



## Nanorobots target search strategies using swarm intelligence

 Rodney Mushininga<sup>(a)\*</sup>  Livashen Moodley<sup>(b)</sup>



<sup>(a)</sup>School of Information Technology, The Independent Institute of Education, IIEMSA, South Africa.

<sup>(b)</sup>Independent Researcher, South Africa

### ARTICLE INFO

#### Article history:

Received 13 January 2023

Received in rev. form 25 Feb. 2023

Accepted 17 March 2023

#### Keywords:

Nanorobots, Swarm Intelligence,  
Emergent Behavior

#### JEL Classification:

D83; M15

### ABSTRACT

*Controlling a swarm of bee-like nanorobots in search of a target in a human body-like environment is a challenge. The challenge lies in the causal factors of emergence. This research focused on designing and developing a bee-inspired algorithm to provide instructions to coordinate simple and naive nanorobots in search of a target. Experiments were conducted through simulations to determine the rate at which myriads of agents converge at a target. A maze environment depicting human blood vessels was used to deploy the nanorobots. Different sample population sizes were used, and the results indicate that the bigger the sample size, the higher the convergence speed. Compared with a single complicated robot, the swarm of intelligently controlled nanorobots proved efficient and effective in finding a target in a human-like environment.*

© 2023 by the authors. Licensee SSBFNET, Istanbul, Turkey. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

## Introduction

Swarm intelligence is the collective behaviour of decentralized, self-organized systems, natural or artificial (Zhang, 2013). Swarm intelligence systems are typically made up of a population of simple agents interacting locally with one another and with their environment (Zhang, 2013). Nanorobots, also known as Nanorobotic agents in this paper, are extremely small and programmed to diagnose and treat various illnesses in a human-like body environment (Manjunath & Kishore, 2014). These nanorobots follow straightforward rules as a collective without any centralized control structure in interacting with each other and their environment (Mushininga & Ogwu, 2013). The nanorobot's simple rule-guided interactions lead to intelligent collective behavior among the nanorobotic agents. The nanorobots can be used in Nano-medicine environments to reduce toxicity, provide delivery of therapeutic molecules and increase the lifespan of the cycle of drugs within the human body (Medical Device Network team, 2020).

The idea of swarm intelligence was inspired by everyday natural examples, such as a bee colony (Lemmens, 2006), and an ant colony (Chibaya, 2014). It can even be seen in the growth of bacteria, where individual organisms behave collectively but still act individually to grow together and find more sustenance to survive and expand (Niu, 2012). In this research paper, we design and develop a search algorithm inspired by bee behavior in search of food. The intelligent behaviour that is observed from a bee colony has given rise to many existing algorithms that implement swarm intelligence, such as the artificial bee colony (ABC) and the bees' algorithm (BA) (Pharm et al., 2006).

In this research, we seek to determine if swarm intelligence can produce better and more efficient ways of completing tasks than random searching. Furthermore, research has shown that swarm intelligence can produce more complex systems models and strategies (Martinoli, 1999). From this point of view, we want to evaluate group behavior among robots themselves and robot-to-environment interaction. Throughout this paper, the researcher aims to investigate the applications and effectiveness of swarm intelligence used to locate a specific target in a human body-like environment. Bee-inspired algorithms will be investigated as benchmarks for our proposed algorithm.

\* Corresponding author. ORCID ID: 0000-0002-8951-6825

© 2023 by the authors. Hosting by SSBFNET. Peer review under responsibility of Center for Strategic Studies in Business and Finance.

<https://doi.org/10.20525/ijrbs.v12i2.2359>

Controlling a swim of bee-like robots toward a target is a challenge. Each bee-like robot is regarded as simple and naïve. Providing instruction sets to allow coordination in a swarm of bee-like robots poses a possible solution to the challenge. Using the instruction sets will then enhance the search for a target in a human-like environment.

In this study, the researchers focused on applying swarm intelligence to search for a target in a human body-like environment. The outcome of this research is to successfully design, develop and implement a bee-inspired swarm intelligent algorithm for simulating the search of a target in a human-like environment. The use of swarm intelligence in controlling nanorobots can present new ways medical operations are performed and viewed. Swarm intelligence-based control mechanisms have the potential to provide more effective ways to complete medical tasks efficiently and effectively.

## Literature Review

This section will provide a comprehensive overview of the related scholarly resources that have been consulted throughout this research. The section will identify the themes, trends, and conflicts in research methodology, design and findings. We will evaluate the contributions of the literature in the specified field of research and thereafter provide an evaluation of the overall strengths and weaknesses in an effort to clearly define the research gaps. We will critically evaluate the Artificial bee colony, Bee Algorithm and the Particle Swarm Optimisation.

