Journal of Computing and Security

Research Article, 2022, Volume 9, Number 2 (pp. 55–64)

http://www.jcomsec.org

Indoor Localization Performance Optimization Using Modified KD-Tree Algorithm

Hossein Ghaffarian [a,*], Seyfollah Soleimani [a], Seyede Habibe Zadsar [a]

[a] *Department of Computer Engineering, Faculty of Engineering, Arak University, Sardasht, Arak, Iran.*

ARTICLE INFO.

Article history:
Received: 6 April 2022
Revised: 3 November 2022
Accepted: 11 November 2022
Published Online: 5 January 2023

Keywords:
Indoor Localization, Fingerprint, Received Signal Strength, KD-Tree.

ABSTRACT

In this paper, we present a new method for improving the efficiency of indoor localization algorithms, in terms of running time and error rate, using the KD-tree data structure. One of the main challenges of indoor localization algorithms in large environments is the high processing overhead of these algorithms due to the high volume of input data and lack of processing resources in users' mobile devices. In the proposed method in this paper, we first cluster the fingerprint database. Then, with the help of a newly proposed method, a modified KD-tree is implemented according to the conditions of the clusters. This tree is a decision-making structure to select one specific cluster where the user stands there. Finally, when a user entered, using a few simple comparisons in the KD-tree, the desired cluster is found and only information about that cluster is passed to the localization algorithm, to compare and predict the user's location. The results of the implementation of this method on the fingerprint data set of the Faculty of Engineering at Arak University show that the proposed method reduces the running times and errors to less than half the values, compared to the time of not using the proposed method.

1 Introduction

One of the most useful technologies in recent years is the technology of locating people, which many applications are designed and worked on based on it. Outdoor localization is done by the Global Positioning System (GPS). However, indoor positioning systems use indoor localization methods; because the signals of GPS cannot penetrate interior spaces [1]. On the other hand, the widespread expansion of smartphones and other mobile wireless devices, along with the growth of business interest in the last few years, has led to the creation of a wide range of services, including indoor localization [2]. Lack of GPS signals, various other signals have been considered for indoor localization, including WiFi, Bluetooth, Zig bee, Broadband, FM radio, Audio, light, magnetic fields, and so on. Among all these, the use of WiFi signals in buildings, due to the pervasive penetration of wireless local area networks and mobile devices with WiFi connectivity, has attracted serious attention. Therefore, the use of WiFi-based location systems is cost-effective due to the lack of need for additional infrastructure investment [3].

During the past decade, several survey papers have been published in the field of indoor localization. The authors of these papers prepare a comprehensive study in this field. Obeidat et al. [4] prepare a high-level

* Corresponding author.
Email addresses: h-ghaffarian@araku.ac.ir (H. Ghaffarian), s-soleimani@araku.ac.ir (S. Soleimani), h.zadsar71@gmail.com (S. H. Zadsar)

https://dx.doi.org/10.22108/JCS.2022.133220.1094
ISSN: 2322-4460

study on different detection technologies in indoor localization algorithms. Those technologies have been compared in terms of accuracy, cost, advantages, and disadvantages. The authors show that although using fingerprinting as a localization technique has some disadvantages such as affecting by multipath effects or needing massive work to create radio maps, it prepares a highly accurate low-cost method. Singh et al. [5] review previously published surveys in indoor localization. Then, they discuss different applications of machine learning techniques in indoor localization. Also, they have a look at performance metrics, public datasets, and open challenges in this field. Roy and Chowdhury [6] prepare a study using machine-learning techniques in indoor localizations to create radio maps and localizations. Another survey is presented in [7]. In addition to reviewing different major indoor localization techniques, the paper prepares a general performance evaluation between them too. A good indoor localization technique must have good accuracy while remaining reliable and stable.

A common localization approach is locating based on the Received Signal Strength (RSS) of Access Points (APs). On a WiFi network, each AP periodically broadcasts control signals. Users detect APs within their range by receiving these signals. Various algorithms such as geometric algorithms or fingerprint methods can be used to determine the user's position in a WiFi network. The fingerprint method provides higher accuracy and is known as an effective approach for localization on a WiFi network [8]. The fingerprint method has two steps. The first step is to collect training data in an offline form. In this case, WiFi fingerprints are collected from all APs in different parts of a building and stored in a database. The second step is localization in an online form. In this case, the WiFi network fingerprints are collected from the user's current location. Then, using localization algorithms, these fingerprints are compared with the stored ones. The results present the user's location.

