**Improved Fingerprint Feature Extraction using Kalman Filter and Deep Learning for Indoor Localization**

|  |
| --- |
| **ABSTRACT** |
| **Objectives:** The emergence of Bluetooth Low Energy (BLE) technology has opened up numerous opportunities in indoor localization. Various methods have been proposed to extract fingerprint features from Bluetooth signals' Received Signal Strength Indicator (RSSI) values. However, these studies have often yielded results with significant errors and instability. Combining filters and deep learning algorithms may provide more optimal outcomes.  **Methods:** In this research, a Kalman filter is employed to enhance the stability of the received RSSI values. Subsequently, Autoencoder and Convolutional Autoencoder models are proposed to extract fingerprint features. Finally, random test points are compared to reference points in the database using normalized cross-correlation. The coordinates of these test points are predicted by averaging the coordinates of the reference points with the highest cross-correlation.  **Results:** The system's performance is evaluated based on the number of reference points with the highest cross-correlation (), average error, and other statistical values. The Convolutional Autoencoder model, combined with the Kalman filter, yields a minor average error of 0.98 meters with  **Conclusion:** The achieved results, characterized by minimal errors, demonstrate that using the Kalman filter and the Convolutional Autoencoder model reduces noise and enhances accuracy in indoor localization. |

1. **Introduction**

Indoor positioning has long been a crucial issue in various large-scale applications today, such as inventory management, equipment tracking, and product monitoring. Currently, there are numerous technologies for indoor positioning, including Wi-Fi, Ultra-Wideband, Bluetooth, optical technology, and infrared. However, these technologies have inherent limitations, such as high production costs, energy consumption, or significant inaccuracies. Bluetooth Low Energy (BLE) technology has been researched and developed to address these challenges. Its advantages include low production costs, energy efficiency, and easy deployment.

Nonetheless, it does not match the precision of Ultra-Wideband (UWB) and lacks the coverage of Wi-Fi. Several BLE-based positioning methods have been proposed, among which BLE fingerprinting stands out for its relatively good accuracy. Hence, this research focuses on developing an indoor positioning method based on BLE fingerprinting.

This approach places some Bluetooth beacons (BC) at fixed locations. After a predetermined time interval, these BCs transmit data packets containing IDs and additional information. Devices in need of positioning continuously gather this information and transmit it to a server for processing. The device's location is predicted based on BLE fingerprint characteristics. This method is divided into two main phases: offline and online. The offline phase collects Received Signal Strength Indicator (RSSI) values from BCs at each reference point (RP). These values are processed to extract features and stored in a fingerprint map database. The online phase consists of collecting RSSI values from BC signal packets. These values are also used to extract fingerprint features, which are then compared with reference points. The reference points with the most similar fingerprint features are selected to predict the coordinates of the target location.

Several BLE fingerprinting methods have been proposed. Zou and colleagues *[1]* applied graph optimization to achieve a best-case accuracy of 1.27 meters. Martin and colleagues *[2]* employed Gaussian kernel-based fingerprinting with an accuracy below 1.5 meters in 90% of cases. Subedi and colleagues *[3]* utilized a two-step fingerprint-based approach with an accuracy of 1.05 meters. Li and colleagues *[4]* utilized an eight-neighborhood template-matching mechanism with a 1-meter accuracy.

This paper proposes an indoor positioning method based on BLE fingerprinting, specifically fingerprint feature extraction. It involves deploying six BCs around a room, with the RSSI values of each reference point stored in the fingerprint database. RSSI measurements are susceptible to noise, so the Kalman filter and deep learning models like Autoencoders and Convolutional Autoencoders are employed to reduce noise and data dimensionality. The Minkowski distance is calculated between the measured fingerprint and reference fingerprint to identify the k nearest reference points with the measured fingerprint. This information is used to calculate coordinates and assess accuracy.

The structure of this article is as follows: Section 2 presents related research on indoor positioning. Section 3 provides an overview of the dataset construction and the algorithms used. Experimental results are presented and compared with previous research findings in Section 4. Conclusions and future directions are discussed in Section 5.

