IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 12 May 2023, accepted 2 June 2023, date of publication 7 June 2023, date of current version 14 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3283592

RESEARCH ARTICLE

Improved Indoor Localization Performance Using a Modified Affinity Propagation Clustering Algorithm With Context Similarity Coefficient

ABDULMALIK SHEHU YARO[1,2], FILIP MALÝ[1], AND KAREL MALÝ[1]

[1]Department of Informatics and Quantitative Methods, Faculty of Informatics and Management, University of Hradec Králové, 50003 Hradec Králové, Czech Republic

[2]Department of Electronics and Telecommunications Engineering, Faculty of Engineering, Ahmadu Bello University, Zaria 810106, Nigeria

Corresponding author: Abdulmalik Shehu Yaro (abdulmalik.yaro@uhk.cz)

This work was supported by the Specifický Výzkum (SPEV) Project 2023 Run at the Faculty of Informatics and Management, University of Hradec Králové, Czech Republic.

ABSTRACT The performance of fingerprint-based indoor wireless localization systems (IWL-Ss) can be enhanced using fingerprint clustering. The localization performance of clustered fingerprint-based IWL-Ss is affected by several factors, including choosing the most optimal initial parameters and the appropriate fingerprint similarity measurement metric. The problem of choosing the best initial parameter is solved by using the affinity propagation clustering (APC) algorithm in this paper, which automatically calculates the number of clusters and cluster centroid vectors. However, the choice of fingerprint similarity measure and the selection of the best cluster centroid when there are multiple potential cluster centroids limit the performance of the APC algorithm. To address this issue, this paper proposes modifying the conventional APC (c-APC) algorithm, which will be referred to as the "m-APC algorithm." The context similarity coefficient (CSC) fingerprint similarity measure replaces the distance-based fingerprint similarity measure used by the c-APC algorithm. Furthermore, the cluster centroids that are generated automatically are replaced by the centroid that is obtained by averaging all fingerprints within a cluster. Using the k-NN localization algorithm and four online fingerprint databases, the performance of the m-APC+CSC algorithm is determined and compared to the c-APC algorithm using cosine, Euclidean, and Shepard distances as fingerprint similarity measures. Based on simulation results, the m-APC algorithm reduced the position root mean square error (RMSE) and mean absolute error (MAE) by about 12% and 8%, respectively, when compared to the c-APC algorithm when both used the CSC as a fingerprint similarity measure. Furthermore, the m-APC+CSC algorithm achieved an 8% and 9%, respectively, position RMSE and MAE reduction over the c-APC algorithm using cosine, Euclidean, and Shepard distances as similarity measurements. The m-APC+CSC algorithm should, however, be used on a reasonably sized fingerprint database with at least four wireless access points (APs) for better localization performance.

INDEX TERMS APC algorithm, context similarity coefficient, fingerprint, k-NN, position RMSE, RSS.

I. INTRODUCTION

The indoor wireless localization system (IWL-S) uses a position-dependent signal parameter (PDSP) obtained from the received signal transmitted by the IU with a localization algorithm to determine the location of an indoor user (IU) [1], [2]. The IWL-S employs several PDSPs, the most common of which is the received signal strength (RSS) [3], [4]. The fingerprinting and trilateration localization algorithms are the two most commonly used RSS-based localization algorithms, and this paper focuses on the fingerprinting algorithm. This is due to its improved localization accuracy, scalability, and robustness to the multipath effect [1], [3]. Fingerprint-based localization using

The associate editor coordinating the review of this manuscript and approving it for publication was Marco Martalo.

RSS measurement is in two phases, namely the offline and online phases [1], [3], [5]. The offline phase involves the generation of an RSS fingerprint database using RSS measurements collected from spatially deployed wireless access points (APs). The fingerprint database is made up of RSS measurement vectors, also known as fingerprints, that have been mapped to the reference locations (RLs) where they were obtained. During the online phase, an IU is instantaneously located using the instantly acquired fingerprint. This is done by comparing the IU's instantly acquired fingerprint with the fingerprint stored in the database. Algorithms such as Kalman filters, k-nearest neighbours (k-NN), support vector machines (SVM), and the Gaussian mixture model (GMM) have been used to determine the location of an IU in the online phase [1], [3], [5], [6], [7].

The localization performance of the fingerprint-based I-WLS depends on several factors, one of which is the size or density of the fingerprint database. The density of the fingerprint database is a function of the number of RL from which the fingerprints are generated, and the number of APs deployed. The larger the size of the fingerprint, the better the localization accuracy; however, this comes at the expense of localization time. The primary objective of any I-WLS is to accurately and instantly determine the location of an IU given the PDSP measurement vector. To solve this localization time and accuracy trade-off, researchers proposed fingerprint database clustering [8], [9], [10], [11], [12], [13]. Fingerprint database clustering is the process of grouping similar fingerprints in the fingerprint database based on their characteristics or features. This process drastically reduces the localization time; however, the accuracy of the clustering process affects the localization accuracy. There are two kinds of clustering algorithms [14], [15], [16]: location-domain clustering algorithms and signal-domain clustering algorithms. Location domain clustering algorithms group fingerprints based on their physical location or spatial features and are commonly used in applications such as indoor localization and mapping, where fingerprints are collected at various points throughout an indoor environment. Location domain clustering algorithms include k-means, c-means, and density-based spatial clustering (DBSCAN) [11], [13], [17], [18]. Signal-domain clustering algorithms, on the other hand, use signal characteristics or waveform features to cluster data. This method is common in applications such as speech recognition, image recognition, and sensor data analysis, where the data points represent signals or waveforms. Signal-domain clustering algorithm examples are spectral clustering and wavelet clustering. This article will focus on location-domain clustering algorithms. Some location domain clustering algorithms have performance limiting factors ranging from the selection of optimal initial parameters to their computational intensity [9], [19], [20]. For example, the number of clusters to be generated and cluster centroid RSS vectors for the k-means and c-means algorithms must be optimally chosen. So also, the DBSCAN also requires optimal selection of initial parameters but not the number of clusters or cluster centroid RSS vectors. The minimum number of fingerprints in the neighbourhood and the radius of the neighbourhood are the parameters that require optimal initial selection. Furthermore, the DBSCAN is computationally intensive. A clustering algorithm that has overcome all these performance limitations associated with k-means, c-means, and DBSCAN is the affinity propagation clustering (APC) algorithm [13], [21], [22]. It does not require the number of clusters to be specified in advance as it does so automatically and simpler to implement than the DBSCAN [13]; as such, it will be adopted in this paper.

