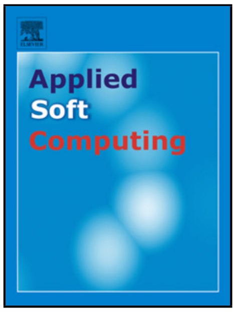


Applied
Soft
Computing

Title: A Hybrid Model using Fuzzy Logic and an Extreme Learning Machine with Vector Particle Swarm Optimization for Wireless Sensor Network Localization

Authors: Songyut Phoemphon, Chakchai So-In, Dusit (Tao) Niyato

| | |
|---|---|
| PII: | S1568-4946(18)30010-3 |
| DOI: | https://doi.org/10.1016/j.asoc.2018.01.004 |
| Reference: | ASOC 4651 |
| To appear in: | *Applied Soft Computing* |
| Received date: | 11-1-2015 |
| Revised date: | 11-10-2017 |
| Accepted date: | 9-1-2018 |

# A Hybrid Model using Fuzzy Logic and an Extreme Learning Machine with Vector Particle Swarm Optimization for Wireless Sensor Network Localization

Songyut Phoemphon[1], Chakchai So-In[1,*] and Dusit (Tao) Niyato[2]

Applied Network Technology (ANT) Laboratory

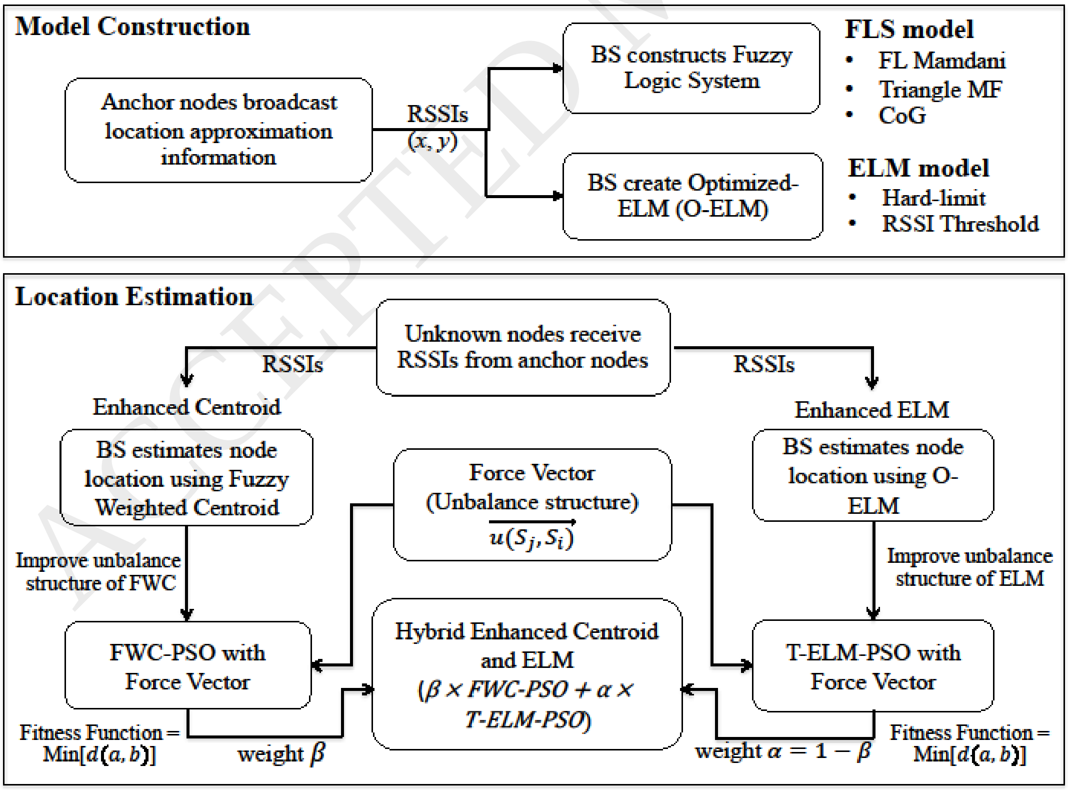
[1]Department of Computer Science, Faculty of Science, Khon Kaen University, Khon Kaen, Thailand

[2]School of Computer Engineering, Nanyang Technological University, Singapore

*songyut_p@kkumail.com*, *chakso@kku.ac.th* and *dniyato@ntu.edu.sg*

**Corresponding Author*

*Graphical abstract*


Model Construction
Anchor nodes broadcast location approximation information
RSSIs
$(x, y)$
BS constructs Fuzzy Logic System
BS create Optimized-ELM (O-ELM)
FLS model
FL Mamdani
Triangle MF
CoG
ELM model
Hard-limit
RSSI Threshold
Location Estimation
Unknown nodes receive RSSIs from anchor nodes
RSSIs
RSSIs
Enhanced Centroid
Enhanced ELM
BS estimates node location using Fuzzy Weighted Centroid
Force Vector (Unbalance structure) $\overrightarrow{u(S_j, S_i)}$
BS estimates node location using O-ELM
Improve unbalance structure of FWC
Improve unbalance structure of ELM
FWC-PSO with Force Vector
Hybrid Enhanced Centroid and ELM $(\beta \times FWC\text{-}PSO + \alpha \times T\text{-}ELM\text{-}PSO)$
T-ELM-PSO with Force Vector
Fitness Function = Min$[d(a,b)]$
weight $\beta$
weight $\alpha = 1 - \beta$
Fitness Function = Min$[d(a,b)]$

Highlights

- We investigate a practical integration of soft-computing for WSN localizations.
- We propose a hybrid model applying Centroid and ELM optimizations.
- We optimize the hybrid model using resultant vector based on PSO.
- Our proposals consider several key factors effecting the location estimation precision, i.e., node density, sensing coverage, and heterogeneous topology.

**Abstract**: *Localization is one of the challenges in wireless sensor networks, especially those without the aid of a global positioning system. Use of a dedicated positioning device incurs additional cost and reduces battery life; therefore, a range-free localization scheme is promising as a cost-effective approach. However, the main limitation of this approach is that the estimation precision can be affected by factors such as node density, sensing coverage, and topology diversity. Thus, this study investigates and proposes a method for improving a traditional range-free-based localization method (centroid) that uses soft computing approaches in a hybrid model. This model integrates a fuzzy logic system into centroid and uses an extreme learning machine (ELM) optimization technique to capitalize on the strengths of both approaches: the former is properly used with low node density and short coverage, while the latter is used for the opposite—to achieve a robust location estimation scheme. The ratios of known nodes within the sensing coverage range to the total known nodes and of the sensing coverage range to the maximum coverage range are used as adaptive weights for this hybrid model. To further improve the efficiency, especially in heterogeneous topologies, the concept of resultant force vectors is applied to this hybrid model over particle swarm optimization to mitigate the effects of irregular deployments. The performance of the proposed method is extensively evaluated via simulations that demonstrate its effectiveness compared to other state-of-the-art soft-computing-based range-free localization schemes (i.e., centroid, a fuzzy logic system, and a support vector machine with a traditional ELM).*

**Keywords:** *Extreme Learning Machine; Fuzzy Logic; Localization; Particle Swarm Optimization; Wireless Sensor Network*

## 1 Introduction

The advancement of microelectromechanical systems (MEMSs) [1-3] has led to various practical applications that have low power consumption and high computing performance on small platforms, sensor nodes, or sensors (motes) that include embedded dedicated storage and transmission logic for multifunctional uses and capabilities. This integration has introduced the concept of self-organizing sensors [3]. In general, multipurpose sensors are used in diverse applications both individually and collectively, as sensor farms. A sensor farm is normally connected through wireless links with embedded transmission logic in an ad hoc manner for ubiquitous access, thus alleviating the deployment and placement issues that must be considered during the design of wireless sensor networks (WSNs) [4,5].

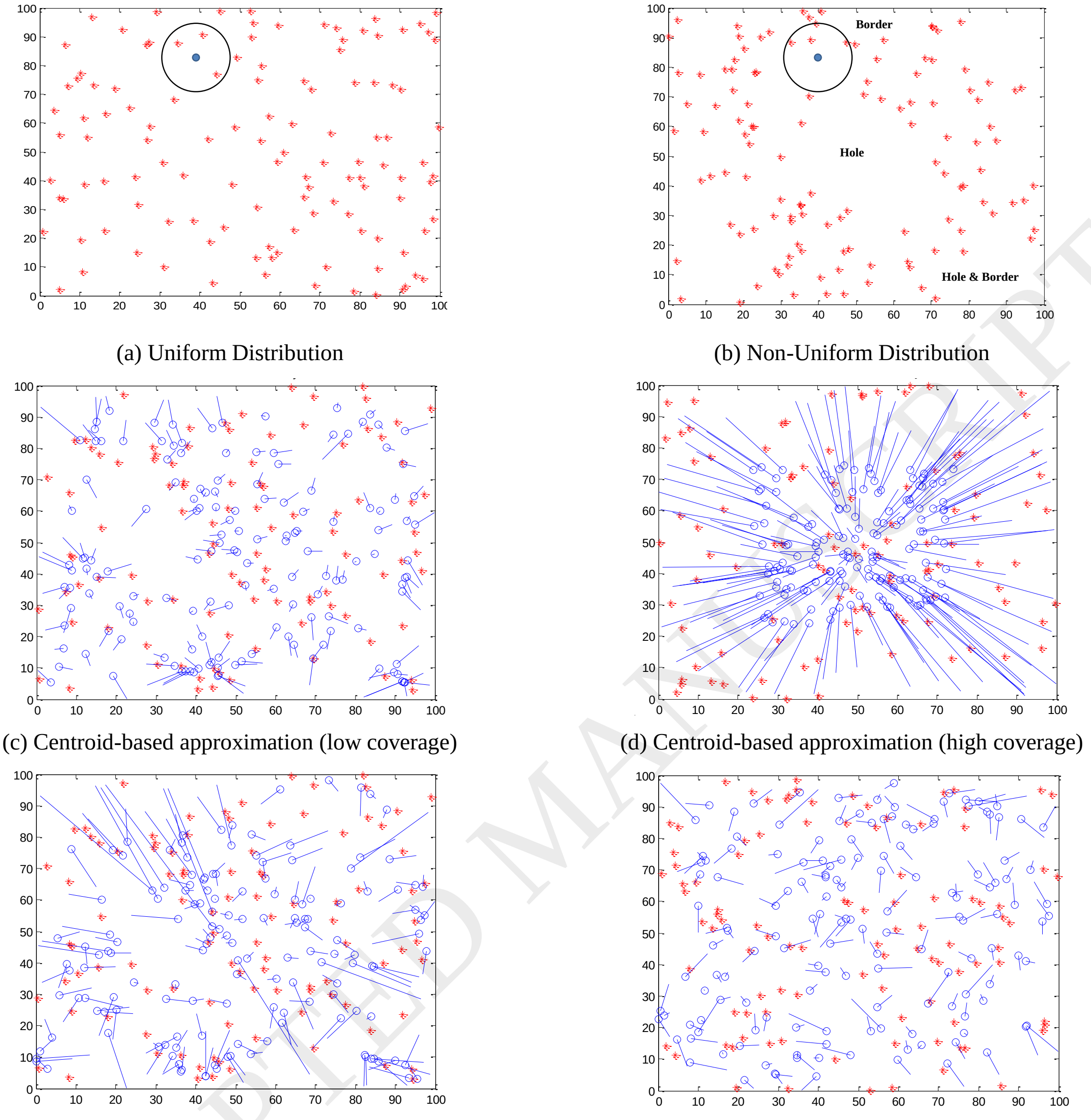
Many applications have recently adopted the concept of WSNs, including security surveillance, tracking and traceability, and civil monitoring [6,7]. This widespread interest is due to several advantages and outstanding features of such networks including self-organization, self-computing, scalability, and fault-tolerance. More specifically, WSNs are suitable for cases in which the sensors are deployed in various environments, including those with harsh conditions or human-inaccessible terrain, which leads to the concept of ubiquitous and pervasive computing. However, along with these features, several aspects require investigation when WSNs are deployed in real-world scenarios. These aspects include quality of service, scalability, data aggregation, time synchronization, energy-aware computation, and real-time communication [8].

One crucial challenge with WSNs is location discovery, especially when hardware that can provide knowledge of a specific location, such as via a global positioning system (GPS), is absent. GPS signals tend to be efficient only when used in open areas with access to GPS satellite signals. In other words, GPS signals may be unobtainable in some scenarios, especially in indoor locations [9]. However, GPS-equipped wireless sensors increase the hardware costs for sensor nodes and reduce battery lifetimes, which reduces sensor lifetimes [9,10]. Thus, a received signal strength indicator (RSSI) is often used as an input to methods that approximate unknown node locations. Such methods have a key advantage: they use only connectivity or proximity information (e.g., the number of hops between sensor nodes and coordinate) and require no additional hardware. One promising approach for location discovery in WSNs is centroid [11].