1. **Related work**

Before introducing our proposed method, we examine various indoor positioning technologies.

There is extensive global research on indoor positioning technology, with methods and technologies outlined in *[5]*. Some non-object-based positioning technologies are mentioned, such as using cameras for detection and location. Object-based positioning technologies include Bluetooth, Wi-Fi, RFID, Ultra-Wideband, or wireless sensor network technologies. Several articles on indoor positioning methods are summarized in Table 1.

Table 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Studies** | **Method** | **Technique** | **Accuracy** |
| Shuang Li et al 2021 | Camera | Use the image and feed it into the detection algorithm |  |
| Liang Ma et al 2018 | RFID tags | Use phase and RSSI signals to feed into POS algorithm |  |
| Weipeng et al 2018 | Wireless Local Area Network | Use existing WLAN infrastructure (APs) |  |
| Samaneh et al 2017 | Wifi Fingerprint | KNN algorthm |  |
| Ashry et al 2019 | Wifi | Trilateration and fingerprinting methods |  |
| Mai et al 2020 | Bluetooth Fingerprint | Pedestrian Dead Reckoning + Fingerprinting + Particle filter |  |
| Alvin Riady et al 2022 | Bluetooth Fingerprint | ANN |  |

The study [5] has identified the most suitable technologies for Indoor Positioning Systems (IPS) and their principles, methods, and algorithms. One of the algorithms uses a mobile phone camera to capture images and apply detection algorithms, as Shuang Li et al. [6] mentioned. However, the results showed a 95% margin of error greater than 2 meters, with an average positioning error of 0.61 meters.

Other algorithms utilize RFID tags, as demonstrated in [7]. Liang Ma et al. [7] employed phase and RSSI signals in their Positioning (POS) algorithm for location calculations.

Most non-invasive positioning technologies rely on infrastructure, such as wireless sensor networks. These sensors detect changes when people or objects move within the area. For instance, in [8] , the authors used a wireless local area network to model their positioning system.

Nonetheless, non-invasive positioning technologies relying on infrastructure can significantly increase system construction costs and may need to be more versatile across different environments. In contrast, technologies like Wi-Fi, Bluetooth, and Ultra Wideband allow for user positioning through their carried devices. In-house wireless positioning systems use RSSI signals to determine coordinates, as discussed in [9] and [10]. Samaneh et al. [9] employed Wi-Fi RSSI signals to create fingerprints, which were then utilized for training and testing. Using fingerprints was also mentioned in [5], based on map analysis. Various algorithms, such as neural networks, KNN, and SVM, can employ fingerprints. Typically, they compute the correlation between RSSI signals during positioning and pre-existing fingerprint datasets to provide the closest location. In [10], the trilateration method and Wi-Fi RSSI fingerprint were used for positioning. [11] combined fingerprint and Pedestrian Dead Reckoning with the Particle Filter method to infer coordinates from signal transmission distances. Alvin et al. [12] also employed fingerprints using the ANN method. Models using physical objects for positioning offer higher accuracy but increase computational complexity and processing time. Fingerprinting has gained traction recently with the use of neural networks, KNN, SVM, and Euclidean distance. However, these studies have yet to address cost and energy efficiency concerns effectively. Therefore, this research proposes a low-energy Bluetooth fingerprinting method to address these issues while maintaining the highest possible accuracy.

1. **Method**

The critical steps of indoor positioning using fingerprint features are illustrated in Figure X. Firstly, specific Reference Points (RPs) are determined in the positioning area, and the RSSI values of BLE beacons are measured at each RP. These RSSI values are then stored in the fingerprint database. Subsequently, the RSSI values are passed through a Kalman filter to eliminate noise while preserving stability in the RSSI values at each reference point. Afterward, Autoencoders (AE) or Convolutional Autoencoders (CAE) models extract fingerprint features at each reference point. Lastly, a normalized cross-correlation method compares the correlation between the point to be located and all the reference points in the database. The coordinates of the points to be determined are calculated as the average of the k points with the highest correlation. Each step in our proposed method is presented in detail below.