The structures of buildings have negative effects on signal propagation which reduces localization accuracy. Zhou et al. [9] focus on these problems. They propose a statistical test approach to detect the signal distribution of APs and find APs' contribution degree. Rizk et al. [10] use a combination of RSS Indication (RSSI) and Round Trip Time (RTT) as inputs of a deep neural network strategy to improve the accuracy of localization. They follow this strategy to overcome signal propagation errors inside buildings. Ali Asgari Renani et al [11] propose a step-by-step tracking strategy. They try to predict the movement of users by combining fingerprint localization methods with the Viterbi algorithm. Fan et al. [12] propose a localization method based on Channel State Information (CSI) fingerprints in Multiple-Input Multiple-Output (MIMO) systems. To do this, they develop a machine learning-based hierarchical localization for MIMO fingerprints. However, CSI data is accessible only on a few wireless network cards. Seçkin and Coşkun [13] use data mining techniques for feature selection and classification to increase the accuracy of positioning. Alshamaa et al. [14] propose a zoning strategy for sensors using the belief function theory. They use kernel density estimation, hierarchical clustering, and similarity divergence, to create a two-level hierarchy, to reduce the number of zones to be classified at a time. At each level of the hierarchy, they use feature selection techniques to reduce the misclassification rate and feature redundancy. However, the goal of the authors in this paper is to look for the zone of sensors, not their exact location.

One of the biggest challenges to advancing conventional methods for locating indoors is the high processing overhead of these methods due to the volume of the fingerprint database and the processing overhead of the algorithms themselves. If the localization method is performed on a server, there is no problem with the overheads. However, in this case, the user needs to have access to the network connected to the server, which is not possible always. Also, because of wireless communication problems inside buildings, such communication may affect the reliability and stability of the algorithms. The second solution is to run an algorithm located on the user's device. In this case, due to the limitations of the battery, memory, and processor of users' mobile devices, processing overheads and complexity of the algorithms pose great challenges. Deep neural network solutions, as one of today's popular solutions, may face performance issues in such platforms. Ghaffarian [15] shows that deleting almost half of the data in a data set under certain conditions can significantly reduce processing overheads while sacrificing a small amount of accuracy. Although these results are valuable, it should be noted that in large buildings and spaces, working with half of the data still puts a lot of overhead on the user's device. Therefore, it is necessary to use methods that reduce the amount of input data to localization algorithms beyond these.

In this article, a new method is introduced in which we try to limit the search spaces by dividing the building into different areas. The purpose of the limitation is to overcome the problem of lack of memory and processing power, by reducing the volume of input data to the positioning algorithms. To do this, the positions inserted in the fingerprint file are first divided into different clusters using the k-means algorithm. The clusters' information is then inserted into a modified KD-tree. With the help of the structure of the KD-tree and based on the conditions of different clusters, a search tree is formed, which helps to divide

and reduce the search space. After using the KD-tree and finding the area surrounding the user, we run the classification process only with the data in that particular area. Implementing the proposed method in MATLAB and evaluating it with real data indicates the optimal performance of the proposed method.

The rest of this paper is organized as follows: In Section 2, a brief review of the KD-tree is presented. Section 3 describes the proposed method for minimizing the amount of data entered into the localization algorithm. Section 4 contains the results of the performance evaluation of the proposed method and finally, Section 5 concludes the paper.

2 KD-Tree

Trees are stands among the best and easiest data structures to present data categorizing. Compared to other methods, especially statistical methods, they have a simple and understandable, yet powerful structure. The KD-tree is a modified binary search tree in multi-dimensional space; invented in 1975 by John Lewis Bentley [16]. The data in each node of the KD-tree is a k-dimensional point in space. This tree is a space-partitioning data structure where each level of the tree makes branching decisions based on its specific conditions and a particular search key associated with that level. In this way, without the need to view each data point in the search environment, it is possible to effectively prune and limit the search space. The basic idea of this tree is to build and search data at any level based on one-dimensional information only. This helps to reduce memory and processing capacity bottlenecks.

Each node of the KD-tree has one of the keys and one of its associated separators. Each separator is an integer between zero and k-1. First, it shows the root of the whole space. Let's suppose x is the root key and i is its delimiter. Space is divided into two regions according to $x[i]$. The y keys with $y[i] < x[i]$ go toward the left tree, and all keys with $y[i] \geq x[i]$ go toward the right tree.

So, generally, a standard KD-tree for a set of next k keys is a binary tree in which:

- Each node has a key and an associated separator as $i \in 0, \ldots, k-1$.
- For each node with the x key and the divider i, each y key is in the left subtree if $y[i] < x[i]$, and each y key is in the right subtree if $y[i] \geq x[i]$.
- The root node has zero depth and zero separators. In general, all nodes at depth d have a separator $d \ mod \ k$ [17].

3 The Proposed Method

The method proposed in this paper is to determine the location inside a building with minimum overheads, using a KD-tree. Since the property of the KD-tree is to break space, and the general idea of the KD-tree is to divide the space into several subspaces using the points on the page, the goal is to break the search space with the help of the KD-tree and find the location in a limited space.

In general, the proposed localization method in this paper includes the following three basic steps:

- Collect and create a building fingerprint database
- Cluster and build the modified KD-tree upon the database
- Run a localization algorithm with restricted inputs in one cluster, defined by the KD-tree output

The first step of the proposed localization method, like other similar methods, is to collect RSS values and create a database of fingerprints from different locations inside the building. In the second step, to speed up the positioning process, clustering and construction of the modified KD-tree are performed. Finally, in the third step, the localization algorithm is applied, whose inputs are limited. In other words, when the user obtains a set of RSS feeds around his location, the set is first clustered based on the KD-tree, and then simply the data from the same cluster with the user's set of RSS feeds are given to the localization algorithm. In this way, with a significant reduction in the volume of input data, it is possible to speed up the localization process.