### Artificial Bee Colony (ABC)

The Artificial Bee Colony (ABC) is an algorithm based on swarm intelligence (Wahab, 2015). The algorithm is inspired by the behaviour of bees when they are foraging for food (nectar). The behaviour entail scouting, employment and recruitment of onlooker bees. In the implementation of this algorithm, the artificial bee agents, are classified into three different classes, the employed bee, the onlooker bee and the scout bees (Karaboga & Akay 2009). Each classification is assigned a role, scout bees responsible for finding a new food source and communicating the location. Another class consist of employed bees responsible for the actual carrying of food home where they will meet a class of onlooker bees. The employed bees will use waggle dance to share direction, distance and quality of the source as a form of recruiting onlooker bees (Karaboga, 2005). Given below are the steps involved in the ABC algorithm (Wahab, 2015).

#### Initialization phase

All the vector of the population source,  $\vec{x}_i$  are initialized ( $i = 1 \dots SN$ , where  $SN$  = population size) by the scout bees and control parameters being set. The following equation is used in this phase:

$$x_i = l_i + rand(0,1) * (u_i - l_i)$$

Where  $l_i$  and  $u_i$  are the lower and upper bound parameters of  $x_i$ .

#### Employed bees phase

Here in order to increase the nectar (food source) around the neighbourhood of the food source,  $\vec{x}_i$ , the search of a new food source,  $\vec{v}_i$  will also increase. When a new food source is identified, the profitability and fitness of said source id evaluated. The following formula is used in this phase:

$$v_i = x_i + \phi(x_i - x_j)$$

Where  $x_j$  is a random selected food source and  $\phi$  is a random number of  $[-a, a]$ . Once the new food source,  $v_i$  is produced, its profitability is measured and a greedy selection is applied between  $x_j$  and  $\vec{v}_i$ . The exploration happens if the difference between  $x_i - x_j$  is large and the exploitation behaviour is when the difference is small. The fitness value of the solution  $fit_i(\vec{x}_i)$  is determined using following equation:

$$fit_i(\vec{x}_i) = \begin{cases} \frac{1}{1 + f_i(\vec{x}_i)} & \text{if } f_i(\vec{x}_i) \geq 0 \\ 1 + abs(f_i(\vec{x}_i)) & \text{if } f_i(\vec{x}_i) < 0 \end{cases}$$

Where  $f_i(\vec{x}_i)$  is the objective function value of the solution ( $\vec{x}_i$ )?

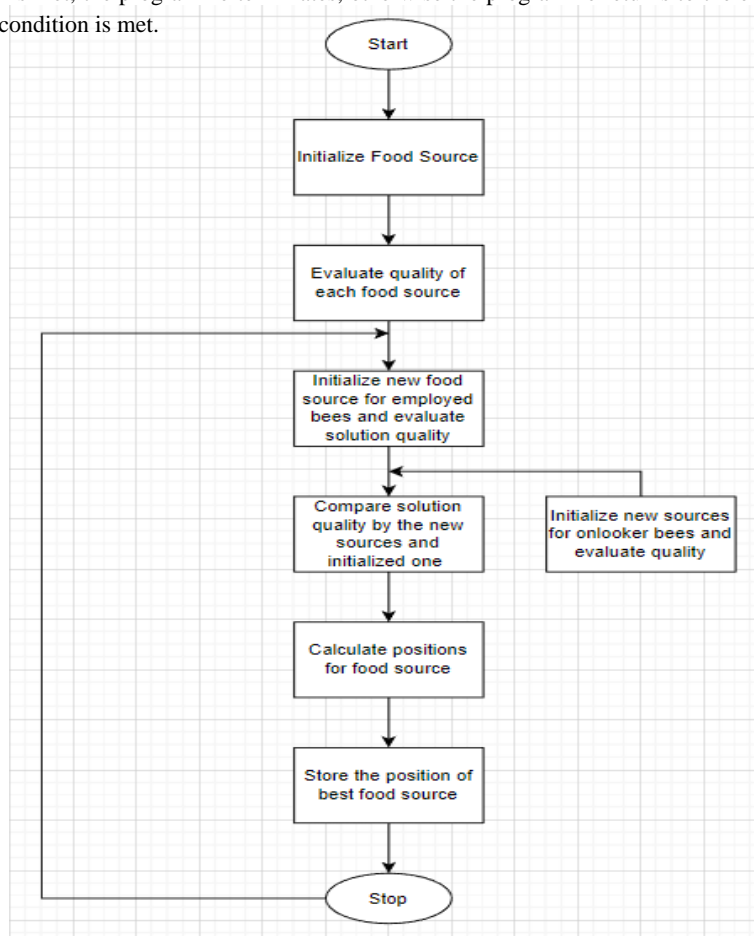
#### Onlooker bees phase

Here the onlooker bees that remain in their hive make their choice of food source based on the probability values measured using the fitness value and the information shared by employed bees. The probability value,  $p_i$ , is measured by the following equation:

