applied sciences

MDPI

Article

A REM Update Methodology Based on Clustering and Random Forest

Mario R. Camana, Carla E. Garcia, Taewoong Hwang and Insoo Koo *

Department of Electrical, Electronic and Computer Engineering, University of Ulsan, Ulsan 44610, Republic of Korea; mario_camana@hotmail.com (M.R.C.); carli.garcia27@hotmail.com (C.E.G.); yuio124@naver.com (T.H.)

* Correspondence: iskoo@ulsan.ac.kr

Abstract: In this paper, we propose a radio environment map (REM) update methodology based on clustering and machine learning for indoor coverage. We use real measurements collected by the TurtleBot3 mobile robot using the received signal strength indicator (RSSI) as a measure of link quality between transmitter and receiver. We propose a practical framework for timely updates to the REM for dynamic wireless communication environments where we need to deal with variations in physical element distributions, environmental factors, movements of people and devices, and so on. In the proposed approach, we first rely on a historical dataset from the area of interest, which is used to determine the number of clusters via the *K*-means algorithm. Next, we divide the samples from the historical dataset into clusters, and we train one random forest (RF) model with the corresponding historical data from each cluster. Then, when new data measurements are collected, these new samples are assigned to one cluster for a timely update of the RF model. Simulation results validate the superior performance of the proposed scheme, compared with several well-known ML algorithms and a baseline scheme without clustering.

Keywords: radio environment map (REM); random forest (RF); machine learning; clustering

Citation: Camana, M.R.; Garcia, C.E.; Hwang, T.; Koo, I. A REM Update Methodology Based on Clustering and Random Forest. *Appl. Sci.* **2023**, *13*, 5362. https://doi.org/10.3390/app13095362

Academic Editors: Yichuang Sun, Haeyoung Lee and Oluyomi Simpson

Received: 4 March 2023
Revised: 17 April 2023
Accepted: 23 April 2023
Published: 25 April 2023

1. Introduction

Recently, the dramatic increase in the number of wireless devices and the required data rates to satisfy QoS for users' applications have made it essential to guarantee high-quality communication links. In this regard, the multipath fading effect is one of the main considerations for wireless signals in indoor scenarios. The presence of obstacles such as walls, roofs, furniture, and so on leads to attenuation of the received signal and variations in the received power on the user side while impacting the coverage area of the wireless transmitter.

To solve this issue, a radio environment map (REM) is proposed to detect shadow areas with a poor quality signal. Construction of the REM is based on real measurements from the area of interest in order to characterize the behavior of the wireless environment by representing it as a temperature map [1]. The detection of shadow areas contributes to successful network planning and leads to an increase in the quality of communication for users. A REM can also provide useful information about the positions of wireless devices, interference levels, and other related information that can be used to improve the performance of the services by using resource allocation schemes [2]. In [3], the authors proposed construction of a REM based on machine learning (ML) methods, which showed better performance compared with conventional statistical models. A REM framework for IoT networks was presented in [1], where the authors considered ML algorithms to construct the REM. Their results proved the superior performance of ML learning models compared with conventional interpolation methods such as Kriging and nearest neighbor. However, the aforementioned work did not consider a methodology to update the REM in a timely manner.

One important aspect of REM management is the need for a timely update mechanism responding to wireless environment changes due to variations in physical element distributions, environmental factors, movements of people and devices, and so on. Updating radio maps can help network planners adjust the network configuration to compensate for these changes. Moreover, in indoor localization systems, updating radio maps can significantly improve the accuracy of localization, which is crucial for applications such as asset tracking and indoor navigation. On the other hand, clustering is considered a meaningful, energy-efficient technique because it contributes to reducing power consumption by organizing user nodes into groups denominated as clusters [4]. Clustering can be used in radio maps to group similar signal strength measurements into clusters based on their proximity to each other in physical space. This allows for a more efficient representation of the radio map since similar signal strength measurements can be processed together to create a more accurate estimate of signal strength in a particular area. Our objective is to update the REM in real time by using a combination of the *K*-means algorithm and a machine learning model. Specifically, the *K*-means algorithm constructs clusters and, for each cluster, we develop a ML model that is trained using actual data measurements specific to that cluster. As new data are collected, they are assigned to their respective cluster to update the corresponding ML model.

