DOI: 10.15439/2017F385
ISSN 2300-5963 ACSIS, Vol. 11

On the Use of Nature Inspired Metaheuristic in Computer Game

Piotr A. Kowalski[1,2], Szymon Łukasik[1,2], Małgorzata Charytanowicz [2,3] and Piotr Kulczycki[1,2]

[1] Faculty of Physics and Applied Computer Science
AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Cracow, Poland
Email: {pkowal,slukasik,kulczycki}@agh.edu.pl

[2] Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6, 01-447 Warsaw, Poland
Email: {pakowal,slukasik,mchmat,kulczycki}@ibspan.waw.pl

[3] Institute of Mathematics and Computer Science
The John Paul II Catholic University of Lublin
Konstantynów 1 H, 20-708 Lublin, Poland
Email: mchmat@kul.lublin.pl

***Abstract*—This paper describes the use of a new swarm-based metaheuristic, namely Krill Herd Algorithm (KHA), in computer gaming. In this work, KHA is employed to find a bots movement strategy in a computer racing game. The complete algorithm is implemented using a Unity Engine in C# language. Herein, the triggering of the metaheuristic optimization task was conducted by the way of a KHA internal parameter investigation. In this approach, the goal of the race (the KHA evaluation function) for both the human and computer player is to finish a lap in the shortest time possible.**

I. Introduction

THe goal of any artificial intelligence algorithm is to create a mechanism that can learn, conclude, and solve problems like a human. In computer games, this creates a form that mimics human behavior, and computer games provide an excellent environment for implementing and even testing artificial intelligence procedures. Developers of computer games are increasingly turning towards creating projects based upon artificial intelligence. Instead of crafting their product through employing predictable algorithms, whose results are identical inside each successive game world, artificial intelligence methods are used to dynamically adapt the behavior of a computer opponent to the player's level of competence.

One of the first games using artificial intelligence tools was released in 1999 by id Software, a first-person shooter game called Quake III Arena. In the production of this, the behavior of the computer player (the so-called bot) was based on an artificial neural network [1]. The bot was able to learn the behavior of its opponents, both the computer generated, and the genuine human player, so as to develop winning strategies.

Swarm intelligence is one of the more important domains of computational intelligence. This group of algorithms is applied in optimisation tasking. Herein, natural environmental processes and behaviours are the main inspiration [2]. Commonly used metaheuristics are: the Genetic Algorithm [3], the Gravitational Search Algorithm [4], Cuckoo Search [5], Earthworm Optimization Algorithm [6], Harmony Search [7], the Firefly Algorithm [8], Particle Swarm Optimization [9], [10], Ant Colony Optimization [11], the Bat Algorithm [12], the Differential Evolution [13] and the Autonomy-oriented computing methodology [14]. Newer algorithms, have been recently introduced for this tasking. These are: the Krill Herd Algorithm [15], [16], [17], Animal Migration Optimization [18], Wolf Search Algorithm [19], The Dragonfly Algorithm [20], Monarch Butterfly Optimization [21] and the Flower Pollination Algorithm [22]. Such bio-inspired metaheuristic algorithms are able to tackle very hard combinatorial optimisation problems [11] as well as, they can be applied for solving optimization problems in continuous space [23].

In this paper, we decided to test the utilization of the KHA within a computer race game. In this type of game, the user competes with computer generated opponents. Titles of such games currently on the market are: Test Drive or Need for Speed. During the project, interesting concepts were developed for the use of swarm intelligence.

The content of this paper has been divided into two main parts - theoretical and implementation. In the first, (Section II), the problem of utilizing artificial intelligence in the implemented computer game was discussed. Above all, the problem of optimization is delved into, as this issue affects the character and behavior of the computer generated opponent. It is to this that the artificial intelligence tools have been applied. We then thoroughly describe the chosen artificial intelligence algorithm. In the second part of the article (Section III and Section IV), aspects of both the implementation and, above all, the details related to the adaptation of individual elements as swarm bots, is described. Following this, we present of the results of selected tests of the proposed algorithm. The article ends with a chapter devoted to the summary and towards further plans for the development of this algorithm.

II. Optimisation based on Krill Herd Algorithm

KHA is an iterative heuristic procedure inspired by the natural phenomena of krill herd behaviour. This method is mainly applied for solving optimization problems in continuous space. Here, the solution of this problem is defined as finding such an argument x°, included in the space under consideration $S \subseteq R^N$, which fulfils the following formula

$$f(x^\circ) = \min_{x \in S} f(x) \tag{1}$$

where $f(x)$ describes the value of the cost function.

The KHA was proposed by Amir Hossein Gandomi and Amir Hossein Alavi in the article [15], and is based on imitating the behaviour of the individual krill moving together as a herd. Individual krill, and the herd itself, move accordingly to diverse environmental factors. Among these are proximity to neighbours (defined by herd density), dispersion of the animal group, food location and several other biological and environmental phenomena.

In order to solve the optimization problem, we introduced the KHA non-deterministic procedure. Herein, particular elements $x_i = x_i^1, \ldots, x_i^N$ are proposed of an N dimensional solutions space, in the form of P individuals. In the kth iteration, the best solution of the this problem as represented by the pth members of swarm is given alternatively by these two equations:

$$x^\circ(k) = arg \min_{p=1,\ldots,P} f(x_p(k)) \quad \text{/for minimalization task/} \tag{2}$$

or

$$x^\circ(k) = arg \max_{p=1,\ldots,P} f(x_p(k)). \quad \text{/for maximalization task/} \tag{3}$$

The above best solution corresponds with the minimal or maximal value of cost function $f^\circ = f(x^\circ)$ given as (2) or (3).

