International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 / https://doi.org/10.14445/22315381/IJETT-V70I10P242
Volume 70 Issue 10, 428-437, October 2022

Original Article

Location Estimation of Multiple Emitting RF Sources Using Supervised Machine Learning Technique

Kamel H. Rahouma[1], Aya S. A. Mostafa[2]

Department of Electrical Engineering, Faculty of Engineering, Minia University, Minia, Egypt

[2]Corresponding Author : ayasami89@yahoo.com

Received: 15 July 2022 Revised: 16 October 2022 Accepted: 20 October 2022 Published: 23 October 2022

Abstract *- Location estimation of many emitting RF sources in space is considered crucial in civilian and military applications. In the present work, many emitter source signals are separated into individual emitting sources, and the location of each source is estimated. Two antenna array stations, A and B, are used to collect the data of the emitting sources. A music algorithm is used to estimate the AOAs. The emitter signals are separated using the Music output angles and array signal processing. The correlation of a source signal received by station A and station B will estimate the TDOA between the two array stations. Thus, a hybrid AOA-TDOA method is used to estimate the location of every individual emitting source. Matlab programming environment is used to design the algorithms used in the geolocation estimation process and present the output results. Supervised machine learning is tested to simplify the calculation complexity and enhance the output results. The present work uses the Matlab 2019b Statistics and Machine Learning Toolbox to build the classification models of emitter station received signals. Different classification algorithms of the tool kit classification learner were tried to get better accuracy. It is found that fine tree and KNN algorithms achieve better results. The resulting output proves that ML could be used to apply multi-emitter geolocation estimation.*

Keywords *- Classification with Matlab toolbox, Emitter-Sensor data collection, Geolocation with machine learning, Machine learning applications, Supervised Machine learning.*

1. Introduction

Geolocating emitting sources may be solved by dividing the multi-emitter into multiple mono-emitters. In the present work, two methods are tested to solve the problem. At first, the analytical method is used, and supervised machine learning is tried for simplification and enhancement. The following assumptions are made to simplify the process of signal detection and data collection [1]: -The sensors are arranged in an antenna array, the antenna elements are placed in the shape of a uniform linear array (ULA), and all elements are identical, and the positions are accurate. - The antenna array is located in the far field region of the source such that the wave is projected to the plane wave. - All emitter signals are narrowband uncorrelated signals having the same center frequency. This assumption ensures that any element in the array can detect the signal. - Signal and noise are uncorrelated, the noise is zero mean white Gaussian noise, and the noise between array elements is zero.

1.1. Analytical Method

An AOA algorithm is firstly performed using the output of the antenna array of elements. Next, TDOA is computed for each emitter using the same technique for the two sensors, but there will be two antenna arrays. Then the location of the emitter is calculated with the previously used hybrid AOA-TDOA algorithm [2]

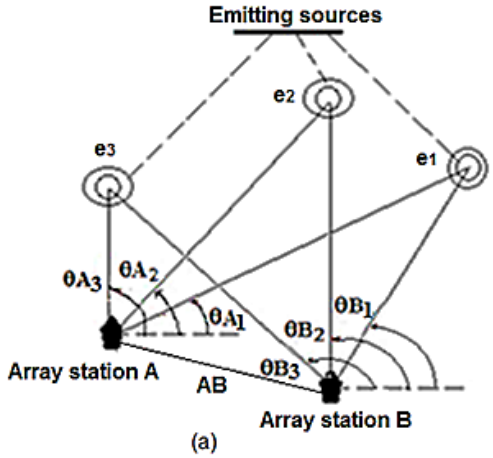
1.2. Previous Work

A variety of methods for locating multi-RF emitting sources have been developed. Ferréol et al. [3] introduced a hybrid AOA-TDOA multistage algorithm to estimate multi-emitters' locations in the x-y plane. Bryan [4] investigated the problem of locating multiple noncooperative radio frequency (RF) emitters using only received signal strength (RSS) data. The novel target acquisition and localization algorithm (TALA) introduced by Hernandez [5] offers a capability for detecting and localizing multiple targets using the intermittent “signals-of-opportunity” (e.g., acoustic impulses or radio frequency transmissions) they generate. Wu et al. [6] introduced a multi-emitter 2D angle-of-arrival (AoA) estimation scheme based on impinging signal spatial sparsity. In the present work, the geolocation process is simplified to be a mono-emitter location problem utilizing the hybrid AOA-TDOA method.

1.3. Antenna Array Received Signals

Let us consider where the RF emitters and the two antenna stations are located, as shown in Figure 1.

Both antenna arrays in stations A and B are identical. Assuming that the antenna array has N elements detecting M emitters such that $N \geq M$ [7].



(a)

	e1	e2	e3
Array station A	θA1 τA1	θA2 τA2	θA3 τA3
Array station B	θ B1 τB1	θ B2 τB2	θB3 τB3

(b)

Fig. 1 (a) emitters-stations geometry, (b) emitters-stations data

According to the stated assumptions, the incident signals of the emitting sources will be as shown in Figure 2, where:e_1:e_M emitting sources, $\theta_1:\theta_M$AOAs of the emitters, $\tau_1:\tau_M$ delay times of the emitters, 1:N antenna array elements, s_1:s_N antenna output signal, d the array elements separation distance, and x(t) the array output signal.

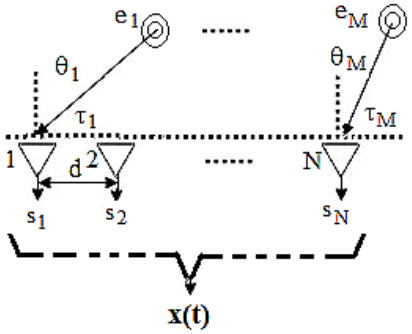


Fig. 2 Emitter signals impinging antenna array elements

According to Figure 2, the M-emitters are e_m ($1 \le m \le M$). In general, the emitter observation vector x(t) at the array output is [8]:

$$\mathbf{x}(t) = \sum_{m=1}^{M} \gamma_m \times \mathbf{a}(\theta_m) \times e_m(t - \tau_m) + \boldsymbol{n}(t) \quad (1)$$

where: τ_m is the TOA(Time Of Arrival) of the m^{th} emitter, γ_m .. the attenuation coefficient of the m^{th} emitter, θ_m is the AOA of the m^{th} emitter, a(θ) is the emitter angle vector in the azimuth θ (steering vector), n(t) is the additive noise vector, $e_m(t)$ is the signal of the m^{th} emitter, and $e_m(t - \tau_m)$ is then associated to the emitter path. Thus, in vector representation expression, Equation 1 becomes:

$$x(t) = A(\theta_e) \times \Gamma_e \times \mathrm{e}(\tau_e, t) + n(t) \quad (2)$$

where $\tau_e = [\ \tau_1, \bullet\bullet\bullet, \tau_M\ ]^T$, is the TOA vector, and $\theta_e = [\ \theta_1, \bullet\bullet\bullet, \theta_M\ ]T$ is the AOA vector $(.)^T$ is the transpose operator and,

$$\mathbf{e}(\boldsymbol{\tau}_e, t) = [e_1(t\ -\ \tau_1), \cdots, e_M(t - \tau_M)\ ]^T \quad (3)$$

$$A(\theta_e) = [\ \mathrm{a}(\theta_1), \ldots, \mathrm{a}(\theta_M)]^T \quad (4)$$

$$\Gamma_e = diag(\gamma_1, \ldots\ldots,\ \ \gamma_M) \quad (5)$$

According to Equation 2, x(t) will be:

$$x(t) = A(\theta_e) \times \Gamma_e \times \mathrm{s}(\tau_e, t) + n(t) \quad (6)$$

1.4. A Multi-Emitters Geolocation Based on AOA-TOA Estimation

AOA/TOA parameters of multi-emitters are used to separate the location problem into single-emitter location problems. The system configuration is illustrated in Figure 3.

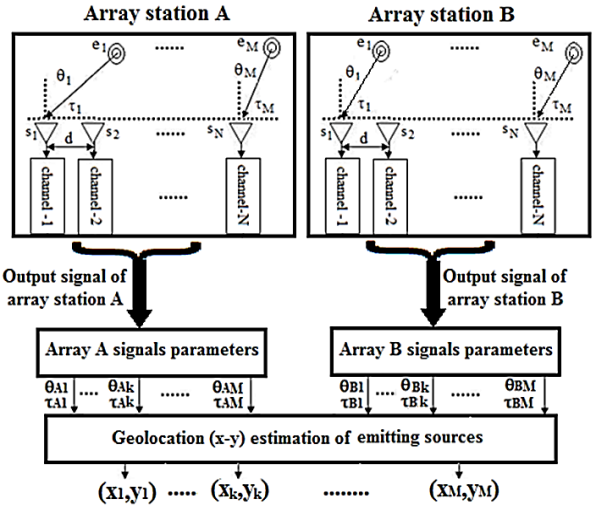


Fig. 3 Multi-emitters geolocation system

Signals $x_A(t)$ and $x_B(t)$ collected by array stations A and B are then equal to:

$$x_A(t) = A(\theta_A) \times \mathrm{s}(\tau_A, t) + n_A(t) \quad (7)$$

The k^{th} components of the system are: θ_A, θ_B, τ_A, and τ_B are θ_{Ak}, θ_{Bk}, τ_{Ak}, and τ_{Bk}, respectively. For example, as shown in Figure 1, M=3. The array of angles received by array station A: θ_A=[θ_{A1}, θ_{A2},θ_{A3}], and that received by B: [θ_{B1},θ_{B2},θ_{B3}]. The measured TOA arrays are: τ_A = [τ_{A1},τ_{A2},τ_{A3}], and τ_B = [τ_{B1},τ_{B2},τ_{B3}] for array stations A, and B respectively.

1.5. AOA Estimation Using the Multiple Signal Classification MUSIC Algorithm

The angles θ are estimated with a subspace method and give estimated values $\tilde{\theta}_A$ and $\tilde{\theta}_B$. Many algorithms are used to estimate the AOA of the received signals. The MUSIC algorithm is a well-known high-resolution algorithm used to estimate the AOA of multiple emitters

and classify the multiple received signals [9-10]. MUSIC stands for Multiple Signal Classification algorithm. It has a low biasing error or ambiguity confusion. It is commonly applied to narrowband signals.

1.6. Implementation of the MUSIC Algorithm

In the first step, the estimation of the input covariance matrix is based on N received signal vectors, as shown in the equation below.

$$R_x = E[x(t)x^H(t)] \tag{8}$$

where R_x is the data covariance matrix, E denotes the expected value, and x^H is the complex-conjugated transpose. A general vector representation of signal x using Equ.6 or 7 is:

$$X = AS+N \tag{9}$$

Substituting (9) in (8):

$$\begin{aligned} R_x &= E[XX^H] \\ &= E[(AS+N)(AS+N)^H] \\ &= AE[SS^H]A^H + E[NN^H] \\ R_x &= AR_sA^H + RN \end{aligned} \tag{10}$$

where Rs=E[SSH] is the signal correlation matrix, R_N= □□I is the noise correlation matrix with noise power=□□□, and I is a unit matrix of size M*M. In practical applications, $\tilde{R}_x$ is estimated to be:

$$\tilde{R}_x = \frac{1}{n}\sum_{i=1}^{n} x(i)x^H(i) \tag{11}$$

The estimation error tends to zero as n $\rightarrow\infty$. Characterized by an array of covariance eigendecompositions, the matrix Rx's eigenvalues are sorted according to the size, which is $\lambda_1 \geq \lambda_2 \geq \ldots.. \lambda_M > 0$, where larger eigenvalues M correspond to the signal while N-M smaller eigenvalues are corresponding to noise.
Get the noise matrix E_n:

$$E_n=[V_{N+1}, V_{N+2}, \ldots., V_M] \tag{12}$$

where V_{N+1}, V_{N+2}, …., V_M are the eigenvalues corresponding to the noise subspace. Calculating the value of the spectrum function $P(\theta)$ in Equation 13. Peaks will be found at the direction of arrival of the source signal (θ).

$$P(\theta) = \frac{1}{a^H(\theta)E_nE_n^Ha(\theta)} \tag{13}$$

The flow chart illustrates the Music algorithm in Figure. 4. As an example, Figure 5 shows the output of the MUSIC algorithm of three incident signals at angles [50, 80, and 140] degrees. The number of array elements N=10, and the number of sources is M=3.