As previously stated, the APC algorithms automatically determine the number of clusters and the centroid RSS vector for each cluster [13], [23], [24]. However, there are instances where a cluster could have more than one potential cluster centroid. If such a thing happens, the algorithm chooses the fingerprint with the highest net responsibility to serve as the centroid. This process is computationally complex and subjective [13], [25] and this will subsequently affect the clustering accuracy of the APC algorithm. To solve this issue and improve clustering accuracy, this paper proposes a modified APC (m-APC) algorithm. The modification made to the conventional APC (c-APC) algorithm has to do with the selection of the centroid RSS vector for each cluster. Regardless of the clustering algorithms used, fingerprint clustering necessitates the use of similarity measurement metrics to determine the similarity between fingerprints [13]. The c-APC algorithm uses the distance similarity metric to cluster fingerprints. The Euclidean distance between fingerprints is calculated, and fingerprint pairs with values close to zero are considered very similar and should be in the same cluster. Cosine and Shepard distance are two other commonly used similarity measure metrics [13]. The distance between fingerprint pairs is insufficient to determine how similar the two fingerprints are. Other considerations include the type of data, data distribution, measurement scale, computational complexity, interpretability, RSS measurement behavior in each fingerprint, and non-linear and linear relationships between fingerprints. As an alternative to the distance-based similarity measure, the context similarity coefficient (CSC) is proposed as a similarity measure metric to further improve the clustering accuracy of the m-APC algorithm. In contrast to the distance-based similarity measure, which relies on closeness, the CSC considers the behavior of each RSS measurement in a fingerprint during clustering [26], [27].

The rest of the paper is structured as follows: Section II contains a brief discussion of fingerprint database clustering using the c-APC algorithm. Section III then presents the proposed m-APC algorithm with CSC-based similarity measure. Section IV presents simulation results and discussion, followed by a conclusion and future work in Section V.

II. FINGERPRINT DATABASE CLUSTERING USING C-APC ALGORITHM

Unlike the k-means and c-means algorithms, the c-APC algorithm is a clustering algorithm that uses a message-passing technique to automatically determine the number of clusters as well as the centroid RSS vector of each cluster given a fingerprint database [23], [24], [25]. The process starts with the creation of a similarity matrix, which is done by calculating the similarity between fingerprint pairs using any of the available similarity measurement metrics, such as Euclidean distance, cosine similarity, and Shepard distance. Following the generation of the similarity matrix, the iterative updating of two matrices, namely the responsibility matrix and the availability matrix, occurs. These two matrices are used to select exemplars for each cluster's centroid RSS vector. The following is a step-by-step implementation of the APC algorithm [21], [25]:

Step 1: Generation and initialization of similarity matrix: The similarity matrix, $S(i, j)$, is first generated using either Euclidean distance, cosine similarity, or Shepard distance as similarity measurement metric. For N number of RLs, the size of $S(i, j)$ is $N \times N$.

Step 2: Generation and initialization of the availability and responsibility matrices: The availability matrix, $A(i, j)$ and responsibility matrix, $R(i, j)$ are also generated and all have the same dimension as $S(i, j)$. At the initial stage, both are set to 0, that is $A(i, j) = 0$ and $R(i, j) = 0$.

Step 3: Iteratively update the $R(i, j)$ and $A(i, j)$ matrices: For each iteration, the $R(i, j)$ and $A(i, j)$ matrices are updated using (1) and (2) respectively as shown below [22].

$$R(i,j) = S(i,j) - max[A(i,k) + S(i,k)],\ k \neq j \tag{1}$$

$$A(i,j) = \min[0, R(i,j) + sum(\max[0, R(k,j)])],\quad k \neq j; k \neq i \tag{2}$$

Step 4: Determine exemplar vector: Calculate the exemplar, $E(i)$ vector using (3) as shown below [22].

$$E(i) = argmax[R(i,k) + A(i,k)],\ \ 1 \leq k \leq N \tag{3}$$

Step 5: Determine the clusters: Once exemplar vectors are determined, fingerprint clusters can be formed. Each fingerprint, i belongs to a cluster with $E(i)$ as the centroid RSS vector.

Step 6: Repeat steps 3 to 5 until convergence: Repeat steps 3 to 5 until the algorithm converges, which is when the $E(i)$ stops changing.

The steps 1–6 mentioned earlier are the basic steps involved in implementing the c-APC algorithm. In the next section, the proposed m-APC algorithm with the CSC-based similarity measure is presented.

III. PROPOSED M-APC ALGORITHM WITH CSC-BASED SIMILARITY MEASURE

The clustering algorithm proposed in this paper is presented in this section of the paper. It is based on a modification of the c-APC algorithm presented in Section II and replaces the distance-based similarity measure matrix with a CSC-based similarity measure matrix. The CSC similarity-based matrix generation process is introduced first. This is followed by the proposed c-APC algorithm modification.