A centroid-based technique is generally employed to estimate the location of a node based on nearby sensors within a predefined radio range used to calculate the mean distance. Centroid's main advantage is its low computational complexity; however, it has a high location estimation error that is dependent on the node density [11]. Recently, several proposals have been designed to estimate the proper weight used to reduce such proximity errors. Some of these promising schemes are based on soft computing methods [12,13] because they solve optimization problems and are appropriate for use in uncertain scenarios and those that require approximation, and they can achieve practicability and robustness with a low-cost solution. Examples are the neural network (NN), evolutionary computation (EC) (e.g., the genetic algorithm (GA)), support vector machine (SVM), and swarm intelligence [14-17]. In addition to NNs and GAs [18], fuzzy logic systems (FLSs) and metaheuristic-based approaches are commonly explored due to their simplicity and low computational costs [19-23].

Recently, these soft-computing-based techniques have generally been combined with a baseline range-free localization method—especially centroid [15,18,24-26]—which provides some beneficial gains with some trade-offs. However, their key limitations are still low accuracy with regard to various parameters and factors that impact the location estimation performance, such as the node density, ratio of known to unknown nodes, network dimension, and sensing coverage [9,27,28]. In addition, one key factor that can degrade the estimation precision is irregular topology (as compared to a uniform sensor distribution scenario—see Figure 1) during deployment, causing a reduction in practicability.

For example, Figure 1(a) shows a uniform node deployment given anchor nodes and unknown nodes. Here, with sufficient anchor nodes (i.e., 3) the approximation error with weighted centroid tends to be low; however, in Figure 1(b), with irregular topology (such as hole and border near the interest area), it is higher. In addition, the examples in Figures 1(c) and 1(d) show that the centroid-based approach tends to result in higher errors based on factors such as the number of anchor nodes and the coverage [11]. Figures 1(e) and 1(f) are similar but involve an extreme learning machine (ELM).



(a) Uniform Distribution

(b) Non-Uniform Distribution

(c) Centroid-based approximation (low coverage)

(d) Centroid-based approximation (high coverage)

(e) ELM-based approximation (low coverage)

(f) ELM-based approximation (high coverage)

Figure 1. Node Deployment Examples: cross (anchor); blue circle (unknown); line from circle (error from the actual position)

Thus, our contributions center on investigating non-uniform deployments in WSNs to improve the location estimation precision. More specifically, we propose a hybrid optimization model consisting of an FLS applied over centroid and an ELM. The former is properly used in low-node-density scenarios, while the latter is used under the opposite conditions. These two techniques are combined and applied using a particle swarm optimization (PSO) technique with resultant force vectors, which adjusts the estimation accuracy to be closer to the actual node locations with regard to the diversity of signal strengths. Then, an additional weight derived from the relationship of known nodes within the sensing coverage area is used to adjust these two hybridized techniques. Hence, we call the proposed scheme a hybrid model using fuzzy logic and extreme learning machine with vector particle swarm optimization for wireless sensor network localization (HVP-FELM).

The remainder of this article is organized as follows. Section 2 briefly surveys recent related studies, especially those that apply soft computing techniques to WSN localizations. Section 3 provides a detailed description of our hybrid technique (HVP-FELM), which uses fuzzy centroid with an optimized ELM as well as vector PSO. The performance of our proposal compared to that of other state-of-the-art soft computing applications—including the traditional centroid-based scheme—is presented and discussed in Section 4. Finally, Section 5 contains conclusions and possible future work.

## 2 Related Work

Localization is a challenge encountered by WSNs in numerous applications. Many techniques have focused on localization optimizations [27,28]; however, this section concentrates on studies that have integrated soft computing techniques to enhance the estimation precision [18,29-35]. These integrated techniques include FL, NNs, ECs (such as GAs), metaheuristics and swarm intelligence (such as PSO), and SVMs, including ELMs [36].

To enhance the precision of the traditional centroid method proposed by Bulusu et al. [11], FL has traditionally been applied because it has the key advantage of low computational complexity for range-free-based location approximation in WSNs. Many FL integrations have been investigated; for example, Yun et al. [26] used the Takagi-Sugeno-Kang method to integrate fuzzy systems into centroid. This fuzzy centroid method used additional weights derived from the fuzzy triangular function while applying a weight average to determine the output with five predefined rules (very low, low, medium, high, very high) to adjust the centroid estimation precision; however, this improvement is more suitable for uniform node distribution.

Instead of Sugeno, Larios et al. [25] proposed the use of FL-Mamdani to derive the two inputs. The first input consists of RSSIs from three nearby sensors with three functions (low, medium, and high). The second input consists of RSSIs from other sensors (second hop) with nine functions (the former includes an interaction between the two levels). Eleven rules were determined to maintain those relationships but at the cost of extra communications in addition to the key limitation of improper weight derivation to account for imbalanced node deployment structures.

It is worth noting that the fuzzy weighted centroid (FWC) methods discussed above considered only grid deployments; consequently, the number of nodes was fixed. These studies achieved performance improvements compared to centroid, but irregular topologies such as imbalances in the anchor nodes may affect the precision of the results. In addition, FWC methods tend to provide good approximation performance when the number of anchors is small or the communication range is short, but their precision tends to be reduced in under the opposite conditions. However, these approaches can be used as baselines for other FL derivatives.

Given their key advantages for high location estimation accuracy, different approaches have applied NNs. For example, Shareef et al. [31] investigated three types of location approximation that involve using NNs: 1) a multilayer perceptron (MLP) consisting of two layers—nine nodes in the first layer with a hyperbolic tangent sigmoid activation function and two nodes in the second (output) layer with a linear activation function (to limit the time complexity, only 200 rounds of training were conducted, and 0.001 was used as an error threshold); 2) recurrent neural networks (RNNs), using a setup similar to that of the first model but with feedback from the output layer to the input layer; and 3) the radial basis function (RBF), with a constant number of nodes in the first layer and a specific standard deviation as the target. An error threshold was determined to ensure robust training. Here, NN with RBF achieved the highest approximation precision but also exhibited high computational complexity and consumed more memory resources.

Rahman et al. [15] proposed a two-phase approximation model that again used RSSIs as inputs. The first part consists of a generalized regression neural network (GRNN). In the second layer, a special linear layer (instead of RBF) was adopted as the activation function to derive four reference nodes. The goal of this approach was to approximate a location using weighted centroid in the second part. Two GRNN models were used for each *x* and *y* coordinate. In general, the location estimation precision of NNs is likely to be higher than that of FL, but only at the cost of comparatively high computational complexity, particularly during the training and location estimation processes, which makes this approach impractical for WSNs.

Thus, N. Li et al. [37] proposed the use of PSO to seek a set of proper parameters for ANN with BP in addition to carrying out preprocessing with principal component analysis to reduce the computational dimension. Although there is a precision gain with computational reduction, compared to a specific type of NN, i.e., a single hidden layer feed-forward network, the error tends to be high in the event of dense deployment and high coverage [30]. In addition, these studies do not evaluate various scenarios that directly impact the location precision (i.e., node density and coverage effects), particularly those with irregular topology (imbalanced topology or non-uniform node distribution). Again, considering these factors will alter the performance results.

Instead of applying FL to derive the weight directly, Huanxiang et al. [19] integrated GAs to adjust the weighted centroid approximation to achieve higher accuracy based on signal strength inputs and localization reference information. Similarly, Yang et al. [20] applied GAs using a filter replenishment strategy to produce better convergence—a technique that can be used with large-scale WSNs. Here, the fitness function was derived from the actual location of the estimated position; however, that information is not available in practice.

The proposals discussed above focus on using only a single technique to improve the precision; however, some studies have also investigated hybridized models (i.e., using two soft computing approaches—in particular, those integrated with centroid). For example, a hybrid approach involving FL and GAs was evaluated and discussed by Yun et al. [18], who enhanced his previous work in [26], which used only an FL approach, by combining those two methods to derive the proper weights. Initially, GA was used to adjust the membership function of FL (using an iterative process) for edge weights based on RSSI information. However, this method has limitations similar to the GA-based localization method: it assumes that the actual position is known when calculating the mean square error. Similarly, a hybrid approach involving NNs and GAs was evaluated and discussed by Chagas et al. [32], in which the GA was used to derive optimal parameters prior to input into the NN, but again, these methods involve a trade-off between accuracy and computational complexity.

Similar to GAs, some studies have utilized PSOs to reduce location errors. For example, Low et al. [22] applied the PSO algorithm to improve the estimation precision and avoid local minima under centroid integration. In that study, a fitness function was determined by translating the RSSIs and distance from each particle (encoded *x* and *y* coordinates) to the anchor nodes. A constant iteration (i.e., 100) was used as a termination criterion to limit the computational complexity. Similarly, adopting the PSO concept, Gopakumar and Jacob [23] proposed a two-step location approximation (weighted centroid was first adopted, and then PSO was applied as a stochastic global optimization) to minimize the location errors of the weighted centroid method based on a fitness function similar to that in [22] for WSNs. A. Kumar et al. [38] similarly enhanced the localization approximation, particularly membership adjustment, via a two-step approximation, using a hybrid-particle swarm optimization (HPSO) and biogeography-based optimization (BBO) to adjust the membership function to achieve a better weight. The use of RSSIs as inputs resulted in an output weight for weighted centroid localization. However, these approaches tend to have low accuracy when there is an imbalance among the known nodes.

SVM-based integration approaches have also been investigated to solve the WSN localization problem, Tran and Nguyen [29] proposed an SVM to determine the relationships between anchor nodes and their locations using RSSIs as inputs and an RBF kernel to map the inputs into a high-dimensional feature space. Then, the location approximation was also enhanced over SVM using a force movement of anchor nodes' locations within a communication range that consists of modified mass-spring optimizations (MSOs). A key advantage of this model is that it can be used with training sets containing outliers.

Samadian and Noorhosseini [33] also applied an attractive/repulsive potential field (ARPoFil) localization technique instead of MSO. Their approach included all the known nodes in the computation to determine the resulting force—not just those within the unknown node coverage area. which achieved a higher location approximation rate. This scheme adopted a probabilistic SVM (PSVM) in which the output was fed to a sigmoid function to normalize it into a 0 to 1 range. Then, the entire area was divided into individual small interest sectors while applying PSVM to classify node locations from nearby nodes before using ARPoFil. Although such SVM optimizations can mitigate the effects of unbalanced structures resulting from node deployments, their estimation precision is typically limited by low node densities [34]. Note that, comparatively, although these optimizations (MSO or ARPoFil or their derivatives) can improve the location approximation precision, NN-based schemes can provide higher accuracy (but with a complexity trade-off).

The ELM, introduced by Huang et al. [36], was recently proposed as a candidate technique that may be superior to SVMs and other methods because of two key advantages: a very fast training phase and increased accuracy. A few studies have focused on integrating ELMs for WSN localization. For example, Yang et al. [35] applied an ELM with sequential online constraints to approximate indoor locations with improved precision using a combination of fingerprint techniques for training that involve several different time-varying received signal strengths. However, this technique has a limitation because the fingerprint information is required. Recently, So-In et al. [30] proposed a model to determine the relationships of RSSIs between anchor nodes and their locations to integrate an ELM with a hard limit activation function for WSN location approximation to reduce the training stage complexity. To enhance the precision of this technique, they also integrated a greedy approach that combines optimized MSO. The performance evaluation found the ELM-based technique to be superior to other techniques (i.e., FL, GAs, NNs, and SVMs). However, this study concentrated solely on uniform and small hole deployments. In addition, it lacked a discussion of the relationship between signal coverage and node density.

## 3 A Hybrid Model using Fuzzy Logic and an Extreme Learning Machine with Vector Particle Swarm Optimization for Wireless Sensor Network Localization

Figure 2 shows the overall scheme of our hybrid approach. In general, it includes four main components:

- Force Vector (see Section 3.1): This stage is used to derive the proper direction with regard to a known node using the maximum received signal strengths together with the concept of resultant force (force vector) on the known nodes within the sensing coverage of the estimated location.
- Enhanced centroid (see Section 3.2): Based on the traditional centroid technique, this stage includes an additional weight derived from a fuzzy logic system to adjust the centroid location approximation precision (i.e., an FWC technique). Then, PSO is applied to mitigate the effect of unbalanced node deployments (FWC-PSO).
- Enhanced ELM (see Section 3.3): After applying ELM to WSN localization (O-ELM), this stage applies a threshold-based scheme to remove unnecessary nodes (those with low received signal strengths) from the approximation computation (T-ELM) before applying PSO (T-ELM-PSO).
- Hybrid enhanced centroid and ELM (see Section 3.4): After obtaining a previously estimated location, this stage applies an additional weight (weighted mean) derived from known nodes within the sensing coverage area to the total known nodes and the sensing coverage to the maximum coverage ($\beta$) to adjust the impact of the centroid and ELM location estimations.