In the rest of this section, the details of the proposed method are presented.

3.1 Clustering

Clustering is one of the most basic ways to isolate a set of heterogeneous objects in an unknown environment. In this paper, we use the $k-means$ clustering algorithm to avoid complicating matters. Due to its simplicity, this algorithm is very practical and popular in various fields of science [18–21]. This algorithm divides the data into unique clusters using the average distance from the cluster heads. In this paper, the silhouette method is used to determine the number of clusters, k. The silhouette criterion focuses on the quality of the clustering performed. This criterion determines the distribution of data in clusters. The higher the silhouette value, the higher the clustering quality. In the average silhouette method, the clustering algorithm is executed for different values of k, and for each execution, the silhouette criterion is calculated for each member of the cluster. The average of the obtained silhouettes is then taken. The optimal

value of k is the value for which the average silhouette is maximized [22, 23].

It should be noted that the maximum value of k in this process will be equal to the number of APs installed in the building; however, according to the conditions and shape of the building, smaller values can be selected as well.

3.2 Modified KD-Tree

In the localization problem, the basic inputs are the RSSs received from different APs in different parts of the building. This means that when we collect fingerprint information of a location, it is necessary to collect and measure the received RSS from all the APs installed in the building. But the important point is that given the effects of environmental barriers inside the building on the propagation of waves, it cannot be expected that in large buildings the signals of all APs will be seen in all parts of the building. So, we can use this point to create a modified KD-tree. A prerequisite for this work is the initial clustering of the building fingerprint databases, mentioned in the previous section.

According to this important point, the division of the building space can be done. The five steps required to create a modified KD-tree in this article are as follows.

In explaining the proposed method of constructing the modified KD-tree, we cluster the data in the first two steps. For each cluster, we then specify which antennas are visible inside the cluster and which antennas are not. Next, in the fourth step, we seek to identify the root element of the tree. In this step, an element is proposed as the root that provides the ability to differentiate between clusters. For this purpose, we are looking to find a powerful antenna that can be seen in some clusters and not in others. It should be noted that based on [15], the power of an antenna here is the average amount of RSS received for an antenna among the RSS values of that antenna among all parts of the cluster, which can be replaced by the maximum amount of RSS of that antenna in the cluster. After finding this antenna, based on whether the clusters see a new root or not, they are divided into two groups. Each of the two created groups is then referred back to the fourth step to complete their sub-trees by creating an iterative process. The tree creation algorithm continues to run until the lists become single-element. During the execution of the algorithm, it is possible that due to the limited lists of left and right or changes in the list of APs installed in the building over time, some elements are unseen in other lists. These elements will be removed from all lists in the fourth step.

If because for any reason at the end of the fourth step, we see that one or both of the lists A and B created are not single-element and the set of unseen antennas between these lists is thoroughly examined, we are forced to enter the fifth step. At this step, firstly, we try to separate the tree through the ones that have been seen in one or more lists and not in the others. If this method is not effective, the antenna with the highest possible RSS difference between two or more clusters will be selected as the new root and new A and B lists would be formed. The process of dividing and creating a tree continues as before until lists A and/or B become single-element lists.

The details of the steps are as follows:

Step 1: Determine the number of clusters using the silhouette method for RSS feeds collected from the building.

Step 2: Separate and cluster the points of the fingerprint database.

Step 3: Create two lists including *seen antennas* (sorted descending based on RSS feed) and *unseen antennas* for each cluster.

Step 4: Start from the first element of the list of *unseen antennas* in cluster 1 and repeat the following comparison with each of the *unseen antennas* in clusters 2 to k:

Step 4-1: If the desired element is unseen in the list of any other clusters, we will delete it from all lists.

Step 4-2: If this antenna is not part of the antennas seen in at least one cluster, then this antenna is compared with the first two options from the list of antennas seen in other clusters. If it is part of the powerful antennas of those clusters, it is selected as the root of the tree and deleted from all lists. Clusters in which the root antenna is not visible in them are in list A, and clusters in which the root antenna is visible in them are in list B.

Step 4-3: For each of the lists A and B as sub-trees on the right and left of the root, we will repeat step 4 until the lists become single-element or completely check all the unseen antennas of different clusters. If list A becomes a single-element list, the existing cluster is selected as the child/leaf to the right of the current node. If list B becomes a single-element list, the existing cluster is selected as the child/leaf to the left of the current node.

Step 5: If for any reason, the lists A or B is not a single element list and at the same time all the unseen antennas in the list of relevant clusters are finished, the following steps are performed in order:

Step 5-1: Calculate the complementary set of an-

Table 1. Lists of antennas seen/unseen in different clusters.