1.7. Emitting Source Separation

The emitter signals s(τ_A, t) of the array A are estimated from the angles $\tilde{\theta}_A$as [11]:

$$\tilde{s}(t) = \mathbf{A}^{\#}\left(\tilde{\theta}_A\right) \times \mathbf{x_A}\,(t) = \begin{bmatrix} \tilde{s}_1(t) \\ \vdots \\ \tilde{s}_k(t) \\ \vdots \\ \tilde{s}_M(t) \end{bmatrix} \tag{14}$$

where $A^{\#}(\theta) = (A^H(\theta)A(\theta))^{-1}A^H(\theta))$, $(.)^H$ is the transpose conjugate and $\tilde{\theta}_A = [\tilde{\theta}_{A1}, \ldots, \tilde{\theta}_{Ak}]^T$. The signal $\tilde{s}(t)$ stands for the estimation of $\Gamma_A s(\tau_A, t)$. Now the signal $\tilde{s}_k(t)$ is associated with the k^{th} components $\tilde{\theta}_{Ak}$ of the vector $\tilde{\theta}_A$. For instance, in Figure 1, $M = 3$ emitters, and the signal $\tilde{s}_k(t)$ is associated with one of the three emitters, e_1, e_2, or e_3. The flow chart shown in Figure 6 describes the emitter signal identification using the emitter AOA (θ) assuming three emitting sources.

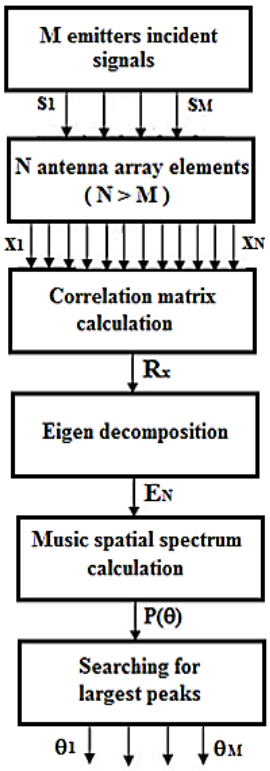


Fig. 4 Steps of MUSIC algorithm

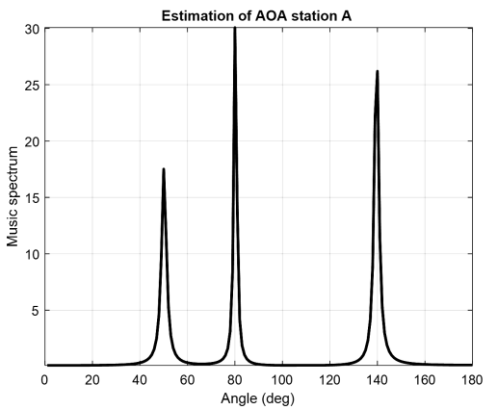


Fig. 5 Angles of the arrival of incident source signals (N=10, M=3)

1.8. TDOA Estimation

The antenna array output signals of stations A and B are cross-correlated in the time domain to get the time difference of arrival of source signal w.r.t. station A and B. For example, if the correlator output is as shown in Figure 7, then the TDOA of sources signals are [219, 290, 346] nanoseconds.

1.9. Geolocation Estimation

The problem of multi-emitters is divided into problems of individual emitters. Knowing the emitting source data, the AOA of the source signal at station A, $\theta_{A,}$ and station B, $\theta_{B,}$ and the TDOA of each emitter signal, the location is estimated using the pre-described algorithm [2].

1.10. Results Analysis of Analytical Method

A novel of the steps of a mathematical method to solve the multi-emitter locating problem is presented. The prescribed analytical method feeds the geolocation algorithm with the AOA and TDOA necessary to estimate each emitter location. In the present work, instead of using a single antenna senor, ULA is used to make the processing of the mixed received signal possible. Two array stations are used to detect the TDOA of each emitter source and hence the signal time delay between stations. The emitting sources example shows that the Geolocation process between many emitting sources is now in hand.

2. Machine Learning Application

We will apply supervised machine learning to simplify the process of multi-emitter Geolocation [11-12]. Once the emitter sensors' signals are associated with each other, the signal parameters are collected, and the geolocation algorithm is directly applied. Sousa and Thomä [19] used Machine Learning and fingerprinting to enhance the localization process. Papageorgiou and Sellathurai [14] tried to fasten the direction of arrival estimation of multiple-target using deep learning. The previously mentioned methods discuss special cases of studies. The present work presents a general yet simple solution to multi-emitter sources' geolocation problem.

The Matlab machine learning classification learner Toolkit 2019b [15] is used to classify and associate the emitter's sensor signals and feed the resulting associated emitter sensor pair to the geolocation estimation algorithm. The system is assumed to consist of two sensors, s1, and s2, that are used to detect three emitters, e1, e2, and e3. The output of sensor1 is the collection of s1e1, s1e2, and s1e3, and the output of sensor2 is the collection of s2e1, s2e2, and s3e3. We need to classify the sensor-emitter pairs to start the geolocation process. The supervised machine learning process consists of two steps; the training step utilizes the output of the multi-emitters geolocation system described above. The trainer will classify the data of the emitters collected by sensor1, s1e1, s1e2, and s1e3, and the data collected by s2, s2e1, s2e2, and s2e3. Once the data are classified, the location could be estimated directly using the machine regression models explained by the authors [16].

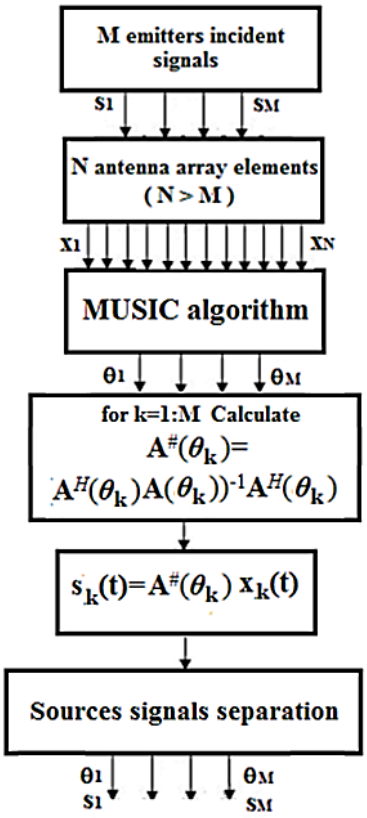


Fig. 6 Sources signals separation.

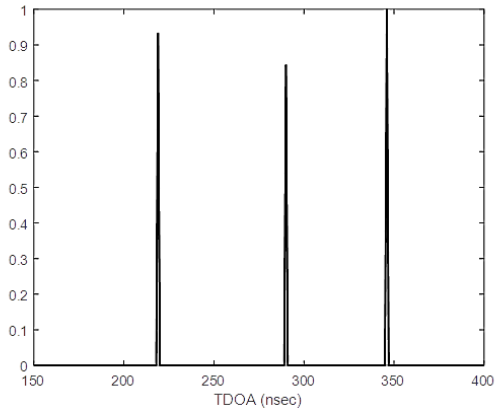


Fig. 7 TDOA between sources signals of array stations A and B.

