Performance Analysis of DFT and COA Algorithms for Wireless Sensor Network Node Localization

Dhuha Gh. Kadhum
Computer department
University of Diyala
Diyala, Iraq
scicompms222311@uodiyala.edu.iq

Muntadher Khamees
Computer department
University of Diyala
Diyala, Iraq
alkarawis@gmail.com

***Abstract*— In a wireless sensor network, nodes localization seeks to use known (main or anchor) nodes to help detect the locations of the unknown (target) nodes. The localization accuracy can have a big impact on a WSN's performance. This paper suggests a strategy for node localization based two recently developed bioinspired algorithms, the Crayfish Optimization Algorithm (COA) and the Flow Direction Algorithm (FDA). In several WSN deployment scenarios, the suggested techniques are contrasted with each other. On the hand of localization error and time of computation, the testing findings demonstrate that the suggested FDA localization scheme outperforms the alternative localization technique.**

***Keywords*— metaheuristic algorithms, WSN, anchor node, target node, localization**

I. Introduction

In recent years, wireless sensor networks (WSNs) have drawn interest from all over the world, especially with the growth of Micro-Electro-Mechanical Systems (MEMS) technology, which has made the creation of smart sensors easier [1]. a metaheuristic is an advanced process or heuristic that is intended to locate, produce, adjust, or choose a heuristic that could offer an adequate answer to an optimization or machine learning issue [2] [3].

In the past several years, there have been numerous research attempts on this issue among the scientific community. It should be noted that the definition of localization is the process of determining an unknown node's position, either by employing connectivity information between the unknown nodes or by utilizing nodes with known positions. Recent research has examined how movement affects localization.[4], [5], [6], real world applications [7], [8], [9], “Anchor Free” and “Anchor Based” localization techniques [10], “Range Based and Range Free” schemes of localization [11], “Non-Cooperative” schemes—where the target nodes only connect with the anchor nodes—and “Cooperative” algorithms—where communication occurs among all nodes[12], “The centralized” scheme localization and “the distributed” scheme, which uses locally collected information to determine each node's position without central supervision.[13][14]. This paper's primary contribution is the first-ever localization of WSN nodes utilizing the DFT and the COA. Analysis and comparison are conducted between these schemes.

Regarding to the time of computing and localization error, results showed that the DFT-based localization schemes outperform the COA localization scheme. The paper's remaining sections are arranged as follows: A selection of the field research projects is covered in Section 2. A brief overview of the swim algorithms used in this work is given in Section 3. The suggested DFT and COA-based localization schemes are presented in Section 4. The findings analysis and conducted experiments are included in Section 5. In Section 6, the paper is finally concluded.

II. Literature Review

Numerous optimization strategies have been used in recent years to solve the node localization issue in WSNs. A brief description and coverage of a few recent pertinent works are provided in this section.

In 2020, Rong Tan and etc.., developed and implemented the DMA node localization algorithm, by, demonstrates its sensor node localization theory and presents a potential foundation for placement to be realized in WSNs and the results showed that the suggested strategy performing are better for energy consumption and accuracy of localization than other popular approaches, which presents a chance to meet the need for high-precision sensor node localization in the development of WSNs [15].

In 2020, Drs. D. Chandirasekaran and S. Sugumaran tried to determine the nodes' positions Using experiments and simulations, using Cat Swarm Optimization (CSO), a novel swarm-based optimization technique modelled after the behaviour of cats and Results using the CSO algorithm have been found to be significantly better than those obtained using Particle Swarm Optimization (PSO), another well-known swarm-based optimization approach. The CSO algorithm's fast searching feature made it possible to locate wireless sensor network node localization more quickly while maintaining the highest level of positioning accuracy and stability [16].

In 2020, Sana Messous and Hend Liouane introduced An Online Sequential DV-Hop technique to improve accuracy localization node to multihop WSNs by progressively calculating node positions. The simulation results demonstrate that the suggested algorithm notably decreases the average localization error of sensor nodes when compared to the original DV-Hop and other localization techniques discussed in the literature [17].

In 2021, Pudi Sekhar and etc.., was developed effective metaheuristic group learning optimization scheme for localization the node GTOA/NL scheme for Wireless sensor network SN-enabled indoor communication. The results obtained guarantee that the model does best than another technique in different transmission range, ranging error, and the number of main nodes [17].

In 2021, Sana Messous and etc.., suggested an updated version of the technique for less the significant error of localization in original DV/Hop scheme. According to experimental findings, the suggested localization method increases localization accuracy while reducing localization error [17].

In 2022, Himanshu and etc.., showed by using a single mobile anchor node, artificial intelligence applications for target node placements in WSNs. To detect ideal locations for target nodes, the following methods are applied independently: Firefly Algorithm (FA), Hybrid PSO (HPSO), and Particle Swarm Optimization (PSO) and the findings demonstrate that, in comparison to current methods, the suggested schemes do best on the hand of energy, time of convergence, and the accuracy [18].