The full KHA procedure as a flow chart description is shown as Figure 1. This procedure begins from an initialization of all its internal parameters, and positions of all P individuals are generated randomly ❶. In next stage ❷, the cost (or fitness) function values are computed for all initial P swarm members using (2) or (3). The subsequent step ❸ is of great importance and is characterized by this technique. It consists of formulas describing the movement of particular individuals. Such motion viv-a-vis each individual krill is determined by three main components. They are:

- movement induced by other krill individuals,

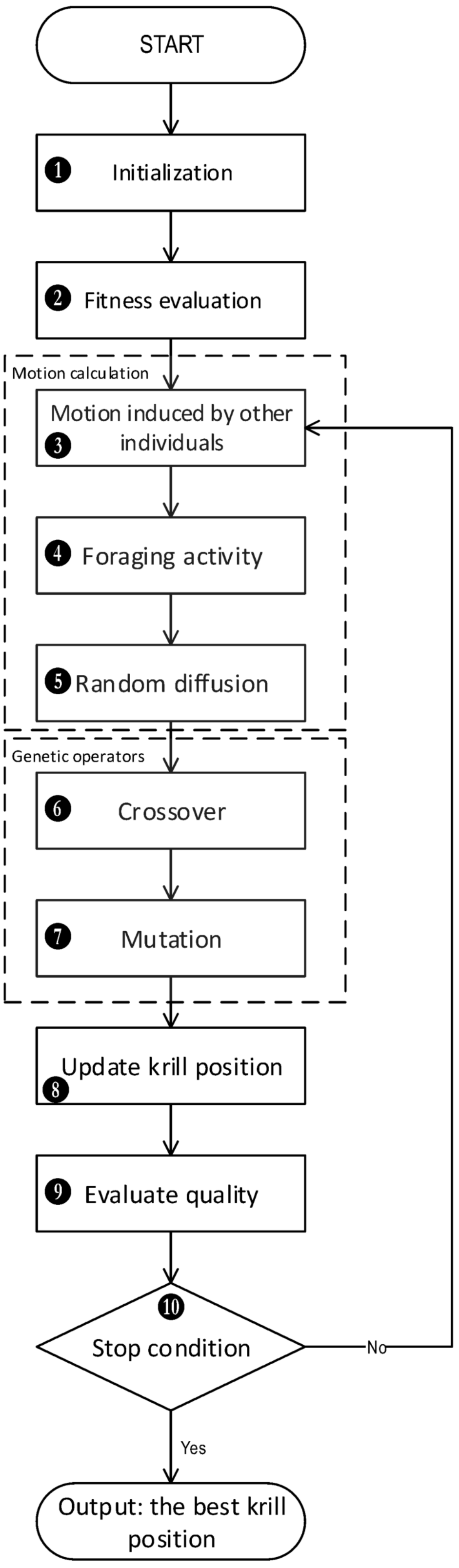

Fig. 1: Flowchart of KHA

- foraging activity,
- random diffusion.

In subsequent iterations, in the KHA technique, a vector of movement for the ith krill is based on the Lagrangian equation:

$$\frac{dx_i}{dt} = N_i + F_i + D_i, \tag{4}$$

where N_i is the motion induced by other krill individuals, F_i denotes the foraging motion and D_i is the physical diffusion of the krill individuals.

The first element ❹ is a reflection of the social inspiration of the individual members of the herd. In the swarm, members are maintained at a high density. Hence, the velocity of each individual is influenced by the movement of others. In consequence, the direction of movement by the α_i parameter is induced by the presence of other herd individuals. This parameter is determined on the basis of the following parts: local effect and target effect. The individual fractions of motion can be notated as:

$$N_i^{new} = N^{max}\alpha_i + \omega_n N_i^{old}. \tag{5}$$

Here N^{max} represents the maximum possible speed that can be induced, ω_n belongs to the interval $[0,1]$, and is defined as the inertia weight of a particular krill and finally N_i^{old} is the motion induced in the previous time step. The α_i parameter is introduced in following way:

$$\alpha_i = \alpha_i^{local} + \alpha_i^{target}, \tag{6}$$

where α_i^{local} describes the local influence of the neighbours of any particular swarm member, whereas α_i^{target} is the target direction. The latter is determined by the position and movement of the best individual in a swarm.

The α_i^{local} parameters are computed according to the formula:

$$\alpha_i^{local} = \sum_{j=1}^{NN} \hat{f}_{ij}\hat{X}_{ij}, \tag{7}$$

where

$$\hat{X}_{ij} = \frac{x_j - x_i}{\|x_j - x_i\| + \epsilon}, \tag{8}$$

and

$$\hat{f}_{ij} = \frac{f_i - f_j}{f^{worst} - f^{best}}. \tag{9}$$

In equation (9), f in provides the cost value (1) of any investigated krill. Consequently f^{worst} and f^{best} represent, respectively, the worst and the best fitness of individuals in swarm. Additionally, NN describes the identification of the number of reachable krill neighbours, and ϵ is a positive number introduced to avoid singularities in the denominator of formula (8).

For determination of distance between particular krills and their neighbours, a parameter designated as being the sensing distance d_s, is proposed. Its value may be formulated as:

$$d_{s,i} = \frac{1}{5P}\sum_{j=1}^{P}\|x_i - x_j\|. \tag{10}$$

What is more, each swarm member incorporates its own target vector. This is formulated as follows:

$$\alpha_i^{target} = C^{best}\hat{f}_{i,best}\hat{x}_{i,best}, \tag{11}$$

where

$$C^{best} = 2\Big(rand + \frac{k}{K^{max}}\Big). \tag{12}$$

Herein, k, K^{max} designate, respectively, the current iteration number and the maximum number of iterations. Moreover, a $rand$ is a random value between 0 and 1, whereas $\hat{f}_{i,best}$ is the best value of fitness function, while $\hat{x}_{i,best}$ provides the location of the best ith individual from the previous time steps.