$$p_i = \frac{fit_i(\vec{x}_i)}{\sum_{i=1}^{SN} fit_i(\vec{x}_i)}$$

### Scout bees phase

The scout bees are generated from the unemployed bees whose fitness values don't improve. This is called the limit or abandonment criteria. The best fitness value and the position associated to that value are memorized. Termination Checking Phase: If the termination condition is met, the programme terminates, otherwise the programme returns to the employed bees phase and repeats until the termination condition is met.



**Figure 1:** Visual Representation of the Steps Involved in the Abc Algorithm :( Metha, 2022)

### Bee Algorithm (BA)

The Bees Algorithm enhances nanorobotics agents searching capabilities for a target in a search environment (Pham et al., 2006). The algorithm combines the neighbourhood search and random search strategies. In its initial stage the Algorithm randomly dispatch a sizable number of scout bee-like robotic devices into the search space equipped with the capping stopping criteria (Pham et al., 2006). A bee is considered a d-dimensional vector in this algorithm and contains the problem variable and represents an appropriate solution the optimization problem. The solution is then represented as a visited site, in this case a food source, and has a fitness value assigned to it. This algorithm uses scout bees that randomly search for new sites and uses recruitment for neighbourhood search in sites with the higher fitness, respectively. This provides the algorithm with a balance in exploration and exploitation. Bees that have high fitness values are categorized as selected bees and the site that they visit, called elite sites, are chosen for neighbourhood search. The algorithm then assigns more bees to these elite sites for searching (recruitment). The parameters involved in this algorithm include; number of scout bees ( $n$ ); number of sites selected ( $m$ ), out of the  $n$  bees; number of elite sites ( $e$ ), out of  $m$  selected sites; number of bees recruited for the best  $e$  elite sites ( $nep$ ); number of bees recruited for other ( $m - e$ ) selected sites ( $nsp$ ) and the radius for neighborhood search ( $ngh$ ).

Bees Algorithm (Parpinelli & Lopes, 2011) in Table 1:

**Table 1:** Bees Algorithm

<b>1</b>	<b>Parameters: <math>n, m, e, nep, nsp, ngh</math></b>
<b>2</b>	Initialize the bees population $\vec{x}_i$ randomly
<b>3</b>	Evaluate fitness $f(\vec{x}_i)$ of the population
<b>4</b>	while stop condition not met do
<b>5</b>	Select $m$ sites from $n$
<b>6</b>	for each $m$ do
<b>7</b>	Select $nsp$ sites from $m$
<b>8</b>	for each $nsp$ do
<b>9</b>	Perform neighbourhood search with radius $ngh$
<b>10</b>	Update bee position according to $f()$
<b>11</b>	end for
<b>12</b>	end for
<b>13</b>	Select $e$ elite sited from $m$
<b>14</b>	for each $e$ do
<b>15</b>	for each $nep$ do
<b>16</b>	Perform neighbourhood search with radius $ngh$
<b>17</b>	Update bee position according to $f()$
<b>18</b>	end for
<b>19</b>	end for
<b>20</b>	Assign remaining $(m - e)$ bees to search randomly and evaluate their fitness
<b>21</b>	Rank all bees and determine the best
<b>22</b>	end while
<b>23</b>	Post process results and visualization

Some advantages that come with implementing either of these algorithms includes being easy to implement, robust and being highly flexible which makes them usable in almost any situation. This high flexibility means that the algorithm only requires two control parameters of maximum cycle number and colony size. This in turn makes adding or removing a bee or particle fairly simple as you do not have to reinitialize the algorithm. However with these advantages come disadvantages. These include the requirement of new fitness tests for the new parameters to improve performance, being quite slow when used in serial processing, and the need for a high amount of objective function evaluations.

### Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) algorithm, which draws inspiration from the flocking of birds or schooling of fish, operates on the premise that the social sharing of information would be more advantageous to the whole swarm. In the natural example of a schooling of fish members of the fish school can profit from other members discoveries and experiences whenever the resource is distributed beyond prediction. This social behaviour will then increase the adaptability of each individual in the swarm (Moriwaki, 2015). In a PSO algorithm particles are a number of simple entities in a search space. Population of particles are created and their individual fitness is measured with an objective function of the problem. Particles are then moved from their current to the next position based on their personal best location, and on the swarm's best location so far. By iterating the moves the swarm gradually reaches an optimal point of the objective function over generations.