The classification model is now generated, and the resulting model is now tested for new unlabeled data. The toolbox classification algorithms are tested to generate the most accurate classification model. The flow chart shown in Figure 8 illustrates the process.

2.1. Importing Data

The classifier input data is a table containing the sensor locations, emitter measured data, AOAs azimuth and elevation, and TOA for each sensor. Figure 9 describes the input data to the classifier. A data set of about1206 observations is used to train the system. The data is fed in a table containing the data of the three emitters collected by the two sensors. After importing the data, the classification learner extracts the workspace predictors and output response. A new classification session is now ready to

start; see Figure 10. Pushing the start button, the classification session starts.

2.2. Classification Training Session

Figure 11 shows the classification learner algorithms used to train the data and generate the classification model. The Toolkit classification learner's library has many applicable classification algorithms. After selecting an algorithm to train the data, the train green button is pushed to start the training session and generate the classification model. Trying most of the ToolBox classification algorithms, it is found from the output results that the two algorithms, Fine decision tree, and KNN algorithms, achieve better results, as shown in Figure 12. The next subsection introduces the decision trees and the KNN methods used for generating the classification models that achieve better results among the ToolBox classification algorithms.

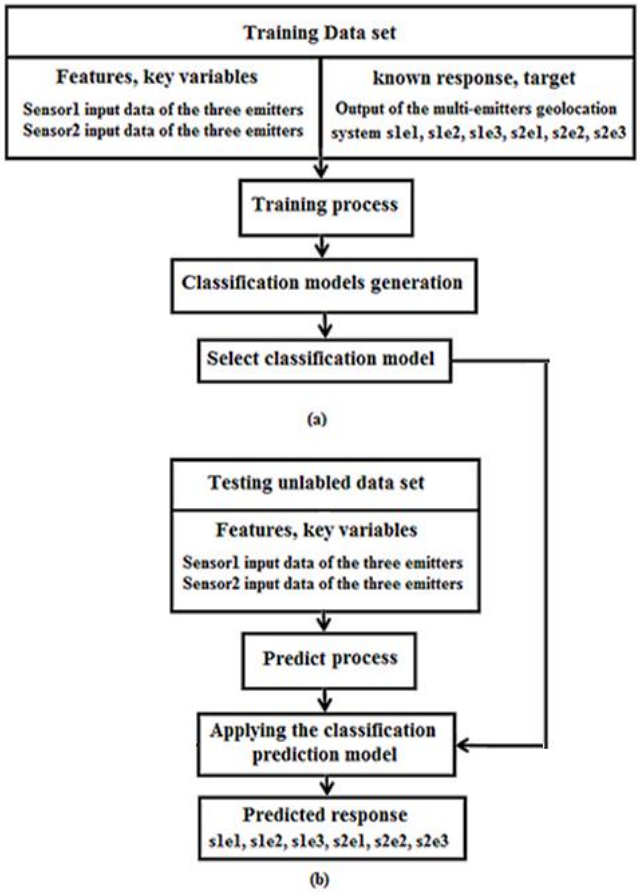


Fig. 8 Classification training (a) and classification testing (b) processes.

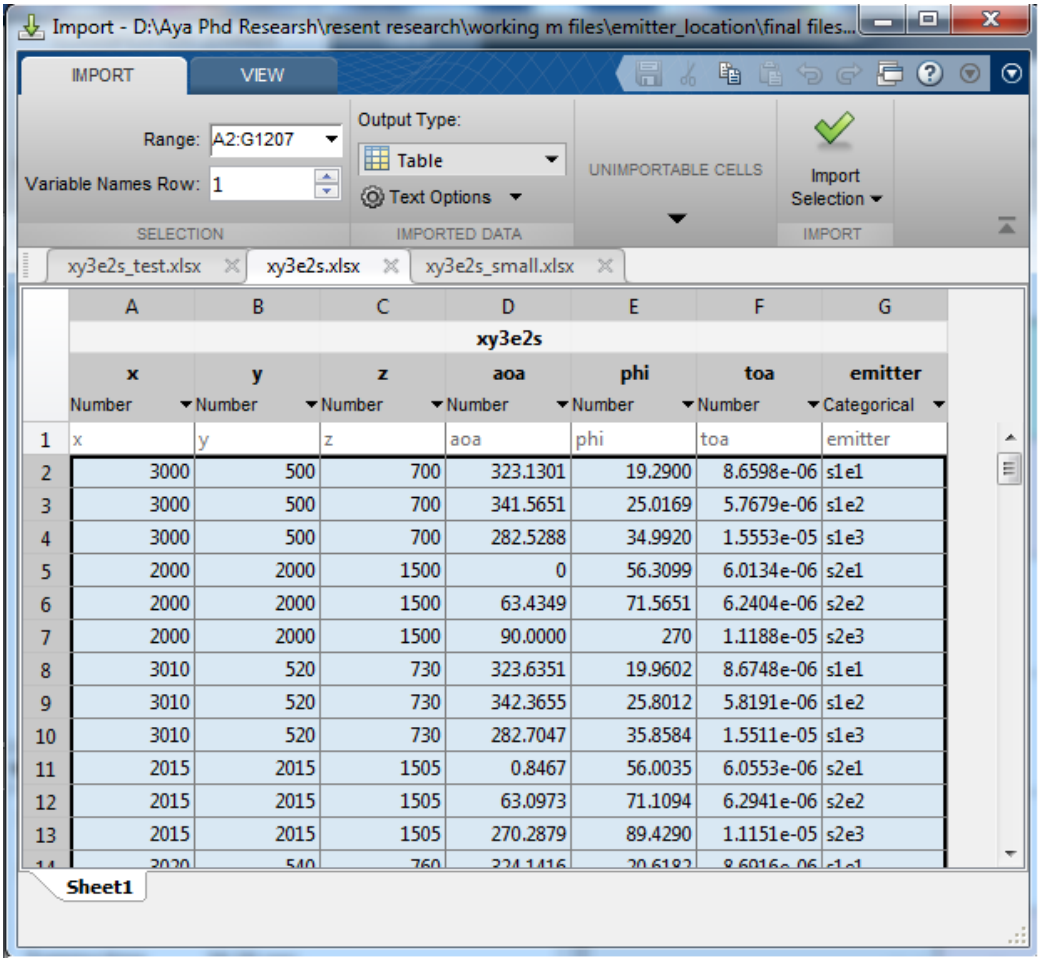
2.3. Decision Trees and KNN Algorithms

Now we will introduce the two best algorithms for this application.