*A diagram of a flowchart

Description automatically generated*

Figure 1: The main steps in the proposed method

* 1. **Data Collection**

Assume there are N reference points within the coverage area. At each reference point, RSSI values are measured over a period and organized into the following matrix:

|  |  |
| --- | --- |
|  | (1) |

In equation (1), represents the RSSI value at reference point n obtained from beacon . Here, , representing the sequential number of reference points, and , the number of beacons used within a defined range.

* 1. **Kalman Filter**

The Kalman filter, introduced by Rudolf E. Kalman and published in 1960 [13] is a widely used tool in control systems. It is employed to estimate the state of a process in the presence of noise in measurements. This method works by determining the estimated state of the process based on actual measurements and the ideal state, to minimize the mean square error between them. The Kalman filter consists of two primary steps: Prediction and Measurement Update [14], [15]. The visualization of the Kalman filter process is depicted in Figure 2.

*A diagram of a algorithm

Description automatically generated*

Figure 2: Implementation of the Kalman filter

*Prediction*

The current state and error covariance matrix of the process are estimated in a general form as:

|  |  |
| --- | --- |
|  | (2) |
|  | (3) |

Where:

- : State transition model matrix

- : Control input model matrix

- : Control vector

- : Process noise covariance matrix

*Measurement Update*

The initial task in the update process is to compute the Kalman Gain, as shown in equation (4):

|  |  |
| --- | --- |
|  | (4) |

Next, the expected state and covariance matrix are updated as per equations (5) and (6):

|  |  |
| --- | --- |
|  | (5) |
|  | (6) |

Here, is the matrix relating to state through the measurement , where is a random variable representing the measurement noise covariance.

The Kalman filter operates recursively: the Prediction process estimates the current provisional state based on the previous state, and then the Measurement Update process adjusts the estimate with an actual measurement. These steps are repeated with previous posterior estimates used to predict new prior estimates [15].

With our collected RSSI data, each vector is passed through the Kalman filter, with the first value as the average of R samples in each vector

|  |  |
| --- | --- |
|  | (7) |

The Kalman filter enhances the stability of our dataset, thereby improving the fingerprint features for each reference point and enhancing training performance.

* 1. **Fingerprint Features Extraction**

*Autoencoder*

Autoencoder (AE) is a neural network model in machine learning and computer vision designed for unsupervised data encoding. It aims to learn a lower-dimensional representation (encoding) for higher-dimensional data, reducing complexity and saving computational resources. AE is often used for dimensionality reduction and feature extraction tasks. Figure 3 provides a visual representation of AE architecture, consisting of Encoder, Code, and Decoder:

- Encoder: Receives input data and transforms it into a lower-dimensional compressed form. The encoder typically consists of a sequence of neuron layers, learning to extract essential information from the data and represent it as a compressed vector. The neuron layers in the encoder often employ activation functions like ReLU, sigmoid, or hyperbolic tangent.

- Code: Contains the compressed data, also known as the output of the encoder. It is a crucial part of the network because it holds the features of the input data.

- Decoder: Receives the compressed data from the encoder and attempts to reconstruct the original data. The decoder also consists of a sequence of neuron layers, transforming the compressed data into the original data while minimizing the reconstruction error.

*A diagram of a code

Description automatically generated*

Figure 3: Proposed Autoencoder model structure

The training process of an Autoencoder aims to minimize the error between the original data and the reconstructed data by adjusting the encoder and decoder weights and parameters. Loss functions commonly include Mean Squared Error (MSE) and Binary Cross-Entropy (BCE).

Each reference point in our database has data vectors of size , which are flattened into vectors to match the input size of the AE model. After passing through the encoder, the data is compressed into a code, which is then decoded to produce an output of . In this study, the Autoencoder model uses the hyperbolic tangent (tanh) activation function, employs the Adam optimization algorithm, and uses Mean Squared Error (MSE) as the loss function.