Cluster 1	Seen APs	AP_2	AP_1	AP_3	
	Unseen APs	AP_4	AP_5	AP_6	
Cluster 2	Seen APs	AP_5	AP_6	AP_3	
	Unseen APs	AP_1	AP_2	AP_4	
Cluster 3	Seen APs	AP_3	AP_1		
	Unseen APs	AP_2	AP_4	AP_5	AP_6

tennas seen in the list of A or B (whichever is not a single-element list). If this set is not empty, the strongest antenna in it is selected as the new root, and while removing it from the list of clusters, we create lists A and B for the new root.

Step 5-2: If the complementary set of antennas seen is empty, then we calculate the RSS difference of each antenna of each cluster with the same RSS of antennas seen by the other clusters in the list. The antenna with the highest relative difference is selected as the root antenna. Then, the clusters whose desired RSS of the antenna is less than half of the difference of antenna power plus the power of the weakest antenna observed in the list of different clusters, are placed as unseen clusters in list A, and the others are placed in list B. If one of the lists A or B is empty in this way, half of the clusters that have the most difference are listed as unseen clusters in list A and the other clusters are listed in list B.

Step 5-3: If lists A or B become single-element lists, their development process will end, otherwise we will repeat step 5.

3.3 An Example

To better understand the proposed algorithm, we provide the following example. Consider a building whose data set is divided into three clusters after collecting and clustering its fingerprints. The building also contains six AP_s, named AP_1 to AP_6, which AP_4 become obsolete over time, but remain on the building's initial list of AP_s. After the evaluations, the order of seen/unseen AP_s in each cluster is shown in Table 1. In this table, the order of antenna power is ascending from left to right.

According to this table, the first AP unseen in cluster 1 is AP_4. This AP is not found in any other clusters; so, we remove it from the list. The second most unseen AP in the cluster 1 list is AP_5, which is the most powerful antenna seen in cluster 2 but does unseen in cluster 3. Therefore, this antenna is selected as the root. We put clusters 1 and 3 in list A and cluster 2 is in list B. Because list B is a single element, cluster 2 is selected as the child/leaf to the left of the root node, and its development stops. In the right subtree and list A, there are two clusters 1 and 3. The third and last unseen AP in the list of cluster 1 is AP_6, which is also in the list of unseen antennas of cluster 3. So, we delete this AP from both lists. Remember that, in the right subtree, we only have clusters 1 and 3. After finishing the list of unseen APs in cluster 1, we go to the unseen AP_s in cluster 3. The first unseen antenna is AP_2, which is among the most powerful antennas seen in cluster 1. Therefore, AP_2 is selected as the root of this subtree and we create two lists A and B for it. Cluster 3 is on list A and cluster 1 is on list B, both of which are single elements. Therefore, cluster 3 is selected as the child/leaf on the right, cluster 1 is selected as the child/leaf on the left, and the construction of the tree ends. Figure 1 shows the final KD-tree.

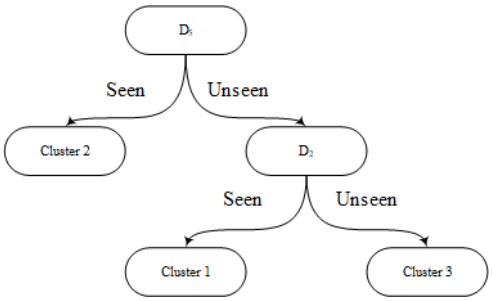


Figure 1. The KD-tree of the example.

4 Performance Evaluation

To implement this proposed method, we used the data set prepared in [11]. This data set contains 365 data points, containing measured RSS data received from 12 APs installed in the building of the Faculty of Engineering at Arak University. The building has 3 floors and includes corridors leading to the classrooms, the administrative department and the professors' room, and the faculty lobby. The fingerprint information of each point is prepared in 3D coordinates (length, width, and height, based on the main plan of the faculty building) and the amount of RSS received from 12 APs. The installed APs are

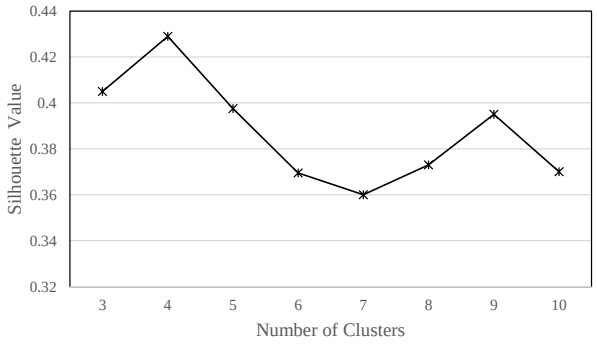


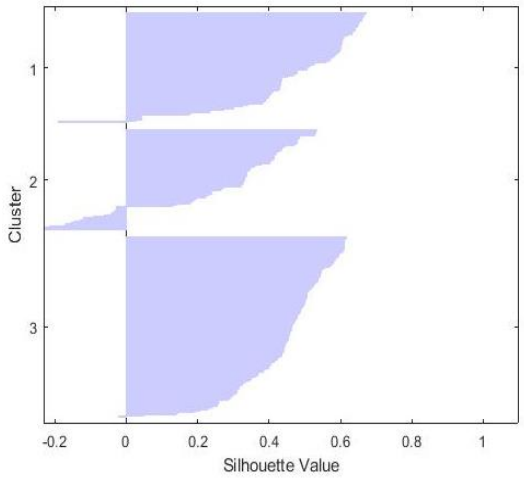
Figure 2. The effect of different numbers of clusters.