In the equation (4), the symbol F_i is connected with the food foraging issue. Herein, F_i is defined in the following way:

$$F_i = V_f\beta_i + \omega_f F_i^{old}, \tag{13}$$

where V_f is the food foraging speed and ω_f describes the inertia of the movement. In equation (13), the food fitness of the ith krill is designated as follows:

$$\beta_i = \beta_i^{food} + \beta_i^{best}. \tag{14}$$

The aforementioned food aspect is defined by way of its location. Therefore, the centre of food concentration is defined via KHA as a virtual point. This conception by the "centre of mass" approach is interpretable. Hence, the food concentration in each iteration is calculated according to formula:

$$X^{food} = \frac{\sum_{i=1}^{P}\frac{1}{f_i}x_i}{\sum_{i=1}^{P}\frac{1}{f_i}}. \tag{15}$$

Here, the food attraction for the ith swarm member is described via:

$$\beta_i^{food} = C^{food}\hat{f}_{i,food}\hat{X}_{i,food}. \tag{16}$$

The food coefficient in (16) expresses the global attraction of the food centre (15), and may be calculated as:

$$C^{food} = 2\Big(1 - \frac{k}{K^{max}}\Big). \tag{17}$$

The second part of equation (14) is as follows:

$$\beta_i^{best} = \hat{f}_{i,best}\hat{x}_{i,best}. \tag{18}$$

In this equation, $f_{i,best}$ expresses the best fit achieved by a given ith individual so far. This is determined by its position $\hat{x}_{i,best}$.

The last element of the Lagrangian equation (4) is connected with random physical diffusion ❺, represented as D_i. In essence, this component has a fully random character. This part of movement is focused upon the diversity in the swarm;

it allows the individual krill to position itself inside the krill swarm so as to be within a situation of local optimum. This part of equation (4), hence, represents a trade-off between exploration and exploitation. The following equation shows these aspects as a random diffusion:

$$D_i \;=\; D^{max}\Big(1-\frac{k}{K^{max}}\Big)\delta, \tag{19}$$

where, D^{max} is the maximum diffusion factor and δ expresses the random directional vector.

The motion aspect of krill activity can now be fully described. Herein, all the aforementioned effective parameters are applied. Thus, the position of the ith individual during the interval t to $t+\Delta t$ is determined by the following equation:

$$x_i(t+\Delta t) \;=\; x_i(t)+\Delta t\frac{dx_i}{dt}. \tag{20}$$

Here, it should be underlined that parameter Δt is very sensitive to the speed and accuracy of the optimisation task. In this respect, the Δt may be interpreted as being a scale factor of krill movement, and can be obtained by way of the following equation:

$$\Delta t = C_t\sum_{j=1}^{N}(UB_j - LB_j). \tag{21}$$

In the above equation, C_t is an empirically found constant number from the interval $[0, 2]$. What is more, UB_j and LB_j constitute, respectively, the upper and lower bounds of the jth feature $(j = 1, \ldots, N)$ of data set $X = x_1, \ldots, x_P$.

The subsequent step of the heuristic algorithm is an implementation of two genetic or evolutionary operators. In step ❻, the crossover function is considered. This operator is controlled by the Cr parameter referred to as the 'crossover probability'. In this approach, this operator is defined randomly, and the crossover is revealed in the change of the mth coordinate of the ith individual. This comes about by applying the following formula:

$$x_{i,m} = \begin{cases} x_{r,m} & \text{for} \quad \gamma \le Cr \\ x_{i,m} & \text{for} \quad \gamma > Cr \end{cases}, \tag{22}$$

where $Cr = 0.2\hat{K}_{i,best}$; $r \in \{1, 2, ..., i-1, i+1, ..., P\}$ and γ is a random number drawn from the interval $[0, 1)$, which is generated via uniform distribution. In this solution, the crossover operator is calculated by way of a single individual.

Finally, the mutation operator ❼ is applied within the last stage of the main loop of the KHA. This changes the m-th coordinate of the i-th individual, as shown below by the formula:

$$x_{i,m} = \begin{cases} x_{gbest,m} + \mu(x_{p,m} - x_{q,m}) & \text{for} \quad \gamma \le Mu \\ x_{i,m} & \text{for} \quad \gamma > Mu \end{cases}, \tag{23}$$

wherein $Mu = 0.05/\hat{K}_{i,best}$; $p, q \in \{1, 2, ..., i-1, i+1, ..., P\}$ and $\mu \in [0, 1)$.

This operation completes all evolutionary procedures. Subsequently, we can now obtain individuals that can be used within the next iteration. In so-doing, in the last step ❽ of the main loop, the cost function for all the swarm members is calculated. Now, the algorithm's termination condition ❿ decides whether the next iteration is to be entered into or the optimization algorithm is to be completed. The form of stop condition applied could be that of a time limit, or the reaching of a desired fitness level or a combination of above two.

More information about this metaheuristic algorithm can be found in [15]. Regarding the procedure's internal parameters, the tuning of the KHA is described in papers: [24], [25] and [26], while publications [17] and [16] introduce some modifications into the algorithm. The KHA procedure has been verified for application within optimization problems in the case of discrete input data [27], while a parallel version of this procedure is put forward in [28]. Furthermore, it has been applied in medical tasks [29], for data base domains [30], in mechanism and machine theory [31], in clustering tasks [32], [33], and also in neural learning processes [34]. Extensive use of this algorithm has been collected in the article [35].

III. Implementation and optimization of Game

The Unity engine [36] is now employed in order to complete the task and to implement the game. This is a tool that allows the creation of games for Windows, Linux, Mac OS, Xbox 360, PlayStation 3, Wii U, iPad, iPhone, Android, Windows Phone 8 and BlackBerry environments. Unity has rapidly gained popularity thanks to its user-friendly interface. It allows for fast development of the game, along with the ability to test existing progress. In optimising Unity for creating cross-platform games, the programmer can use any of three programming languages: C# for the Mono platform, JavaScript, or the Boo-inspired language, Python. All implementations described in this work have been written in C#.

