						
ξ	**1**	**5**	**10**	**25**	**50**	**100**
0.05	0	26.9	92	263.8	522.8	1978.7
0.1	0	13	44	142	394	1329
0.2	0	13	25	114	286	753
Energy-Effectiveness						
ξ	**1**	**5**	**10**	**25**	**50**	**100**
0.05	100%	98.4%	94.3%	85.3%	73.9%	34.3%
0.1	100%	98.6%	94.8%	84.2%	68.7%	31.0%
0.2	100%	97.5%	94.7%	80.2%	63.8%	30.9%

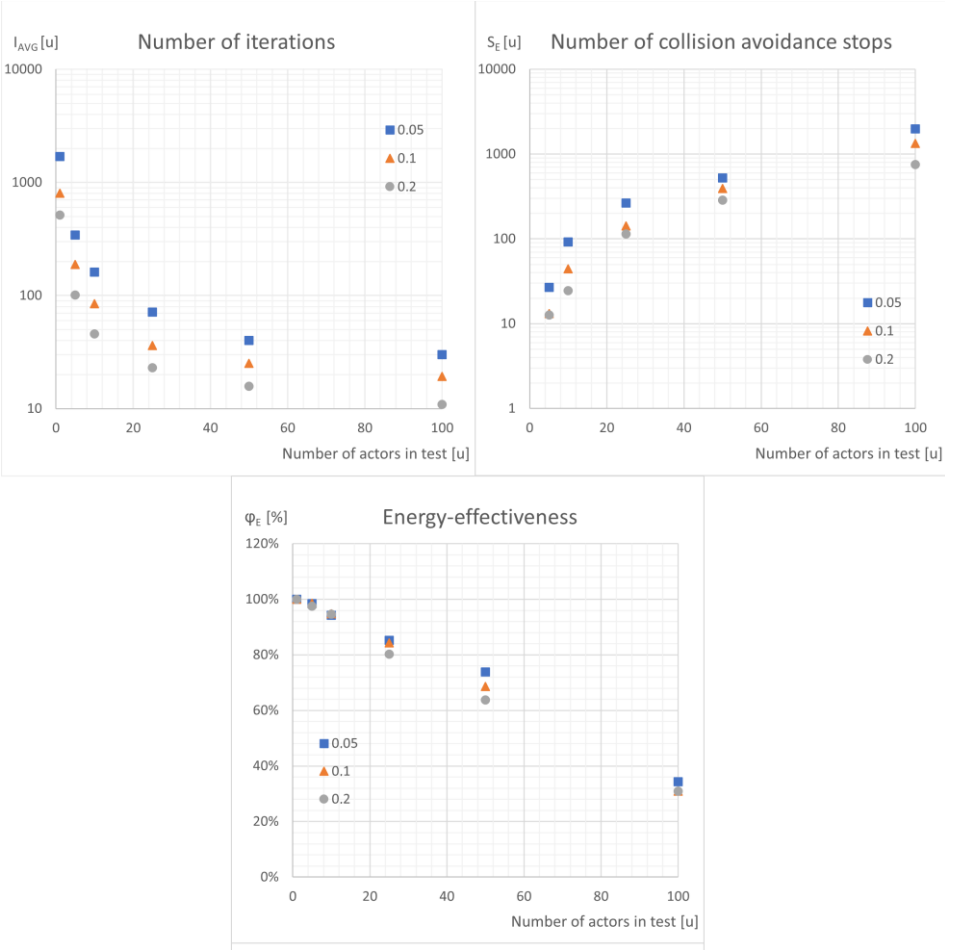

Fig.5.5. Influence of varying a size of an area scanned in every step

The bigger area searched with every step, the quicker task is done. For a constant Velocity and a Safe Distance, the number of Collision Avoidance Stops is directly proportional to time, regardless of the radius r_{scan}. Due to this, energy-effectiveness is almost the same in every case, which means that the size of the area, scanned with every step, influences the total time but not the energy-effectiveness of the swarm.

6. CONCLUSIONS

The presented algorithm allows performing the Coverage Task. The task is performed correctly, but the way it is done is not optimised. In case of the possible collision, boids stop what momentarily prevents them from continuing the task. The more collisions occur, the less efficient the swarm is. Furthermore, in this version, the algorithm could not be used on any real object, as it does not take into account any dynamics.