2.3.1. Decision Trees

Decision Trees are important types of algorithms for predictive modeling and machine learning [20]. The classical decision tree algorithms have been around for decades. The decision tree algorithm is called CART, which stands for Classification and Regression Trees. Decision Tree Analysis is a general predictive modeling tool with several applications in different areas. Decision trees are generally constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised machine learning. Decision Trees are a non-parametric supervised learning method for classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules collected from the data features. The decision rules are generally in the form of if-then-else statements.

Decision trees are easy to interpret, fast for fitting and prediction, and low on memory usage, but they can have low predictive accuracy. Try to grow simpler trees to prevent overfitting. Control depth with the maximum number of split settings. Classification and Regression Trees can be used for classification or regression predictive modeling problems. You train trees to predict responses to data. To predict a response, follow the decisions in the tree from the root (beginning) node down to a leaf node. The leaf node contains the response. Statistics and Machine Learning Toolbox trees are binary. Each step in a prediction involves checking the value of one predictor (variable). Figure 13 illustrates the classification tree of the Fine Tree algorithm.



	x	y	z	aoa	phi	toa	emitter
1	x	y	z	aoa	phi	toa	emitter
2	3000	500	700	323.1301	19.2900	8.6598e-06	s1e1
3	3000	500	700	341.5651	25.0169	5.7679e-06	s1e2
4	3000	500	700	282.5288	34.9920	1.5553e-05	s1e3
5	2000	2000	1500	0	56.3099	6.0134e-06	s2e1
6	2000	2000	1500	63.4349	71.5651	6.2404e-06	s2e2
7	2000	2000	1500	90.0000	270	1.1188e-05	s2e3
8	3010	520	730	323.6351	19.9602	8.6748e-06	s1e1
9	3010	520	730	342.3655	25.8012	5.8191e-06	s1e2
10	3010	520	730	282.7047	35.8584	1.5511e-05	s1e3
11	2015	2015	1505	0.8467	56.0035	6.0553e-06	s2e1
12	2015	2015	1505	63.0973	71.1094	6.2941e-06	s2e2
13	2015	2015	1505	270.2879	89.4290	1.1151e-05	s2e3

Fig. 9 Input data of the sensors and collected emitter parameters.