In 2022, Wenyan Liu et al. suggested a node localization technique depend on the strategy location of main node selection to more effectively address the conflict between the placement of anchor nodes in wireless sensor networks, localization accuracy, and coverage of localization. The suggested technique outperforms the standard algorithms that are already in use in terms of localization coverage and accuracy, according to the result [19].

In 2023, Yuxiao Cao and etc.., colleagues introduced a DV Hop based location scheme for WSNs that makes use of optimal anchor nodes subsets and the outcomes show that, in a variety of network scenarios, the OANS DV Hop scheme has a better accuracy of localization than both the first DV Hop and other enhanced DV Hop schemes [20].

In 2023, Bahadur and etc.., introduced a novel genetic algorithm-based approach to optimize energy usage in wireless sensor networks. The comparison's findings suggested that these logarithms and techniques might lessen that energy, albeit to different degrees. It has been found that using the recommended methods could result in a 50% decrease in energy use [21].

In 2023, The Multi-population Firefly Algorithm Based Node Deployment in Underwater Wireless Sensor Networks, which Annapurna presented, demonstrates how MFA may produce more residual energy and improved deployment accuracy with lower error and expense [22].

III. Intelligent Swarm Algorithms

The collective activities of self-organized systems are the foundation of intelligence of swarm. Ant Colony System (ACS), Artificial Bee Colony (ABC), Bacteria Foraging (BF), Stochastic Diffusion Search (SDS), Particle Swarm Optimization (PSO)… and other common SI systems are examples. In addition to its applications in traditional optimization problems, SI may also be utilized in the control of robotics and unmanned vehicles, prediction of social behaviours, improvement of communications and computer networks, and more. Swarm optimization can be effectively utilized in several domains such as engineering and social sciences [23][24]. In this work, we examine two swarm intelligence schemes for optimization problems and several comparisons are made between these algorithms.

A. Crayfish Optimization Algorithm (COA)

The crayfish has a hard shell and resembles a shrimp. It is a member of the Decapoda, Crustacea, and Arthropoda groups in animal taxonomy. It is typically regarded as an important species for freshwater habitats [25]. The behaviors of crayfish during foraging, summer vacations, and competition inspire the COA. COA's exploitation stages include foraging and competition, while the exploration stages are represented by the summer resort periods. At the start of the process, the crawfish colony X is defined to represent the features of swarm intelligence optimization. The ith crayfish's location, Xi, denotes a solution. (Xi = {Xi,1, Xi,1, Xi,1...Xi,dim}, where dim, usually referred to as dimension, is the characteristic quantity of the optimisation issue). The function f (·) is introduced by Xi in order to get the fitness value, or solution [26]. Temperature, a random constant representing the environmental temperature, regulates COA's exploration and exploitation. When temperature rises very high, COA enters summer resort stage or the competitive. In summer resort stage, the new solution is updated based on the cave position Xshade and the individual position Xi. When the temperature is optimal, COA transitions to the foraging stage. During foraging, the best position or optimal solution is where the food is located.

Food size is determined by the current solution, fitnessfood (obtained by the best or optimal solution), and the optimal solution, fitness (obtained by Xi). Crayfish obtain new foods based on their position (Xi), food position (Xfood), and food intake (constant p) when the food is suitable. The crayfish divides up enormous meals with its claw foot and then eats with its second and third walking feet, alternately [26].

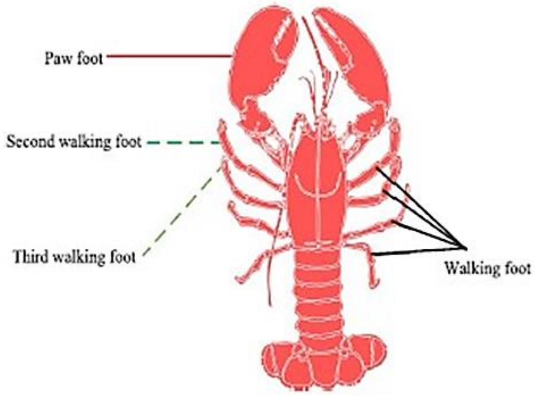

Fig. 1. Structure diagram of COA [26]

1) Initialization of population

Every crayfish in the multidimensional optimization problem is represented by a 1 × dim matrix. Each column matrix represents the problem's solution. The upper and lower bounds of the set of variables (Xi,1, Xi, 2..., Xi,dim) must contain all of the variables Xi.

A set of potential solutions X is randomly generated as the COA's initialization in the space.

It is suggested that the solution candidate X be used depended on the number of the population N and the dimension of area dim [27]. The initialization of COA scheme is showed in Equation (1).

$$X = [X_1, X_2, \dots, X_N] = \begin{bmatrix} X_{1,1} & \dots & X_{1,j} & \dots & X_{1,dim} \\ \vdots & \cdots & \vdots & \dots & \vdots \\ X_{i,1} & \dots & X_{i,j} & \dots & X_{i,dim} \\ \vdots & \cdots & \vdots & \dots & \vdots \\ X_{N,1} & \dots & X_{N,j} & \dots & X_{N,dim} \end{bmatrix} \quad (1)$$

where N is populations number, dim is dimension of the population, and $X_{i,j}$ describes the location of individual i in the j dimension. The value of $X_{i,j}$ is obtained from Equation (2).

$$X_{i,j} = lb_j + (ub_j - lb_j) \times rand \quad (2)$$

where $rand$ is a random number and lb_j and ub_j denote the lower and upper bounds of the jth dimension, respectively [28].