named Def79, Def137, Def138, Def140, Def141, Def142, Def143, Def144, Def146, Def147, Def148 and Def149, respectively. To implement the methods and evaluate the results, we used MATLAB 2019 software on a laptop with 8 GB of RAM and a 64-bit version of Windows 10.

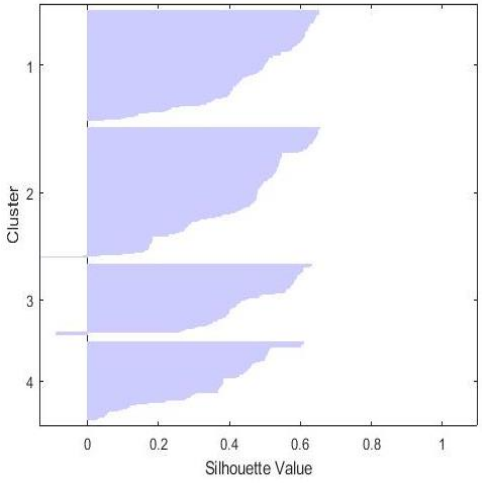
After collecting the data, we evaluated the collected database of fingerprints in different parts of the building, using the silhouette function. We used different numbers of clusters from 3 to 10. As shown in Figure 2, the best possible number of clusters for the data in this dataset is 4 clusters. In 4 clusters formation, the first cluster contains 105 records related to the corridors of the administrative department and the professors' rooms, respectively. The second cluster contains 127 records related to the corridors of the publishing department, buffet, and 3xx series classrooms. The third cluster contains 70 records related to the corridors of the 1xx and 2xx series classrooms. The last cluster contains 64 records related to lobby records. Figure 3 shows the results of calculating Silhouette values for 3, 4, and 8 clusters. Negative values show wrong clustering.

After clustering the data, we performed AP categorization for the seen and unseen APs at each cluster. Table 2 presents the results. It should be noted that in preparing this list if an AP is seen only in a small number of points in a cluster, due to the measurement of the average value, the name of that AP is practically removed from the list of seen APs in that cluster. Figure 4 shows this. For example, in Cluster 1, Def141 is seen only in two data points. There are similar conditions for Def143 too. So, their names are inserted in the unseen list of this cluster. Def 79 and Def144 in Cluster 2, Def141, Def142, and Def148 in Cluster 3, and Def 138, Def142, and Def147 in the Cluster 4 have similar conditions.

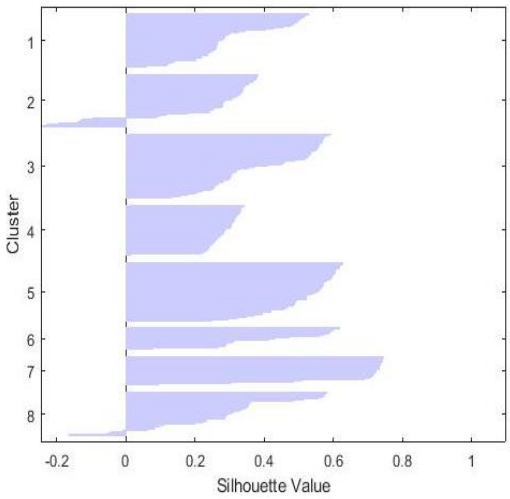
That is interesting that the length of seen and unseen lists in all clusters is equal. This happened by following the proposed clustering and the KD-tree structure, not by external forces.



a. 3 clusters



b. 4 clusters

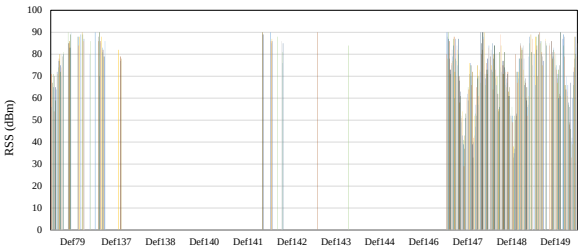


c. 8 clusters

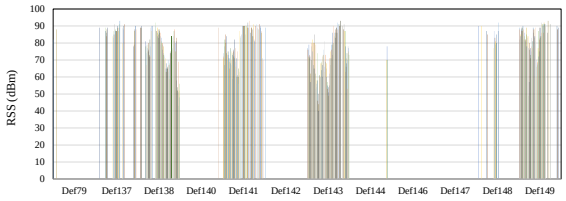
Figure 3. Silhouette value calculation.