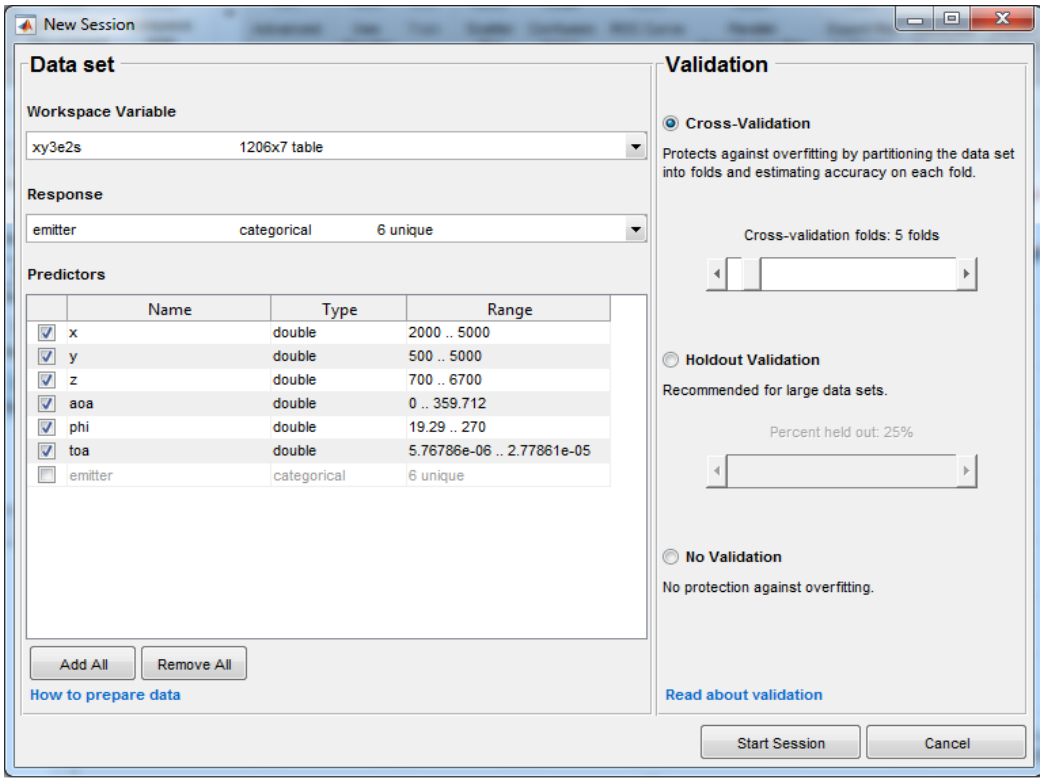


Fig. 10 Data set predictors and response

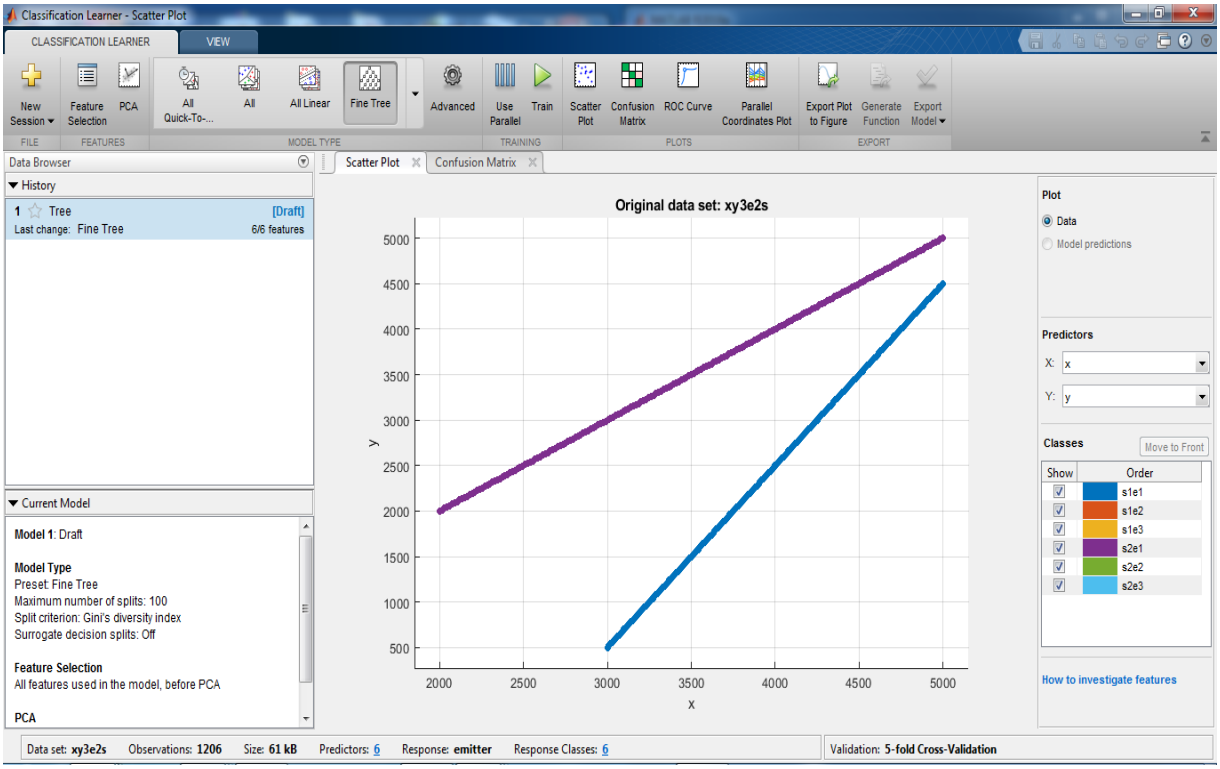


Fig. 11 Selecting a classification training algorithm

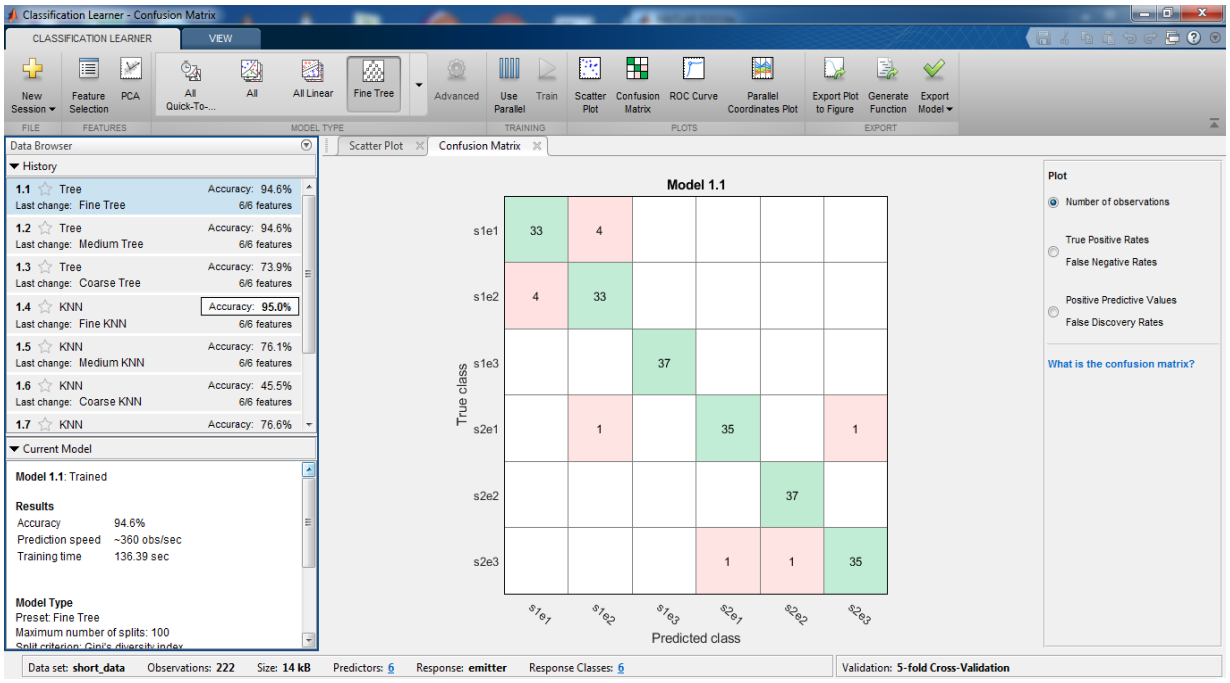


Fig. 12 Classification algorithms training results

2.3.2. K-Nearest Neighbors Algorithm

The KNN algorithm is used in the classification and regression of many applications in supervised machine learning **[18]**. The key idea behind its machine learning applications is that points tend to share the properties of nearby points (the distance function from one point to another often depends on the context; some common ones include Euclidean distance between particles in space, Hamming distance between words, etc.). In a classification setting, a majority vote on the labels of the KNN is often used to determine the label of a point. In a regression setting (where regression is a machine learning technique commonly used to obtain continuous outputs as opposed to discrete outputs in classification), an average (or maximum or minimum) of the KNN is typically used to determine the value of the variable being regressed. Due to its wide applicability and simplicity, KNN is simple and commonly used in many machine learning applications.

There are many approaches to finding the KNN of a given point, usually known as a query point. Assuming a suitable distance function, one approach is to iterate through all points and compute distances from the query point. The nearest k distances can then be selected. However, each query takes a linear time in the dataset size. The k-nearest neighbor (KNN) algorithm is a supervised ML algorithm that can be used for both classifications and regression predictive problems. However, it is mainly used for classifying predictive problems in the industry. The following two properties would define KNN well:

- Lazy learning algorithm − KNN is a lazy learning algorithm because it does not have a specialized training phase and uses all data for training classification.
- Non-parametric learning algorithm
- KNN is also a non-parametric learning algorithm because it does not assume anything about the underlying data.