Variable in the algorithm:

```

Number of particles =  $i$ 
Number of dimensions =  $n$ 
Fitness function =  $f(x_i)$ 
Particles =  $x_i$ 
Current velocity =  $v_i$ 
Individual particles best =  $p_i$ 
Global particles best =  $p_g$ 
Inertia component =  $w * v_i(t)$ 
Cognitive component =  $c_1 * r_1 * (p_i - x_i(t))$ 
Social component =  $c_2 * r_2 * (p_g - x_i(t))$ 
Velocity adjustment =  $v_i(t+1) \leftarrow \text{Inertia} + \text{Cognitive} + \text{Social}$ 
Position adjustment =  $x_i(t+1) \leftarrow x_i(t) + v_i(t+1)$ 

```

PSO algorithm outline:

1. Initialise the particle population array  $x_i$
2. Loop
3. For each particle, calculate the fitness using the fitness function
4. Compare the current fitness value with its best  $p_i$ . Replace the best with the current value  $x_i$  if it is better than the best.
5. Check the swarm's best particle from the individual particles best and assign the best array to the global best.
6. Calculate the velocity and update the position of the particles to  $x_i(t+1)$
7. If a criterion is met, exit loop.
8. End loop

**Source:** Moriwaki, 2015

After a critical evaluation of the three algorithms, the ABC and BA and PSO algorithms, we derived some components to embed in the formulation of our own swarm intelligence algorithm mainly inspired by the behaviour of bees. From the ABC algorithm the researchers are more interested in the initial scouting, employment and recruitment. However, recruitment will not be done at the hive but on the search field. When considering the BA algorithm, the researcher is interested in its ability to use scout bees that randomly search for new sites and uses recruitment for neighbourhood search in sites with the higher fitness, respectively. This provides the algorithm with a balance in exploration and exploitation. Bees that have high fitness values are categorized as selected bees and the site that they visit, called elite sites, are chosen for neighbourhood search. Lastly from the PSO algorithm the researcher is interested in the way that each individual member of the swarm can profit from other member's discoveries and experiences whenever the resource is distributed beyond prediction. This social behaviour will then increase the adaptability of each individual member in the swarm.

To formulate our own algorithm literature similar to our work has been reviewed. In the review primitive instructions being used by other algorithms were identified, some of those closely related to our work were considered in the formulation of our algorithm. The concepts generated from the various algorithms will aid in effectively developing a sound simulation that will be able to map out a maze and successfully locate a specific target through the various concepts used in the program. The researcher will then measure the performance by the rate of accuracy, meaning how accurate the program is at identifying the target and the speed at which the particles move and relay the information regarding the target.

## Research Method

To implement our search algorithm we started off by designing the human like environment in a form of a maze. Bee like nanorobotics devices will be deployed in this environment with the mandate of locating a target. To simulate internal structure of a complex human body like environment with veins and arteries maze like environment with dozens of intertwined paths. In order to achieve an optimal maze complexity that closely mimics a human body like environment a measurement of the maze complexity will be done by taking into account various aspects such as:

- i. Size – dimensions of the maze environment
- ii. Shape – Shape or visual representation of the maze environment
- iii. Pathways – Number of available pathways that exist in the maze environment, which makes up the path complexity.
- iv. Obstacles – How densely populated the maze environment is, containing the Nanorobotic agents and other objects.

The generated maze structure is made up of various cells of  $n, m$  as well as an entry and exit point representing the entry point of the Nanorobotic agents and their target respectfully. The maze structure have only one possible solution to reach the target location and the pathways can only be travelled in one direction as to mimic the human body's blood circular flow. The Nanorobotic agents

attempt to find the most optimal solution to find the target, being the exit point of the maze. The researchers evaluated the performance of the overall program by taking into account the various different maze environments and variables present. The movement of the swarm is represented by partial replacement of the path constituting the temporary solution of the generated maze structure. Thereafter path is recorded in the form of vectors related to the cells of the path. Then when viewing the individuals in the swarm they will move in the direction of their temporary individuals, regardless of the temporary path predetermined. The searching procedure of the maze ceases if the stopping criteria of the program reaching the maximum number of iterations is met. Lastly at the end of the experiments the performance will be measured by the rate of accuracy, meaning how accurate the program is at identifying the target and the speed at which the particles move and relay the information regarding the target.

```

Maze algorithm:
initialize:
    maze parameters:  $n, m$  (maze size)
    parameters: population size( $k$ ), visual representation( $visual$ ),
    maximum number of iterations( $Maxit$ ), optimal complexity( $P_g$ )
for  $i=1$  to  $k$  do
    evaluate the quality of the maze complexity //determine temporary
    path
end for
find the best maze complexity ( $P_g$ ) in initial population //start
individual swarm movement
while  $numIteration < Maxit$  do
    for each individual  $i$  do
        for each individual  $j$  do
            if  $j$  complexity > than  $i$  complexity within  $visual$ 
                then direct  $i$  towards  $j$ ; end if
            if complexity = local optimum within  $visual$ 
                then direct  $i$  towards  $P_g$ ; end if
        end for
    end for
    if new complexity is better than  $P_g$ 
        then update  $P_g$ ; end if
    for  $i = 1$  to  $k$  do //random dispersion
        move the  $i$ th individual towards new solution; end for
    if new complexity is better than  $P_g$ 
        then update  $P_g$ ; end if
end while