2) Define the temperature and feed of crayfish

Crayfish will exhibit behavioral changes and progress through distinct stages due to temperature shifts. Equation (3) defines temperature. When the temperature reaches above 30°C in the summer, crayfish will choose a cool location. When the temperature is right again, the crayfish will resume feeding. The quantity of crayfish that feed is influenced by temperature. The optimal feeding range for crayfish is between 15 and 30 degrees Celsius. As a result, it is possible to roughly estimate how much crayfish to feed according to their regular distribution, with temperature having an impact. Because between 20 and 30 °C, crayfish exhibit robust feeding behavior. As a result, the COA specifies a temperature range of 20 to 35 °C [26]. Equation (4) displays the crayfish intake mathematical model.

$$temp = rand \times 15 + 20 \quad (3)$$

where temp, is the crayfish's location's temperature.

$$p = C_1 \times \left(\frac{1}{\sqrt{2\times\pi * \sigma}} \times exp\left(-\frac{(temp-\mu)^2}{2\sigma^2}\right)\right) \quad (4)$$

Among them, μ denotes the perfect crayfish temperature, and C_1 and σ are utilized to organize crayfish intake in various temperatures.

3) Stage of summer resort (exploratory phase)

The crayfish take sanctuary in a cave during their summer season when the temperature rises beyond thirty degrees. Here is how the cave X_{shade} is described:

$$X_{shade} = (X_G + X_L)/2 \quad (5)$$

where X_L denotes the optimal position of the current population and X_G is the optimal position reached thus far based on the number of iterations.

Random fights break up between crayfish over caverns. The crawfish will enter the cave unhindered and be prepared for summer when rand is less than 0.5, which means that no other crawfish are vying for the cave. The crayfish plans to spend the summer in the cave by using (6) [29].

$$X_{i.j}^{t+1} = X_{i.j}^{t} + C_2 \times rand \times \left(X_{shade} - X_{i.j}^{t}\right) \quad (6)$$

According to (7), C_2 is a declining curve, where t denotes the iteration number of the current generation and t+1 the iteration number of the next generation.

$$C_2 = 2 - (t/T) \quad (7)$$

where T is the maximum number of iterations that are permitted.

Crayfish try to go to the cave, which stands for the best course of action, as they progress through the Summer Resort stage. The crayfish will now begin to migrate in the direction of the cave, increasing COA's potential for exploitation and bringing them closer to the best course of action [30]. This procedure helps the algorithm to converge more quickly.

4) The Exploitation Phase, or Competition Stage

When rand ≥ 0.5 and temp > 30, it indicates that other crayfish are interested in the cave. They are going to battle over who gets to keep the cave. Equation (8) is used by the crayfish to compete for the cave [27].

$$X_{i.j}^{t+1} = X_{i.j}^{t} - X_{z.j}^{t} + X_{shade} \quad (8)$$

where according to equation (9), z stands for the random individual of crayfish.

$$z = round\left(rand \times (N-1)\right) + 1 \quad (9)$$

During the Competition stage, crayfish compete with one another.

with crayfish Xi changing positions in reaction to $Xz's$ position. This positioning change expands COA's search range, which strengthens the algorithm's capacity for exploration [26].

5) Phase of foraging (exploitation)

It's best to feed crayfish when the temperature is below thirty degrees. The crayfish will start to move towards the food. Once they've located it, they'll measure it. The crayfish will use their claws to break up huge prey before using their second and third walking legs to eat it [30]. X_food is a food location that is described as:

$$X_{food} = X_G \quad (10)$$

Q represent the food size that defined as:

$$Q = C_3 \times rand \times \left(fitness_i / fitness_{food}\right) \quad (11)$$

The food component, $C3$, always has a value of 3, which stands for the largest food. The ith crayfish's fitness value is indicated by the notation $fitness_i$, but the food location's fitness value is shown as $fitness_{food}$.

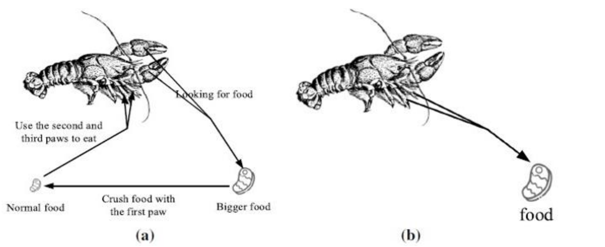

Fig. 2. (a) Before eating, crayfish shred their food. (b) Crayfish consume food right away [26]