Swarm intelligence is applied in our study application for optimizing the travel time by way of adjusting driving performance. Firstly, the track was divided into sectors that consist of curves of similar characteristics. In doing so, a racing line was formed along which the bots are to move. Figure 2 shows the waypoints which are densely distributed throughout the route.

In completing this task, some parameters are introduced. These are considered as being the same parameters that affect the coordinates of individual krill within the herd. The most important parameter is undoubtedly the maximum speed in the sector. If a bot is currently located in a straightaway or where steering arcs are long, and where the steering input angle is low, a high maximum speed is desirable. In turn, if the bot enters within a twisting and winding section of the track or where the curves are tight and steering input is intense, then the speed must be properly limited, otherwise the car loses its grip, resulting in drift, a wider line of travel and, consequently slower travel times. The second parameter is related to the angle between the car and the next point of the race line. If it is greater than the value for a given sector, then the computer

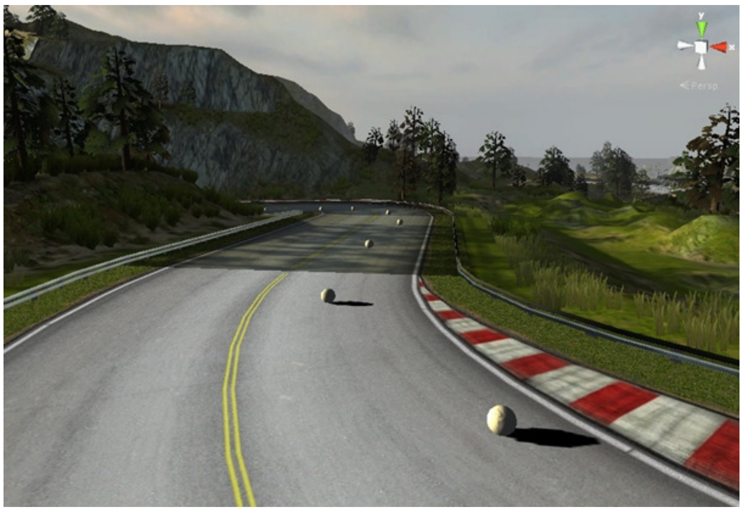

Fig. 2: Race line with waypoints

player starts to understeer or oversteer and must accelerate or decelerate. This value should be greater for straightforward sectors as it reduces the gliding effect of the car on the track. The third and last parameter of the driving characteristic is the time it takes to make a turn. For winding sectors, the value is less than that for simple sectors, because a faster response is needed. The above parameters are transferred to the algorithms for bots controlling in the game engine.

In presenting the implementation of swarm intelligence, the passage time within a particular sector of the track is optimized by adjusting the driving parameters of a given computer player. Therefore, the algorithm should be run only when all the computer players overcome the sector. Detecting the moment when players finish the passage of a given sector takes place using the so-called 'collider' (Fig. 3). Thus, when the last computer player completes a subsequent sector, the *Run(int)* function is called up for the sector identifier that it is responsible for when executing one iteration of the algorithm.

A. Application of KHA to game

For the implementation of the KHA, each computer-based object was coded as a *MKrill* class component representing one individual in the population.

Each object has the following attributes: an identifier in the form of an integer, of times in the current round; sector records which store the performance characteristics of the computer player used in the current lap; the parameters described above (maximum speed in the sector, angle between the car and the next point and time to make a turn); best parameters array; induced vectors, foraging vectors and diffusion vectors as components of KHA; and, finally, lower bounds and upper bounds. All the aforementioned are used to store the lower and upper limits of the respective main driving parameters.

Another important element is the determination of the value of the cost function (1). This is the first step in executing each iteration of the KHA. In this implementation, however, cost function is not calculated explicitly, because the value of this function is the passage time within the sector for which the algorithm is being executed at the moment.

Now, the individual designated Lagrangian (4) components are calculated (see Section II). Firstly, the determination of the motion induced by other krill individuals is accomplished by applying the formulas (5)-(12). In this part of the algorithm, the displacement vector for each individual in the population is generated. In doing this, in each iteration, α^{local} and α^{target} based on equations (7) and (11) are first calculated, and then summed according to equation (6). Thereafter, in each iteration, the appropriate vector (5) is determined, taking into account the following parameters, N^{max} and ω_n.

In the next step of the algorithm, the food foraging movement is ascertained as per notation (13). In this part of the iteration, equations (14)-(18) are applied. This process is similar to that of the N_i calculation. Due to the optimization of travel time in the presented version of the algorithm, it is not possible to easily determine the value of the cost function for food (the value appearing in equation (16)). Thus, the solution is to assign to its value, the activity adapted by an individual closest to the food.

Finally, with regard to moment computing, a random physical diffusion, notated as D_i , is performed. In this case, equation (19) is used.

After all the above effective motion parameters are calculated, the change of each i-th krill position can be ascertained through employing equation (20) and applying notation (21).

Finally, it is worth observing that basic KHA utilizes several other evolutionary operators such as mutation (23) and crossover (22) for swarm member modifications. In the present iteration of the paper, these were not applied.