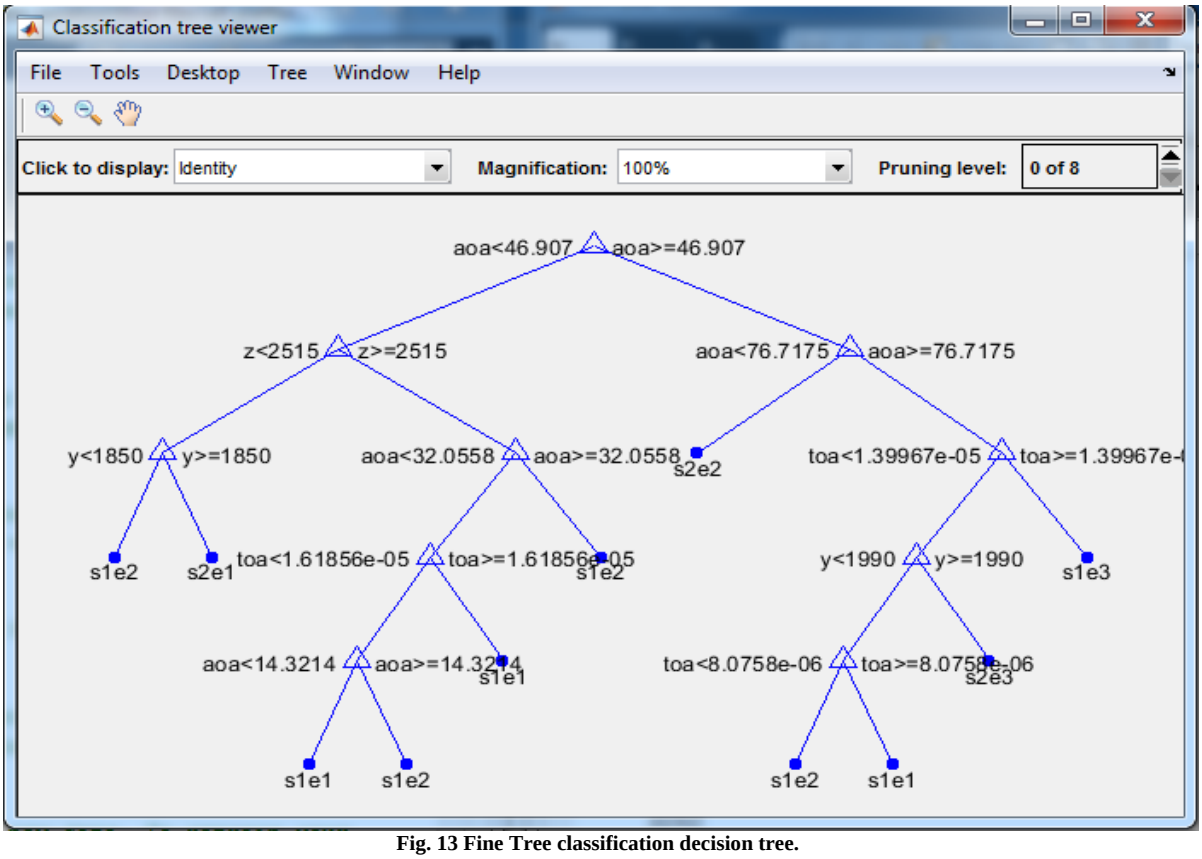


Fig. 13 Fine Tree classification decision tree.

2.3.3. Implementing KNN Algorithm

K-nearest neighbors (KNN) algorithm uses 'feature similarity' to predict the values of new data points, which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. We can understand its working with the help of the following steps:

- Feeding algorithm with both training and testing data sets.
- Compute the Euclidean distances for each test data using Equation 15:

$$\text{Euclidean distance} = (\text{testing data} - \text{training data})^2 \quad (15)$$

- Sort the calculated distances (ascending).
- For each distance, detect its position.
- For a specified test observation, select the nearest neighbor positions according to the predefined number k.
- For those positions, detect the responses of the training observations.
- The frequently repeated response is selected to be the response of the test observation.

Figure 14 describes the KNN example of classifying tested data (black) among the training data (red and blue).

2.4. Testing the Classification Model

Testing the models is done using two sets of data. As part of the training data, random observations are selected and tested. The output results accuracy reached about 96%, as shown in the confusion matrices in Figure 15 of the two algorithms. After testing the selected classification models using known data set, a new data set is now tested for convenience. The new observations are fed to the two classification models, and the output results are shown in Figure 16. The accuracy reaches about 87%, which is considered an acceptable result.

2.5. Machine Learning Results Analysis

Mathematical methods used to estimate multi-emitter Geolocation suffer from many difficulties like time and calculation overheads. It is proved that machine learning is

used to classify the emitter's sensors directly and feed the data directly to the simple geolocation algorithm to draw the emitter's positions. The emitter's sensors platforms severely affect the geolocation process in mathematically designed systems. Using ML, it is sufficient to know the data of the received signal at each sensor.

Classification models can do that task. The training accuracy of a set of data of random samples used in the training model accuracy reaches 96%. Testing the model with a new data set, the accuracy reaches 87%, which is considered an acceptable result. The above ML application results prove that the machine learning system can deal efficiently with the multi-emitter geolocation problem.

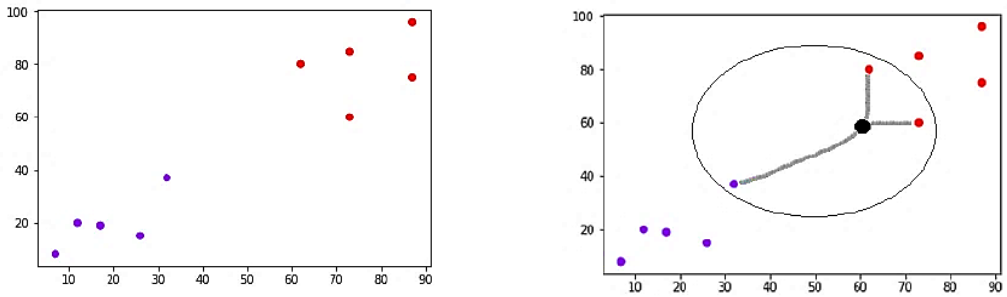


Fig. 14 KNN algorithm principle (k=3).