```

After development of the proposed algorithm, it is crucial that the researcher implement the behaviour for the movement of the individuals, being the swarm particles or Nanorobotic agents, present in the maze environment. The individual movement of the swarm will move in the direction of the temporary path solution or will tend towards another individual in the swarm.

#### Individual swarm movement algorithm:

```

If  $pointNum(path) > visual$ 
then select 2 random points;
update  $i$ th path vector =  $j$ th path vector of the 2 points;
move to path which replaced the segment of path
    if path segment not true
        then redirect to nearest available path; end if
    if corresponding path not equal temporary path
        then update 1 random path element = not true with a
        corresponding path value; end if
    if cell element = path
        then update 1 random path element = a corresponding path;
        end if
end if

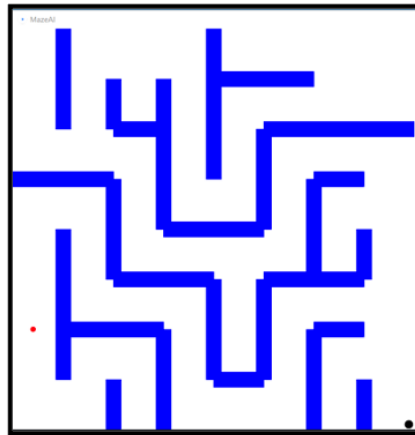
```

## Experiment

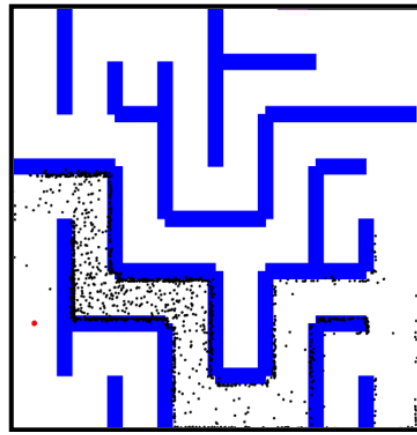
Our algorithm is implemented using visual studio code and the Processor IDE as the algorithm will be implemented using the Java coding language. The hardware used for this research was a single workstation operating on a Windows10 OS powered by the Ryzen 9 processors paired with 16GB RAM which is more than capable of effectively running the proposed algorithm built on the selected software.

#### Environment setup

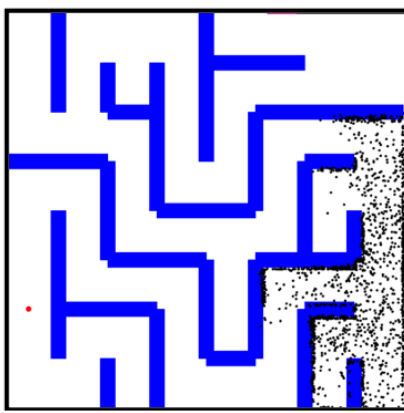
The initial phase of the research is to generate a complex maze environment in relation to the swarm population size in line with the proposed swarm based algorithm. Thus, the collected results will be evaluated based of the process and output of the overall algorithm. To reiterate the methodology used to conduct these experiments will generate the mazes at a 25x25 cell size and will assume that the  $visual = 10$  and most importantly that the maximum number of iterations was set to stop after the desired results were achieved. Lastly the population size was set to 2000 individual swarm particles and the velocity of the swarm was limited to 40m/s. The figures below represent the operation of the implemented algorithm starting from the maze initialization phase and ending once the researcher has gathered the desired results.



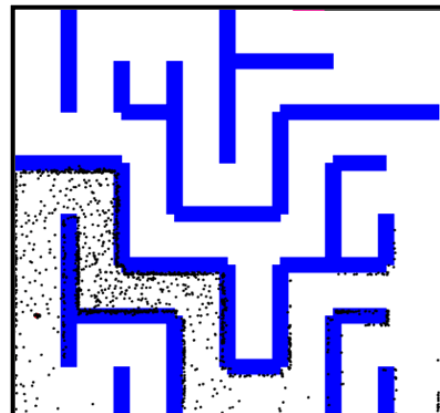
**Figure 2:** Maze initialization phase.