In summation, it should be noted that utilizing KHA is, in a sense, a way of optimising the computer game activity. The presented implementation of the KHA has its advantages and disadvantages. The disadvantage is the inability to explicitly

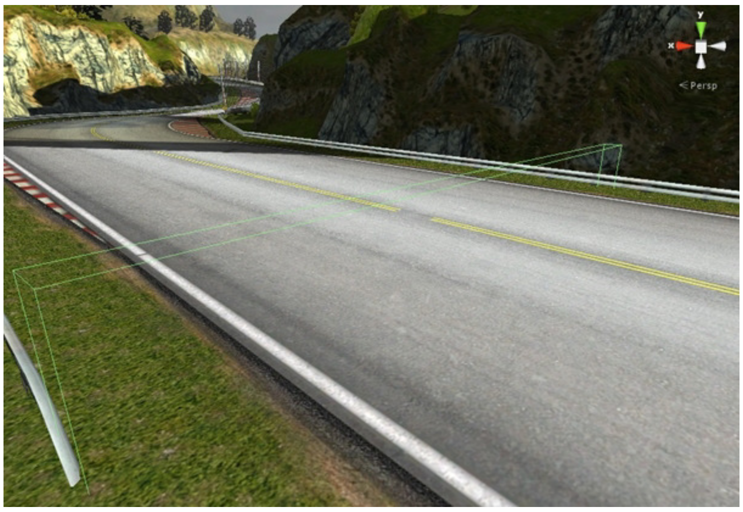

Fig. 3: A collider located at the end of the sector

calculate the value of the cost function. This value is the time of sector passage, and it can only be known when the computer player has overcome the sector through applying the parameters specified. Therefore, only the estimation of the effect of food position for the krill movement was applied.

The advantage of this implementation is, undoubtedly, its scalability. When needed, it is easy to take into account additional factors that can influence the nature of the player's computer. Moreover, this implementation is not computationally demanding, because each one iteration takes place when the last competitor crosses the boundary of a given sector. In the case of a track, as used in the game, and assuming that the players are moving close together, this means that one iteration every 1.5 – 2.0 seconds is performed.

IV. Numerical simulations

While researching the effectiveness of the proposed method, we analysed the impact of KHA internal parameters on quality of solution. Herein, we saw that the quality of the solution can be greatly influenced by the impact of internal parameters [24], [37].

The enclosed figures show the results of the tests that were applied to assess the quality and speed of the solution according to KHA parameters. In this case, the subject quantities were C_t, ω_n, ω_f and N.

In the test of the first parameter, C_t, the remaining parameter values are $\omega_n = 0.5$, $\omega_f = 0.5$ and $N = 5$. The results have been visualized in Figure 4. Here, each line shows the best results (i.e., the shortest time of the lap of the bot-car) for the investigated C_t value.

From the above tests, it can be inferred that an increase in the value of the variable Ct resulted in an increase in the difference between the times gained by the individual players in the first phase of the test. This indicates that even the far-fetched points in the solution space are represented. Finally, the best time was reached at $C_t = 1.5$. This was 93.27 s.

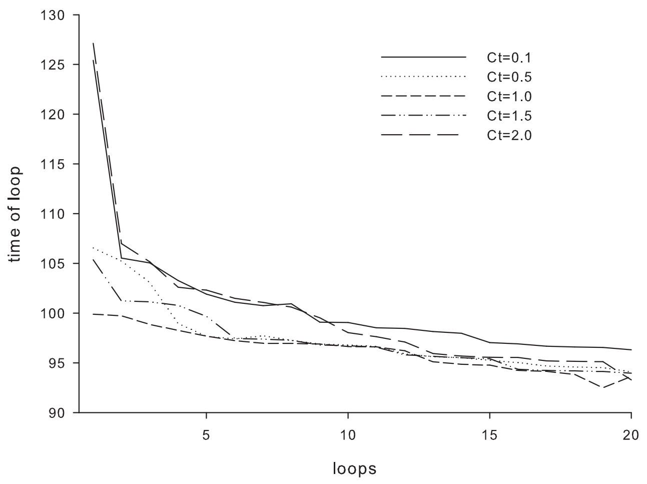

Fig. 4: Convergence of the optimisation procedure with various C_t parameters

In the next test, the ω_n parameter is modified through the application of a number in the range $(0.0; 1.0)$. This action represents the influence of neighbors in the creating of the movement vector. The obtained results are shown in Figure 5.

In this case, increasing ω_n, slowed the computer players in achieving better results. This means that through introducing the calculated influence of the neighbors, a larger value of ω_n results in a more accurate search within the krill environment while reducing its pace of approaching the global minimum. The best time passed in the test was for the case of $\omega_n = 0.2$. Herein, the time value of 92.30 s was achieved.

In our study, the parameter ω_f was modified (Figure 6). This is a number in the range of $(0.0; 1.0)$, and it represents the effect of the phase of the food search on the movement