*Convolutional Autoencoder*

The Convolutional Autoencoder (CAE) combines convolutional neural network principles with an autoencoder. It is often used for unsupervised learning tasks. Like an autoencoder, the CAE architecture consists of an Encoder, Code, and Decoder [16], [17]. The proposed CAE architecture in this study is illustrated in Figure 4.

*A diagram of a puzzle

Description automatically generated with medium confidence*

Figure 4: Proposed Convolutional Autoencoder model structure

The encoding part processes the input as a matrix using convolutional layers to produce lower-dimensional output than the input matrix. The decoding part takes the lower-dimensional representation from the encoding part and transforms it back to the original matrix size using decoding layers. The training process of the Convolutional Autoencoder is similar to that of the Autoencoder, with the aim of minimizing the difference using Mean Squared Error (MSE) as the loss function.

As the input of the CAE model is a matrix, each data vector is transformed into a matrix with 96 zero-padding elements.

*A screenshot of a computer game

Description automatically generated*

Figure 5: The data vector is converted to matrix form for input to the Convolutional Autoencoder model.

* 1. **Coordinate Prediction**

1. *Correlation*

Signal correlation is a crucial aspect in signal research and analysis. In this study, we employ a correlation system to compute and compare the input signal with an available fingerprint dataset. Considering two discrete signals and we calculate their correlation, denoted as using the following formula:

|  |  |
| --- | --- |
|  | (8) |

Where and represent specific time intervals for calculating the correlation between the two signals [18]. In a special case where the two signals are identical, it can be observed that in this case, the main correlation is the signal's energy:

* 1. *Normalized Cross-Correlation*

Normalized Cross-Correlation (NCC) is used in signal processing to measure the degree of similarity or correlation between two signals. NCC is typically employed to search for a specific signal pattern within a larger signal.

This research proposes using the NCC coefficient to compare the input signal with a pre-existing fingerprint database to determine the most accurate coordinates. NCC between two signals and is determined by the following formula [18]:

|  |  |
| --- | --- |
|  | (9) |

This formula normalizes the aggregate correlation by dividing the numerator by the product of the energy of two signals, and The result falls within the range of -1 to 1, indicating the level of similarity between the two signals. A value of 1 typically represents complete correlation, while -1 indicates complete inverse correlation. A value close to 0 generally indicates low or no correlation between the two signals.

Using this approach, we identify k reference points with the closest distance. Eventually, the point coordinates to be determined are predicted as the centroid of these reference points. Different values of result in different predicted coordinates, calculated using formula (10):

|  |  |
| --- | --- |
|  | (10) |

1. **Experimentation and results** 
   1. **Data Collection**

The experiment was conducted on the 6th floor of the Tạ Quang Bửu Library at Hanoi University of Science and Technology. Six BLE beacons were placed at coordinates (0,0), (0,4), (0,8), (8,0), (8,4), (8,8) within an area, as described in Figures 6 and 7. The beacons and Bluetooth signal strength receiving devices were on the same floor plane.

A screenshot of a cell phone

Description automatically generated with medium confidence

Figure 6

A diagram of a grid with red dots and blue objects

Description automatically generated

Figure 7: Arrange the experiment to collect RSSI values from beacons at each reference point.

There are a total of 75 reference points on the map. We collected 200 RSSI value samples for a specific beacon at each reference point. Additionally, 20 random test points were collected to assess the performance of the fingerprint feature extraction model, as depicted in Figure 7.

A diagram of a grid with green and blue dots

Description automatically generated

Figure 8: Test scores are collected randomly.

*A graph with blue and orange lines

Description automatically generated*

Figure 9: The RSSI value of the same beacon is obtained at different reference points.

*A graph with blue and orange lines

Description automatically generated*

Figure 10: RSSI value obtained from two different beacons at the same reference point.

Figure 8 presents a specific example of RSSI data from 200 samples recorded at two reference points (RP1 and RP2) for a specific beacon. Conversely, Figure 9 illustrates RSSI data from 200 samples obtained from two different beacons (BC1 and BC2) at a reference point. These data illustrate the uneven signal variations. This inconsistency may be due to the influence of the surrounding environment and factors causing random errors during the experimental process. This issue poses a significant challenge for indoor localization methods relying on BLE signals.