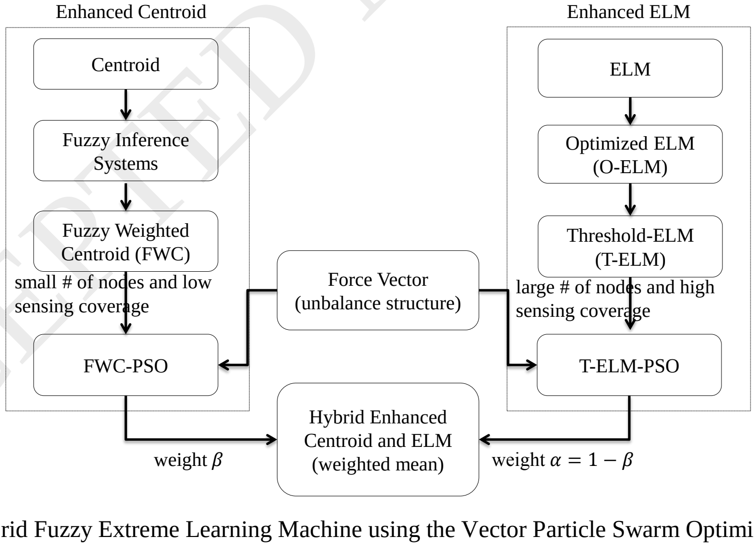

Enhanced Centroid
Centroid
Fuzzy Inference Systems
Fuzzy Weighted Centroid (FWC)
small # of nodes and low sensing coverage
FWC-PSO
Enhanced ELM
ELM
Optimized ELM (O-ELM)
Threshold-ELM (T-ELM)
large # of nodes and high sensing coverage
T-ELM-PSO
Force Vector (unbalance structure)
Hybrid Enhanced Centroid and ELM (weighted mean)
weight $\beta$
weight $\alpha = 1 - \beta$

Figure 2. Hybrid Fuzzy Extreme Learning Machine using the Vector Particle Swarm Optimization Process

### 3.1 Force Vector PSO

Our proposal employs the unit vector concept, which is the vector over its amplitude, denoted by $\vec{u}/|u|$ in $(x, y)$ coordinates ($S$ anchor nodes) in the two-dimensional area of the WSNs. Then, the force vector is defined as the summation of the unit vectors in terms of the resultant force [29] as follows.

$$\overrightarrow{u(S_j,S_i)} = \frac{x_j - x_i}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}}, \frac{y_j - y_i}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}}, \tag{1}$$

$$\overrightarrow{F(S_j)} = \textstyle\sum_{i=1}^{N} \overrightarrow{u(S_j,S_i)}, \tag{2}$$

where the unit vector *u(S<sub></sub>* $u(S_j, S_i)$ is derived from the coordinate between nodes $S_i$ and $S_j$ such that $S_i$ is the anchor node and $S_j$ denotes the anchor node with the maximum RSSI within the sensing radius of the unknown nodes. A force vector $F(S_j)$ is the integration of the unit vectors of the neighboring anchor nodes, *N*. Figure 3 shows an example of this concept. There are 5 anchor nodes ($S_1$ to $S_5$) corresponding to their coordinates ($x_1$, $y_1$) to ($x_5$, $y_5$), respectively. Here, ($x_j$, $y_j$) is the anchor node with the maximum RSSI. The unit vectors are $u_1$ to $u_5$, each of which can be computed from the equations above and then used to calculate the direction of the resultant forces.

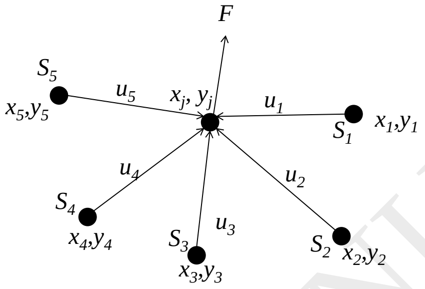


Figure 3. Force Vector Examples

This force vector is used to determine the direction of the moving location approximation so that it becomes close to the actual location, which is used during the PSO iterations. PSO is a heuristic method [39] used to determine the best solution under given a set of constraints. As stated in Algorithm 1, after the initial population is set up, a particle in the swarm is given a random position (*S*) and velocity (*v*) (lines 2–5). Then, each particle is evaluated by the fitness function, and the best personal fitness value (*pb*) is updated (lines 7–11). The fitness function is used for population selection. It determines the shortest distance between two particles in *n* dimensions, as stated in Equation (3) below, using Euclidean distance (ED).

$$d(a,b) = \sqrt{\textstyle\sum_{i=1}^{n}(a_i - b_i)^2}, \tag{3}$$

where *a* and *b* are the positions of the two particles.

In addition, the best overall fitness value will be selected as the global best fitness value (*gb*) (lines 12–14). The velocity and position will then be updated (lines 15–18). Lines 16 and 17 state how to derive and update the particle velocity and position as shown in the equations below. Note that this process continues until a certain number of iterations are completed (line 19):

$$v_i(t+1) = iw_i \times v_i(t) + c_1 \times rand_1\big(pb_i - S_i(t)\big) + c_2 \times rand_2(gb_i - S_i(t)) \tag{4}$$

$$S_i(t+1) = S_i(t) + v_i(t+1) \tag{5}$$

$$iw = iw_{max} - \frac{iw_{max} - iw_{min}}{t_{max}} \times t, \tag{6}$$

where the velocity of particle *i* in the *t*th iteration is both the personal best solution (*pb*) and global best solution (*gb*). Here, $c_1$ and $c_2$ denote acceleration factors, and $rand_1$ and $rand_2$ are random effects that are normally in the range between 0 and 1 [39]. Additionally, to maintain a balance between global and local search, an additional weight, the inertial weight (*iw*), is generally applied as a multiplier factor of $v_i(t)$ [39]. The weight $iw_{max}$ is the initial inertia weight, $iw_{min}$ denotes the termination inertia weight, and $t_{max}$ is the maximum number of iterations.

Algorithm 1: Particle Swarm Optimization

```
Select Initial Population = {S}
for (i = 1 to size(S))
    Randomly set the velocity of particle $S_i$
    Randomly set the position of particle $S_i$
end for
do
    for (i = 1 to size(S))
        if Fitness($S_i$) < Fitness($pb_i$) then
            $pb_i = S_i$
        end if
    end for
    if $pb_i < gb_i$ then
        $gb_i = pb_i$
    end if
    for (i = 1 to size(S))
        Calculate the velocity of particle $S_i$
        Update the position of particle $S_i$
    end for
while ($t$++ < $t_{max}$)
```

### 3.2 Enhanced Centroid

The main advantage of the traditional centroid localization scheme is its simplicity and, thus, its low computational complexity; however, its estimation precision is rather low [14]. Consequently, many proposals have applied this concept but optimized the traditional centroid-based approach. Most of these proposals focused on the introduction of weights [40,41]. In this study, an FL system is selected to derive the additional weights. The concept of a resultant force over the unit vectors is also used to adjust the location of the fuzzy process.

#### A. Centroid Localizations

In general, the centroid-based scheme is used as a baseline for the other integrated techniques. In the WSN evaluation process, there is an assumption that the positions of the known nodes (anchor nodes) and unknown nodes are mixed throughout the network and that all the nodes are randomly deployed. The location approximation is generally calculated based on the RSSI and the coordinates of the anchor nodes. The transmission protocol and other overhead costs are left as further implementation aspects. Similarly, the radio propagation is theoretically isotopic (i.e., the radio transmission ranges are all identical) [7,42]. Then, the main concept of the centroid in the WSN localization is to compute the approximate locations of nodes based on informational (beacon) announcements from the known nodes that are within the RSSI threshold as stated in the equation below.

$$(x_{est}, y_{est}) = \left(\frac{\sum_{i=1}^{n} x_i}{n}, \frac{\sum_{i=1}^{n} y_i}{n}\right), \tag{7}$$

where $x_{est}$ and $y_{est}$ form the estimated position in *x, y* coordinates, and $x_i$ and $y_i$ are the positions of the anchor nodes (*1* to *n*) that are within the acceptable RSSI threshold.

### B. Fuzzy Inference Systems

The FLS is used to manage the logic, which involves an approximation of location. In general, the true value will be within the range of 0 to 1 [24-26]. Figure 4 shows an overview of the four main processes of an FLS given the input RSSIs to derive a proper output, which is then used to determine the adjustable weights for weighted centroid location approximation schemes.

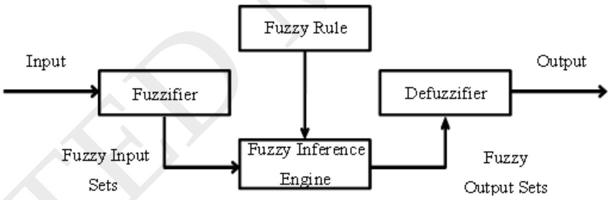


Figure 4. Fuzzy Logic System [18]

- Fuzzifier: This process is used to transform the input data (here, the RSSIs of known nodes within an unknown node’s coverage range) by mapping them to the defined membership functions to determine the crossing point of each function. In this research, after intensively evaluating the performance of different membership functions (MF) [30], we found that the triangular function was superior and used it in our subsequent research. We used five criteria for the inputs and weights: very low (VL), low (L), medium (M), high (H), and very high (VH) [42].

  For example, Figure 5 shows the relationship between the input range given maximum and minimum RSSIs based on a recommendation provided by [43]. Here, the RSSIs are between -150 and 0; the output is between 0 and 1. The degree of membership corresponds to the triangular function. For example, the five criteria for the inputs and outputs are as follows: VL (-150.00, -150.00, -112.50), L (-150.00, -112.50, -75.00), M (-112.50, -75.00, -37.50), H (-75.00, -37.50, 0.00), VH (-37.50, -4.441e-16, 0.00); and VL (0.00, 0.00, 0.25), L (0.00, 0.25, 0.50), M (0.25, 0.50, 0.75), H (0.50, 0.75, 1.00), VH (0.75, 1.00, 1.00). Similarly, the generated weight range will be from 0 to 1 (see Figure 6).

- Fuzzy Rule: Similar to [18], this process is used to create a mapping between the fuzzy input (RSSI) from the anchor node (not a distance measurement) and the fuzzy output (generated weight) given its membership function. The fuzzy model has the following form:

$$R^l\text{: IF } x \text{ is } A^l \text{ THEN } y \text{ is } B^l \tag{8}$$

Here, $R^l$ is the $l^{\text{th}}$ rule; $x$ is the RSS from the anchor nodes given a value in the interval [$RSSI_{min}$, 0], where $RSSI_{min}$ is the minimum RSSI value (= -150); $y$ is the $l^{\text{th}}$ output variable; $A^l$ is the fuzzy membership function for inputs; and $B^l$ is the fuzzy membership function for outputs in the case of Mamdani-type systems. An input variable RSS is mapped onto five membership functions (using the triangular function), as shown in Figures 5 and 6. Note that similar to the examples shown in Table 1 (Table 2 also presents the fuzzy associative matrix), we followed the recommendations for rule construction presented in Gu et al. [41] by dividing the rules into five different inference rules; however, the technique is not limited to this approach.

- Fuzzy Inference Engine: This process is used to derive the fuzzy implication (the output weight ranging between 0 and 1) for each fuzzy rule based on the fuzzy input and fuzzy output. Then, each will be combined with aggregation methods such as maximization operations, as illustrated in Figure 7. For example, given the input RSSI = -70, the crossing point traverses the 5 fuzzy rules (from the five fuzzy inputs to derive five fuzzy outputs or weights), each of which is applied to the implication method to generate five output weights.
- Defuzzifier: After the FL inference, the CoG will be calculated to determine the actual output over the derived output weights (e.g., 0.67 in this example). As recommended by Kumar et al. [24], we evaluated different fuzzy inference systems (i.e., Sugeno and Mamdani) in our research and the results showed that FL-Mamdani (using CoG instead of the average) is promising based on a simulation result from grid deployments.

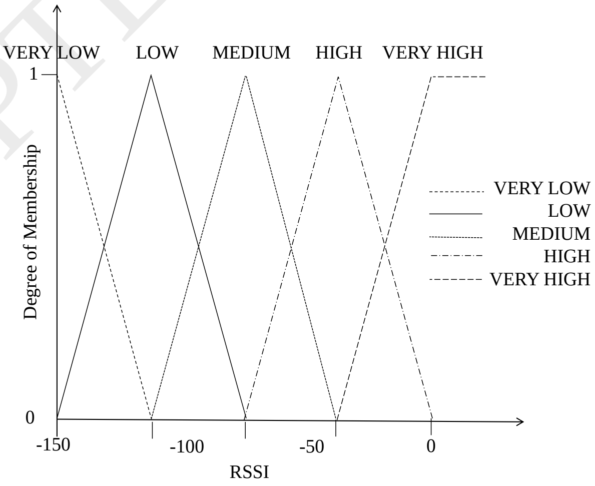


Figure 5. Fuzzy Logic: Triangular Function of the Input (RSSI)

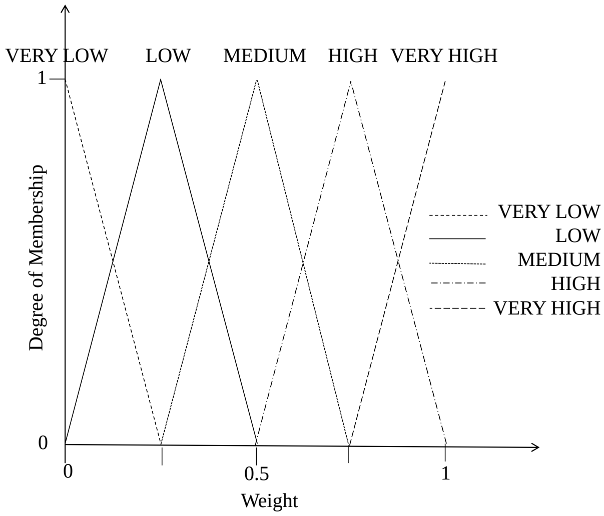


Figure 6. Fuzzy Logic: Triangular Function of the Output (weight)