Compared to the Blanket method, the provided algorithm requires a similar level of simplicity of a single drone but it does not require that high amount of objects. The task can be performed even by a single boid regardless of the research area's size.

The presented algorithm is more related to the Sweep Method. In both solutions, drones are in constant movement until the task is done but most Sweep Coverage solutions require the movement to be somehow determined.

This implicates that the boids must be more advanced objects. In this project, we assumed stochastic movement of the boids which does not require high computing power and the object might be as simple as possible.

On the other hand, the random movement of boids comes with some flaws. There can be seen a large results dispersion, especially for less numerous groups. In a real-life solution using this algorithm, there should be an additional system that would control the area coverage. Without additional control, one might never be certain what part of the area was truly covered, it can be only assumed by the results provided in simulations.

Furthermore, finishing the task takes more time if it is performed by randomly moving objects. The same amount of boids could perform the task much faster if they moved along the precalculated, optimal path.

According to the presented results, the more boids there are, the faster they finalise the task, but their effectiveness decreases. Furthermore, the change in time, needed to complete the task, decreases with an increase of boids in the swarm. From some amount of boids, a further increase in quantity causes disproportionate changes in iterations which means, that beyond that point the effectiveness of subsequent boids is reduced. The results show that there can be designated an optimal number of objects in the group, that would perform the task in the best time and the best effectiveness. It can be useful while developing a real, physical swarm where a smaller amount of devices means smaller costs.

Calculating the energy effectiveness allowed us to designate additional information about optimisation of the number of boids in the swarm. The results show, that there can be evaluated the best solutions for certain assumptions. There can be found what number of boids would perform the task in the shortest possible time with assumed energy-effectiveness.

There can also be found a group with the best time to an energy-effectiveness index that would perform the task with the best combination of the shortest time and consumed energy.

Additional tests, with changing the variables, allowed us to determine their impact on the result. They showed that the Velocity and Safe Distances are combined but a change in the Velocity is more significant for smaller swarms when a change in the safe distance has a greater influence on more numerous groups. This concludes that for small groups better are faster boids and for more numerous swarms more useful would be boids that can move in closer proximity. Varying the radius of the area, scanned with every step impacts the total performance but has almost no influence on the energy-effectiveness of the swarm.

The presented solution is the first version of this algorithm. As mentioned earlier, at this stage it might not be used to simulate real objects as it does not consider any physical parameters. This indicates furthers steps for the project. In subsequent version of the algorithm, there will be included variables such as mass and acceleration.

Also, as this algorithm is intended to simulate UAVs, there will be added the height of flight and hence energy consumption should be divided into the Potential Energy and Kinetic Energy.

FUNDING

The authors received no financial support for the research, authorship, and/or publication of this article.

REFERENCES

[1] Chakraborty Amrita, Arpan Kumar Kar. 2017. „Swarm Intelligence: A Review of Algorithms". *Nature-Inspired Computing and Optimization* 10 : 475-494. DOI: 10.1007/978-3-319-50920-4_19

[2] Krause Jonas, Jelson Cordeiro, Rafael Stubs Parpinelli, Heitor Silveiro Lopes. 2013. 7 - A Survey of Swarm Algorithms Applied to Discrete Optimization Problems. In *Swarm Intelligence and Bio-Inspired Computation, Theory and Application*. Elsevier, ISBN 9780124051638.

[3] Alaliyat Saleh, Harald Yndestad, Filippo Sanfilippo. 2014. Optimisation of Boids Swarm Model Based on Genetic Algorithm And Particle Swarm Optimisation Algorithm (Comparative Study). In *Proceedings of the 28th European Conference on Modelling and Simulation, ECMS 2014*. DOI: 10.7148/2014-0643.

[4] Larsson Max, Sebastian Lundgern. 2017. *Perception of Realistic Flocking Behavior in the Boid Algorithm* (Bachelor Thesis). Karlskrona, Sweden: Blekinge Institute of Technology.

[5] Teng Hu, Ishtiaq Ahmad, Alamgir Msm, Kyung Hi Chang. 2020. "3D Optimal Surveillance Trajectory Planning for Multiple UAVs by Using Particle Swarm Optimization with Surveillance Area Priority". *IEEE Access* 8 : 86316-86327 DOI: 10.1109/ACCESS.2020.2992217.