* 1. **Utilizing Kalman Filtering**

As explained in the previous section, noise factors can significantly affect the process of fingerprint feature extraction and BLE signal-based localization. Therefore, the database we collected was subjected to Kalman filtering to partially eliminate some of the noise as mentioned above values. Moreover, it enhances the feature characteristics of RSSI values at each reference point. Figure 10 below illustrates the difference before and after employing Kalman filtering. It is evident that, after passing through the Kalman filter, the RSSI data eliminates noisy values, resulting in new, more stable data.

A graph with blue and orange lines

Description automatically generated

Figure 11: Raw RSSI value and after passing it through the Kalman filter

* 1. **Experimental results**

Table 2: Statistical parameters of the proposed methods with different k values (unit: m).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Methods** | **Statistics** | **k = 3** | **k = 4** | **k = 5** | **k = 6** | **k = 7** |
| CAE\_Kalman | Mean | 1.19 | 0.98 | 1.24 | 1.37 | 1.41 |
| Min | 0.33 | 0.12 | 0.20 | 0.37 | 0.52 |
| Max | 2.33 | 1.60 | 2.24 | 2.39 | 2.33 |
| Median | 1.05 | 1.02 | 1.22 | 1.20 | 1.42 |
| Var | 0.31 | 0.21 | 0.28 | 0.26 | 0.26 |
| Std | 0.55 | 0.46 | 0.53 | 0.51 | 0.51 |
| CAE | Mean | 1.12 | 1.07 | 1.21 | 1.18 | 1.37 |
| Min | 0.23 | 0.2 | 0.27 | 0.33 | 0.40 |
| Max | 2.57 | 2.15 | 2.41 | 2.27 | 2.59 |
| Median | 1.05 | 1.02 | 1.22 | 1.12 | 1.25 |
| Var | 0.41 | 0.28 | 0.32 | 0.28 | 0.46 |
| Std | 0.64 | 0.53 | 0.56 | 0.53 | 0.68 |
| AE\_Kalman | Mean | 1.25 | 1.16 | 1.21 | 1.37 | 1.46 |
| Min | 0.33 | 0.25 | 0.28 | 0.47 | 0.87 |
| Max | 3.07 | 2.85 | 2.61 | 2.69 | 3.03 |
| Median | 1.05 | 1.09 | 1.09 | 1.31 | 1.38 |
| Var | 0.44 | 0.29 | 0.25 | 0.24 | 0.26 |
| Std | 0.66 | 0.54 | 0.50 | 0.49 | 0.51 |
| AE | Mean | 2.82 | 2.60 | 2.76 | 2.69 | 2.72 |
| Min | 0.94 | 0.25 | 0.28 | 0.17 | 0.77 |
| Max | 5.49 | 4.75 | 5.07 | 4.78 | 4.90 |
| Median | 2.96 | 2.65 | 2.66 | 2.71 | 2.71 |
| Var | 1.94 | 1.51 | 1.95 | 1.70 | 1.64 |
| Std | 1.39 | 1.23 | 1.40 | 1.31 | 1.28 |

Table 2 displays the results using two methods, one incorporating the Kalman filter and the other without the filter, with different values of . Parameters in the table include the mean, median, maximum, and minimum error values. Firstly, we compare two methods, AE and CAE: the average error values for different values with the CAE method are significantly smaller than those with the AE method. The AE method provides the smallest average error of 2.60m with , while the CAE method yields the smallest average error of 1.07m with the same value. When combined with the Kalman filter, it can be observed that the accuracy of the localization task improves. Specifically, with , the AE\_Kalman method achieves the smallest average error of 1.16m, which is an improvement compared to AE (2.6m), and the CAE\_Kalman method delivers the smallest average error of 0.98m compared to CAE's 1.07m. Among the four methods we experimented with, CAE\_Kalman proved to be the most stable method, with the smallest localization error of 0.12m and the largest error of 2.39m In contrast, the AE method shows a maximum localization error of 5.49m at .