Table 1. Fuzzy Rule: Example [41]

| **Rule** | **IF: State of the input (RSSI) is** | **THEN: State of the weight is** |
|---|---|---|
| **1** | Very low | Very low |
| **2** | Low | Low |
| **3** | Medium | Medium |
| **4** | High | High |
| **5** | Very high | Very high |

Table 2. Fuzzy Rule: Matrix

| **RSSI**<br>**Weight** | **Very Low** | **Low** | **Medium** | **High** | **Very High** |
|---|---|---|---|---|---|
| **Very Low** | High | Medium | Low | Low | Low |
| **Low** | Medium | High | Medium | Low | Low |
| **Medium** | Low | Medium | High | Medium | Low |
| **High** | Low | Low | Medium | High | Medium |
| **Very High** | Low | Low | Low | Medium | High |

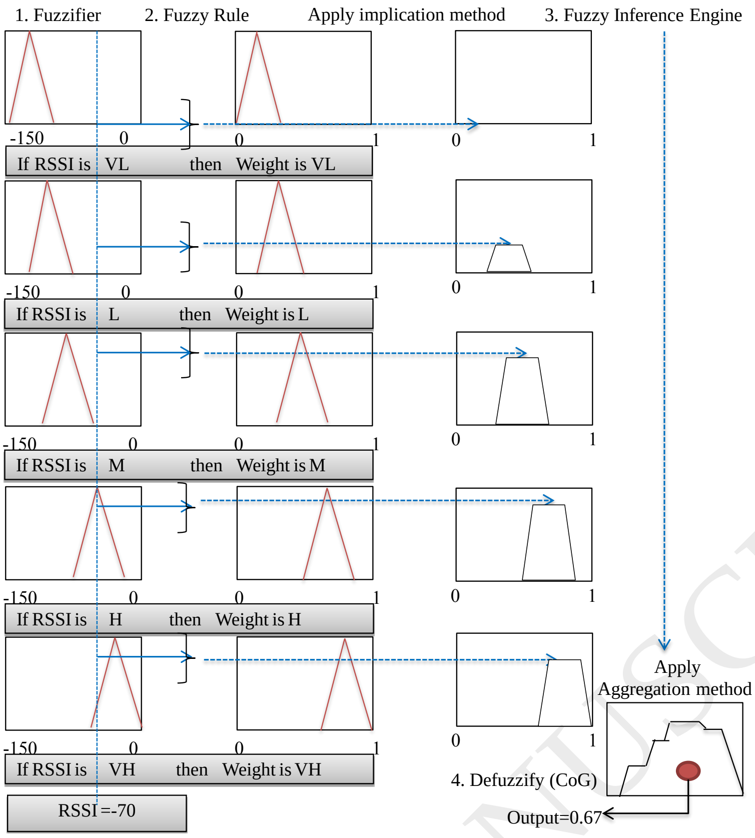

1. Fuzzifier
2. Fuzzy Rule
Apply implication method
3. Fuzzy Inference Engine
-150
0
0
1
0
1
If RSSI is VL then Weight is VL
If RSSI is L then Weight is L
If RSSI is M then Weight is M
If RSSI is H then Weight is H
If RSSI is VH then Weight is VH
RSSI =-70
Apply
Aggregation method
4. Defuzzify (CoG)
Output=0.67

Figure 7. Fuzzy Logic Systems: Fuzzy Inference Engine and Defuzzifier

### C. Fuzzy Weighted Centroid (FWC)

After deriving the output from the fuzzy inference system, the positions of the unknown nodes can be estimated using the output as the weight, as stated in Equation (9). In contrast, a traditional centroid is only the average of the two-dimensional positions of the reference nodes:

$$(x_{est}, y_{est}) = \left(\frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}, \frac{\sum_{i=1}^{n} w_i y_i}{\sum_{i=1}^{n} w_i}\right), \qquad (9)$$

where $w_i$ is an additional weight that is normally used to further improve the location error estimation.

Algorithm 2 shows an overview of the fuzzy centroid optimization to approximate the node location in the WSN such that the output weight will be derived to adjust the weight as follows. First, a set of node perimeters are considered given a threshold (lines 1–6). Note that line 4 is also used to compute an output weight (*w_out*) for each anchor node within a communication range of unknown nodes (*RSS*). Finally, these weights will be used to adjust the centroid-based estimation using Equation (9) (line 7).

Algorithm 2: Fuzzy Weighted Centroid for WSN Localizations (FWC)

```
Input: $RSSI[N], RSSI_{Threshold}$
Output: $x_{est}, y_{est}$
for $i$ = 1:$N$
    if $RSSI_i \leq RSSI\_Threshold$ then
        $RSS = RSS \cup RSSI_i$
        Calculate the output weight ($w\_out[i]$) from the Fuzzy Inference Systems in Section 3.2B
    end if
end for
Calculate the location estimation ($x_{est}, y_{est}$) according to Equation (9), where $w$ = $w\_out$
```

### D. Fuzzy Weighted Centroid-Particle Swarm Optimization (FWC-PSO)

Although FWC can improve the precision of the location estimation, the accuracy of FWC depends largely on the adjustable weights for each anchor node. More specifically, the estimation errors are higher for sensor nodes close to the border because there is an imbalance in the sensed information. Therefore, in this research, we also apply the resultant force to determine the proper direction, and then, we employ PSO to move the approximate node location close to the actual location (i.e., moving closer to the first anchor node, which has the maximum RSSI ($x_f$, $y_f$)). The output from the previous step (FWC) is then be used as the input for the PSO.

Algorithm 3 shows the details of the optimization. Based on the resultant force over the unit vectors (line 1) $v_x$ and $v_y$ derived from $\overrightarrow{F(S_J)}$ (line 2), the algorithm iterates a predefined number of times to determine the best approximation. Lines 4 to 5 state the weight derivations based on Equation (6). Here, $iw_{max} = 2^{\frac{rmax}{r}} - 1$, where *rmax* is the maximum sensing coverage (radius *r*), which is 100 m, and $iw_{min} = 1$. Then, lines 6 to 7 calculate the update velocities ($v_{x+1}$ and $v_{y+1}$), including the random effects that lead to the location update ($pb_x$, $pb_y$) in line 8. Here, $c_1$ is set to 1 based on our intensive evaluation.

In FWC-PSO, *gb* is the same as *pb* because this process finds only one estimation point in each round. The fitness function (distance) of *pb* based on Equation (3) will then be derived (line 9) and used to select the most suitable population given the fitness evaluation of the other particles ($x_i$, $y_i$), as stated in lines 10–18. The fitness function used for population selection involves the shortest distance between the estimated nodes' positions (unknown) and the anchor nodes' positions (the maximum RSSI).

```
Algorithm 3: FWC-PSO for WSN Localizations
Input: $(x_f, y_f), (x_i, y_i), (x_{est}, y_{est}), r, t_{max},$
Output: $pb_x, pb_y$
1. Calculate the resultant force based on Equation (2)
2. Initialize $v_x = \overrightarrow{F_x(S_J)}$ and $v_y = \overrightarrow{F_y(S_J)}$
```

```
while ($t$++ < $t_{max}$)
    Initialize the max and min of the inertia weights $iw_{max}$ and $iw_{min}$
    Calculate the inertia weight based on Equation (6)
    Calculate $v_{x+1} = iw \times v_x + c_1\text{rand}(-1, 1)$
    Calculate $v_{y+1} = iw \times v_y + c_1\text{rand}(-1, 1)$
    Update the new approximate position $pb_x = x_{est} + v_{x+1}$ and $pb_y = y_{est} + v_{y+1}$
    Evaluate the fitness function on the distance $(d_f)$ based on Equation (3) with parameters $pb$ and $(x_f, y_f)$
    for $i$ = 1: $N$-1
        Evaluate the fitness function on the distance $(d_i)$ based on Equation (3), with parameters $(x_i, y_i)$ and $(x_f, y_f)$
        if $(d_i < d_f)$ then
            $x_{est} = pb_x$ and $y_{est} = pb_y$
        else
            $pb_x = x_{est}$ and $pb_y = y_{est}$
            break
        end if
    end for
end while
```

### 3.3 Enhanced ELM

The enhanced centroid method (discussed previously in Sections 3.1 and 3.2) can effectively generate an estimated location for unknown nodes; however, its estimation precision is limited in high node density scenarios. Thus, this section describes the application of ELM to overcome that issue. The traditional ELM architecture was initially proposed as a single hidden layer feed-forward network (SLFN) [36,44], the main feature of which is the ability to use random independent nonlinear feature transformations. Additionally, in an SLFN, unlike a traditional NN, there is no requirement to adjust the hidden nodes' weights, which results in a fast training algorithm. Here, we propose using ELM to optimize the WSN node localizations. The concept of the resultant force using unit vectors is also employed to adjust the locations from the optimized ELM.

#### A. ELM

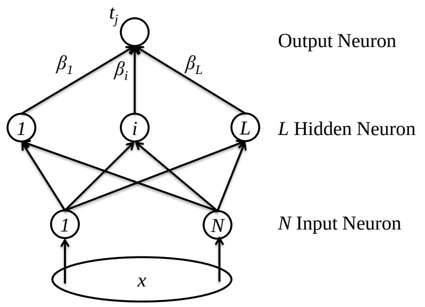


Figure 8. ELM (SLFN)

Similar to NN, Figure 8 shows an overview of the ELM; however, the details of the process are somewhat different, especially without the backpropagation procedure. A total of $N$ input neurons in $(x_i, t_i)$ format such that $i = 1, 2, 3, \ldots, N,$ and the input $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^T$ enters the network, which is given the target $T$ or $t_i = (t_{i1}, t_{i2}, \ldots, t_{im})^T$, where $n$ denotes the input dimension and $m$ is the targeted class. The output weight $\beta$ is used to convert the nonlinearity and is derived from Equation (10) for training purposes. The output weight $\beta$ can be obtained by solving the least-squares solution shown in Equation (11):

$$\beta = H^{\dagger}T \tag{10}$$

$$min \sum_{i=1}^{N} \|\beta_i \cdot h_i - t_i\|, \tag{11}$$

where $H$ is a hidden node function, i.e., $\{h_{ij}\} \mid (i = 1, \ldots, N$ and $j = 1, \ldots, L)$ derived from $h_{ij} = g(w_j x + b_j)$. Here, $L$ is the number of hidden nodes and $H^{\dagger}$ is a Moore-Penrose matrix such that the hidden nodes' input weights $w_j = (w_{j1}, w_{j2}, \ldots, w_{jN})^T$ have the bias $b_j$. Moreover, $g(x)$ denotes the activated function, e.g., hard limit, sigmoid, radial basis, triangular basis, and sine [45].

Additionally, during the testing stage, given the output weight $\beta$ from the training, the unknown input $x$ will be fed into $h_{ij} = g(w_j x + b_j)$ given the defined weight ($w$) (from training) before applying the activation function. Then, the reverse equation will be employed to compute the predicted target, as shown in the equation below.

$$T = H\beta. \tag{12}$$

### B. Optimized ELM in Wireless Sensor Networks

The ELM is applied in WSN localizations via Algorithm 4, which includes a detailed ELM training phase. The RSSI is mainly used as the input from position ($x$, $y$) of the known nodes (anchors) $N$ given the actual node position or target $T$. The output of this training phase will be used as the input weight $w$, output weight $\beta$, and bias $b$. First, the input weight and bias are randomly generated (lines 1–2). Next, the activation function is performed based on its characteristics (lines 3–12). Then, a training matrix will be determined with a summation of the result matrix from previous process ($C$) with bias. This matrix is used as an input to compute an activation function ($G$) (line 14). Finally, the output weight will be derived according to Equation (10) (line 15).

```
Algorithm 4: ELM in the Training Phase for WSN Localizations (ELM_TRAIN)
Input: $RSSI[N \times N], T[N]$
Output: $w[L \times N], \beta[L], b[L \times N]$
```

```
Generate a random input weight $w[L \times N]$ in the range [-1, 1]
Generate a random bias $b[L \times N]$ in the range [0, 1]
Let $C$ be a new matrix of the appropriate size
for $i$ from $1$ to $N$
    for $j$ from $1$ to $L$
        $sum = 0$
        for $k$ from $1$ to $m$
            $sum \leftarrow sum + w_{ik} \times RSSI_{kj}$
    end
        Set $C_{ij} \leftarrow sum$
    end
end
Calculate $F\_Train = C[L \times N] + b[L \times N]$
Calculate the activation function $H\_Train = G(F\_Train[L \times N])$ | $G$ is the hard limit function
Calculate the output weight $\beta = H\_Train^{\dagger}T$ where $T = [t_{i1}, t_{i2}, ..., t_{im}]^{T}$
```

In the testing stage, Algorithm 5 shows that after the RSSI normalization process, similar processes to those for training will be performed. However, at this point, in additional to the input weight $w$, output weight $\beta$, and bias $b$ derived from the training process, the *RSSI_Test* (the RSSI of unknown nodes) is used to compute the approximate location ($x_{est}$).