A. GENERATION OF CSC-BASED SIMILARITY MATRIX

The similarity matrix in the c-APC algorithm is generated using distance-based similarity measure metrics such as the Euclidean distance, cosine, and Shepard distance. However, for large fingerprint datasets, these similarity measure metrics are not computationally efficient. Furthermore, they do not consider the behavior of each RSS measurement in the fingerprints being compared, nor do they capture the non-linear relationship between the fingerprints [26], [27]. When looking for similarities between fingerprints, these factors must be considered. As a result, this study proposes using a CSC-based similarity measure to determine fingerprint similarity. The CSC similarity measure is a pattern-based measure that considers the behavior of each RSS measurement in the fingerprint as well as the linear and non-linear relationship between the fingerprints being compared. Additionally, it is more interpretable, robust to noise, and computationally efficient for sizable datasets [26].

Unlike distance-based methods, which use closeness to determine fingerprint similarity, the CSC uses patterns in the fingerprints to determine similarity. For example, in Figure 1, four RSS vectors obtained at four RLs from three APs are shown.

The RSS vectors 1 and 3 share the same pattern and thus have a high degree of similarity. RSS vectors 2 and 4 have the same RSS pattern as well, making them very similar. The CSC uses a normalized scale of 0 to 1 to determine the degree of similarity between fingerprints. A CSC value of 0 denotes high similarity, whereas a CSC value of 1 denotes dissimilarity. The degree of similarity, that is the CSC value between two fingerprint vectors $\mathbf{r}_i$ and $\mathbf{r}_j$ of size M is determine as follows [26]:

First determine the term-wise total vector, $\mathbf{t}_{ij}$ between vectors $\mathbf{r}_i$ and $\mathbf{r}_j$ using (4) as follows:

$$\mathbf{t}_{ij} = \mathbf{r}_i + \mathbf{r}_j \ \text{ for } i \neq j \tag{4}$$

Determine the probability of the outcome, p of vector $\mathbf{r}_i$ using (5).

$$p = \frac{\sum_{n=1}^{M} \mathbf{r}_i(n)}{\sum_{n=1}^{M} \mathbf{t}_{ij}} \tag{5}$$

Determine the expected value, $< \mathbf{r_i}\ (n) >$ for each RSS in vector $\mathbf{r}_i$ using (6).

$$< \mathbf{r_i}\ (n) >= p \times \mathbf{t}_{ij} \tag{6}$$

Determine the error for each RSS in vector $\mathbf{r}_i$ using (7).

$$error\ (n) = \frac{< \mathbf{r}_i\ (n) > - \mathbf{r}_i\ (n)}{\sqrt{\mathbf{t}_{ij}(n) \times p \times (1-p)}} \tag{7}$$

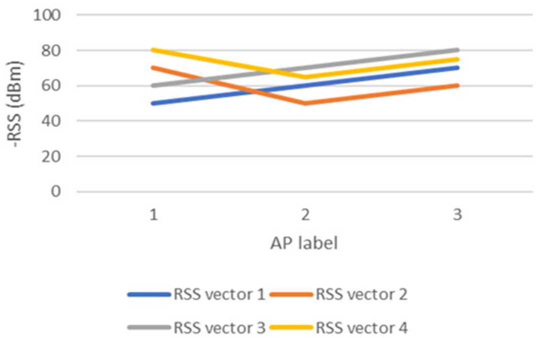

FIGURE 1. **Fingerprint RSS vector pattern for 4 RLs.**

The CSC value between vectors $\mathbf{r}_i$ and $\mathbf{r}_j$ is determined using (8).

$$CSC(r_i, r_j) = \frac{\sum_{n=1}^{M} \left((error(n))^2 \times \sqrt{\mathbf{t}_{ij}(n)}\right)}{\sum_{n}^{M} \sqrt{\mathbf{t}_{ij}(n)}} \quad (8)$$

The CSC value obtained from (8) indicates the degree of similarity between vectors $\mathbf{r}_i$ and $\mathbf{r}_j$. The CSC value of each fingerprint pair in the database is calculated and used to create the CSC-based matrix that will be used with the APC algorithm. Given a fingerprint database with N fingerprint vectors, the CSC-based similarity matrix is obtained as (9).

$$S_{csc}\left(r_i, r_j\right) = \begin{bmatrix} CSC(r_1, r_1) & \cdots & CSC(r_1, r_N) \\ \vdots & \ddots & \vdots \\ CSC(r_N, r_1) & \cdots & CSC(r_N, r_N) \end{bmatrix}$$
$$1 \leq i \leq N; 1 \leq j \leq N \quad (9)$$

where $CSC(r_i, r_j)$ is the CSC value between the i-th and j-th fingerprint vectors obtained using (8).

The diagonal elements of $S_{csc}\left(r_i, r_j\right)$ are replaced with the median of all the elements in $S_{csc}\left(r_i, r_j\right)$ as shown in (10) to eliminate self-similarity and enable the APC algorithm to converge fast.

$$diag\left(S_{csc}\left(r_i, r_j\right)\right) = median\left(S_{csc}\left(r_i, r_j\right)\right) \quad (10)$$

Thus, the final CSC-based similarity matrix for use with the APC algorithm, $S_{csc\_apc}\left(r_i, r_j\right)$ is obtained as (11).

$$S_{csc\_apc}\left(r_i, r_j\right) = \begin{bmatrix} median(S_{csc}\left(r_i, r_j\right)) & \cdots & CSC(r_1, r_N) \\ \vdots & \ddots & \vdots \\ CSC(r_N, r_1) & \cdots & median(S_{csc}\left(r_N, r_N\right)) \end{bmatrix} \quad (11)$$

The CSC-based generated similarity matrix is used instead of the distance-based generated similarity matrix. The modification to the c-APC algorithm is presented in the following subsection.