The crayfish will use its first claw foot, as shown in Fig. 2(a), to break down the meal. The following is the mathematical formula [29]:

$$X_{food} = exp\left(-\frac{1}{Q}\right) \times X_{food}, \quad (12)$$

The food will be picked up and put in the mouth using the second and third paws as it gets smaller and shreds. Using simulation, the alternating process is recreated by combining the sine and cosine functions. As shown in Fig. 2(b). Additionally, because crayfish intake and food availability are correlated, the foraging equation is as follows:

$$X_{i.j}^{t+1} = X_{i.j}^{t} + X_{food} \times P \times (\cos(2\times\pi\times rand) - \sin(2\times\pi\times rand)) \quad (13)$$

When $Q \le (C3+1)/2$, the crayfish are just interested in approaching and immediately devouring the food:

$$X_{i.j}^{t+1} = \left(X_{i.j}^{t} - X_{food}\right) \times P + P \times rand \times, X_{i.j}^{t} \quad (14)$$

Crayfish use a range of feeding strategies during the foraging stage, depending on the size of their meal Q, the most effective strategy is using the food X_{food}. The crayfish will come over and eat it when it is tiny enough. A large value for Q suggests a substantial discrepancy between the optimal solution and the actual situation. X_{food} must thus be lowered and moved closer to the food source. COA will advance towards the best option during the foraging phase, increasing the scheme's potential for exploitation and encouraging notable convergence [29].

B. *Flow Direction Algorithm (FDA)*

The quantity of precipitation that falls on the surface of the land without penetrating the soil is referred to as excess or effective rainfall in a drainage basin. After precipitation, this is essentially the water that remains on the surface (direct runoff), taking into account losses from evapotranspiration, infiltration, and interception. Various methods have been proposed thus far to determine direct runoff, one of which is the ϕ-index method [30].

The index ϕ, expressed in cm/hr., indicates the average amount of water lost during rainfall. Rainfall that falls above this level turns into runoff right away. In other words, the high-level total of the index ϕ equals the height of the direct runoff. Subtracting the index ϕ from the rainfall at each time period yields the direct runoff.

Fig. 3 presents the idea of the ϕ-index [31]. The direct runoff calculation method is expressed as follows:

$$r_d=\sum_{m=1}^{M}(r_m - \phi\Delta t) \quad (13)$$

where the parameters r_d, r_m, Δt, and M indicate in that order, the amounts of rainfall, time interval, number of time steps, and direct runoff.

Precipitation losses, including infiltration, evapotranspiration, and interception, are deducted from the total precipitation to calculate direct runoff. The slope of the terrain determines how this runoff travels to the basin's exit. The drainage basin can be divided into multiple cells to simulate this process, with each cell transferring its runoff to neighbouring cells based on their height and slope.

The D8 approach [34], which makes the assumption that flow goes to one of the eight surrounding cells [35], is one of the most used techniques for estimating runoff direction. Every cell has eight neighbors with this method, and each neighbor has a unique height and distance from the cell. The direction of flow is determined by calculating the height and distance disparities between each cell and its surrounding cells.

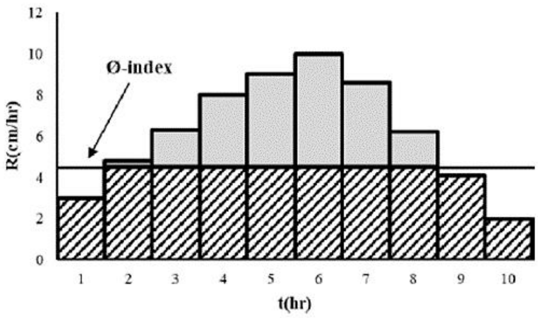

Fig. 3. Ø-index diagram

The flow from each cell is then directed towards the cell with the steepest slope once the slope of each cell is ascertained. Fig. 4 shows the flow pattern as well as the eight surrounding cells. Fig. 5 depicts the D8 technique's schematic design.

Lastly, the flow direction throughout the basin is ascertained by using the D8 algorithm. Following the specification of the flow direction, a value representing the number of cells flowing into each cell is considered, with the maximum number occurring at the basin output point. A cell is also said to have a depression (or hole) if it needs to be filled and has a lower height than the cells around it. Fig. 6 illustrates the location of a depression inside a canal.

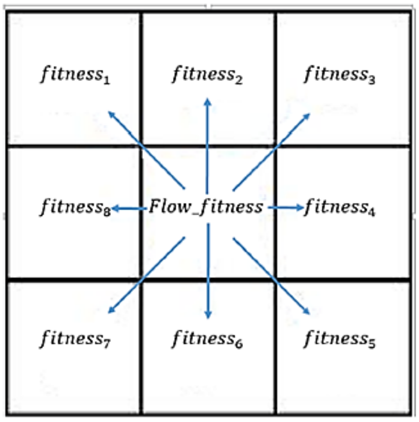

Fig. 4. Flow and the eight positions around it

1) *Principal Concept*

After rainfall is transformed to runoff, the FDA algorithm estimates the direction of flow in a drainage basin using the D8 technique. Initially, this process creates a population in the problem's drainage basin or search space. The flows then go in that direction in an attempt to achieve the optimal solution or the lowest altitude outlet point.