Table 2. Lists of seen/unseen antennas in 4 different clusters in the Faculty of Engineering at Arak University.

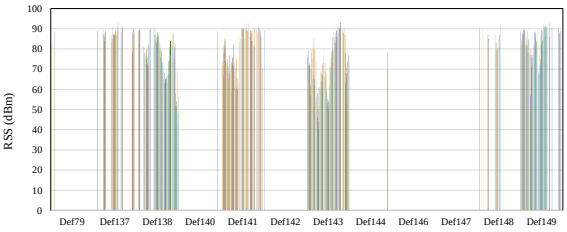
Cluster 1	Seen APs	Def147	Def148	Def149	Def79	Def137	Def142
	Unseen APs	Def138	Def140	Def141	Def143	Def144	Def146
Cluster 2	Seen APs	Def143	Def141	Def138	Def149	Def137	Def148
	Unseen APs	Def140	Def142	Def147	Def146	Def144	Def79
Cluster 3	Seen APs	Def140	Def137	Def138	Def144	Def143	Def149
	Unseen APs	Def79	Def141	Def142	Def146	Def147	Def148
Cluster 4	Seen APs	Def79	Def137	Def148	Def149	Def144	Def143
	Unseen APs	Def138	Def140	Def141	Def142	Def146	Def147



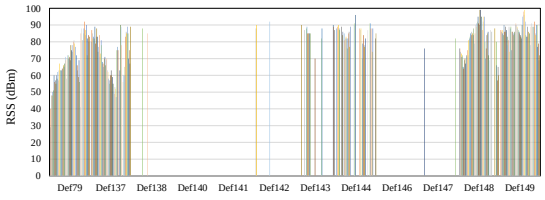
a. Cluster 1



b. Cluster 2



c. Cluster 3



d. Cluster 4

Figure 4. RSS values in different clusters.

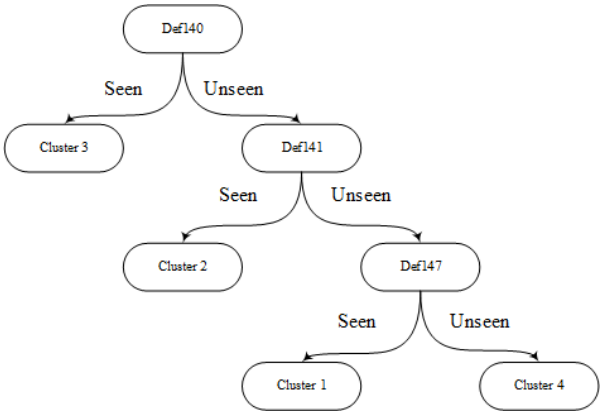


Figure 5. The KD-tree of the faculty.

Accordingly, we create the modified KD-tree for this dataset which is shown in Figure 5. As can be seen in this figure, only data from three AP_s named Def140, Def141, and Def147 are needed to separate the points of 4 different clusters from each other. Thus, according to the clusters' size, by starting from the root of the tree, in the first comparison, at least 19% of the information, in the second comparison, at least 54%, and in the third comparison, at least 71% of the information can be deleted before entering the localization algorithm.

To evaluate the results, we performed localization with three popular Weighted K-Nearest Neighbors (WKNN) [24], Support Vector Machine (SVM) [25], and Naive Bayes (NB) [25] algorithms. These algorithms are among the most used classification algorithms in indoor localization studies [4–7]. For classification, we divided the data set into two parts: 70% for training and 30% for testing, which is applied to all parts of the faculty building. Also, to better ensure the results, we repeated the localization process 4 times.

Table 3 shows the results of the running time of localization algorithms with/without applying the proposed method for space division. In this table, the KDT symbol indicates the application of the proposed method before the application of the localization al-

Table 3. Running time of localization algorithms with/without applying the proposed method.

Algorithm	Localization Running Time (s)				
	Run#1	Run#2	Run#3	Run#4	Average
WKNN	0.002385	0.000484	0.00111	0.000456	0.00110875
SVM	2.012793	1.076879	1.124207	1.410924	1.40620075
SVM+KDT	0.398812	0.23075	0.006437	0.058532	0.17363275
NB	0.0007192	0.0005935	0.0005013	0.0007695	0.000645875
NB+KDT	0.0005592	0.0005175	0.000428	0.000614	0.000529675

Table 4. Error of localization algorithms with/without applying the proposed method.

Algorithm	Localization Error (cm)				
	Run#1	Run#2	Run#3	Run#4	Average
WKNN	603.8896	591.3708	615.0501	589.3617	599.91805
WKNN+KDT	405.9888	452.4086	531.2732	431.8835	455.388525
SVM	580.8307	694.1192	577.3461	882.742	683.7595
SVM+KDT	497.8558	507.6408	680.0176	641.7521	581.816575
NB	627.0982	767.3609	627.098	767.3616	697.229675
NB+KDT	563.7757	527.0536	705.4126	551.6301	586.968

gorithm. As can be seen in Table 3, from the average running time point of view, the application of the proposed method, compared to when it is not used, resulted in an improvement of 326% for the WKNN algorithm, 810% for the SVM algorithm and 122% for the NB algorithm at the execution time. Reviewing the results shows that the proposed method has the best performance when used with the SVM. Using Stochastic Gradient Descent (SGD) in heart of the SVM makes it computationally expensive and complex. The complexity of the SVM is $O(n^3)$ which shows the high impact of the number of input values on the running time in this algorithm. So, when the proposed approach breaks the inputs into smaller volumes, the running time of the algorithm improves fast.