B. DEVELOPMENT OF THE M-APC ALGORITHM

The c-APC algorithm described in Section II generates the exemplars that represent the cluster centroid RSS vectors automatically. Each cluster may have several exemplars. This makes determining which of the exemplars should be used as the cluster centroid more difficult. Another issue is that a member of one cluster may also be an exemplar of another. The c-APC algorithm is modified to address these issues. The automatically generated exemplar for each cluster is removed and replaced with another RSS vector generated by averaging all fingerprints in each cluster. The step-by-step implementation of the m-APC algorithm is presented as follows:

Step 1: Perform steps 1 through 6 of the c-APC algorithm in Section II.
Step 2: Find the mean RSS vector for each cluster by taking the mean of all fingerprints in that cluster.
Step 3: Use the mean RSS vector obtained in Step 2 as the centroid RSS vector for each cluster.

The m-APC algorithm requires further execution of the c-APC algorithm using the additional steps 2 and 3 as presented in this section of the paper after the c-APC algorithm using steps 1-6 in Section II with (11) as the similarity matrix.

The performance of the m-APC+CSC algorithm is assessed in the following section, and its results are contrasted with those of the c-APC algorithm using similarity metrics such as cosine, Euclidean distance, and Shepard distance.

IV. SIMULATION RESULT AND DISCUSSION

This section determines the performance of the proposed algorithm in Section III. The simulation setup and performance analysis parameters are presented first, followed by a comparison of localization performance.

A. SIMULATION SETUP AND PARAMETER

The localization performance of the algorithms presented in Sections II and III will be evaluated using experimentally generated fingerprint databases found in [28]and [29]. Additionally, both algorithms will also be evaluated using two publicly available databases, INCR_IndoorLoc [30] and MSI_IndoorLoc [31] from the International Conference on Indoor Positioning and Indoor Navigation (IPIN) of 2016 and 2017, respectively. The characteristics of these four databases can be seen in Table 1.

The SEUG_IndoorLoc database was generated using Wi-Fi technology in a small meeting space containing only tables and chairs, so there was no unnecessary interference from other transmitting equipment or the environment [28]. A coverage area of 33 m^2 is provided by the meeting room, which measures approximately 6 m by 5.5 m in size. There are 3 APs in total, and 49 RL fingerprints were collected at 4 m intervals. The IIRC_IndoorLoc database was generated in a room with a coverage area of about 161 m^2 using ZigBee technology [29]. A total of 4 Zigbee nodes were used, and fingerprint measurements were collected at about 194 RLs. Researchers from the Italian National Council of Research (INCR) in Pisa generated the INCR_IndoorLoc database, one of the databases used at the IPIN 2016 International conference [30]. The fingerprint measurements were made along two perpendicular sections of the corridor inside a structure

TABLE 1. Characteristics of the four databases considered for performance analysis.

Database	Characteristics			
	Wireless technology	No. AP	Coverage area	Number of RLs
SEUG_IndoorLoc [28]	Wi-Fi	3	33 m^2	49
IIRC_IndoorLoc [29]	Zigbee	4	161.12 m^2	194
INCR_IndoorLoc [30]	Wi-Fi	127	1260 m^2	325
MSI_IndoorLoc [31]	Wi-Fi	11	1000 m^2	4973

that was 36 m by 36 m in size. The database contains fingerprint measurements obtained from 325 RLs using a total of 127 Wi-Fi APs. Another database used at the 2017 IPIN international conference is MSI_IndoorLoc. It was developed as part of a research initiative to create an indoor positioning system for automated industrial vehicles [31]. The fingerprint measurements were taken using 11 APs over a total area of 1000 m^2 at a university building that resembles an industrial floor plant. The database contained 4973 RLs' fingerprint measurements.

For the localization algorithm, the k-NN algorithm is considered with k = 3. The performance comparison of all clustering algorithms is carried out using a computer with the following characteristics: an Intel (R) Core (TM) i5-2400 CPU at 3.10 GHz, 12 GB of RAM, the Windows 10 operating system, and MATLAB R2020a.

B. LOCALIZATION PERFORMANCE COMPARISON

The localization performance of the m-APC+CSC algorithm is determined in this section of the paper. This is first compared to the c-APC algorithm using CSC as a similarity measure (termed the "c-APC+CSC algorithm"). Second, using cosine, Euclidean distance, and Shepard distance as similarity measurements, the m-APC+CSC algorithm is compared to the c-APC algorithm. For the comparisons, position mean absolute error (MAE) and position root mean square error (RMSE) will be used as the localization performance metrics [32], [33], [34].

1) COMPARISON BETWEEN M-APC+CSC AND C-APC+CSC ALGORITHMS

This subsection employs CSC as a fingerprint similarity measure to assess the enhancement in localization performance brought about by the modification to the c-APC algorithm. The number of clusters, sizes of clusters, and members per cluster are identical for m-APC and c-APC. The cluster centroid represents the only difference. The position RMSE and MAE obtained using the m-APC+CSC algorithm are obtained and compared with the c-APC+CSC algorithm using the 4 fingerprint databases with characteristics shown in Table 1 and k-NN as the localization algorithm. With graphic representations shown in Figures 2 and 3, respectively, Tables 2 and 3 compare the RMSE and MAE of the two algorithms.

TABLE 2. Position RMSE error comparison between m-APC+CSC and c-APC+CSC algorithms.

Database	Position RMSE (m)	
	c-APC+CSC	m-APC+CSC
SEUG_IndoorLoc	1.30	1.02
IIRC_IndoorLoc	2.15	1.95
INCR_IndoorLoc	1.37	1.30
MSI_IndoorLoc	2.50	2.17

In all four fingerprint databases, the m-APC+CSC algorithm outperforms the c-APC+CSC algorithm in terms of position RMSE and MAE, as shown in Tables 1 and 2. The position RMSE obtained by the m-APC+CSC and c-APC+CSC algorithms, for example, are 1.02 m and 1.30 m, respectively, representing a 21% reduction, while the position MAEs are 1.25 m and 1.30 m, respectively, representing a 4% reduction, using the SEUG_IndoorLoc database. For the IIRC_IndoorLoc, INCR_IndoorLoc, and MSI_IndoorLoc databases, respectively, the percentage reduction in position RMSE achieved by the m-APC+CSC algorithm over the c-APC+CSC algorithm is 10%, 5%, and 15%. Additionally, for the IIRC_IndoorLoc, INCR_IndoorLoc, and MSI_IndoorLoc databases, respectively, the percentage reduction in position MAE achieved by the m-APC+CSC algorithm over the c-APC+CSC algorithm is 9%, 4%, and 15%.