**Figure 4:** Iteration 5 of the algorithm which shows significant improvement in the swarm.



**Figure 3:** First iteration of the algorithm.



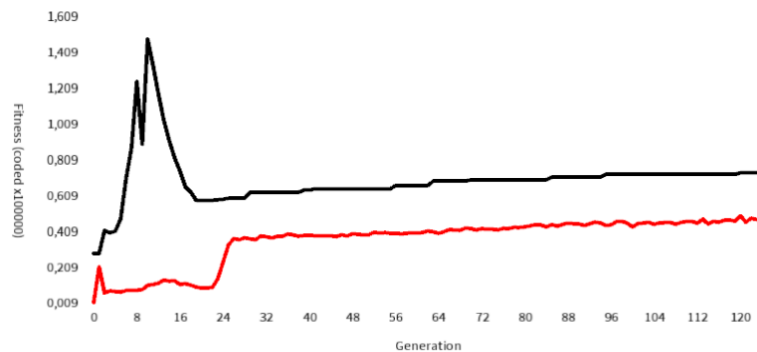
**Figure 5:** Iteration 21 which shows that the swarm particles have reached their target goal.

The maze system is made up of various randomly generated obstacles which are represented by the blue boundary walls. This allowed the researcher to successfully generate a testing environment which could possibly mimic the human body environment. Located outside the blue boundary walls are the entry point for the swarm and their target goal. The black dot located in the bottom right of the maze represents the cluster of swarm particles at the entry point of the maze, whereas the red dot located on the right-hand side of the maze represents the end goal of the swarm particles.

**Table 2:** Data collected from the conducted experiments

Generation/Iteration	Total number of steps	Fewest steps
0	60	0
1	80	0
2	100	0
3	120	0
4	140	0
5	160	0
21	480	404
24	540	381
30	660	374
42	900	360
45	960	357
56	1180	353
68	1420	352
90	1760	350
100	1960	350
110	2160	350
125	3600	350

In the tables above the first column represents the generation or iteration of the program, which is how many times the swarm, has gone through the maze. The second column represents the total number of steps taken by the whole swarm system which is meant to increase by 20 after every generation. Lastly the third column represents the fewest steps taken by the swarm to reach the target. This number only increases after the 21st iteration as the program needs time to learn the best route taken to the end goal, which should also decrease at a steady rate and achieve a constant best after a certain amount of iterations.

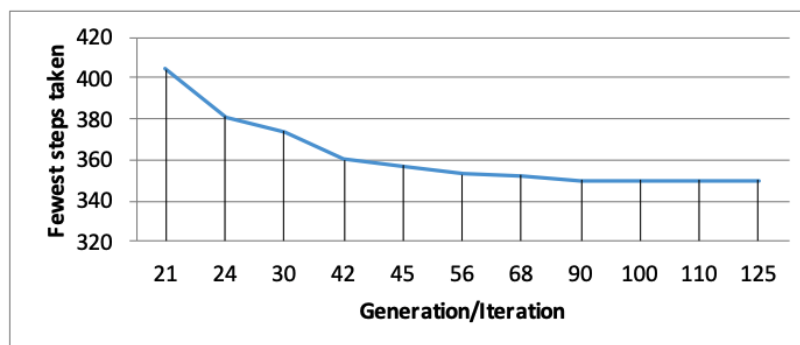


**Figure 6:** A graph representing the calculated fitness of the swarm population.

In the above graph, used to show the calculated fitness of the swarm population, the fitness is shown on the y axis of the graph and the generation or iteration number is shown on the x axis. On the graph there are two distinct lines which we can gather information from. The black line, which will always be more than the red one, represents the maximum fitness of all the iterations and the red line is used to represent the average fitness of the swarm population.

## Analysis and Findings

In order to critically analyse the data collected the researcher will make use of an appropriate statistical analysis tool namely being Microsoft Excel. Firstly the data collected from the simulations will be critically evaluated using various evaluation methods, such as presenting the collected data in a line graph format, which allows the researcher to visually represent the collected data and said data's performance. This is shown by figure 7 below.



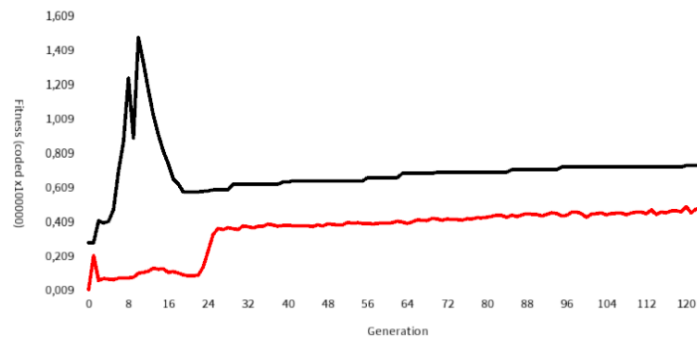
**Figure 7:** Graph representing the fewest steps taken by the swarm.