[6] Prabakaran Veerajagadheswar, Rajesh Elara Mohan, Vinu Sivanantham, Thejus Pathmakumar, Suganya Sampath Kumar. 2018. "Tackling Area Coverage Problems in a Reconfigurable Floor Cleaning Robot Based on Polyomino Tiling Theory". *Applied Science* 8 (3) : 342-1-21, DOI:10.3390/app8030342.

[7] Gage W Douglas. 1992. "Command Control for Many-Robot Systems". *Unmanned Systems* 10 (4) : 28–34.

[8] Beal Jacob, Nicolaus Correll, Leonardo Urbina, Jonathan Bachrach. 2009. Behavior Modes for Randomized Robotic Coverage. In *Proceedings of the 2nd International ICST Conference on Robot Communication and Coordination*. DOI: 10.4108/ICST.ROBOCOMM2009.5863.

[9] Wang Qiuzhen, Hai Zhang. 2021. "A Self-Organizing Area Coverage Method for Swarm Robots Based on Gradient and Grouping". *Symmetry* 13 (4) : 680-1-24, DOI:10.3390/sym13040680.

[10] Miao Kun, Qian Feng, Wei Kuang. 2021. "Particle Swarm Optimization Combined with Inertia-Free Velocity and Direction Search". *Electronics* 10 (5) : 597-1-26. DOI:10.3390/electronics10050597.

[11] Singha Arindam, Anjan Kumar Ray, Arun Baran Samaddar. 2021. "Neural dynamics based complete grid coverage by single and multiple mobile robots". *SN Applied Sciences* 3 : 543, DOI: 10.1007/s42452-021-04508-5.

[12] Jevtić Aleksandar, Diego Andina. 2007. Swarm intelligence and its applications in swarm robotics. In *Proceedings of the 6th WSEAS Int. Conference on Computational Intelligence, Man-Machine Systems and Cybernetics*. Tenerife, Spain, December 14-16, 2007.

[13] Tehrani-Saleh Ali, Christoph Adami, Randal Olson. 2016. Flies as Ship Captains? Digital Evolution Unravels Selective Pressures to Avoid Collision in Drosophila. In *Proceedings of the Fifteenth International Conference on the Synthesis and Simulation of Living System*, 554-561. DOI: 10.7551/978-0-262-33936-0-ch089.

Metoda optymalizacji zachowania się roju na podstawie autorskiego algorytmu

Krzysztof FALKOWSKI, Michał DUDA

Wojskowa Akademia Ttechniczna,
Wydział Mechatroniki, Uzbrojenia i Lotnictwa
ul. gen. Sylwestra Kaliskiego 2, 00-908 Warszawa

Streszczenie. W artykule przedstawiono metody znalezienia optymalnej wielkości roju dla danego zadania. Głównymi ocenianymi czynnikami są czas i zużycie energii. Autorskie rozwiązanie algorytmiczne pozwoliło na wyznaczenie różnych przypadków i zbadanie wpływu różnych parametrów pojedynczego boida na zachowanie całego roju. Obliczenie efektywności energetycznej pozwoliło na wyznaczenie dodatkowych informacji o optymalizacji liczby boidów w roju. Wyniki pokazują, że można ocenić najlepsze rozwiązania dla określonych założeń. Można znaleźć, jaka liczba boidów wykonałaby zadanie w jak najkrótszym czasie przy założonej energooszczędności. Można również znaleźć grupę z najlepszym czasem do uzyskania wskaźnika efektywności energetycznej, która wykonałaby zadanie przy najlepszej kombinacji najkrótszego czasu i zużytej energii. Dodatkowe testy ze zmieniającymi się zmiennymi pozwoliły określić ich wpływ na wynik. Wykazano, że prędkość i bezpieczna odległość są ze sobą połączone, ale zmiana prędkości jest bardziej znacząca dla mniejszych rojów, gdy zmiana bezpiecznej odległości ma większy wpływ na liczniejsze grupy. Wynika z tego, że dla małych grup lepsze są szybsze boidy, a dla liczniejszych rojów bardziej przydatne byłyby boidy, które mogą poruszać się bliżej. Zmienianie promienia obszaru skanowanego na każdym kroku wpływa na ogólną wydajność, ale prawie nie ma wpływu na efektywność energetyczną roju.
Keywords: optymalizacja, rój, algorytm roju