When considering all four databases, the m-APC+CSC algorithm demonstrated an average percentage reduction of 12% and 8% in position RMSE and MAE compared to the c-APC+CSC algorithm, respectively. This conclusion is supported by the position RMSE and MAE results presented in Tables 2 and 3. These findings indicate that the modification to the cluster centroid assignment process in the c-APC algorithm, which resulted in the m-APC algorithm, was responsible for the observed reductions of 12% and 8% in position RMSE and MAE, respectively.

2) M-APC+CSC ALGORITHM VS C-APC ALGORITHM WITH COMMONLY USED SIMILARITY MEASURE

The previous subsection compared the performance of the m-APC and c-APC algorithms using CSC as a similar measure for both algorithms. The c-APC algorithm modification reduced position RMSE and MAE by 12% and 8%, respectively. The m-APC+CSC algorithm's localization performance is compared to the c-APC algorithm in this section using cosine, Euclidean, and Shepard similarity measurements. Tables 4 and 5 compare the m-APC+CSC algorithm's position RMSE and MAE to the c-APC algorithm using cosine, Euclidean, and Shepard distances as similarity measurements for all fingerprint databases in Table 1, with graphical representations in Figures 4 and 5.

The results from Table 4 indicate that the m-APC+CSC algorithm generally outperforms the three c-APC algorithms in terms of reduced position RMSE for all four databases. The position RMSE result for the SEUG_IndoorLoc dataset

TABLE 3. Position MAE error comparison between m-APC+CSC and c-APC+CSC algorithms.

Database	Position MAE (m)	
	c-APC+CSC	m-APC+CSC
SEUG_IndoorLoc	1.30	1.25
IIRC_IndoorLoc	2.41	2.19
INCR_IndoorLoc	1.67	1.60
MSI_IndoorLoc	2.93	2.51

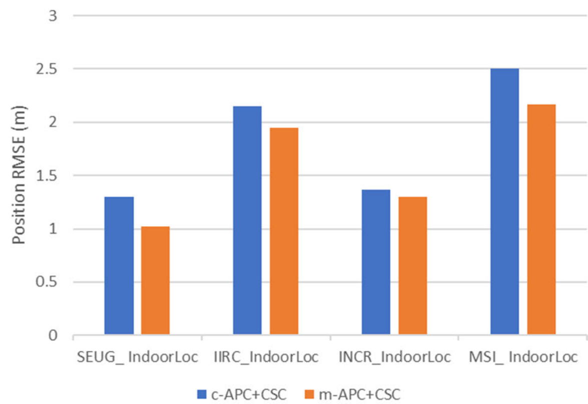

FIGURE 2. Position RMSE comparison between m-APC+CSC and c-APC+CSC algorithms.

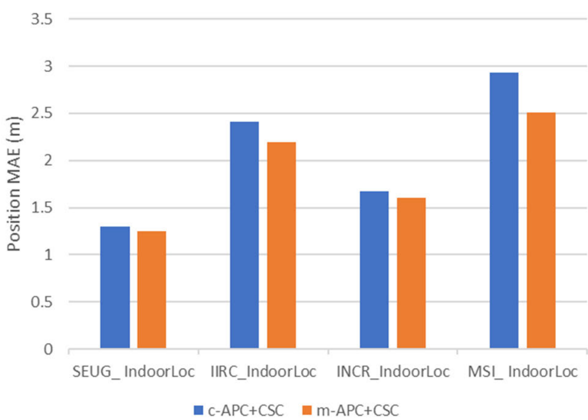

FIGURE 3. Position MAE comparison between m-APC+CSC and c-APC+CSC algorithms.

shows that all four algorithms perform similarly, with position RMSE values ranging from 1.01 m to 1.02 m. This implies that all four algorithms have comparable localization performance. However, there is a noticeable difference in the IIRC_IndoorLoc database. The position RMSE for the m-APC+CSC algorithm is 1.95 m, while for the c-APC+cosine, c-APC+Euclidean, and c-APC+Shepard algorithms, it ranges from 3.00 m to 3.35 m. This shows that the m-APC+CSC algorithm outperforms the other algorithms with an average reduction of about 38% in position RMSE. The performance variations between the algorithms are less pronounced for the INCR_IndoorLoc and MSI_IndoorLoc databases. All four algorithms' position RMSE values are generally within a small range. Even though m-APC+CSC consistently achieves the lowest or similar

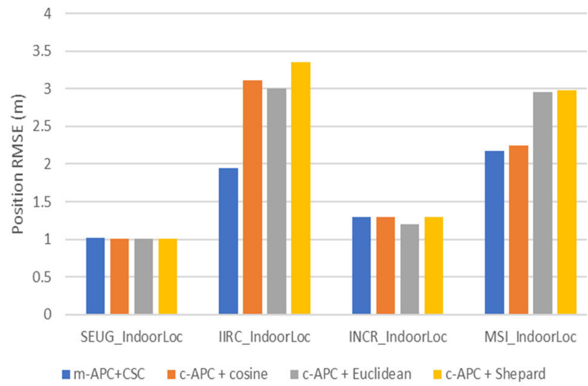

FIGURE 4. Position RMSE comparison between the m-APC+CSC algorithm and the c-APC algorithm using Cosine, Euclidean, and Shepard as fingerprint similarity measure.

position RMSE values, the variations between the algorithms are typically not very significant.