1.1. Related Work

For indoor localization systems, mobile crowd sensing has become a popular method for updating radio maps [5–12]. In [5–7], the authors propose an approach where mobile users generate reports at their current locations to update the radio maps. The radio map is based on a database, and the predicted location for a query point is obtained by using the nearest neighbors. The method to update the radio map is based on simply appending the new fingerprints into the database and removing those that are older than a certain period. In [8], the authors use an integrity check algorithm to determine whether to update the radio map used for indoor localization. In this method, several new fingerprints are accumulated before updating the received signal strength indicator (RSSI) value at a particular location. The update is performed by using the average value of the accumulated points to update the database.

In [9,10], the authors propose a method for adapting radio maps used for indoor localization to changes in the environment. The proposed method utilizes data gathered from typical wireless users' devices that were stationary at certain locations. RSSI values received from several reference locations are used to update the mapping of the RSSI value to a particular location. In [11], the decision of whether to update the radio map for each reference point is made by a periodic adaptive estimate algorithm. The authors represent the radio map using a matrix expression, and the update process involves updating the fingerprints in the radio map with the average of valid RSSI measurements collected from users. However, the success of the previous approaches heavily depends on the location and behavior of mobile users to gather new measurements, and their proposed algorithms require collecting new data for each location to update the predictive relationship between the RSS value and corresponding positions. In [12], the authors propose an updating method for signal maps based on Bayesian compressive sensing (BCS). Several crowdsourced samples are first mapped to the nearest reference point, and the BCS-based approach computes the signal change for each reference point based on the correlation between the new samples and reference points. In the aforementioned works, the radio map construction is based on collecting data and organizing it into a database, and the methodology to update the radio is focused on indoor localization approaches, where new measurements in each of the reference locations are required to correctly update the radio map. Unlike the previously mentioned methods, our proposed method only requires newly collected data in specific sectors to update a large area of interest. Additionally, the prediction of RSSI values is performed using powerful ML algorithms.

The authors in [13] propose a scheme for REM updates based on hypothesis testing, which is used to decide whether to update the REM each time. In [14], the authors propose a REM update mechanism based on Siamese neural networks to determine the level of similarity between an already-constructed REM and a new REM. However, previous work only considered a simple average of collected measurements to construct the REM. In [15], the authors propose a REM update scheme based on clustering and Gaussian process regression (GPR). They manually collected RSSI data with a smartphone in particular areas of the experiment room, and they use the collected dataset to predict the RSSI at specific reference points by using GPR with clustering. However, the objective of the GPR-based scheme is the RSSI prediction of only reference points to later be used for indoor localization, where a complete prediction of the whole area of interest is not obtained. Moreover, the authors only consider scenarios by varying the number of training samples and do not analyze the errors under changes in the wireless conditions such as in the presence of new obstacles or under the relocation of the AP. This is contrary to our proposed method, which is able to obtain the RSSI prediction of any point of the area of interest and represent it as a temperature map. Moreover, we extensively evaluate the proposed scheme by considering several comparative ML methods and several different scenarios, including the presence of obstacles and relocation of the AP.

1.2. Main Contributions

In this paper, we propose a REM update methodology based on clustering and ML algorithms. In particular, we consider the *K*-means algorithm to create *K* clusters in the area of interest, and the random forest (RF) algorithm is applied to predict the quality of the wireless signal for each cluster. The proposed REM update can be applied in different scenarios, such as technology industries, where several sensor nodes interact to carry out different tasks, such as maintenance or scheduled programming. Many times, it is difficult to collect measurements in the whole area to evaluate coverage prediction and build the REM because sensor nodes need to move frequently. Similarly, in hospitals, users storing measurements on every floor for coverage prediction may disturb other users, and measurement collecting tasks cannot be done as many people walk around the area. Therefore, the application of REM updating via clustering can reduce time-consuming tasks and optimize network resources. The main contributions of this paper are summarized as follows:

- We propose an efficient methodology to update a REM based on clustering and RF in a timely manner. In the proposed scheme, the *K*-means algorithm is applied to divide the area of interest in *K* clusters, where one RF model is deployed per cluster. The REM is constructed to cover every point within the area of interest, where the prediction of the RSSI values for each location is obtained by the corresponding RF model in each cluster.
- The RSSI measurements were collected by a mobile robot, which can reduce the risk of human error because the robot can be programmed to move in a controlled manner. This can help ensure that RSSI measurements are taken at consistent intervals and under consistent conditions, while improving the accuracy and reliability of the measurements. Moreover, mobile robots can operate autonomously, which can save time and resources compared to manual data collection methods.
- In the REM construction, to avoid abrupt changes in the border areas between the clusters, we propose a methodology that utilizes the weighted average of the RF model predictions from the two nearest centroids to determine the RSSI value of the points within the border areas. Moreover, when new measurements are available, only the RF models for clusters that have enough measurement samples are updated.
- We extensively evaluate the proposed scheme for different scenarios, including the presence of obstacles and relocating the AP, and we consider several comparative ML methods, including the case without clusters. Moreover, the computational complexity of the proposed scheme is analyzed along with the comparative schemes.

The simulation results demonstrated the superior performance of the proposed scheme compared to the baseline methods in effectively adapting to changes in the wireless environment. Moreover, the proposed approach requires only newly collected data in specific sectors to update a large area of interest.

The rest of this paper is organized as follows. Section 2 describes the measurement methodology. Section 3 presents the proposed REM updating framework. In Section 4, we present comparative models and an evaluation of simulation results. Finally, Section 5 concludes the paper.

2. Measurement Methodology

The scenario considered for indoor REM analysis was Room 302 in the Engineering Building at the University of Ulsan. The equipment used for the analysis included the TurtleBot3 [16] mobile robot, which is equipped with sensors, navigation systems, and decision-making algorithms that enable it to operate and complete tasks without human intervention. In particular, TurtleBot3 is composed of several parts, including an embedded controller, a light detection and ranging (LiDAR) sensor (laser distance sensor LDS-02), an inertial measurement unit (IMU) sensor, an encoder, a single board computer (SBC), and the robot operating system (ROS). A Raspberry Pi powers the SBC, which configures algorithms in a Linux environment and relies on ROS to enable communication between different processes. Meanwhile, the embedded controller, which utilizes OpenCR, is responsible for controlling the movement of the mobile robot through various sensors. The LiDAR sensor utilizes laser beams to calculate the distance to objects in its surroundings. By scanning the laser beams in a 360-degree horizontal field of view, LiDAR can generate a 2D point cloud map of the robot's surroundings. This map can be utilized for obstacle detection, mapping, and localization. Moreover, TurtleBot3 utilizes the odometry technique that employs sensor data from the encoders and the IMU to estimate the position and orientation of the robot in relation to its starting position. The odometry data are then combined with the LiDAR data to provide a more accurate and robust localization system, where the coordinates of the location points are presented in a 2D Cartesian coordinate system [17].

RSSI was utilized to estimate the link quality between the transmitter and receiver. RSSI is a measurement of the power present in a received radio signal, and it is commonly used in wireless communication systems like Wi-Fi, Bluetooth, and cellular networks to determine the signal strength received from a transmitter. The RSSI value is typically expressed in dBm (decibel-milliwatts) and reflects the power of the signal received by the receiver's antenna. A higher RSSI value indicates a stronger signal, while a lower RSSI value indicates a weaker signal. In our experiments, we used the built-in Wi-Fi module of the Raspberry Pi to collect RSSI data, along with the SSID, signal frequency, and link quality. The RSSI and location data were assembled in Ubuntu 22.04LTS through a shell script and synchronized based on timestamps.