The execution of the algorithm is based on the following assumptions:

1. Every flow has a precise height and place.
2. Every flow is surrounded by β locations, each with an objective function or height.
3. There is a strong correlation between slope and flow movement velocity.
4. The flow moves in the direction of the lowest altitude with a velocity V.
5. The basin departure point is determined by the flow position that has the best objective function.

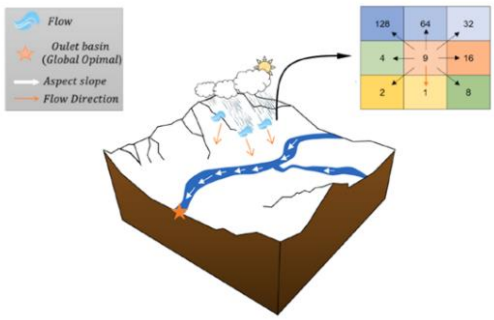

Fig. 5. Diagram showing the D8 technique and the movement of flow to the basin's exit

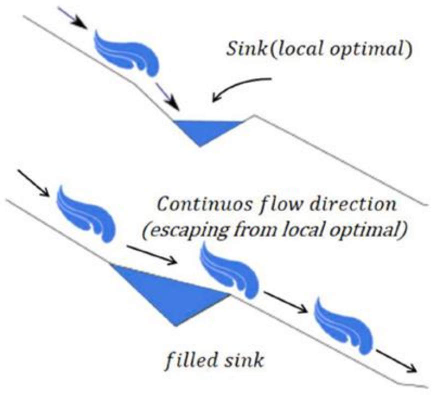

Fig. 6. Sink placement both before and after filling

The neighborhood radius Δ, population number α, and number of neighbors β make up the algorithm's initial parameters. The equation below is utilized in relation to the FDA algorithm in order to ascertain the flowing beginning position:

$$Flow\_X(i) = lb + rand \times (ub - lb) \tag{16}$$

If the position flow is represented by $Flow\_X(i)$, then the lower and upper bounds of the decision variables are denoted by lb and ub, respectively, and a random value uniformly distributed between zero and one is represented by *rand*. Furthermore, it is believed that every flow has β neighbourhoods, whose locations are established by the connection given below:

$$Neighbor\_X(j) = Flow\_X(i) + randn \times \Delta \tag{17}$$

Neighbor_ X denotes the location of the jth neighbor, and randn random number with a standard deviation of one, a mean of zero, and a normal distribution. A small value of Δ results in a narrower search range, whereas a big value of Δ opens up a wider range of possible search results. Finding near-optimal solutions is more likely when a broad search is conducted (global search).

It is imperative to strike a balance between these two strategies: A more concentrated search radius makes it easier to locate the global optimal solution more precisely when the algorithm's solutions approach the global optimum (local search). If one uses solely global search operators, the algorithm might not be able to locate the global optimum precisely enough. On the other hand, the algorithm may become stuck in local optima if just local search is carried out.

In order to balance local and globally search, this work employs a formula that linearly decreases the value of Δ from large too small. This ensures greater variation and directs the search towards random positions.

$$\Delta = \left(rand \times Xrand - rand \times Flow\_X(i)\right) \times \| Best\_X - Flow\_X(i) \| \times W \tag{18}$$

W is a nonlinear weight with a random integer between 0 and ∞, $Xrand$ is a random location generated by relation (15), and $rand$ is a random number with a uniform distribution. The first term of this connection states that $Flow\_X(i)$ goes to a random place ($Xrand$).

As the number of iterations increases in the second term, the Euclidean distance between $Best\_X$ and $Flow\_X(i)$ decreases to zero, making local search impossible. The third term's W computation looks like this:

$$W = \left(\left(1 - \frac{iter}{Max\_Iter}\right)^{(2\times randn)}\right) \times \left(\overline{rand} \times \frac{iter}{Max\_Iter}\right) \times \overline{rand} \tag{19}$$

where $\overline{rand}$ is random vector has a uniform distribution.

The flow moves towards the neighbor with the lowest goal function at a speed of V, as was previously mentioned. Moreover, the slope has a direct impact on the flow's velocity towards its neighbours. As a result, the flow velocity vector can be found using the following equation:

$$V = randn \times S_0 \tag{20}$$

where S_0 is sloping vector between the flow's neighbour and current position. The global search space is enlarged and a variety of solutions are generated by the random number generator, or randn. The slope vector of the ith flow with respect to its jth neighbour can be found using the following equation:

$$S_0 = \frac{Flow\_fitness(i) - Neighbor\_fitness(j)}{\|Flow\_x(i,d) - Neighbor\_x(j,d)\|} \tag{21}$$

where $Neighbor\_fitness(j)$ and $Flow\_fitness(i)$, represent, respectively, the flow i's and the neighbor j's goal values. The parameter d indicates the dimensions of the problem. The new position of the flow is found using the following formula:

$$Flow\_newX(i) = Flow\_X(i) + V \times \frac{Flow\_X(i) - Neighbor\_X(j)}{\|Flow\_x(i) - Neighbor\_x(j)\|} \tag{22}$$

where $Flow\_newX(i)$.represents flow i's new location.