The position RMSE comparison results in Table 4 demonstrate, in summary, that the m-APC+CSC algorithm outperforms other algorithms, particularly on the IIRC_IndoorLoc and MSI_IndoorLoc databases, where it achieves a noticeably lower position RMSE. In total, the m-APC+CSC algorithm reduced position RMSE by 29%, considering these two databases. For the SEUG_IndoorLoc and INCR_IndoorLoc databases, the performance variations between the algorithms in terms of position RMSE are less noticeable.

By extending the analysis to Table 5 for the position MAE, it is possible to come to the same conclusion from the position RMSE result analysis. All algorithms demonstrated comparable position MAE performance for the SEUG_IndoorLoc and INCR_IndoorLoc databases, with MAE values ranging from 1.23 m to 1.25 m for the SEUG_IndoorLoc database and 1.60 m to 1.67 m for the INCR_IndoorLoc database. For the IIRC_IndoorLoc database, the m-APC+CSC algorithm obtained an MAE of 2.19 m, which is on average 37% lower than the MAE of the other algorithms. The m-APC+CSC algorithm has the lowest position MAE in the MSI_IndoorLoc database, with a value of 2.51 m, while the MAE values for the other algorithms range from 2.57 m to 2.77 m. This results in a position MAE reduction of about 19%.

In conclusion, based on the results of the position RMSE and MAE metrics, it can be inferred that the m-APC+CSC algorithm outperforms the c-APC+cosine, c-APC+Euclidean, and c-APC+Shepard algorithms on the IIRC_IndoorLoc and MSI_IndoorLoc databases. Its performance is slightly comparable with the other algorithms on the SEUG_IndoorLoc and INCR_IndoorLoc databases. The percentage improvements in position RMSE achieved by the m-APC+CSC algorithm over the other algorithms for SEUG_IndoorLoc, IIRC_IndoorLoc, INCR_IndoorLoc, and MSI_IndoorLoc databases are −1%, 35%, −8%, and 4%, respectively. This amounts to an overall percentage reduction of approximately 8% in position RMSE compared to the other algorithms across all the databases. As for the position MAE,

TABLE 4. Position RMSE comparison between the m-APC+CSC algorithm and the c-APC algorithm with cosine, Euclidean, and Shepard as fingerprint similarity measure.

Database	Position RMSE (m)			
	m-APC+CSC	c-APC+cosine	m-APC+CSC	c-APC+Shepard
SEUG_IndoorLoc	1.02	1.01	1.01	1.01
IIRC_IndoorLoc	1.95	3.11	3.00	3.35
INCR_IndoorLoc	1.30	1.30	1.20	1.30
MSI_IndoorLoc	2.17	2.25	2.96	2.98

TABLE 5. Position MAE comparison between the m-APC+CSC algorithm and the c-APC algorithm with cosine, Euclidean, and Shepard as fingerprint similarity measure.

Database	Position MAE (m)			
	m-APC+CSC	c-APC+cosine	c-APC+Euclidean	c-APC+Shepard
SEUG_IndoorLoc	1.25	1.24	1.23	1.23
IIRC_IndoorLoc	2.19	3.49	3.25	3.67
INCR_IndoorLoc	1.60	1.67	1.64	1.67
MSI_IndoorLoc	2.51	2.59	2.76	2.77

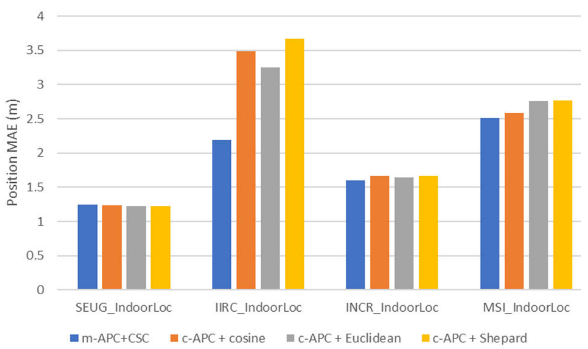

FIGURE 5. Position MAE comparison between the m-APC+CSC algorithm and the c-APC algorithm using Cosine, Euclidean, and Shepard as fingerprint similarity measure.

the percentage of reduction achieved by m-APC+CSC over the other algorithms for SEUG_IndoorLoc, IIRC_IndoorLoc, INCR_IndoorLoc, and MSI_IndoorLoc databases is −0.8%, 33%, 2%, and 3%, respectively. This amounts to an overall percentage reduction of approximately 9% in position MAE compared to the other algorithms across all the databases.

The modifications made to the c-APC algorithm in the m-APC+CSC algorithm, specifically the cluster centroid assignment process and the use of a CSC-based fingerprint similarity measure, have led to a notable improvement in the accuracy of indoor localization across all four databases. The CSC is a pattern-based similarity measure algorithm, and the more APs there are, the more distinctive the fingerprint patterns are, and the more accurately the algorithm can measure the similarity between two fingerprint vectors. The APC algorithm is well known for its extremely poor performance and high computational complexity for large fingerprint database sizes. As a result, the m-APC+CSC algorithm is suggested for use on a moderately sized fingerprint database with at least four APs.

V. CONCLUSION AND FUTURE WORKS

This paper proposes a clustering algorithm based on the modification of the cluster centroid assignment process to the c-APC algorithm and the use of a CSC-based similarity matrix in place of the conventional distance-based similarity measure matrix. Contrary to distance-based similarity, the CSC similarity measure is based on patterns and has a number of benefits. These benefits include the behavior of fingerprints and the non-linear relationship between fingerprint considerations. The localization performance of the proposed m-APC+CSC algorithm is assessed using the k-NN localization algorithm and contrasted with the c-APC algorithm using cosine, Euclidean distance, and Shepard distance. The m-APC+CSC algorithm performed better with fingerprint databases generated with at least four APs. Future work will focus on enhancing the m-APC+CSC algorithm to achieve outstanding performance with larger fingerprint database sizes as well as evaluating the performance of the clustering algorithm against other machine learning localization algorithms.