Table 4 shows the average localization error too. To calculate the error, we use the Root Mean Square Error Rate (RMSE) [11]:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2} \qquad (1)$$

where N is the number of points in a scenario, x_i is the actual position of point i, and $\widehat{x}_i$ is the estimated position of point i. As can be seen in this table, the use of the proposed method, in addition to reducing runtime, has also had a positive effect on increasing the execution accuracy of algorithms. According to the information in this table, by applying the proposed method, the accuracy of the WKNN algorithm has been improved by 132%, the accuracy of the SVM algorithm by 118%, and the accuracy of the NB algorithm by 119%. These results show that using the proposed method and creating smaller segments makes the data entered into the classification algorithms more accurate and appropriate. This issue increases the accuracy of the output of the classifiers.

5 Conclusions

In this paper, we present a new KD-tree-based method. The purpose of this method is to quickly minimize the amount of data input to the localization algorithms. For this purpose, first, the fingerprint database is clustered and then through the proposed algorithm, the modified KD-tree is created with the aim of quick and easy decision-making to find a small subset of searchable data. Finally, when the user sends his fingerprint information to this system, first through the tree, the desired cluster is determined and then the search is performed only within the same cluster. The evaluation results of the proposed method indicate that the application of this method on conventional localization algorithms improves their performance, in terms of running times and error rates, by at least 2 times.

References

[1] Yungeun Kim, Hyojeong Shin, Yohan Chon, and Hojung Cha. Crowdsensing-based Wi-Fi radio map management using a lightweight site survey. *Computer Communications*, 60:86–96, 2015. ISSN 0098-5589. doi:10.1016/j.comcom.2014.11.006.

[2] Faheem Zafari, Athanasios Gkelias, and Kin K Leung. A Survey of Indoor Localization Systems and Technologies. *IEEE Communications Surveys & Tutorials*, 21(3):2568–2599, 2019. ISSN 1553-877X. doi:10.1109/COMST.2019.2911558.

[3] Suining He and S-H Gary Chan. Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons. *IEEE Communications Surveys & Tutorials*, 18(1):466–490, 2015. ISSN 1553-877X. doi:10.1109/COMST.2015.2464084.

[4] Huthaifa Obeidat, Wafa Shuaieb, Omar Obeidat, and Raed Abd-Alhameed. A Review of Indoor Localization Techniques and Wireless Technologies. *Wireless Personal Communications*, 119(1): 289–327, 2021. doi:10.1007/s11277-021-08209-5.

[5] Navneet Singh, Sangho Choe, and Rajiv Punmiya. Machine Learning Based Indoor Localization Using Wi-Fi RSSI Fingerprints: An Overview. *IEEE Access*, 9:127150 – 127174, 2021. ISSN 2169-3536. doi:10.1109/ACCESS.2021.3111083.

[6] Priya Roy and Chandreyee Chowdhury. A Survey of Machine Learning Techniques for Indoor Localization and Navigation Systems. *Journal of Intelligent & Robotic Systems*, 101(3):1–34, 2021. doi:10.1007/s10846-021-01327-z.

[7] Tian Yang, Adnane Cabani, and Houcine Chafouk. A Survey of Recent Indoor Localization Scenarios and Methodologies. *Sensors*, 21 (23), 2021. doi:10.3390/s21238086.

[8] Teemu Roos, Petri Myllymäki, Henry Tirri, Pauli Misikangas, and Juha Sievänen. A Probabilistic Approach to WLAN User Location Estimation. *International Journal of Wireless Information Networks*, 9(3):155–164, 2002. doi:10.1023/A:1016003126882.

[9] Mu Zhou, Yaohua Li, Muhammad Junaid Tahir, Xiaolong Geng, Yong Wang, and Wei He. Integrated Statistical Test of Signal Distributions and Access Point Contributions for Wi-Fi Indoor Localization. *IEEE Transactions on Vehicular Technology*, 70(5):5057 – 5070, 2021. ISSN 0018-9545. doi:10.1109/TVT.2021.3076269.

[10] Hamada Rizk, Ahmed Elmogy, and Hirozumi Yamaguchi. A Robust and Accurate Indoor Localization Using Learning-Based Fusion of Wi-Fi RTT and RSSI. *Sensors*, 22(7), 2022. doi:10.3390/s22072700.

[11] Fatemeh Ali Asgari Renani, Hossein Ghaffarian, and Mastaneh Chegeni. Indoor Path Tracking Using Combination of Viterbi and WKNN. *Journal of Computing and Security*, 8(2):19–26, 2021. doi:10.22108/jcs.2021.129843.1078.