Algorithm 5: ELM in the Testing Phase for WSN Localizations (ELM_TEST)

Input: $RSSI\_Test[N \times N], w[L \times N], \beta[L], b[L \times N]$

Output: $x_{est}$

```
Let $C$ be a new matrix of the appropriate size
for $i$ from $1$ to $N$
    for $j$ from $1$ to $L$
        $sum = 0$
        for $k$ from $1$ to $m$
            $sum \leftarrow sum + w_{ik} \times RSSI\_Test_{kj}$
        end
        Set $C_{ij} \leftarrow sum$
    end
end
Calculate $F\_Test = C[L \times N] + b[L \times N]$
Calculate the activation function $H\_Test = G(F\_Test[L \times N])$ | $G$ is the hard limit function
Estimate the position $x_{est} = (H\_Test)^{T} \times \beta[L]$
```

Figure 9 also shows an example of how to map the ELM in both the training and testing stages for location approximation according to Algorithms 4 and 5. The former is used for training using the RSSIs of anchor node sets by including the locations of the anchor nodes as targets; the latter is used to apply the output from the former algorithm to conduct an approximation using RSSIs as the inputs (unknown nodes received from the anchor nodes within a communication range).

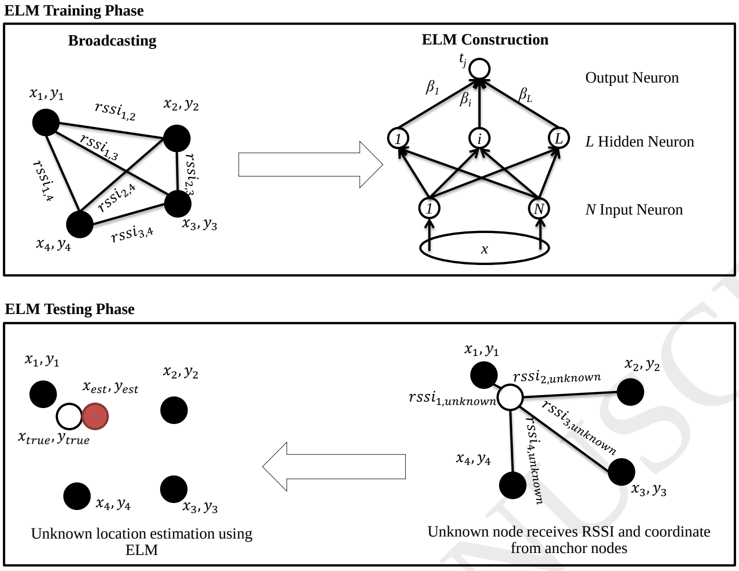


Figure 9. ELM Training and Testing: Example

## C. Threshold-based ELM in Wireless Sensor Networks

When applying the ELM to WSN localizations, when the sensing coverage is increased, the estimation precision can be reduced due to the ELM computation, which could conceivably include anchor nodes that are far from the actual node position, as shown in Figure 10. Here, the error tends to diminish as the radius increases from 20 to approximately 60 m; however, outside this radius range (see error fluctuations), the error increases as the radius increases from approximately 60 to 80 m. Thus, in this research, the introduction of threshold-based ELM (T-ELM) is also proposed to limit the boundary of interest for the anchor nodes, i.e., *RSSI_Threshold*.

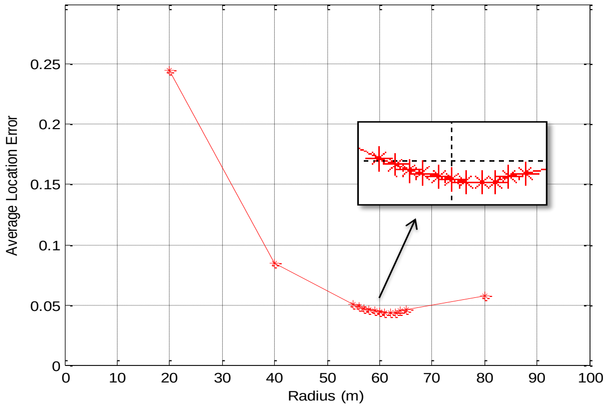


Figure 10. ELM Localization Error Behavior

Algorithm 6 shows the details of T-ELM. Given a set of RSSIs from the anchors of interest (*RSSI_Test*), we limit the minimum and maximum signal strengths (*RSSI_min* and *RSSI_max*) so that they do not affect the ELM estimation position (lines 1–5). Note that the resultant set will feed the ELM approximation to derive the primary target ($T_1$) in line 6. After predefining the output threshold, *RSSI_min* will be updated given the RSSI window (*window_size*), for example, 1 m. Then, a process similar to that in lines 1 to 5 will be applied given the updated minimum RSSI (lines 10–14). The secondary target ($T_2$) is computed in line 15. This process continues until the minimum RSSI is less than the maximum or until the secondary target is beyond the primary target as shown in the fitness functions (lines 8 and 16–21), including the updates to these two targets.

```
Algorithm 6: Threshold-based ELM in the Testing Phase for WSN Localizations (T-ELM)
Input: $RSSI\_Test[N \times N], w[K \times N], \beta[K], b[K \times N]$, RSSI_min, RSSI_max, window_size
Output: RSSI_Threshold
1. for ($i$ = 1 to size(RSSI_Test))
2.        if ($RSSI\_Test_i$ < $RSSI\_min$) then
3.               $RSSI\_Test_i$ = $RSSI\_min$
4.        end if
5. end for
6. Calculate the primary target $T_1$ = ELM_TEST($RSSI\_Test[N \times N], w[K \times N], \beta[K], b[K \times N]$)
7. RSSI_Threshold = RSSI_min
8. while (RSSI_min > RSSI_max)
9.        Update $RSSI\_min = RSSI\_min - window\_size$
10.       for ($i$ = 1 to size(RSSI_Test))
11.              If ($RSSI\_Test_i$ < $RSSI\_min$) then
12.                     $RSSI\_Test_i$ = $RSSI\_min$
13.              end if
14.       end for
15.       Calculate the secondary target $T_2$ = ELM_TEST($RSSI\_Test[N \times N], w[K \times N], \beta[K], b[K \times N]$)
16.       if ($T_1 < T_2$) then
17.              break
18.       else
19.              RSSI_Threshold = RSSI_min
20.              $T_1 = T_2$
21.       end if
22. end while
```

## D. Threshold-based ELM-Particle Swarm Optimization (T-ELM-PSO)

Although T-ELM can improve the precision of the location estimation, the accuracy of T-ELM also depends on the number of nodes of interest and the sensing coverage. More specifically, T-ELM can efficiently improve the performance in a hole scenario, but its precision degrades when there is an imbalance of the sensing information in the hole. Therefore, in this research, similar to FWC-PSO, we also applied the resultant force to determine the proper direction. We capitalized on the PSO behavior by gradually moving the estimated node location toward the actual location (i.e., closer to the first anchor node) using the maximum RSSI ($x_f$, $y_f$). The output from the previous step (T-ELM) is then used as the input for the PSO (T-ELM-PSO).

Algorithm 7 shows the details of this optimization. Based on the resultant force over the unit vectors (line 1) and the $v_x$ and $v_y$ derived from $\overrightarrow{F(S_j)}$ (line 2), the algorithm iterates for up to a predefined number of iterations to determine the best approximation. Here, $iw_{max} = \frac{rmax}{r}$, where *rmax* is the maximum sensing coverage (radius *r*), which is 100 m, and $iw_{min} = 1$. After the initial personal best solution (line 4) is found, lines 5–6 calculate the weight derivations derived from Equation (6). Then, the updated velocities ($v_{x+1}$ and $v_{y+1}$) are derived based on the random effects and the difference between the personal best solution and the current estimation (lines 7–8). Finally, the new approximated location will be updated ($pb_x$, $pb_y$) (line 9).

In T-ELM-PSO, *gb* is the same as *pb* because this process finds only one estimation point in each round. The fitness function (distance) of *pb* and the anchor node ($x_f$, $y_f$) that has the maximum RSSI based on Equation (3) will then be derived (line 10), after which a similar computation is also performed but using the estimated unknown node ($x_{est}$, $y_{est}$) in addition to the specific anchor node (line 11). Lines 12–15 show the update of *pb* before iteratively performing the approximation toward the threshold.

```
Algorithm 7: Threshold-based ELM-PSO in the Testing Phase for WSN Localizations (T-ELM-PSO)
Input: $(x_f, y_f), (x_{est}, y_{est}), t_{max}, r$
Output: $pb_x, pb_y$
1. Calculate the resultant force $\overrightarrow{F(S_j)}$ based on Equation (2)
2. Initialize $v_x = \overrightarrow{F_x(S_j)}$ and $v_y = \overrightarrow{F_y(S_j)}$
3. while ($t$++ < $t_{max}$)
4.     Initialize $pb_x = x_{est}, pb_y = y_{est}$
5.     Initialize the max and min of the inertia weights $iw_{max}$ and $iw_{min}$
6.     Calculate the inertia weight based on Equation (6)
7.     Calculate $v_{x+1} = iw \times v_x + c_1 \text{rand}(-1, 1) \times (pb_x - x_{est})$
8.     Calculate $v_{y+1} = iw \times v_y + c_1 \text{rand}(-1, 1) \times (pb_y - y_{est})$
9.     Update the new approximate position $pb_x = x_{est} + v_{x+1}$ and $pb_y = y_{est} + v_{y+1}$
10.    Evaluate the fitness function on the distance ($d_{fnew}$) based on Equation (3), with parameters $pb$ and $(x_f, y_f)$
11.    Evaluate the fitness function on the distance ($d_{fold}$) based on Equation (3), with parameters $(x_{est}, y_{est})$ and
       $(x_f, y_f)$
12.    if ($d_{fold} < d_{fnew}$) then
13.        $pb_x = x_{est}$ and $pb_y = y_{est}$
14.        break
15.    end if
16. end while
```

When soft computing is applied as a heuristic approach to the localization problem, there is an assumption that the computational work is performed at a base station (BS). There are two types of nodes excluding BSs: known and unknown location nodes. There are also two phases: a training phase and a testing phase. In the training phase, all the known information such as the exact location and RSSI within the sensing radius is sent back to the BS. We do not specifically limit the routing techniques; in fact, there are several approaches to WSN routing that consider the data-centric, location-based, hierarchical, network flow and QoS-aware protocols [16], as described earlier in the literature review [8,46,47]. However, taking the routing into account could require additional calculations for the node communicators. When the required information (from the known nodes) is sent to the BS, the soft computing computation will be performed that generates the output to the training model. For testing, the unknown node will also send the information to the nearest known location, which acts to forward the information to the BS. Finally, the BS recomputes the estimation process and sends the approximation back to the unknown node.

### 3.4 Hybrid Centroid and ELM Optimizations

As previously stated, the centroid- and ELM-based derivations have advantages and disadvantages. Our final contribution is to propose a hybrid model over these two combinations, i.e., hybrid centroid and ELM optimizations, as follows (Figure 2):

- Centroid-based model: The input RSSIs are fed into FL-Mamdani using a triangular membership function with five rules. Then, the weight derived from the fuzzy inference system is applied to adjust the location estimation of the centroid in the process called FWC. The force vector is then applied to determine the direction before iteratively moving the approximation so that it is closer to the actual location with a metaheuristic scheme (i.e., PSO, called FWC-PSO).
- ELM-based model: After the traditional ELM steps with WSN localization parameters (O-ELM), a modification over the approximated distance is derived (T-ELM). The improvement further proceeds with a technique similar to FWC-PSO, called T-ELM-PSO.
- Hybrid model: With knowledge of the number of known nodes or anchor nodes, by considering the node density and sensing the coverage while considering an irregular topology, we applied an additional parameter, an example of which is shown in the equation below:

$$\beta \times \mathit{FWC\text{-}PSO} + \alpha \times \mathit{T\text{-}ELM\text{-}PSO}, \tag{13}$$

where $\alpha$ is a summation of the ratio of the number of anchor nodes with sensing coverage over the total number of anchor nodes and the ratio of the sensing coverage over the maximum coverage (the IEEE 802.15.4 range). Here, $\beta$ is $1 - \alpha$ and is used as a normalization factor between the centroid and ELM to estimate the unknown node position. In other words, in dense scenarios or areas with wide coverage, ELM has a higher accuracy than centroid-based approaches.

### 3.5 Complexity Analysis

The proposed hybrid model has two main stages: enhanced centroid and enhanced ELM. For the former, the computational complexity is $O(n)$ given only a single iteration, as shown in Equation (9). Using the weights derived from an FLS, Kim et al. [48] provided an analysis of the fuzzy process and reported that there are three fuzzy stages: fuzzification, inference, and defuzzification. Based on the five operation methods (i.e., addition, subtraction, comparison, multiplication, and division), the number of operations for each state is as follows: $(59+31N_{IF})N_I$ with triangular, $(63M_{OD}+37N_I+19)L+6$ with minimum inference, and $(39M_{OD}+5)L+15$ with CoG. Here, $N_I$, $N_{IF}$, $M_{OD}$, and $L$ are the number of inputs ($I$) and input fuzzy sets ($IF$), the discretization of output ($OD$), and rules ($L$), respectively. Assuming that these are $n$s, the maximum complexity is $O(n^2)$.