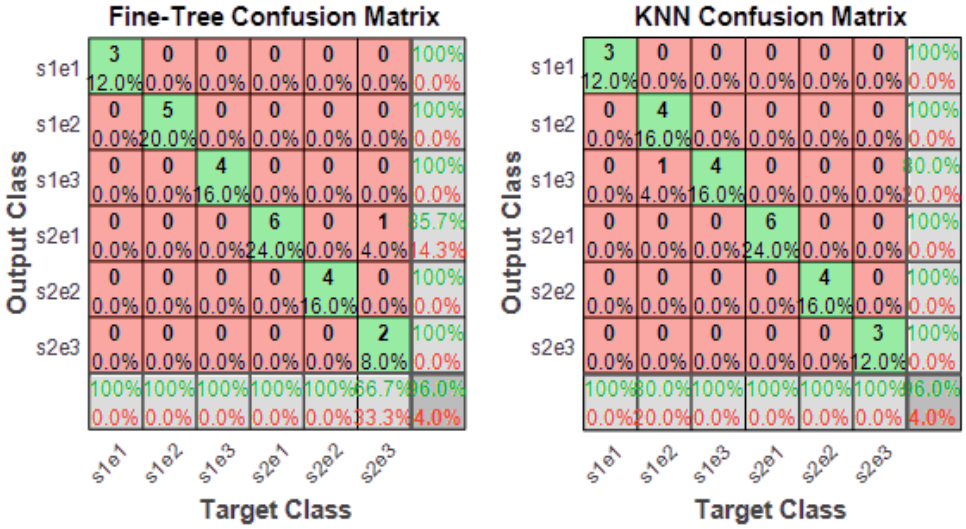


Fig. 15 Confusion matrices classifying known data

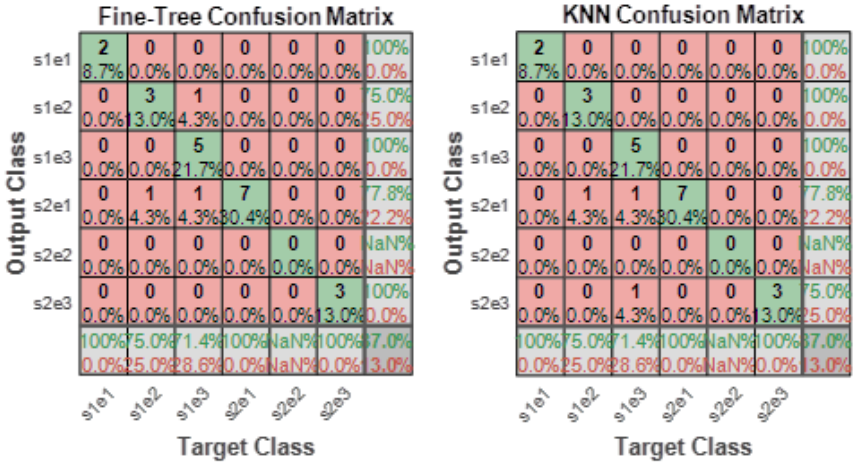


Fig. 16 Confusion matrices classifying new data

References

[1] Alon Amar, and Anthony J. Weiss, "Fundamental Limitations on the Number of Resolvable Emitters Using a Geolocation System," *IEEE Transactions on Signal Processing,* vol. 55, no. 5, pp. 2193, 2007.

[2] Kamel H. Rahouma and Aya S. A. Mostafa, "Analysis and Simulation of RF Source Location Estimation Using Hybrid AOA and TOA Technique," *International Journal of Advanced Science and Technology,* vol. 29, no. 3, pp. 972-985, 2020.

[3] Anne Ferréol, Jonathan Bosse, and Pascal Larzabal, "A Low-Cost AOA-TDOA Approach for Blind Geolocation in Multi-Paths Context," *8th European Signal Processing Conference*, Denmark, 2010.

[4] K. Bryan and D. J. Walter, "Geolocation of Multiple Noncooperative Emitters Using Received Signal Strength: Sparsity, Resolution, and Detectability," *in IEEE Access*, vol. 8, pp. 121999-122012, 2020.

[5] Hernandez M, "Novel Maximum Likelihood Approach for Passive Detection and Localisation of Multiple Emiters," *EURASIP Journal on Advances in Signal Processing,* vol. 36, 2017.

[6] C. Wu and J. Elangage, "Multiemitter Two-Dimensional Angle-of-Arrival Estimator via Compressive Sensing," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 56, no. 4, pp. 2884-2895, 2020.

[7] M. Viberg and B. Ottersten, "Sensor Array Processing Based on Subspace Fitting," *IEEE Transactions on Signal Processing,* vol. 39, no. 5, pp. 1110–1121, 1991.

[8] Ralpho 0. Schmidt, "Multiple Emitter Location and Signal Parameter Estimation," *IEEE Transactions on Antennas and Propagation,* vol. AP-34, no. 3, pp 276-280, 1986.

[9] Mushtaq Ahmad, and Xiaofei Zhang, "Performance of Music Algorithm for DOA Estimation," *ICAYS Engineering Mechanics and Interdisciplinary,* China, 2016.

[10] Mahmoud Mohanna, Mohamed L. Rabeh, Emad M. Zieur, Sherif Hekala, "Optimization of MUSIC Algorithm for Angle of Arrival Estimation in Wireless Communications," *NRIAG Journal of Astronomy and Geophysics*, vol. 2, no. 1, pp. 116-124, 2013.

[11] Sio-Iong Ao, Burghard Rieger, and Mahyar A. Amouzegar, "Machine Learning and Systems Engineering," *Springer*, 2010.

[12] Kamel H. Rahouma and Aya S. A. Mostafa, "Location Estimation of RF Emitting Source Using Supervised Machine Learning Technique," *Advances in Intelligent Systems and Computing*, *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics*, *Springer,* vol. 1261, 2020.

[13] Abha Tiwari, Manish Kumar, "A Review of Range Based Localization Techniques in Wireless Sensor Networks," *SSRG International Journal of Electronics and Communication Engineering,* vol. 8, no. 12, pp. 1-5, 2021. Crossref, https://doi.org/10.14445/23488549/IJECE-V8I12P101.

[14] G. K. Papageorgiou and M. Sellathurai, "Fast Direction-of-arrival Estimation of Multiple Targets Using Deep Learning and Sparse Arrays," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* Barcelona, Spain, pp. 4632-4636, 2020.

[15] MathWorks, Statistics and Machine Learning Toolbox, R2019b.

[16] Kamel H. Rahouma, and Aya S. A. Mostafa, "Location Estimation of RF Emitting Source Using Supervised Machine Learning Technique," *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics, Springer,* pp. 26-37, 2020.

[17] Dinh Tho Long, To Van Binh, Roan Van Hoa, Le Van Anh, Nguyen Van Toan, "Robotic Arm Simulation by using Matlab and Robotics Toolbox for Industry Application," *SSRG International Journal of Electronics and Communication Engineering,* vol. 7, no. 10, pp. 1-4, 2020. Crossref, https://doi.org/10.14445/23488549/IJECE-V7I10P101.

[18] Cheng D., Zhang S., Deng Z., Zhu Y., Zong M, "kNN Algorithm with Data-Driven k Value," *Advanced Data Mining and Applications, ADMA 2014, Lecture Notes in Computer Science, Springer,* vol. 8933, pp. 499-512, 2014.

[19] N. de Sousa, M., and S. Thomä R, "Enhancement of Localization Systems in NLOS Urban Scenario with Multipath Ray Tracing Fingerprints and Machine Learning," *Sensors,* vol. 18, pp. 4073, 2018.

[20] Shucheng Gong and Hongyan Liu, "Constructing Decision Trees for Unstructured Data," *Advanced Data Mining and Applications, ADMA 2014, Lecture Notes in Computer Science, Springer,* vol. 8933, pp. 475-498, 2014.