[12] Jiancun Fan, Susu Chen, Xinmin Luo, Ying Zhang, and Geoffrey Ye Li. A Machine Learning Approach for Hierarchical Localization Based on Multipath MIMO Fingerprints. *IEEE Communications Letters*, 23(10):1765–1768, 2019. ISSN 1089-7798. doi:10.1109/LCOMM.2019.2929148.

[13] Ahmet Çağdaş Seçkin and Aysun Coşkun. Hierarchical Fusion of Machine Learning Algorithms in Indoor Positioning and Localization. *Applied Sciences*, 9(18), 2019. ISSN 1089-7798. doi:10.3390/app9183665.

[14] Daniel Alshamaa, Farah Mourad-Chehade, and Paul Honeine. A Weighted Kernel-Based Hierarchical Classification Method for Zoning of Sensors in Indoor Wireless Networks. In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5. IEEE, 2018. ISBN 978-1-5386-3512-4. doi:10.1109/SPAWC.2018.8445918.

[15] Hossein Ghaffarian. Reducing Search Area in Indoor Localization Applications. *Wireless Personal Communications*, 117(2), 2021. ISSN 1243–1258. doi:10.3390/app9183665.

[16] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975. doi:10.1145/361002.361007.

[17] Maria Mercè Pons Crespo. Design, analysis and implementation of new variants of kd-trees. *Master's thesis, Universitat Politècnica de Catalunya*, 2010.

[18] Dengxiu Yu, Hao Xu, CL Philip Chen, Wenjie Bai, and Zhen Wang. Dynamic Coverage Control Based on K-Means. *IEEE Transactions on Industrial Electronics*, 69(5):5333–5341, 2021. ISSN 0278-0046. doi:10.1109/TIE.2021.3080205.

[19] Jesús Estupiñán Ricardo, Jorge Juan Domínguez Menéndez, Ignacio Fernando Barcos Arias, Jorge Manuel Macías Bermúdez, and Noel Moreno Lemus. Neutrosophic K-means for the analysis of earthquake data in Ecuador. *Neutrosophic Sets and Systems*, 44(1), 2021. ISSN 2331-608X.

[20] Zhen-Song Chen, Xuan Zhang, Witold Pedrycz, Xian-Jia Wang, Kwai-Sang Chin, and Luis Martínez. K-means clustering for the aggregation of HFLTS possibility distributions: N-two-stage algorithmic paradigm. *Knowledge-Based Systems*, 227:107230, 2021. doi:10.1016/j.knosys.2021.107230.

[21] Shashank Reddy Vadyala, Sai Nethra Betgeri, Eric A Sherer, and Amod Amritphale. Predic-

tion of the number of COVID-19 confirmed cases based on K-means-LSTM. *Array*, 11:100085, 2021. doi:10.1016/j.array.2021.100085.

[22] Malika Charrad, Nadia Ghazzali, Veronique Boiteau, and Azam Niknafs. Determining the Best Number of Clusters in a Data Set. `https://cran.rproject.org/web/packages/NbClust/NbClust.pdf`, Date Accessed: Dec. 30, 2022.

[23] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[24] Gints Jekabsons and Vadim Zuravlyov. Refining Wi-Fi based indoor positioning. In *Proceedings of 4th International Scientific Conference Applied Information and Communication Technologies (AICT), Jelgava, Latvia*, pages 87–95, 2010.

[25] Jian Pei Jiawei Han, Micheline Kamber. *Data Mining: Concepts and Techniques*. Elsevier, 2011.

Hossein Ghaffarian received his BSc in software engineering from Shahid Bahonar University of Kerman, Iran, in 2003. Also, he received his M.Sc. and Ph.D. from Iran University of Science and Technology, Tehran, Iran in 2008 and 2013 respectively. Currently, he is with the department of computer engineering, faculty of engineering at Arak University, Arak, Iran. Dr. Ghaffarian's main research interests are wired/wireless computer networks, Intelligent Transportation Systems (ITS), and applied AI.

Seyfollah Soleimani was born in January 1977 in the city of Marvdasht in the south of Iran. He finished his high school studies there. In 1995, he took the universities entrance exam and was accepted to study Computer Engineering at Shiraz University. He finished his studies there in 1999 when he participated in the master's entrance exam of universities and was accepted into the master's program of Artificial Intelligence at Iran University of Science and Technology in Tehran. Afterward, he worked as a faculty member at Arak University in 2002. In 2008 he went to Belgium and started his Ph.D. at Ghent University. After receiving his Ph.D., he came back to Iran in 2014 and resumed working as a faculty member at Arak University in the computer engineering department till now.

Seyede Habibe Zadsar received her bachelor's degree in software engineering in 2014 from Industrial University of Sirjan, Kerman, Iran. She received her master's degree in software engineering in 2019 from Arak University, Arak, Iran. Mrs. Zadsar's main research interest is indoor localization.