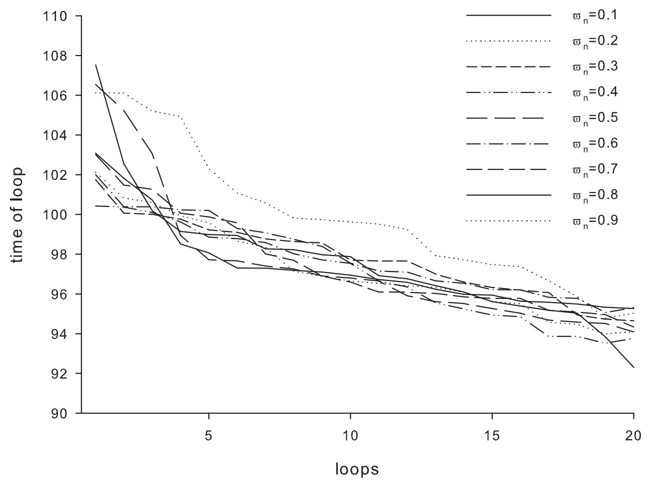

Fig. 5: Convergence of the optimisation procedure with various ω_n parameters

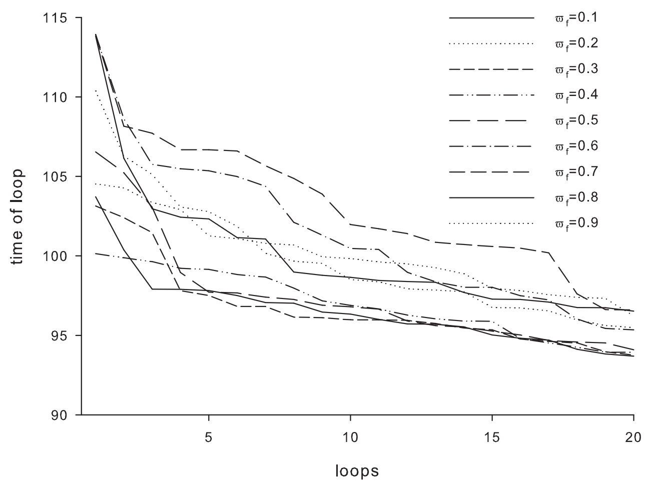

Fig. 6: Convergence of the optimisation procedure with various ω_f parameters

vector value. Increasing the value of ω_f, therefore, resulted in better player performance. For $\omega_f = 0.1$, the best time is 93.69, while for $\omega_f = 0.9$, this increased to 95.02. However, it is important to take into account that in this implementation, the approximate value of the cost function of the food is one that was calculated to reduce the efficiency of the entire phase of the food search.

The last test was to change the size of the krill population. In this research, a limited amount of the swarm are employed as bots. The main reason for this is that each krill represents one car on the race route. So a large number of bots in one place holds the implication that their collective movement is similar to that of a krill herd, and a great number of collisions will take place. The results of racing 3, 5 and 7 bots are shown in Figure 7.

One can observe from Figure 7 that increasing population numbers, increases the speeds in which better lap times are acquired by the computer players. When the population had three individuals, it was only after 10 laps that all players started to regularly achieve lap times less than 100 seconds. In the case of a population of five, this came about on the 7th lap. The main reason for such results is that the parameters for each krill are generated according to a uniform distribution. In other words, increasing the population, increases the probability that one of the individuals will be closer to the global minimum. The second reason is the development of synergies between the herd participants.

In conclusion, the modification of the studied parameters can influence the behaviour of the KHA. Increasing the C_t parameter speeds up the exploration of the solution space, but at $C_t > 1.0$, the incremental value of the movement vector may be too large, which in turn, can lead to better solutions. Increasing the value of variables ω_n and ω_f clearly slows the pace to gaining better results. Increasing the size of the population, in addition to having impact on the speed of the solution, also affects the quality of the solution, as more agents can better search for better solutions.

V. Summary

Experimental results indicate that the proposed solution can be used in a professional computer game, but only for one of low and medium difficulty. Thus, the level of computer opponents in this approach could be a challenge only for lesser and intermediate players. In order to streamline the implementation, a number of modifications would have to be made. Among these are the incorporation of target users' game results, as this would help improve the performance of the computer players. In order to eliminate the fluctuations of the final travel times, it would be useful to include the current best path, which would have an impact on the routing of the car. An alternative to improving the algorithm is to reduce the random factor generated through the method of determining a new food distribution, by replacing it with a deterministic algorithm or by manually selecting a developer designated fixed location during the game design. The implementation of the game presented in this paper can be further developed in many different ways. The most interesting directions are to find a better selection of krill algorithm parameters in order to be more efficient; to implement an improved version of the KHA, ie Lévy-flight KHA [37]; or to implement a learning mechanism based on human player experience.

Acknowledgment

The authors are thankful to Jacek Kurlit and Mateusz Harla (students of CUT) for the helpful implementation of the investigated problem.