It is crucial to remember that any neighbor's objective function must be less than the flow's own in order to identify the direction of flow. This idea is comparable to how a washbasin fills. To simulate this circumstance, the FDA technique randomly chooses a new flow, which moves in the direction of the present flow if its objective function is lower. If not, it will travel in the direction of the slope that is most prevalent. The way to describe the flow direction under these conditions is shown by the following equation:

$$\begin{cases} if\ Flow\_fitness(r) < Flow\_fitness(i) \\ Flow\_newX(i) = Flow\_X(i) + \overline{randn} \times \left(Flow\_X(r) - Flow\_X(i)\right) \\ else \\ Flow\_newX(i) = Flow\_X(i) + 2randn \times \left(Best\_X - Flow\_X(i)\right) \\ end \end{cases} \tag{23}$$

where a is a random integer.

IV. WSN Localization Problem Formulation

The problem of localization for wireless sensor network nodes may be defined as a single hop range-based distribution strategy, which involves estimating the target (unknown) nodes'(X, Y) position with the help of the main nodes' coordinates (x,y), which act the location of the known nodes. Because main nodes come with GPS unit, they can figure out where they are on their own. Because GPS is so expensive, the majority of WSN nodes are not outfitted with it. The steps used are shown below in order to calculate the coordinates of the N target (unknown) nodes.

Step 1: Within communication range (R), randomly establish M anchor nodes and N unknown nodes. Anchor nodes use positional awareness to tell their neighbours their coordinates. The node that settles at the conclusion of each cycle is referred to as the reference node, and it serves as the anchor node.

Step 2: A node is deemed localized if 3 or more main (anchor) nodes are present inside the range of its connection.

Step 3: Assign (x,y) to the target node's coordinates and d_i to the separation between both of target (un known) and ith anchor (main) node.

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2} \tag{25}$$

Step 4: The localization problem's error is minimized by formulating the optimization problem. The localization problem's objective function is expressed as:

$$f(x, y) = min\left(\sum_{i=1}^{M}\left(\sqrt{(x - x_i)^2 + (y - y_i)^2}\right)^2\right) \tag{26}$$

where M denotes main nodes that are inside the target node's transmission range.

Step 5: After determining each unknown localized node (N_L), the total error of localization is computed as mean square of the difference between the predestined and the really coordinates node x_i, y_i, for i = 1,2, 3... N_L:

$$E_L = \frac{1}{N_L}\sum_{i=1}^{L}\left(\sqrt{(x_i - X_i)^2 + (y_i - Y_i)^2}\right) \tag{27}$$

Step 6: Go back to step 2 and repeat through 5 till no more nodes can be located or until all unknown/target nodes have been localized.

V. Experimental Analysis

This section compares the performance of the proposed WSN schemes in terms of localization error and localization time. The strategy is assessed under different situations. The calculation of various algorithms is performed using MATLAB R22021b on Intel Core (TM) i5 CPU, Windows10 operating system and 8 GB RAM. The parameters of the shipping point values are shown in Table 1.

TABLE I. THE SETTING OF THE PARAMETERS

The Parameters	**The Values**
The target (un known) nodes	differ on $\sum_{k=1}^{6} k * 25$
The main (anchor) nodes	differ with increase k=k+5
Range of transmission	30 meters
Space of work	100 meters × 100 meters

VI. The Comparison Between Schemes

In this part, DFT and COA schemes have been examined regard to the time of localization, and the error of localization under various conditions. Table 2 displays the outcomes of the two schemes that were obtained.

TABLE II. COMPRESSION OF THE TWO LOCALIZATION SCHEMES RESULTS

No. of target nodes	**No. of anchor nodes**	**No. of iteration**	**COA**		**DFT**	
			Time (s)	E_L	**Time (s)**	E_L
25	10	25	27.72	0.0385	23.10	0.0345
		50	53.28	0.0382	48.23	0.0339
		75	82.60	0.0340	69.41	0.0322
		100	109.82	0.0335	93.96	0.0322
50	15	25	111.24	0.0134	51.55	0.0130
		50	199.76	0.0132	111.26	0.0122
		75	271.70	0.0134	160.42	0.0118
		100	361.58	0.0132	229.97	0.0113
75	20	25	182.01	0.0112	93.72	0.0100
		50	460.54	0.0110	184.91	0.0092
		75	603.20	0.0107	276.38	0.0079
		100	840.81	0.0100	387.93	0.0075
100	25	25	413.52	0.0069	154.47	0.0069
		50	788.64	0.0060	269.26	0.0069
		75	1003.72	0.0055	419.31	0.0044
		100	1298.42	0.0050	576.58	0.0035
125	30	25	627.58	0.0029	202.47	0.0029
		50	1180.40	0.0027	396.33	0.0027
		75	1617.52	0.0018	578.22	0.0018
		100	2398.37	0.0015	719.72	0.0015
150	35	25	699.08	0.0030	243.85	0.0011
		50	1295.98	0.0028	518.05	0.0011
		75	1968.03	0.0022	788.95	0.0010
		100	3254.45	0.0018	1034.61	0.0010

It is shown that for the localization strategies, in all cases (the number of the unknown (target) nodes and the number of the main (anchors) nodes), increasing in iteration leads to a reduction in localization error, but an increase in the number of localizations and computation time. This seems to cover the goal, as more iterations equal more calculations and take longer to complete. On the contrary, the greater the number of actions, the more likely we are to reach a better solution. As a result, there are more localized nodes and the localization error value that explains the location error (E_L) for relation between these main and unknown nodes. However, as the number of targets and anchors increases, it turns out that DFT outperform COA in this area. Regard to the computation time, it was observed that increase number for both target (un known) and anchor (main) nodes increases time of computing for the whole localization's schemes. But still: Compared with COA, DFT's computation time is better. The next two figures show schematic diagrams of experiments conducted at different scales.