The ELM is based on the matrix operations shown in Equations (10) to (12). For training, given Equation (9), the matrix transpose ($H^T$), multiplication ($H^TH$), and inverse $(H^TH)^{-1}$ operations are used. Thus, the analyzed complexity is $O(n^3)$ [49], which results in an output ($w$, $b$, and $\beta$) via traditional computation. For testing, Equation (12) uses only the former operations, i.e., the multiplication ($H$) between the RSSI and $w$ and another multiplication between ($H + b$) and $\beta$. The same complexity was applied. Equation (11) shows the error between the actual location and its estimation. At this step, only $n$ operations are required. These three main stages result in a complexity of $O(n^3)$; however, this complexity can be reduced using other known methods such as the Strassen algorithm or Coppersmith–Winograd algorithm [50,51].

With the optimization of the resultant force vector stated in Equation (2), the complexity is $O(n)$. This vector is also applied in both centroid and ELM in sequence. However, with PSO integration, Chuang et al. [52] discussed the complexity of $O(MX)$ and stated that $M$ and $X$ are the numbers of initial particles and iterations, respectively, required to reach the global optimum, or $O(nt_{max})$, as shown in Algorithm 1.

As a result, the enhancement of centroid, i.e., FWC-PSO, as shown in Algorithm 3, is $O(n^2 + nt_{max})$ (Fuzzy + PSO). In addition, for the ELM enhancements, T-ELM, as shown in Algorithm 6, has $O(n + cn^3)$ complexity. Here, $c$ is the conditional check comparing the maximum and minimum RSSIs. Finally, with the PSO integration (T-ELM + PSO), Algorithm 7 has a complexity of $O(t_{max} + n^3)$. Therefore, the total computational complexity of HVP-FELM, including the arithmetic operations in Equation (13), is $O(n^2 + nt_{max} + t_{max} + n^3)$.

## 4 Performance Evaluation

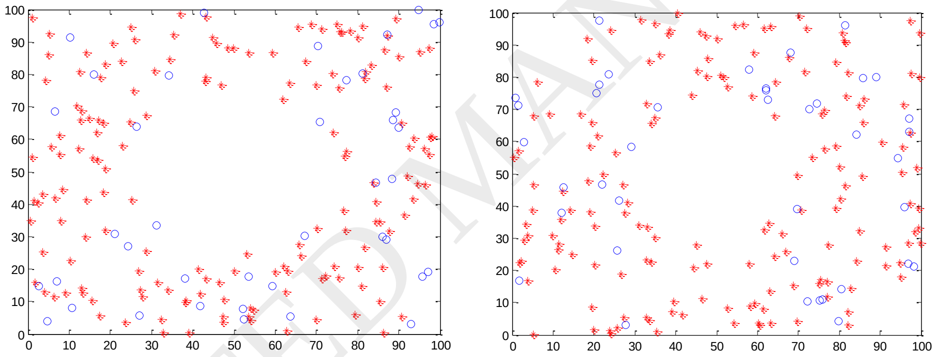
This section discusses the performance of soft computing-based approaches for WSN localizations, including traditional centroid localization. Our proposed scheme, HVP-FELM, is also evaluated and compared with the traditional centroid method, its derivative, i.e., fuzzy centroid [41], SVM [53], and ELM [30].

### 4.1 Simulation Configurations

The evaluation testbed is a standard configuration of the Windows 7 (64-bit) operating system running on an Intel(R) Core (TM) Quad Q8400 2.66 GHz CPU, with 2048×2 MB DDR-SDRAM memory and a 320 GB 7200

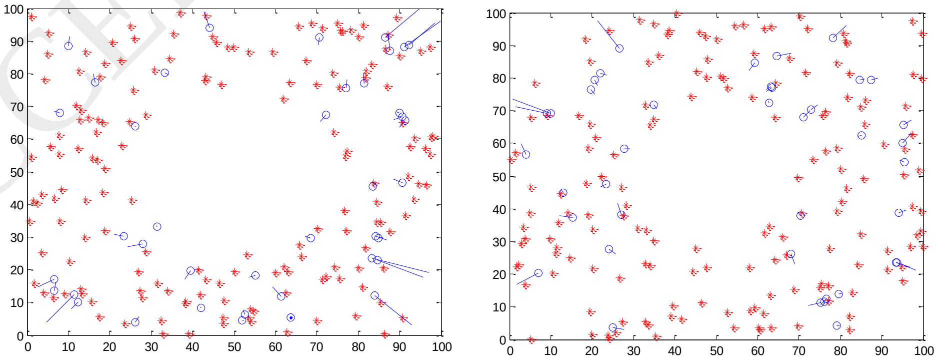
RPM hard disk. For comparison purposes, the simulation model is based on MATLAB [54] with the standard library, which is similar to the system described by Gu et al. [41]. For the sake of simplicity, the testing process is performed over a 100×100 m$^2$ simulated area in which both the anchors and the unknown nodes are placed randomly. After deployment, no node mobility is involved. In all the evaluations, the key metric is the average over ten trials at a 95% confidence level.

There are two main scenarios with regard to the non-uniform deployments: one hole and five holes [29], as shown in Figures 11 and 12, which are examples of node placements and location errors. The investigation focuses on the node density and the sensing coverage (i.e., the number of nodes in each scenario is varied from 100 to 250 nodes by an increment of 50 nodes) (see also Table 3). The effect of anchor node density is also evaluated by varying the unknown to known node ratio between 0.2, 0.4, 0.6, and 0.8; however, graphs are shown at 100 and 250 nodes to represent small and large node densities. Again, each scenario will also be evaluated with regard to the coverage (i.e., the signal radius) by varying it from 20 to 80 m with a 20-m step size over a log-distance path loss model. However, graphs will be shown at 20 and 80 m to represent the lowest and highest coverage levels, respectively [41]. Note that we evaluated all performances at increases of 50 nodes and with a 20-m step increment; however, the average location error ratio (ALR) follows the same trend (see Figures 13-20).



(a) one hole (b) five holes

Figure 11. Node Distribution Deployment: cross (anchor) and circle (unknown)



(a) one hole (b) five holes

Figure 12. Localization Error: cross (anchor), circle (unknown), line from circle (error from the actual position)

Table 3. Simulation Parameters

| **Parameters** | **Symbol** | **Scenario 1** | **Scenario 2** |
|---|---|---|---|
| Area of SNs | M × M | 100 m × 100 m | |
| Number of SNs | N | 100 and 250 nodes | |
| Communication radius | r | 20 m and 80 m | |
| Ratio of unknown to known nodes | m/N | 0.2, 0.4, 0.6, and 0.8 | |

The energy consumption of the computing node and transmission logic is not considered in this localization model evaluation [9]. Again, no routing protocol limitations exist; the focus here is only on localization, similar to Yun et al. [18,26], Tran and Nguyen [29], Chagas et al. [32], Gu et al. [41] and Gholami et al. [55]. Several WSN routing techniques are available, as stated in the literature review [8,46,47]. Therefore, communication metrics are not reflected in our evaluation; other metrics remain for future investigation. Two primary measurements are used here to evaluate the performance of WSN localizations for various soft computing models, namely, ALR [21–23], as shown in Equation (14), and computational time (seconds). The reason for the use of ALR rather than analyzing only the error difference is due to the two-dimensional data and the sum of squared errors (SSE) metric. Note that the first metric is used to measure the precision of each approach, while the second metric can be divided into testing and training phases, as captured from MATLAB. Here, the given computational time is the average over the number of estimations (unknown nodes).

$$ALR = \frac{\sum_{i=1}^{m} \sqrt{\left(x_{i\_est} - x_{i\_sensor}\right)^2 + \left(y_{i\_est} - y_{i\_sensor}\right)^2}}{m \times r}, \tag{14}$$

where ($x_{i\_sensor}$, $y_{i\_sensor}$) is the actual node position of the $i^{th}$ node corresponding to its approximation ($x_{i\_est}$, $y_{i\_est}$), *m* denotes the total number of unknown nodes, and *r* is the signal coverage.

For a comparative evaluation, we consider the centroid derivative and fuzzy centroid, which applies Fuzzy-Mamdani with a triangular membership function using the five rules provided in Tables 1 and 2, as recommended by Gu et al. [41]. In addition, Table 4 shows the detailed configuration for the PSO parameters used in the Fuzzy-PSO and ELM-PSO algorithms for reproducibility.

Table 4. Parameter Configuration of the Fuzzy-PSO and ELM-PSO Methods

| **Parameter** | **Fuzzy-PSO** | **ELM-PSO** |
|---|---|---|
| $c_1$ | 1 | 1 |
| $c_2$ | 0 | 0 |
| $t_{max}$ | 100 | 100 |
| $iw_{min}$ | 1 | 1 |
| $iw_{max}$ | 31 | 5 |

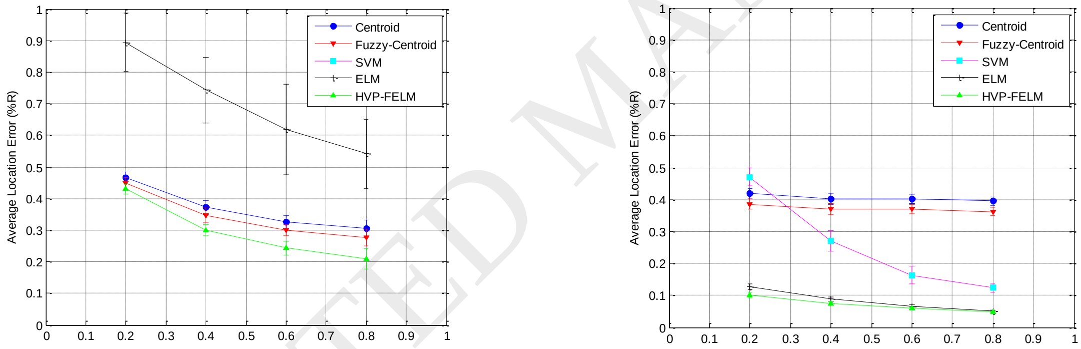
Considering that the SVM [29] and libSVM [53] versions in our evaluation platform use the RBF as a kernel function $K(x, x') = \exp(-\gamma\|x - x^{'}\|^2)$, the parameters $c$ (the regularization parameter) and $\gamma$ are set to 10 and

0.1, respectively. Note that based on our intensive evaluation [30] of the RBF, linear, polynomial, and sigmoid kernel functions, the performance of SVM-RBF was superior. In addition, based on the ELM library developed by Huang et al. [36], the hard limit activation function is used due to its outstanding performance, and the number of hidden nodes is selected to be 20,000 [30].

In our intensive evaluation based on our previous work [30], in which the actual setup is evaluated in a range from 1,000 to 40,000 nodes with various kernel functions (hard limit, sigmoid, radial basis, triangular basis, and sine) and with 20,000 nodes and hard limit, the results showed that the location precision is in a stable stage at approximately 20,000 nodes. Although there is a reduction in the trend, at 20,000 nodes, the results have high accuracy with low computational time. Our proposal also used the ELM/hard limit and fuzzy centroid/Mamdani as baselines with additional optimization steps for HVP-FELM.

## 4.2 Simulation Results

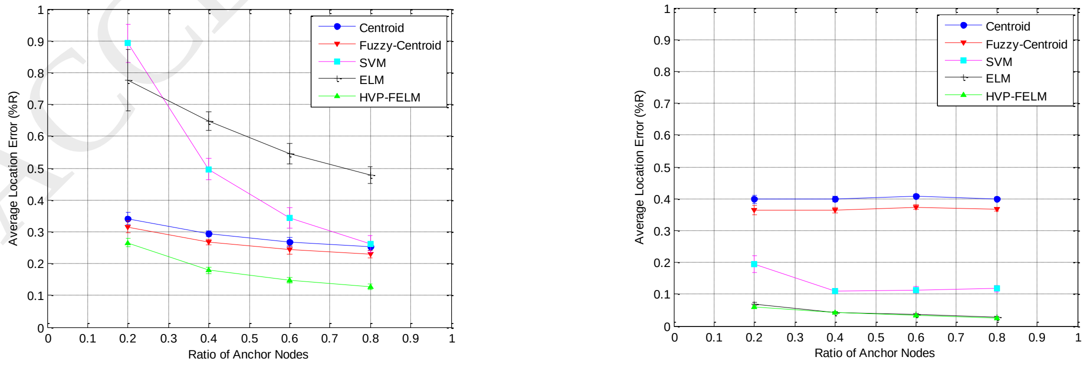
Figures 13 and 14 show the simulation results in the first scenario (i.e., one hole) as the number of nodes varied from 100 to 250 with various anchor node ratios (0.2 to 0.8) with a sensing coverage between 20 and 80 m. For the sake of simplicity, only the 20 and 80 m coverage distances are pictured (see Table 4 for details). In general, when the node density is high, the location estimation accuracy is also high, as shown in Figures 13(a), 14(a), 13(b), and 14(b).