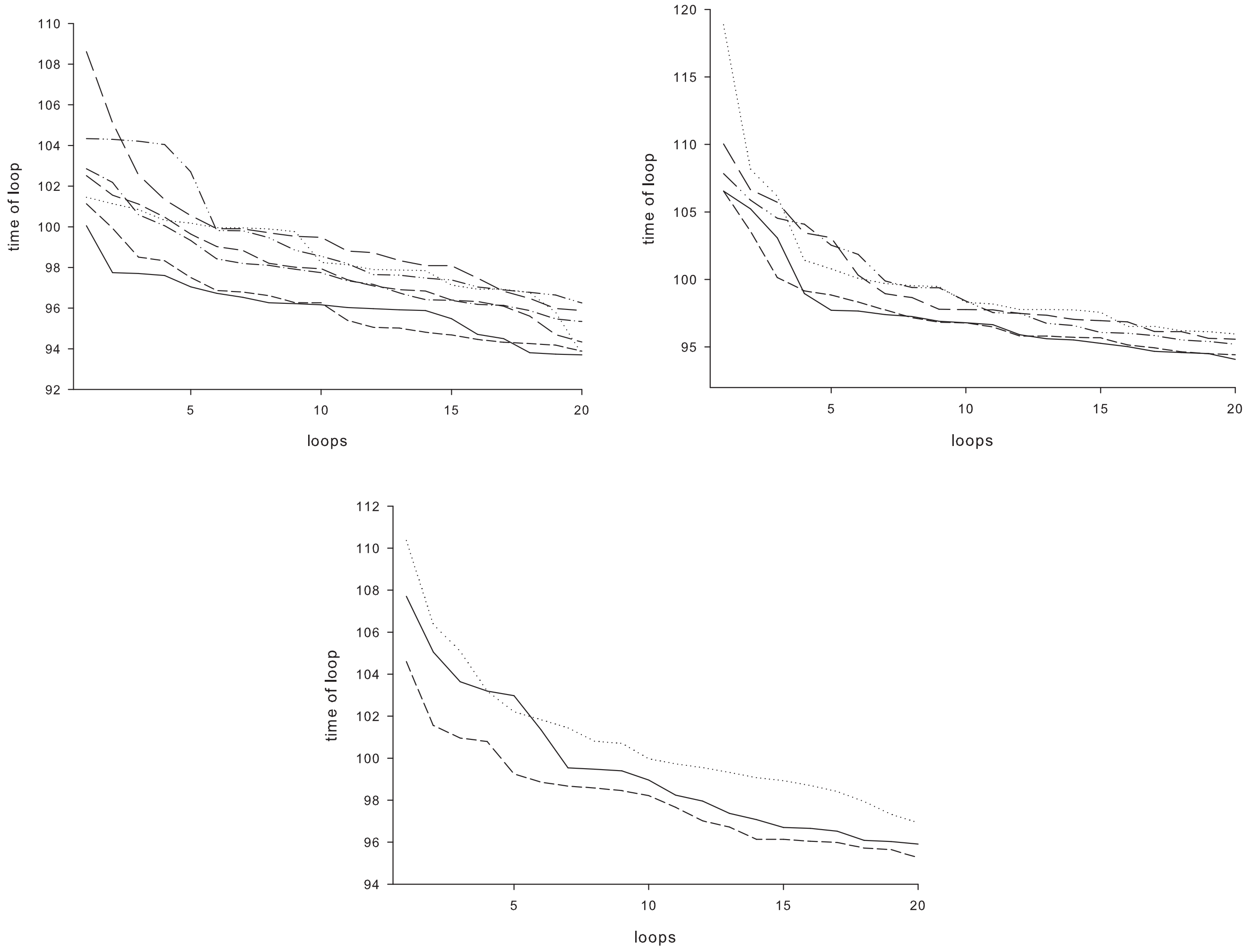

Fig. 7: Results of simulation for 7, 5 and 3 members of swarm respectively.

References

[1] J. van Waveren, "The quake iii arena bot," *University of Technology Delft*, 2001.

[2] X. Yang, *Nature-Inspired Optimization Algorithms*. London: Elsevier, 2014.

[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989. ISBN 0201157675

[4] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "Gsa: A gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232 – 2248, 2009. doi: https://doi.org/10.1016/j.ins.2009.03.004 Special Section on High Order Fuzzy Sets. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020025509001200

[5] X. S. Yang and S. Deb, "Cuckoo search via levy flights," in *2009 World Congress on Nature Biologically Inspired Computing (NaBIC)*, Dec 2009. doi: 10.1109/NABIC.2009.5393690 pp. 210–214.

[6] G.-G. Wang, S. Deb, and L. Coelho, "Earthworm optimization algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *International Journal of Bio-Inspired Computation*, 2015.

[7] Z. W. Geem, J. H. Kim, and G. Loganathan, "A new heuristic optimization algorithm: Harmony search," *SIMULATION*, vol. 76, no. 2, pp. 60–68, 2001. doi: 10.1177/003754970107600201

[8] X. Yang, "Firefly algorithm, stochastic test functions and design optimisation," *Int. J. Bio-Inspired Comput.*, vol. 2, no. 2, pp. 78–84, Mar. 2010. doi: 10.1504/IJBIC.2010.032124. [Online]. Available: http://dx.doi.org/10.1504/IJBIC.2010.032124

[9] S. Łukasik and P. A. Kowalski, "Fully informed swarm optimization algorithms: Basic concepts, variants and experimental evaluation," in *2014 Federated Conference on Computer Science and Information Systems*, Sept 2014. doi: 10.15439/2014F377 pp. 155–161.

[10] S. K. Panigrahi, A. Sahu, and S. Pattnaik, "Structure optimization using adaptive particle swarm optimization," *Procedia Computer Science*, vol. 48, pp. 802 – 808, 2015. doi: http://dx.doi.org/10.1016/j.procs.2015.04.218. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915007279

[11] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, no. 2, pp. 243 – 278, 2005. doi: http://dx.doi.org/10.1016/j.tcs.2005.05.020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397505003798

[12] X.-S. Yang and X. He, "Bat algorithm: Literature review and applications," *Int. J. Bio-Inspired Comput.*, vol. 5, no. 3, pp. 141–149, Jul. 2013. doi: 10.1504/IJBIC.2013.055093. [Online]. Available: http://dx.doi.org/10.1504/IJBIC.2013.055093

[13] D. Zou, J. Wu, L. Gao, and S. Li, "A modified differential evolution algorithm for unconstrained optimization problems," *Neurocomputing*, vol. 120, pp. 469 – 481, 2013. doi: https://doi.org/10.1016/j.neucom.2013.04.036 Image Feature Detection and Description. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231213005717

[14] J. Liu, X. Jin, and K. C. Tsui, "Autonomy-oriented computing (aoc): formulating computational systems with autonomous components," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 35, no. 6, pp. 879–902, Nov 2005. doi: 10.1109/TSMCA.2005.851293

[15] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012. doi: 10.1016/j.cnsns.2012.05.010. [Online]. Available: http://dx.doi.org/10.1016/j.cnsns.2012.05.010

[16] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "Stud krill herd algorithm," *Neurocomputing*, vol. 128, pp. 363–370, 2014. doi: 10.1016/j.neucom.2013.08.031. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2013.08.031

[17] L. Guo, G.-G. Wang, A. H. Gandomi, A. H. Alavi, and H. Duan, "A new improved krill herd algorithm for global numerical optimization," *Neurocomputing*, vol. 138, pp. 392–402, 2014. doi: 10.1016/j.neucom.2014.01.023. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2014.01.023

[18] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: an optimization algorithm inspired by animal migration behavior," *Neural Computing and Applications*, vol. 24, no. 7, pp. 1867–1877, 2014. doi: 10.1007/s00521-013-1433-8. [Online]. Available: http://dx.doi.org/10.1007/s00521-013-1433-8