ACKNOWLEDGMENT

The authors would like to thank the management of the University of Hradec Králové (UHK), Czech Republic, as well as the Faculty of Informatics and Management (FIM), for providing the resources and support for this study. The technical support provided by Ing. Kruncik is also gratefully acknowledged.

REFERENCES

[1] A. S. Yaro, F. Maly, and P. Prazak, "A survey of the performance-limiting factors of a 2-Dimensional RSS fingerprinting-based indoor wireless localization system," *Sensors*, vol. 23, no. 5, p. 2545, Feb. 2023, doi: 10.3390/s23052545.

[2] A. S. Yaro, F. Maly, and P. Prazak, "Outlier detection in time-series receive signal strength observation using Z-score method with Sn scale estimator for indoor localization," *Appl. Sci.*, vol. 13, no. 6, p. 3900, Mar. 2023, doi: 10.3390/app13063900.

[3] S. Shang and L. Wang, "Overview of WiFi fingerprinting-based indoor positioning," *IET Commun.*, vol. 16, no. 7, pp. 725–733, Apr. 2022, doi: 10.1049/cmu2.12386.

[4] M. Kolakowski, "Automatic radio map creation in a fingerprinting-based BLE/UWB localisation system," *IET Microw., Antennas Propag.*, vol. 14, no. 14, pp. 1758–1765, Nov. 2020, doi: 10.1049/iet-map.2019.0953.

[5] K. Fronckova and P. Prazak, "Possibilities of using Kalman filters in indoor localization," *Mathematics*, vol. 8, no. 9, p. 1564, Sep. 2020, doi: 10.3390/math8091564.

[6] X. Feng, K. A. Nguyen, and Z. Luo, "A survey of deep learning approaches for WiFi-based indoor positioning," *J. Inf. Telecommun.*, vol. 6, no. 2, pp. 163–216, Apr. 2022, doi: 10.1080/24751839.2021.1975425.

[7] F. Alhomayani and M. H. Mahoor, "Deep learning methods for fingerprint-based indoor positioning: A review," *J. Location Based Services*, vol. 14, no. 3, pp. 129–200, Jul. 2020, doi: 10.1080/17489725.2020.1817582.

[8] S. H. Marakkalage, B. P. L. Lau, Y. Zhou, R. Liu, C. Yuen, W. Q. Yow, and K. H. Chong, "WiFi fingerprint clustering for urban mobility analysis," *IEEE Access*, vol. 9, pp. 69527–69538, 2021, doi: 10.1109/ACCESS.2021.3077583.

[9] D. Quezada-Gaibor, J. Torres-Sospedra, J. Nurmi, Y. Koucheryavy, and J. Huerta, "Lightweight Wi-Fi fingerprinting with a novel RSS clustering algorithm," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Nov. 2021, pp. 1–8, doi: 10.1109/IPIN51156.2021.9662612.

[10] P. Sadhukhan, S. Gain, K. Dahal, S. Chattopadhyay, N. Garain, and X. Wang, "An efficient clustering with robust outlier mitigation for Wi-Fi fingerprint based indoor positioning," *Appl. Soft Comput.*, vol. 109, Sep. 2021, Art. no. 107549, doi: 10.1016/j.asoc.2021.107549.

[11] H. Zhao, "Design and implementation of an improved K-means clustering algorithm," *Mobile Inf. Syst.*, vol. 2022, pp. 1–10, Sep. 2022, doi: 10.1155/2022/6041484.

[12] A. Alitaleshi, H. Jazayeriy, and J. Kazemitabar, "Affinity propagation clustering-aided two-label hierarchical extreme learning machine for Wi-Fi fingerprinting-based indoor positioning," *J. Ambient Intell. Humanized Comput.*, vol. 13, no. 6, pp. 3303–3317, Jun. 2022, doi: 10.1007/s12652-022-03777-1.

[13] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Ann. Data Sci.*, vol. 2, no. 2, pp. 165–193, Jun. 2015, doi: 10.1007/s40745-015-0040-1.

[14] J. Bi, L. Huang, H. Cao, G. Yao, W. Sang, J. Zhen, and Y. Liu, "Improved indoor fingerprinting localization method using clustering algorithm and dynamic compensation," *ISPRS Int. J. Geo-Inf.*, vol. 10, no. 9, p. 613, Sep. 2021, doi: 10.3390/ijgi10090613.

[15] J. Bi, H. Cao, G. Yao, Z. Chen, J. Cao, and X. Gu, "Indoor fingerprint positioning method with standard particle swarm optimization," in *Proc. China Satell. Navigat. Conf. (CSNC)*. Singapore: Springer, 2021, pp. 403–412, doi: 10.1007/978-981-16-3138-2_38.

[16] E. L. Gomes, M. Fonseca, A. E. Lazzaretti, A. Munaretto, and C. Guerber, "Clustering and hierarchical classification for high-precision RFID indoor location systems," *IEEE Sensors J.*, vol. 22, no. 6, pp. 5141–5149, Mar. 2022, doi: 10.1109/JSEN.2021.3103043.

[17] W. Jiang, X. Fang, and J. Ding, "Gaussian kernel fuzzy C-Means algorithm for service resource allocation," *Scientific Program.*, vol. 2020, pp. 1–6, Dec. 2020, doi: 10.1155/2020/8889480.

[18] J. Bi, H. Cao, Y. Wang, G. Zheng, K. Liu, N. Cheng, and M. Zhao, "DBSCAN and TD integrated Wi-Fi positioning algorithm," *Remote Sens.*, vol. 14, no. 2, p. 297, Jan. 2022, doi: 10.3390/rs14020297.

[19] L. Klus, D. Quezada-Gaibor, J. Torres-Sospedra, E. S. Lohan, C. Granell, and J. Nurmi, "RSS fingerprinting dataset size reduction using feature-wise adaptive k-means clustering," in *Proc. 12th Int. Congr. Ultra Modern Telecommun. Control Syst. Workshops (ICUMT)*, Oct. 2020, pp. 195–200, doi: 10.1109/ICUMT51630.2020.9222458.