|  |  |
| --- | --- |
| A graph with a line graph and a red line  Description automatically generated  Figure 12 | A graph with green and orange lines  Description automatically generated  Figure 13 |
| A graph with a line graph and a red line graph  Description automatically generated  Figure 14 | A graph with blue and green lines  Description automatically generated  Figure 15 |
| A graph of a number of boxes  Description automatically generated with medium confidence  Figure 16 | |

Figures 12, 13, 14, and 15 illustrate cumulative distribution function (CDF) curves for the following methods: CAE-Kalman, CAE, AE-Kalman, and AE. It can be observed that CAE-Kalman and CAE have similar CDF curves, while AE-Kalman and AE exhibit similar curves. When the Kalman filter is applied, CAE-Kalman outperforms CAE, and AE-Kalman outperforms AE. CAE-Kalman and CAE have a lower error range of less than 2m, whereas AE-Kalman and AE have an error range of less than 4m. Using Kalman filtering improves the performance of fingerprint-based localization and reduces the error range.

Figure 16 represents the localization error box and whisker plots for the four methods we used in this study, with a value of . It is evident that the CAE method generally outperforms the AE method, and using the Kalman filter in the data processing phase proves to be more effective than not.

The CAE method combined with the Kalman filter is compared to studies using native BLE fingerprint-based localization. The methods are compared in various aspects, such as the number of beacons used, the area scale, and minimum, average, and maximum localization errors. As explained in section 2, Mai et al [11] used fingerprinting combined with Pedestrian Dead Reckoning and Particle filter to achieve a minimum average error of 1.18m. Alvin Riady et al [12], with a larger localization scale and a greater number of beacons than our method, achieved minimum average and maximum errors of 1.1178m and 3.3601m, respectively. Li et al [4] used the ENTM method, developed by the KNN and WKNN methods, and achieved an average error of 1m. Table 4 compares the CAE method combined with the Kalman filter and other methods. The comparison table shows that the proposed CAE method combined with the Kalman filter achieved an average error of 0.96m, significantly outperforming the compared methods.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Studies** | **Methods** | **Number of Becons** | **Area size (m x m)** | **Minimum Error**  **(m)** | **Average Error**  **(m)** | **Maximum Error**  **(m)** |
| Mai et al [11] | Pedestrian Dead Reckoning + Fingerprinting + Particle filter | 8 | 35,25 | --- | 1.18 | --- |
| Alvin Riady et al [12] | ANN | 23 |  | 0.1055 | 1.1178 | 3.3601 |
| Mingfeng Li et al [4] | ENTM | 4 |  | --- | 1 | --- |
| This study | CAE + Kalmam filter | 6 |  | 0.10 | 0.96 | 1.77 |

1. **Conclusions**

In this research, we employed four distinct methods to assess the performance of indoor localization: CAE\_Kalman, CAE, AE\_Kalman, and AE. Our study results reveal that the CAE model outperforms the AE model, highlighting the superiority of the CAE model in fingerprint feature extraction for indoor localization. Additionally, we examined the impact of applying the Kalman filter to both models. The results demonstrate that using the Kalman filter significantly enhances the performance of both models compared to not using the filter. This underscores the effectiveness of improving the stability and accuracy of RSSI values obtained from BLE beacon signal transmitters. In summary, this research has elucidated the excellence of the CAE model and the positive effects of the Kalman filter in enhancing the performance of fingerprint feature extraction for indoor localization.

**References**

[1] Z. Zuo, L. Liu, L. Zhang, and Y. Fang, “Indoor Positioning Based on Bluetooth Low-Energy Beacons Adopting Graph Optimization,” *Sensors*, vol. 18, no. 11, Art. no. 11, Nov. 2018, doi: 10.3390/s18113736.

[2] P. Martins, M. Abbasi, F. Sa, J. Celiclio, F. Morgado, and F. Caldeira, “Intelligent beacon location and fingerprinting,” *Procedia Comput. Sci.*, vol. 151, pp. 9–16, Jan. 2019, doi: 10.1016/j.procs.2019.04.005.