As seen from figure 7 above the researcher can deduce that after every successful iteration or generation, the swarm particles can be seen to take less and less steps towards reaching the target goal and after a certain amount of iterations the graph above starts to even out as the algorithm has reached the most optimal path from the entry point of the maze to the end target location. It can also be seen that initially the algorithm does not successfully reach the target goal. Only after the 21st iteration does the experiments yield the fewest steps taken. The cause of this is due to the fact that the algorithm first needs to successfully locate the target and thereafter train the algorithm in remembering the best optimal route taken after every generation. The graph also shows that the algorithm gradually gets better and there are no inconsistencies or spikes in the data collected. This means that the algorithm successfully increases the best fitness of the swarm population after every successful iteration. The researcher then evaluated the data collected from these experiments and applied various evaluation metrics in order to obtain the improvement percentage (%) of the swarm population in locating its target goal based on previous generation's ability to locate the target goal. Only the data for the generations that successfully reached the target location will be evaluated with the exception of the first 2 iterations being used as a benchmark evaluation.

**Table 3:** Table representing the improvement % of the swarm population.

Generation/Iteration	Total number of steps	Fewest steps	Improvement percentage (%)
0	60	0	0%
1	80	0	0%
21	480	404	100%
24	540	381	6.04%
30	660	374	1.87%
42	900	360	3.89%
45	960	357	0.84%
56	1180	353	1.13%
68	1420	352	0.248%
90	1760	350	0.57%
100	1960	350	0%
110	2160	350	0%
125	3600	350	0%

From table 3 above the researcher can deduce that the algorithm experiences a high amount of improvement at the initial iteration where the swarm population successfully reaches its target location. Thereafter the improvement gradually decreases with certain spikes but at the end it evens out to a 0% percentage as that is the best improvement that the swarm population can take. Lastly the researcher collected the data regarding the fitness of the swarm population implemented with the proposed algorithm. This is shown in figure 7 as well as figure 8 below.

**Figure 8:** Calculated fitness of the swarm population

The fitness of the swarm population was calculated after each successful iteration of the program. From the experiments only the swarm particles with the best fitness were carried over for use in the next iteration. This ensures that the fitness of the swarm particles would always increase. As seen in the graph above, the fitness starts off very slow initially, but then experiences a sudden spike in the fitness. This means that only a few of the swarm particles completed the task of locating the target goal. There after the fitness drops drastically before consistently increasing at a gradual rate. This means that the more of the swarm particles with each generation achieved a best fitness required to carry over to the next generation. This ensures that the fitness and overall algorithm constantly achieves a better result after every new iteration.

## Conclusions

The developed algorithm was successfully implemented to simulate nanorobots searching for a target in a maze representing a human-like environment. The main aim of the research has been achieved by using swarm intelligence to control bees like Nanorobotic agents to locate a specific target in the human body-like environment. In the process, a complex maze system was created. For the Nanorobotic agents to successfully locate their target goal, the swarm population had to be trained by measuring the fitness of the best swarm particles and then carrying them over to the next iteration while dropping the particles with low fitness. Through the experiments conducted, it was proven that by following the algorithm and rules set, the overall improvement seen by the swarm population in locating their target goal experiences a constant increase in said improvement. This means that the swarm population has a bearing on the accurate location of a target in the complex maze system. This will reduce the risk of errors and improve the overall efficiency of the scheduling process. In the future, the use of other performance metrics, such as quality of emergence, robustness and scalability, will be considered. The developed algorithm was successfully implemented to simulate nanorobots searching for a target in a maze representing a human-like environment. The main aim of the research has been achieved by using swarm intelligence to control bees like Nanorobotic agents to locate a specific target in the human body-like environment. In the process, a complex maze system was created. For the Nanorobotic agents to successfully locate their target goal, the swarm population had to be trained by measuring the fitness of the best swarm particles and then carrying them over to the next iteration while dropping the particles with low fitness. Through the experiments conducted, it was proven that by following the algorithm and rules set, the

overall improvement seen by the swarm population in locating their target goal experiences a constant increase in said improvement. This means that the swarm population has a bearing on the accurate location of a target in the complex maze system. This will reduce the risk of errors and improve the overall efficiency of the scheduling process. In the future, the use of other performance metrics, such as quality of emergence, robustness and scalability, will be considered.