[19] S. Fong, S. Deb, and X.-S. Yang, "A heuristic optimization method inspired by wolf preying behavior," *Neural Computing and Applications*, vol. 26, no. 7, pp. 1725–1738, 2015. doi: 10.1007/s00521-015-1836-9. [Online]. Available: http://dx.doi.org/10.1007/s00521-015-1836-9

[20] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Computing and Applications*, vol. 27, no. 4, pp. 1053–1073, 2016. doi: 10.1007/s00521-015-1920-1. [Online]. Available: http://dx.doi.org/10.1007/s00521-015-1920-1

[21] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Computing and Applications*, pp. 1–20, 2015. doi: 10.1007/s00521-015-1923-y. [Online]. Available: http://dx.doi.org/10.1007/s00521-015-1923-y

[22] X.-S. Yang, M. Karamanoglu, and X. He, "Multi-objective flower algorithm for optimization," *Procedia Computer Science*, vol. 18, pp. 861 – 868, 2013. doi: http://dx.doi.org/10.1016/j.procs.2013.05.251 2013 International Conference on Computational Science. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050913003943

[23] S. Łukasik and S. Żak, *Firefly Algorithm for Continuous Constrained Optimization Tasks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 97–106. ISBN 978-3-642-04441-0. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-04441-0_8

[24] P. A. Kowalski and S. Łukasik, "Experimental study of selected parameters of the krill herd algorithm," in *Intelligent Systems'2014*. Springer Science Business Media, 2015, pp. 473–485. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11313-5_42

[25] G. P. Singh and A. Singh, "Comparative study of krill herd, firefly and cuckoo search algorithms for unimodal and multimodal optimization," *IJISA*, vol. 6, no. 3, pp. 35–49, 2014. doi: 10.5815/ijisa.2014.03.04. [Online]. Available: http://dx.doi.org/10.5815/ijisa.2014.03.04

[26] P. K. Adhvaryyu, P. K. Chattopadhyay, and A. Bhattacharjya, "Application of bio-inspired krill herd algorithm to combined heat and power economic dispatch," in *2014 IEEE Innovative Smart Grid Technologies - Asia*. IEEE, 2014. doi: 10.1109/isgt-asia.2014.6873814. [Online]. Available: http://dx.doi.org/10.1109/isgt-asia.2014.6873814

[27] G.-G. Wang, S. Deb, and S. M. Thampi, *Intelligent Systems Technologies and Applications: Volume 1*. Cham: Springer International Publishing, 2016, ch. A Discrete Krill Herd Method with Multilayer Coding Strategy for Flexible Job-Shop Scheduling Problem, pp. 201–215. ISBN 978-3-319-23036-8. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-23036-8_18

[28] A. Nowosielski, P. A. Kowalski, and P. Kulczycki, "Increasing the Speed of the Krill Herd Algorithm through Parallelization," in *Information Technology, Computational and Experimental Physics*. AGH University of Science and Technology Press, 2016, pp. 117–120. ISBN 978-83-7464-838-7

[29] A. Mohammadi, M. S. Abadeh, and H. Keshavarz, "Breast cancer detection using a multi-objective binary krill herd algorithm," in *Biomedical Engineering (ICBME), 2014 21th Iranian Conference on*, Nov 2014. doi: 10.1109/ICBME.2014.7043907 pp. 128–133.

[30] A. Nowosielski, P. A. Kowalski, and P. Kulczycki, "The column-oriented database partitioning optimization based on the natural computing algorithms," in *2015 Federated Conference on Computer Science and Information Systems, FedCSIS 2015, Łódź, Poland, September 13-16, 2015*, 2015. doi: 10.15439/2015F262 pp. 1035–1041. [Online]. Available: http://dx.doi.org/10.15439/2015F262

[31] R. R. Bulatović, G. Miodragović, and M. S. Bošković, "Modified krill herd (mkh) algorithm and its application in dimensional synthesis of a four-bar linkage," *Mechanism and Machine Theory*, vol. 95, pp. 1 – 21, 2016. doi: http://dx.doi.org/10.1016/j.mechmachtheory.2015.08.004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0094114X15001895

[32] P. Kowalski, S. Łukasik, M. Charytanowicz, and P. Kulczycki, "Clustering based on the krill herd algorithm with selected validity measures," in *Proceedings of the 2016 Federated Conference on Computer Science and Information Systems*, ser. Annals of Computer Science and Information Systems, G. M., M. L., and P. M., Eds., vol. 8. IEEE, 2016. doi: 10.15439/2016F295 pp. 79–87. [Online]. Available: http://dx.doi.org/10.15439/2016F295

[33] ——, "Comparison of krill herd algorithm and flower pollination algorithm in clustering task," *ESCIM 2016*, pp. 31–36, 2016.

[34] P. Kowalski and S. Łukasik, "Training neural networks with krill herd algorithm," *Neural Processing Letters*, 2015. doi: 10.1007/s11063-015-9463-0

[35] P. Kowalski, S. Łukasik, and P. Kulczycki, "Methods of collective intelligence in exploratory data analysis: A research survey," in *Proceedings of the International Conference on Computer Networks and Communication Technology (CNCT 2016)*, ser. Advances in Computer Science Research, P. Kowalski, S. Łukasik, and P. Kulczycki, Eds., vol. 54. Xiamen (China): Atlantis Press, December 2016. doi: 10.2991/cnct-16.2017.1 pp. 1–7.

[36] J. Craighead, J. Burke, and R. Murphy, "Using the unity game engine to develop sarge: a case study," in *Proceedings of the 2008 Simulation Workshop at the International Conference on Intelligent Robots and Systems (IROS 2008)*, 2008.

[37] S. Łukasik and P. A. Kowalski, "Study of flower pollination algorithm for continuous optimization," in *Intelligent Systems'2014*. Springer Science Business Media, 2015, pp. 451–459. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11313-5_40