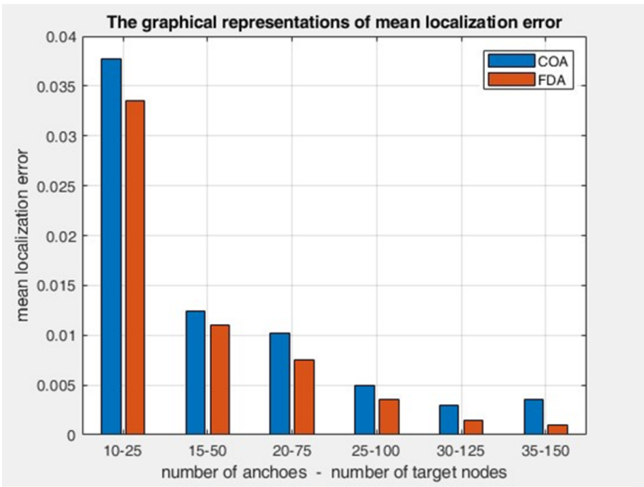

Fig. 7. The mean localization error of the localization algorithms across many deployments of wireless sensor networks

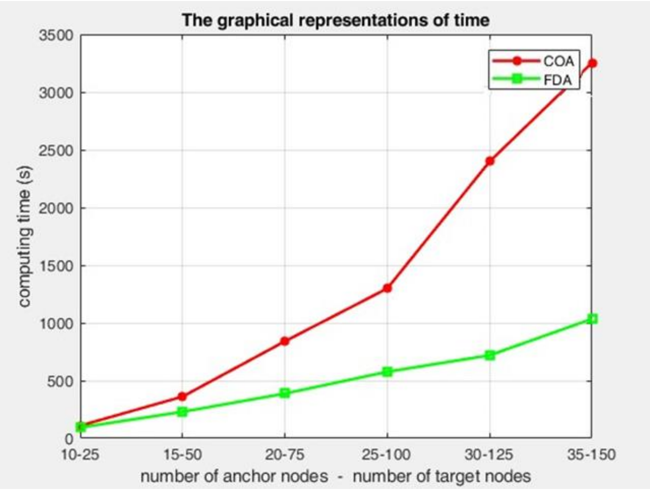

Fig. 8. The computation time of various algorithms in various Wireless Sensor Network configurations

VII. CONCLUSION

Precise node placement is a major challenge for many WSN adoption applications. In this work, the node localization problem was approached as an optimization problem, and a node localization technique was developed using the Flow Direction Algorithm and Crayfish Optimization Algorithm, two unique bioinspired algorithms. The proposed schemes have been executed and checked on many WSN installations with different numbers of unknown and main nodes. Additionally, the suggested schemes have been compared in terms of time of localization and localization error. Experimental findings show that, with respect to the different performance criteria, the Flow Direction Algorithm performs better than the other localization scheme.

REFERENCES

[1] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," Comput. networks, vol. 52, no. 12, pp. 2292–2330, 2008.

[2] M. Khamees, A. Albakry, and K. Shaker, "Multi-objective feature selection: Hybrid of salp swarm and simulated annealing approach," in International conference on new trends in information and communications technology applications, 2018, pp. 129–142.

[3] M. Khamees and A. A.-B. Rashed, "Hybrid SCA-CS optimization algorithm for feature selection in classification problems," in AIP conference proceedings, 2020, vol. 2290, no. 1.

[4] X. Wang, Y. Liu, Z. Yang, K. Lu, and J. Luo, "Robust component-based localizationin sparse networks," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 5, pp. 1317–1327, 2013.

[5] S. Salari, S. Shahbazpanahi, and K. Ozdemir, "Mobility-aided wireless sensor network localization via semidefinite programming," IEEE Trans. Wirel. Commun., vol. 12, no. 12, pp. 5966–5978, 2013.

[6] B.-F. Wu and C.-L. Jen, "Particle-filter-based radio localization for mobile robots in the environments with low-density WLAN APs," IEEE Trans. Ind. Electron., vol. 61, no. 12, pp. 6860–6870, 2014.

[7] C. Wu, Z. Yang, and Y. Liu, "Smartphones based crowdsourcing for indoor localization," IEEE Trans. Mob. Comput., vol. 14, no. 2, pp. 444–457, 2014.