[3] S. Subedi, H.-S. Gang, N. Y. Ko, S.-S. Hwang, and J.-Y. 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.

[4] M. Li, L. Zhao, D. Tan, and X. Tong, “BLE Fingerprint Indoor Localization Algorithm Based on Eight-Neighborhood Template Matching,” *Sensors*, vol. 19, no. 22, p. 4859, Nov. 2019, doi: 10.3390/s19224859.

[5] L. Batistić and M. Tomic, “Overview of indoor positioning system technologies,” in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2018, pp. 0473–0478. doi: 10.23919/MIPRO.2018.8400090.

[6] S. Li, B. Yu, Y. Jin, L. Huang, H. Zhang, and X. Liang, “Image-Based Indoor Localization Using Smartphone Camera,” *Wirel. Commun. Mob. Comput.*, vol. 2021, p. 3279059, Jul. 2021, doi: 10.1155/2021/3279059.

[7] L. Ma, M. Liu, H. Wang, Y. Yang, N. Wang, and Y. Zhang, “WallSense: Device-Free Indoor Localization Using Wall-Mounted UHF RFID Tags,” *Sensors*, vol. 19, no. 1, Art. no. 1, Jan. 2019, doi: 10.3390/s19010068.

[8] W. Guan, S. Wen, H. Zhang, and L. Liu, “A Novel Three-dimensional Indoor Localization Algorithm Based on Visual Visible Light Communication Using Single LED,” in *2018 IEEE International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, Nov. 2018, pp. 202–208. doi: 10.1109/AUTEEE.2018.8720798.

[9] S. Amirisoori, S. Mohd Daud, N. A. Ahmad, S. Natasha, N. Sa’at, and N. Noor, “WI-FI Based Indoor Positioning Using Fingerprinting Methods (KNN Algorithm) in Real Environment,” *Int. J. Future Gener. Commun. Netw.*, vol. 10, pp. 23–36, Sep. 2017, doi: 10.14257/ijfgcn.2017.10.9.03.

[10] A. E. M. E. Ashry and B. I. Sheta, “Wi-Fi based indoor localization using trilateration and fingerprinting methods,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 610, no. 1, p. 012072, Sep. 2019, doi: 10.1088/1757-899X/610/1/012072.

[11] T.-M. T. Dinh, N.-S. Duong, and K. Sandrasegaran, “Smartphone-Based Indoor Positioning Using BLE iBeacon and Reliable Lightweight Fingerprint Map,” *IEEE Sens. J.*, vol. 20, no. 17, pp. 10283–10294, Sep. 2020, doi: 10.1109/JSEN.2020.2989411.

[12] A. Riady and G. P. Kusuma, “Indoor positioning system using hybrid method of fingerprinting and pedestrian dead reckoning,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 9, pp. 7101–7110, Oct. 2022, doi: 10.1016/j.jksuci.2021.09.005.

[13] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960, doi: 10.1115/1.3662552.

[14] B. Alsadik, “Chapter 10 - Kalman Filter,” in *Adjustment Models in 3D Geomatics and Computational Geophysics*, vol. 4, B. Alsadik, Ed., in Computational Geophysics, vol. 4. , Elsevier, 2019, pp. 299–326. doi: 10.1016/B978-0-12-817588-0.00010-6.

[15] M. George and K. Vadivukkarasi, “Kalman filtering for RSSI based localization system in wireless sensor networks,” vol. 10, pp. 16429–16440, Jan. 2015.

[16] A. Kargar-Barzi, E. Farahmand, A. Mahani, and M. Shafique, “CAE-CNNLoc: An Edge-based WiFi Fingerprinting Indoor Localization Using Convolutional Neural Network and Convolutional Auto-Encoder”.

[17] “Convolutional Autoencoders,” Paperspace Blog. Accessed: Sep. 17, 2023. [Online]. Available: https://blog.paperspace.com/convolutional-autoencoder/

[18] “EECS 206 Labs.” Accessed: Oct. 09, 2023. [Online]. Available: https://www.eecs.umich.edu/courses/eecs206/public/lab/