(a) 20 m signal radius

(b) 80 m signal radius

Figure 13. ALR over the ratio of anchor nodes with 100 nodes (non-uniform, one hole)



(a) 20 m signal radius

(b) 80 m signal radius

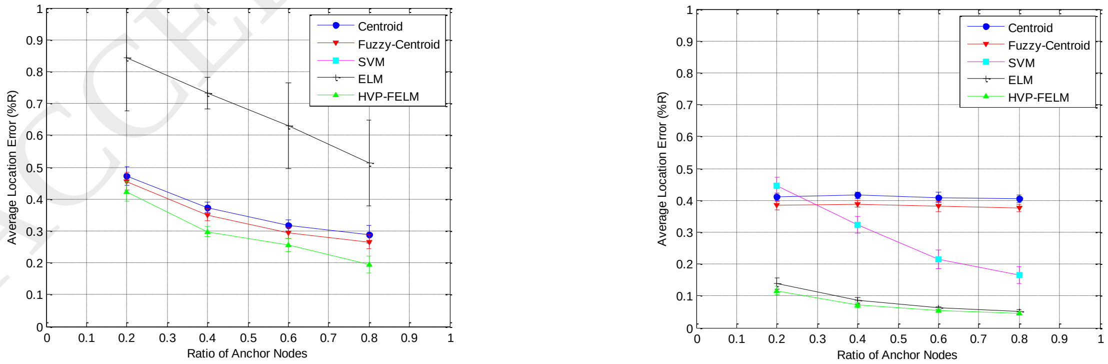
Figure 14. ALR over the ratio of anchor nodes with 250 nodes (non-uniform, one hole)

These trends are also observed when the ratio of anchor nodes increases (i.e., a smaller location estimation error is obtained). However, when varying the signal radius, the precision of the traditional centroid scheme tends to be lower, unlike the other schemes, as shown in each subfigure (i.e., (a) and (b)). In general, HVP-FELM outperformed the other schemes in all the scenarios. We also applied this approach using a uniform distribution, which we ran in a simulation to demonstrate the efficacy of HVP-FELM as a robust algorithm.

Considering the 100-node case shown in Figure 13, HVP-FELM has an outstanding performance, achieving an error range of less than 0.1 to 0.4. ELM has a low accuracy when the radius is small (20 m) of between 0.5 and 0.9; however, its precision improves as the radius increases. It is close to the precision of HVP-FELM at a sensing coverage of 80 m—less than 0.2. In contrast, the accuracy of the traditional centroid decreases as the coverage radius increases, achieving approximately 0.4 at 80 m, but less than that at a shorter sensing range. The centroid derivation fuzzy centroid mostly follows the behavior of the traditional centroid but achieves a higher accuracy—typically less than 0.4.

Here, the approximate error of the SVM falls between those of the HVP-FELM and the centroid derivatives (i.e., between 0.1 and 0.4); however, at short-range communications, the mapping space of the SVM leads to an impractical computational burden and the accuracy suffers, such as in the 20 m case (see Figure 13). In addition, with the maximum number of nodes (250 nodes) Figure 15 shows the same behaviors as those in the low-node-density cases; for example, less than 0.25, 0.65, 0.35, 0.3, and 0.95 for HVP-FELM, ELM, centroid, fuzzy centroid, and SVM, respectively.

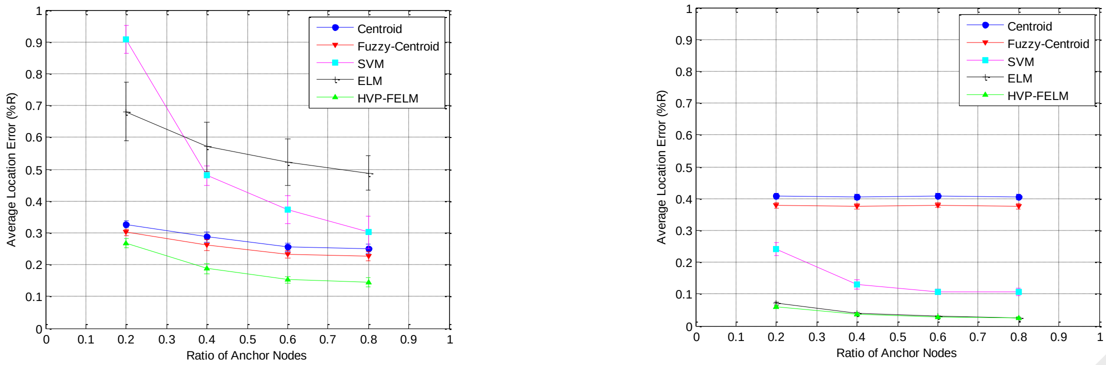
Figures 15 and 16 show the location estimation errors in the second scenario, which has five holes. In general, similar to the first scenario (one hole), HVP-FELM outperforms the other schemes in all scenarios. When the node density is high, its location estimation is also high, as shown in Figures 15(a) and 16(a). These trends are also observed when the ratio of anchor nodes is increased or a smaller location estimation error exists. Again, increasing the signal radius leads to lower accuracy for centroid-based approaches but not for other approaches.



(a) 20 m signal radius

(b) 80 m signal radius

Figure 15. ALR over the ratio of anchor nodes with 100 nodes (non-uniform, five holes)



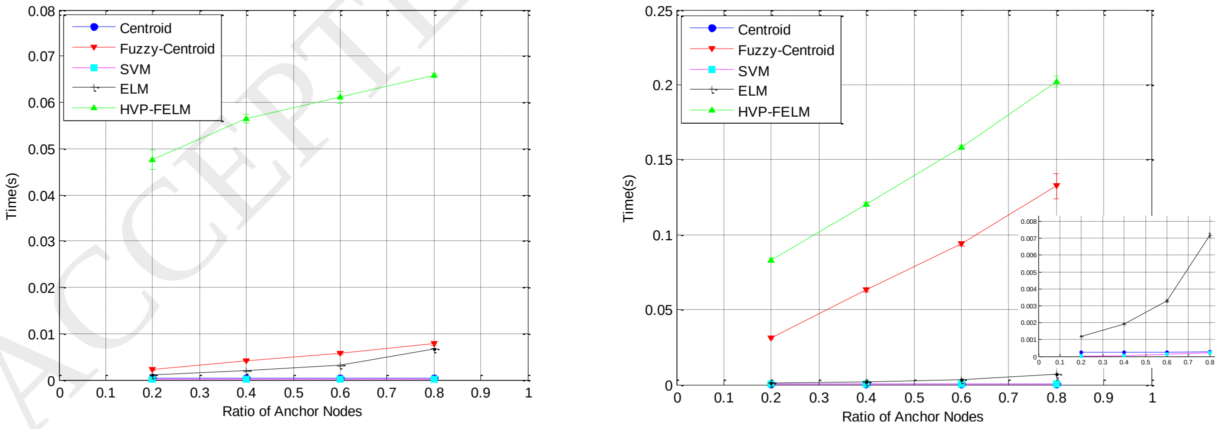
(a) 20 m signal radius

(b) 80 m signal radius

Figure 16. ALR over the ratio of anchor nodes with 250 nodes (non-uniform, five holes)

In addition, with the same configurations except for the number of holes, the accuracy in the scenario with five holes tends to be higher than that in the scenario with a single large hole. In addition, considering the lower node density of the five-hole configuration, as shown in Figure 15 with 100 nodes, the accuracies of HVP-FELM, ELM, centroid, fuzzy centroid, and SVM range from less than 0.1 to 0.45, 0.1 to 0.85, 0.25 to 0.45, 0.22 to 0.42, and 0.15 to incomputable (20 m), respectively. Again, Figure 16 shows trends similar to Figure 15 but likely has smaller estimation errors due to the higher node density.

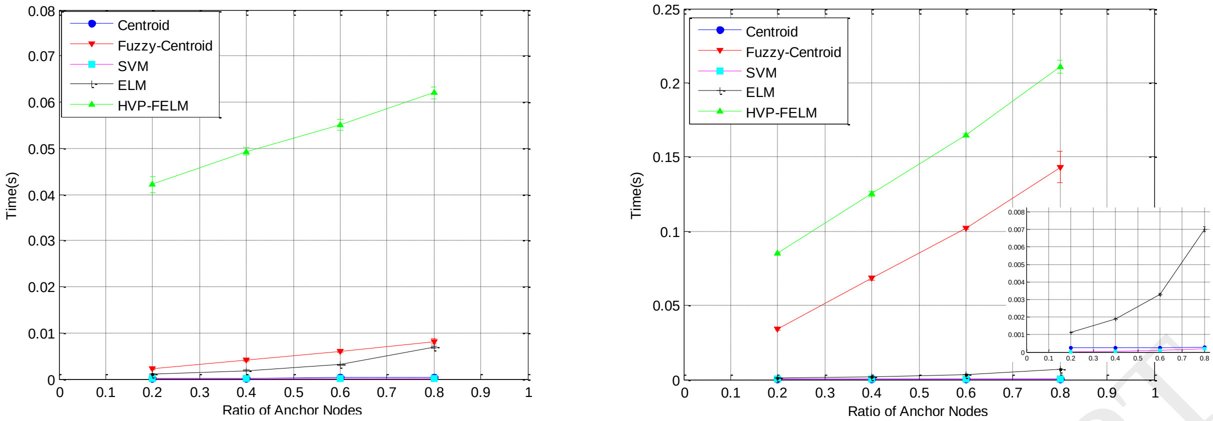
In addition to the location estimation errors, Figures 17 and 18 show the computational complexity when testing the cases of the lowest and highest complexity in the one- and five-hole scenarios. In the former, Figure 17 shows that although HVP-FELM required the most computational time, considering the accuracy yield trade-off, it required less than 0.2 additional seconds compared to the other schemes. Note that the complexity of fuzzy centroid is also higher here (with a higher ratio of anchor nodes)—close to 0.15 seconds—and this effect also led to the increased computational time required by our hybrid approach, HVP-FELM. Figure 18 shows a result similar to that of Figure 17 in both scenarios.



(a) 100 nodes and 20 m signal radius

(b) 250 nodes and 80 m signal radius

Figure 17. Computational testing time (seconds) over the ratio of anchor nodes (non-uniform, one hole)

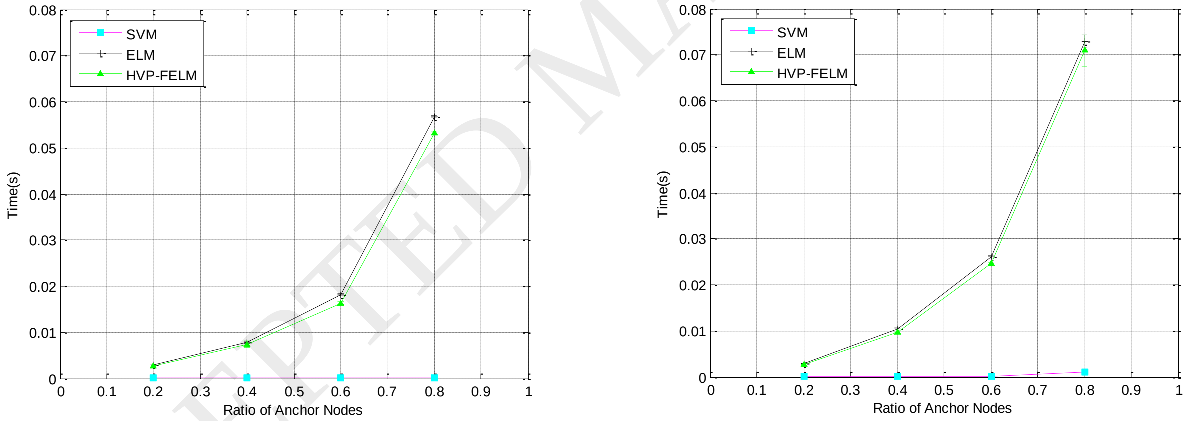


(a) 100 nodes and 20 m signal radius

(b) 250 nodes and 80 m signal radius

Figure 18. Computational testing time (seconds) over the ratio of anchor nodes (non-uniform, five holes)

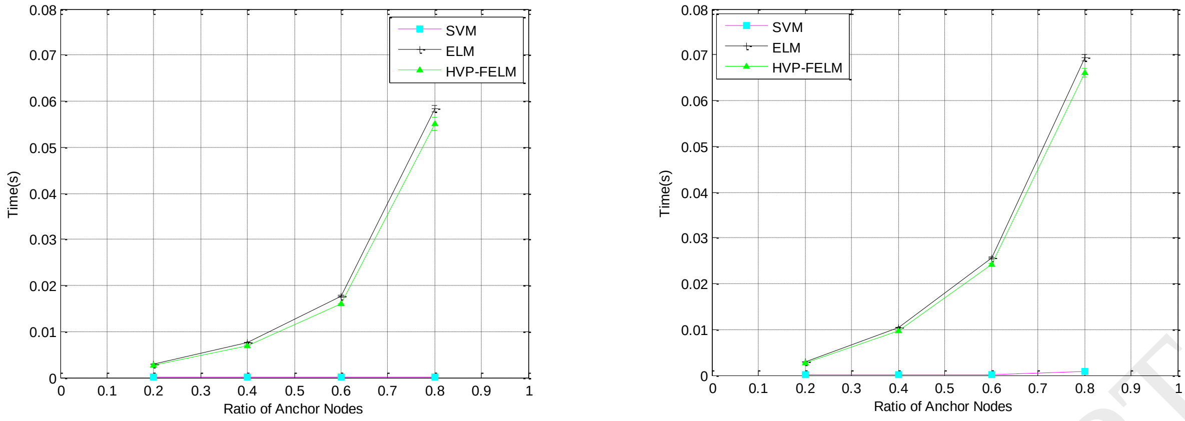
The SVM and ELM derivatives also require a training step; therefore, Figures 19 and 20 show the computational complexity in this step (one hole and five holes, respectively). Figure 19 shows that the SVM has the shortest training time (less than 0.01 seconds), but that of HVP-FELM is similar to that of the ELM (less than approximately 0.075 seconds). However, when the number of nodes increases, the training time also increases (i.e., less than 0.1 seconds in 250 nodes with the same trends and behaviors). Figure 20 shows the training time in the five-hole scenario. The computational times of all the algorithms follow similar trends as those in the one-hole scenario.