[20] G. Zou, L. Ma, Z. Zhang, and Y. Mo, "An indoor positioning algorithm using joint information entropy based on WLAN fingerprint," in *Proc. 5th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2014, pp. 1–6, doi: 10.1109/ICCCNT.2014.6963033.

[21] F. Li, M. Liu, Y. Zhang, and W. Shen, "A two-level WiFi fingerprint-based indoor localization method for dangerous area monitoring," *Sensors*, vol. 19, no. 19, p. 4243, Sep. 2019, doi: 10.3390/s19194243.

[22] S. Subedi, H. Gang, N. Y. Ko, S. Hwang, and J. Pyun, "Improving indoor fingerprinting positioning with affinity propagation clustering and weighted centroid fingerprint," *IEEE Access*, vol. 7, pp. 31738–31750, 2019, doi: 10.1109/ACCESS.2019.2902564.

[23] I. E. Givoni and B. J. Frey, "A binary variable model for affinity propagation," *Neural Comput.*, vol. 21, no. 6, pp. 1589–1600, Jun. 2009, doi: 10.1162/neco.2009.05-08-785.

[24] F. Shang, L. C. Jiao, J. Shi, F. Wang, and M. Gong, "Fast affinity propagation clustering: A multilevel approach," *Pattern Recognit.*, vol. 45, no. 1, pp. 474–486, Jan. 2012, doi: 10.1016/j.patcog.2011.04.032.

[25] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007, doi: 10.1126/science.1136800.

[26] A. Kulkarni, V. Tokekar, and P. Kulkarni, "Discovering context of labeled text documents using context similarity coefficient," *Proc. Comput. Sci.*, vol. 49, pp. 118–127, Jan. 2015, doi: 10.1016/j.procs.2015.04.235.

[27] P. Mudgil, P. Gupta, I. Mathur, and N. Joshi, "A novel similarity measure for context-based search engine," in *Proc. Int. Conf. Innov. Comput. Commun.* Singapore: Springer, 2023, pp. 791–808, doi: 10.1007/978-981-19-2535-1_65.

[28] S. Sadowski, P. Spachos, and K. N. Plataniotis, "Memoryless techniques and wireless technologies for indoor localization with the Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10996–11005, Nov. 2020, doi: 10.1109/JIOT.2020.2992651.

[29] T. Alhmiedat, "Fingerprint-based localization approach for WSN using machine learning models," *Appl. Sci.*, vol. 13, no. 5, p. 3037, Feb. 2023, doi: 10.3390/app13053037.

[30] P. Barsocchi, A. Crivello, D. L. Rosa, and F. Palumbo, "A multi-source and multivariate dataset for indoor localization methods based on WLAN and geo-magnetic field fingerprinting," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2016, pp. 1–8, doi: 10.1109/IPIN.2016.7743678.

[31] A. Moreira, I. Silva, F. Meneses, M. J. Nicolau, C. Pendao, and J. Torres-Sospedra, "Multiple simultaneous Wi-Fi measurements in fingerprinting indoor positioning," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2017, pp. 1–8, doi: 10.1109/IPIN.2017.8115914.

[32] J. Bi, M. Zhao, G. Yao, H. Cao, Y. Feng, H. Jiang, and D. Chai, "PSOSVRPos: WiFi indoor positioning using SVR optimized by PSO," *Expert Syst. Appl.*, vol. 222, Jul. 2023, Art. no. 119778, doi: 10.1016/j.eswa.2023.119778.

[33] T. Yang, A. Cabani, and H. Chafouk, "A survey of recent indoor localization scenarios and methodologies," *Sensors*, vol. 21, no. 23, p. 8086, Dec. 2021, doi: 10.3390/s21238086.

[34] F. Potortì et al., "Off-line evaluation of indoor positioning systems in different scenarios: The experiences from IPIN 2020 competition," *IEEE Sensors J.*, vol. 22, no. 6, pp. 5011–5054, Mar. 2022, doi: 10.1109/JSEN.2021.3083149.

ABDULMALIK SHEHU YARO was born in Kaduna, Nigeria, in 1990. He received the B.Eng. degree in electrical engineering from Ahmadu Bello University (ABU), Zaria, and the M.Eng. degree in telecommunications engineering and the Ph.D. degree in electrical engineering from the University of Technology, Malaysia, in 2014 and 2018, respectively. He is currently a Senior Researcher with the Department of Informatics and Quantitative Methods, University of Hradec Králové, Czech Republic, as well as an Assistant Professor with the Department of Electronics and Telecommunications Engineering, ABU. His articles have appeared in over 20 international peer-reviewed journals and conference proceedings. His research interests include wireless localization and sensor networks.

FILIP MALÝ was born in Czech Republic, in 1981. He received the master's degree in information management from the University of Hradec Králové, Czech Republic, in 2005, and the Ph.D. degree in information and knowledge management, in 2009. Since 2014, he has been an Associate Professor of systems engineering and informatics with the University of Hradec Králové. He is currently a Senior Researcher and a Teacher with the Department of Informatics and Quantitative Methods, University of Hradec Králové. His research interests include indoor localization, wireless sensor networks, programming languages, and operating systems.

KAREL MALÝ was born in Czech Republic, in 1973. He received the master's degree in cybernetics and the Ph.D. degree in artificial intelligence and biocybernetics from the Faculty of Electrical Engineering, Czech Technical University in Prague (CTU), Czech Republic, in 1998 and 2005, respectively. Since 2000, he has been an Assistant Professor and a Researcher with the Department of Informatics and Quantitative Methods, University of Hradec Králové, Czech Republic. His research interests include artificial intelligence, robotics, indoor localization, programming languages, and Unix operating systems.