[8] Z. Chaczko and F. Ahmad, "Wireless sensor network based system for fire endangered areas," in Third International Conference on Information Technology and Applications (ICITA'05), 2005, vol. 2, pp. 203–207.

[9] B. Jankuloska, M. Zahariev, A. Mateska, V. Atanasovski, and L. Gavrilovska, "Traffic regulations monitoring using vsns," in 17th Telecommunication forum (TELFOR), 2009, pp. 1474–1477.

[10] N. R. Council, Grand challenges in environmental sciences. National Academies Press, 2001.

[11] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller, "Anchor-free distributed localization in sensor networks," in Proceedings of the 1st international conference on Embedded networked sensor systems, 2003, pp. 340–341.

[12] A. Coluccia and F. Ricciato, "RSS-based localization via Bayesian ranging and iterative least squares positioning," IEEE Commun. Lett., vol. 18, no. 5, pp. 873–876, 2014.

[13] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative localization in wireless networks," Proc. IEEE, vol. 97, no. 2, pp. 427–450, 2009.

[14] P. Oğuz-Ekim, J. P. Gomes, J. Xavier, M. Stošić, and P. Oliveira, "An angular approach for range-based approximate maximum likelihood source localization through convex relaxation," IEEE Trans. Wirel. Commun., vol. 13, no. 7, pp. 3951–3964, 2014.

[15] R. Tan, Y. Li, Y. Shao, and W. Si, "Distance mapping algorithm for sensor node localization in WSNs," Int. J. Wirel. Inf. Networks, vol. 27, pp. 261–270, 2020.

[16] D. Chandirasekaran and S. Sugumaran, "A REAL TIME-BASED OPTIMIZED NODE LOCALIZATION TECHNIQUE FOR WIRELESS SENSOR NETWORKS," J. Electr. Eng. Technol., vol. 11, no. 8, pp. 62–73, 2020.

[17] S. Messous, H. Liouane, O. Cheikhrouhou, and H. Hamam, "Improved recursive DV-hop localization algorithm with RSSI measurement for wireless sensor networks," Sensors, vol. 21, no. 12, p. 4152, 2021.

[18] R. Khanna and A. Kumar, "Artificial intelligence applications for target node positions in wireless sensor networks using single mobile anchor node," Comput. Ind. Eng., vol. 167, p. 107998, 2022.

[19] W. Liu, X. Luo, G. Wei, and H. Liu, "Node localization algorithm for wireless sensor networks based on static anchor node location selection strategy," Comput. Commun., vol. 192, pp. 289–298, 2022.

[20] Y. Cao and J. Xu, "DV-Hop-based localization algorithm using optimum anchor nodes subsets for wireless sensor network," Ad Hoc Networks, vol. 139, p. 103035, 2023.

[21] D. J. Bahadur and L. Lakshmanan, "A novel method for optimizing energy consumption in wireless sensor network using genetic algorithm," Microprocess. Microsyst., vol. 96, p. 104749, 2023.

[22] R. Annapurna and A. C. Sudhir, "Multi-population Firefly Algorithm Based Node Deployment in Underwater Wireless Sensor Networks," Wirel. Pers. Commun., vol. 130, no. 1, pp. 635–649, 2023.

[23] A. Tharwat, E. H. Houssein, M. M. Ahmed, A. E. Hassanien, and T. Gabel, "MOGOA algorithm for constrained and unconstrained multi-objective optimization problems," Appl. Intell., vol. 48, pp. 2268–2283, 2018.

[24] A. G. Hussien, E. H. Houssein, and A. E. Hassanien, "A binary whale optimization algorithm with hyperbolic tangent fitness function for feature selection," in 2017 Eighth international conference on intelligent computing and information systems (ICICIS), 2017, pp. 166–172.

[25] A. Kouba, A. Petrusek, and P. Kozák, "Continental-wide distribution of crayfish species in Europe: update and maps," Knowl. Manag. Aquat. Ecosyst., no. 413, p. 5, 2014.

[26] H. Jia, H. Rao, C. Wen, and S. Mirjalili, "Crayfish optimization algorithm," Artif. Intell. Rev., vol. 56, no. Suppl 2, pp. 1919–1979, 2023.

[27] Z. Liao, Z. Chen, and S. Li, "Parameters extraction of photovoltaic models using triple-phase teaching-learning-based optimization," IEEE Access, vol. 8, pp. 69937–69952, 2020.

[28] S. Carbas, A. Toktas, and D. Ustun, Nature-inspired metaheuristic algorithms for engineering optimization applications. Springer, 2021.

[29] H. Jia, X. Zhou, J. Zhang, L. Abualigah, A. R. Yildiz, and A. G. Hussien, "Modified crayfish optimization algorithm for solving multiple engineering application problems," Artif. Intell. Rev., vol. 57, no. 5, p. 127, 2024.

[30] N. Covic and B. Lacevic, "Wingsuit flying search—A novel global optimization algorithm," IEEE Access, vol. 8, pp. 53883–53900, 2020.

[31] L. Wang and L. Li, "An effective differential evolution with level comparison for constrained engineering design," Struct. Multidiscip. Optim., vol. 41, pp. 947–963, 2010.