## Acknowledgements

I would like to thank the co-author of this paper for the effort he put in this research to ensure successful formulation and eventual submission of this paper. All authors have read and agreed to the published version of the manuscript.

**Author Contributions:** Conceptualization, Moodle L.; Methodology, Mushininga R and Moodley L.; Formal analysis, Mushininga R and Moodle L.; Investigation Mushininga R and Moodley L.; Writing--original draft preparation, Mushininga R and Moodley L.; Writing--review and editing, Mushininga R and Moodley L.;

**Funding:** This research was funded by The Independent Institute of Education, MSA, South Africa,

**Informed Consent Statement:** Informed consent was obtained from all subjects involved in the study.

**Data Availability Statement:** The data presented in this study are publicly available in this article. There are no data access restrictions.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Chibaya, C. (2014). An Investigation into Xsets of Primitive Behaviour for Emergent Behaviour In Stigmegic and Message Passing Ant-like Devices. PhD thesis Submitted at Rhodes University, South Africa.
- Frey, B. B. (2018). Pragmatic paradigm. The SAGE encyclopedia of educational research, measurement, and evaluation. SAGE: <https://methods.sagepub.com/>
- Karaboga,D.,Akay, B. (2009). Artificial bee colony (abc), harmony search and bee algorithm on numerical optimization. In 2009 innovative production machines and systems virtual conference (IRPOMS 2009).*Behavioral Ecology*, Volume 18, Issue 2, March 2007, Pages 384–392, <https://doi.org/10.1093/beheco/arl095>
- Lemmens N. (2006) To bee or not to bee: A comparative study in swarm intelligence. In Thesis submitted at Maastricht University.
- Liu, Y., & Passino, K. M. (2000). Swarm intelligence: Literature overview. *Department of electrical engineering, the Ohio State University*. <http://www2.ece.ohio-state.edu/>
- Manjunath, A., & Kishore, V. (2014). The promising future in medicine: nanorobots. *Biomedical Science and Engineering*, 2(2), 42-47. <https://doi.org/10.12691/bse-2-2-3>
- Martinoli, A. (1999). *Swarm intelligence in autonomous collective robotics: From tools to the analysis and synthesis of distributed control strategies* (Doctoral dissertation, Verlag nicht ermittelbar). 181. <https://citeseerx.ist.psu.edu/>
- Medical Device Network team. (2020). Nanotechnology in Medicine – Thematic Research. *GlobalData Thematic Research*, 28.
- Metha, S. A. (2022). Artificial bee colony and its applications to optimization problems. *Developers Corner*, 50.
- Moriwaki, K. (2015). *Swarm Intelligence: Coding and Visualising Particle Swarm Optimisation in Python*. Retrieved September 25, 2022, from towardsdatascience: <https://towardsdatascience.com/swarm-intelligence-coding-and-visualising-particle-swarm-optimisation-in-python-253e1bd00772>
- Mushininga, R & Ogwu, F.J. (2013).Nanorobotic Agents Communication Using Bee-Inspired Swarm Intelligence, *Wireless Sensor Network*, Vol. 5 No. 10, 2013, pp. 208-214. <https://doi.org/10.4236/wsn.2013.510024>.
- Muthukumaran, G., Ramachandraiah, U., & Samuel, D. H. (2015). Role of nanorobots and their medical applications. *Advanced Materials Research*, 1086, 61-67. <https://doi.org/10.4028/www.scientific.net/AMR.1086.61>
- Niu, B., & Wang, H. (2012). Bacterial colony optimization. *Discrete Dynamics in Nature and Society*, 2012. <https://doi.org/10.1155/2012/698057>
- Parpinelli, R. S., & Lopes, H. S. (2011). New inspirations in swarm intelligence: a survey. *International Journal of Bio-Inspired Computation*, 3(1), 1-16. <https://doi.org/10.1504/IJBIC.2011.0387>
- Pham, D. T., Ghanbarzadeh, A., Koç, E., Otri, S., Rahim, S., & Zaidi, M. (2006). The bees algorithm—a novel tool for complex optimisation problems. In *Intelligent production machines and systems* (pp. 454-459). Elsevier Science Ltd. <https://doi.org/10.1016/B978-008045157-2/50081-X>
- Zhang, Y., Agarwal, P., Bhatnagar, V., Balochian, S., & Yan, J. (2013). Swarm intelligence and its applications. *The Scientific World Journal*, 2013. <https://doi.org/10.1155/2013/528069>

**Publisher's Note:** SSBFNET stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2023 by the authors. Licensee SSBFNET, Istanbul, Turkey. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

International Journal of Research in Business and Social Science (2147-4478) by SSBFNET is licensed under a Creative Commons Attribution 4.0 International License.