(a) 100 nodes and 20 m signal radius

(b) 250 nodes and 80 m signal radius

Figure 19. Computational training time (seconds) over the ratio of anchor nodes (non-uniform, one hole)



(a) 100 nodes and 20 m signal radius

(b) 250 nodes and 80 m signal radius

Figure 20. Computational training time (seconds) over the ratio of anchor nodes (non-uniform, five holes)

## 5 Conclusions and Future Work

In this research, we studied the application of soft computing techniques to range-free localization approximation for WSNs, specifically, the effects of node density and sensing coverage including irregular topology, especially unbalanced node deployment. Our main contribution was to propose a novel hybrid model incorporating modified versions of both fuzzy centroid and ELM, called HVP-FELM.

The selection criterion was based on the evaluation of a fuzzy system and ELM such that the estimation precision of fuzzy centroid is better with a smaller coverage radius and/or more nodes, but the estimation precision of ELM is better under the opposite conditions. In addition to this trade-off, both techniques have limitations when a border occurs with a hole and when only a hole occurs (node unbalance).

There are two main components. First, FLS is used to adjust the weight of the traditional centroid; second, ELM is proposed to optimize the location estimation precision. These two hybrid approaches were further optimized by PSO, using the resultant force to move the approximate node location closer to the actual position. The iterative nature of PSO moves the estimated position toward the known node using the maximum RSSI. These techniques are combined to create the hybrid optimization approach of Fuzzy-ELM with PSO.

In our extensive evaluation, the performance of HVP-FELM on irregular topology, is outstanding (as determined by the lowest location estimation error with a comparatively low complexity trade-off) compared to other soft-computing-based localization approaches (i.e., traditional centroid, fuzzy centroid, SVM, and ELM). In a one-hole scenario, our average performance improvement is higher than the others at 62.2%, 58.5%, 46.7%, and 54.9%, respectively, while for the five-hole scenario it is 62.2%, 58.5%, 42.0%, and 54.9%, respectively.

To maintain the precision in heterogeneous scenarios, additional investigations, assumptions, and constraints should be explored, such as QoS-aware mechanisms and data aggregation techniques. Other hybrid schemes and optimizations using soft computing should also be investigated, including comprehensive simulations and analyses. The factors to be studied include scalability in terms of the network density and diversity, network dimensions, and signal propagation as well as heterogeneous data traffic—all while considering the additional transmission protocol overhead. All these aspects will be addressed in ongoing research.

## Conflicts of Interest

The authors declare that there are no conflicts of interests regarding the publication of this research.

## Acknowledgments

This research was supported by a grant from the Department of Computer Science, Faculty of Science, Khon Kaen University.

## References

[1] L. Atzori, A. Iera, G. Morabito, The Internet of things: a survey, Comput. Netw. 54 (2010) 2787-2805.

[2] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of things (IoT): a vision, architectural elements, and future directions, Future Gener. Comp. Syst. 29 (2013) 1645-1660.

[3] V. Lindroos, M. Tilli, A. Lehto, T. Motooka, Handbook of Silicon Based MEMS Materials and Technologies, William Andrew, Burlington, 2010.

[4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Commun. Mag. 40 (2002) 102-114.

[5] J. Yick, B. Mukherjee, D. Ghosal, Wireless sensor network survey, Comput. Netw. 52 (2008) 2292-2330.

[6] H. Lee, K. Chung, K. Jhang, A study of Wireless Sensor Network Routing protocols for maintenance access hatch condition surveillance, J. Info. Process. Syst. 9 (2013) 237-246.

[7] G. Zhou, T. Yi, Recent developments on wireless sensor networks technology for bridge health monitoring, Math. Probl. Eng. 2013 (2013) 1–33.

[8] N. A. Pantazis, S. A. Nikolidakis, D. D. Vergados, Energy-efficient routing protocols in wireless sensor networks, IEEE Commun. Survey & Tutorials 15 (2013) 551-591.

[9] G. Han, H. Xu, T. Q. Duong, J. Jiang, T. Hara, Localization algorithms of Wireless Sensor Networks: a survey, Telecommun. Syst. 52 (2013) 2419-2436.

[10] C. Fuchs, N. Aschenbruck, P. Martini, M. Wieneke, Indoor tracking for mission critical scenarios: a survey, Pervasive Mobile Comput. 7 (2011) 1-15.

[11] N. Bulusu, J. Heidemann, D. Estrin, GPS-less low-cost outdoor localization for very small devices, IEEE Pers. Commun. 7 (2000) 28-34.

[12] D. K. Chaturvedi, Soft Computing Techniques and its Applications in Electrical Engineering, Springer-Verlag, Berlin Heidelberg, 2008.

[13] H. Jang, E. Topal, A review of soft computing technology applications in several mining problems, Appl. Soft Comput. 22 (2014) 638–651.

[14] G. Molina, E. Alba, Location discovery in Wireless Sensor Networks using metaheuristics, Appl. Soft Comput. 11 (2011) 1223-1240.

[15] M. S. Rahman, Y. Park, K. Kim, RSS-Based indoor localization algorithm for Wireless Sensor Network using generalized regression neural network, Arab J. Sci. Eng. 37 (2012) 1043–1053.

[16] A. M. Zungeru, L. Ang, K. P. Seng, Classical and swarm intelligence based routing protocols for wireless sensor networks: a survey and comparison, J. Netw. Comput. Appl. 35 (2012) 1508-1536.

[17] X. Liu, D. He, Ant colony optimization with greedy migration mechanism for node deployment in wireless sensor networks, J. Netw. Comput. Appl. 39 (2014) 310-318.

[18] S. Yun, J. Lee, W. Chung, E. Kim, S. Kim, A soft computing approach to localization in wireless sensor networks, Expert Syst. Appl. 36 (2009) 7552-7561.

[19] J. Huanxiang, W. Yong, T. Xiaoling, Localization algorithm for mobile Anchor Node based on genetic algorithm in wireless sensor networks, in: 2010 International Conference on Intelligent Computing and Integrated Systems (ICISS), IEEE, 2010, pp. 40-44.

[20] G. Yang, Z. Yi, N. Tianquan, Y. Keke, X. Tongtong, An improved genetic algorithm for wireless sensor networks, in: Proceedings of IEEE Int. Conf. of the Bio-Inspired Computi.: Theories and Appl., 2010, pp. 439-443.

[21] V. R. Maneesha, P. L. Divya, V. K. Raghavendra, M. Rekha, A swarm intelligence based distributed localization technique for wireless sensor network, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ACM, 2012, pp. 367-373.

[22] K. S. Low, H. A. Nguyen, H. Guo, A particle swarm optimization approach for the localization of a wireless sensor network, in: IEEE International Symposium on Industrial Electronics, IEEE, 2008, pp. 1820-1825.

[23] A. Gopakumar, L. Jacob, Localization in wireless sensor networks using particle swarm optimization, in: Proceedings of the IET International Conference on Wireless, Mobile and Multimedia Networks, 2008, pp.227-230.

[24] V. Kumar, A. Kumar, S. Soni, A combined Mamdani-Sugeno fuzzy approach for localization in wireless sensor networks, in: Proceedings of the of the International Conference & Workshop on Emerging Trends in Technology, ACM, 2011, pp. 98-803.

[25] D. F. Larios, J. Barbancho, F. J. Molina, C. León, LIS: localization based on an intelligent distributed fuzzy system applied to a WSN, Ad Hoc Netw. 10 (2012) 604-622.

[26] S. Yun, J. Lee, W. Chung, E. Kim, Centroid localization method in wireless sensor networks using TSK fuzzy modeling, in: Proceedings of the Int. Symposium on Advanced Intelligence Systems, 2005, pp. 971-974.

[27] L. Cheng, C. Wu, Y. Zhang, H. Wu, M. Li, C. Maple, A survey of localization in Wireless Sensor Network, Int. J. Distrib. Sens. Netw. 2012 (2012) 1–12.

[28] N. A. Alrajeh, M. Bashir, B. Shams, Localization techniques in wireless sensor networks, Int. J. Distrib. Sens. Netw. 2013 (2013) 1–9.

[29] D. A. Tran, T. Nguyen, Localization in wireless sensor networks based on support vector machines, IEEE Trans. Parallel Distrib. Syst. 19 (2008) 981-994.

[30] C. So-In, S. Permpol, K. Rujirakul, Soft computing-based localizations in wireless sensor networks, Pervasive Mobile Comput. 29 (2016) 17–37.

[31] A. Shareef, Y. Zhu, and M. Musavi, Localization using neural networks in wireless sensor networks, in: 2008 1st International Conference on Mobile Wireless Middleware, Operating Systems, and Applications, 2008, pp. 1–7.

[32] S. H. Chagas, J. B. Martins, L. L. de Oliveira, An Approach to Localization Scheme of Wireless Sensor Networks Based on Artificial Neural Networks and Genetic Algorithms, in: 2012 10th International Conference of New Circuits and Systems Conference, IEEE, 2012, pp. 137-140.

[33] R. Samadian, M. Noorhosseini, Improvements in support Vector machine based localization in Wireless Sensor Networks, in: 2010 5th International Symposium on Telecommunications (IST), IEEE, 2010, pp. 237–242.

[34] S. Phoemphon, C. So-In, T.G. Nguyen, An enhanced wireless sensor network localization scheme for radio irregularity models using hybrid fuzzy deep extreme learning machines,Wireless Networks (2016) doi:10.1007/s11276-016-1372-2

[35] G. Yang, L. Junfa, C. Yiqiang, J. Xinlong, Constraint online sequential extreme learning machine for lifelong indoor localization system, in: Proc. of the International Joint Conference on Neural Networks. IEEE, Los Alamitos, 2014, pp. 732-738.

[36] G. Huang, Q. Zhu, C. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (2006) 489-501.

[37] N. Li, J. Chen, Y. Yuan, X. Tian, Y. Han, M. Xia, A Wi-Fi Indoor localization strategy using particle swarm optimization based artificial neural networks, International Journal of Distributed Sensor Networks. 12 (2016) 1-9.

[38] A. Kumar, A. Khosla, J. S. Saini, S. S. Sidhu, Range-free 3D node localization in anisotropic wireless sensor networks, Appl. Soft Comput. 34(2015) 438-448.

[39] J. Kennedy, R. C. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, vol. 4, IEEE, 1995, pp. 1942-1948.

[40] S.Y. Kim, O.H. Kwon, Location estimation based on edge weights in wireless sensor networks, J. of Korean Institute of Communications and Information Sciences 30 (2005) 938-948.

[41] S. Gu, Y. Yue, C. Maple, C. Wu, Fuzzy logic based localization in Wireless Sensor Networks for disaster environments, in: 2012 18th International Conference on Automation and Computing, IEEE, 2012, pp. 1-5.

[42] A. Matic, A. Popleteev, V. Osmani, O. Mayora-Ibarra, FM radio for indoor localization with spontaneous recalibration, Pervasive Mobile Comput. 6 (2010) 642-646.

[43] G. Zhou, T. He, S. Krishnamurthy, J. A. Stankovic, Models and solutions for radio irregularity in wireless sensor networks, ACM T. Sensor Network. 2 (2006) 221-262. doi: 10.1145/1149283.1149287.

[44] H. Chen, J. Peng, Y. Zhou, L. Li, Z. Pan, Extreme learning machine for ranking: generalization analysis and applications, Neural Netw. 53 (2014) 119-126.

[45] F. Zorriassatine, J.D.T. Tannock, A review of neural networks for statistical process control, J. Intell. Manuf. 9 (1998) 209-224.

[46] W. Guo, W. Zhang, A survey on intelligent routing protocols in wireless sensor networks, J. Netw. Comput. Appl. 38 (2014) 185-201.

[47] K. Akkaya, M. Younis, A survey on routing protocols for wireless sensor networks, Ad Hoc Netw. 3 (2005) 325–349.

[48] Y. H. Kim, S. C. Ahn, W. H. Kwon, Computational complexity of general fuzzy logic control and its simplification for a loop controller, Fuzzy Sets Syst. (2000) 111 215-224.

[49] S. C. Althoen, R. McLaughlin, Gauss–Jordan reduction: a brief history, Am. Math. Mon. 94, (1987) 130-142.

[50] D. Coppersmith, S. Winograd, Matrix multiplication via arithmetic progressions, J. Symbolic Comput. 9 (1990) 251-258.

[51] A. Schönhage, V. Strassen, Schnelle Multiplikation großer Zahlen, Comput. J. 7 (1971) 281-292.

[52] C. L. Chuang, C. H. Jen, C. M. Chen, G. S. Shieh, A pattern recognition approach to infer time-lagged genetic interactions, BioInformatics 24 (2008) 1183-1190.

[53] C. Chang, C. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 27.

[54] MathWorks, Matlab toolbox, 2014. Available at: www.matlab.com.

[55] M. Gholami, N. Cai, R. W. Brennan, An artificial neural network approach to the problem of wireless sensors network localization, Robot Cim.-Int. Manuf. 29 (2013) 96-109.