The access point (AP) for the experiments was the IPTime N704M at 2.4 GHz. Figure 1 is a floor plan of the room used for the experiments, indicating the location of the AP. The floor plan presented in Figure 1 was obtained with the Hovermap HF1 [18], which is a mobile LiDAR 3D scanner, to map GPS-denied environments. Hovermap uses innovative simultaneous localization and mapping (SLAM) algorithms along with LiDAR data to produce 3D point clouds of the scanned area.

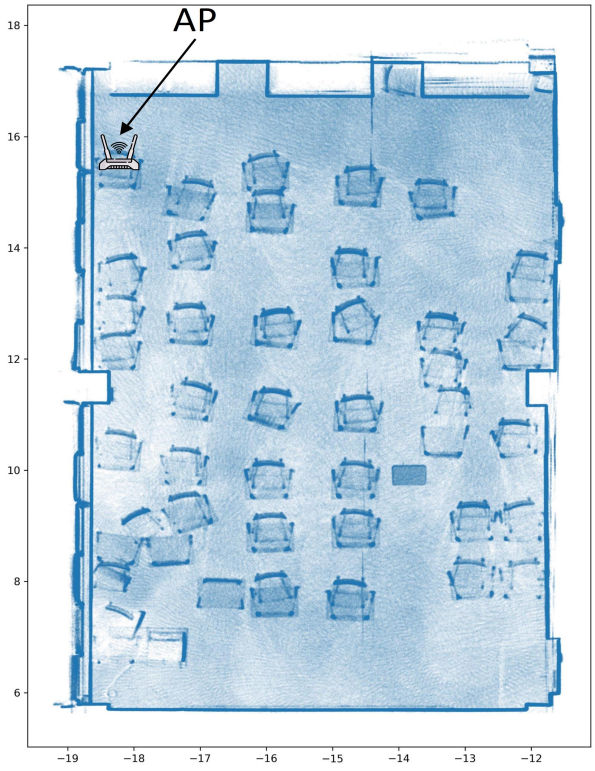


Figure 1. Floor plan of the room used for the experimental evaluations.

3. Proposed Approach for REM Updates

3.1. Overview

Figure 2 illustrates an overview of the proposed approach to REM updates based on clustering and the RF algorithm. First, assume the initial dataset is $D_H = \{\mathbf{z}_1, ..., \mathbf{z}_n, ..., \mathbf{z}_N\}$, where N is the total number of collected historical measurements and $\mathbf{z}_n = \{x_n, y_n, R_n\}$, in which x_n is the location in the x-axis, y_n is the location in the y-axis, and R_n is the RSSI value at position (x_n, y_n). The initial module of the proposed approach conducts clustering. In particular, we consider the K-means algorithm where N samples are separated into K groups. A detailed description of the K-means algorithm can be found in Section II of [19]. The K-means algorithm is applied to initial dataset D_H to obtain K cluster centers, which are used to assign each n-th sample to one of the K clusters based on the nearest centroid.

The second module, based on the RF algorithm, is where we train one RF model per cluster. Once each n-th measurement of dataset D_H has been assigned to a cluster, we have $D_k = \{\mathbf{z}_{1_k}, ..., \mathbf{z}_{m_k}, ..., \mathbf{z}_{M_k}\}$ at the k-th cluster with $k = 1, ..., K$ and $\mathbf{z}_{m_k} = \{x_{m_k}, y_{m_k}, R_{m_k}\}$. Then, K RF models are trained based on the corresponding dataset D_k. A detailed description of the RF algorithm is presented in Section 3.2.

Once we have the K-trained RF models, we construct a grid, $G = \{\mathbf{f}_1, ..., \mathbf{f}_h, ..., \mathbf{f}_H\}$, where H is the total number of samples in the grid, and $\mathbf{f}_h = \{x_h, y_h\}$. The grid is created to cover the whole area of interest and corresponds to the positions to be estimated by the RF algorithm. Next, each $\mathbf{f}_h$ is assigned to one cluster, and the corresponding RSSI value in that cluster is predicted by the RF model. Then, we construct the REM by using the location coordinates and by mapping the RSSI values to a specific color in a color map. A detailed description of REM construction is presented in Section 3.3.

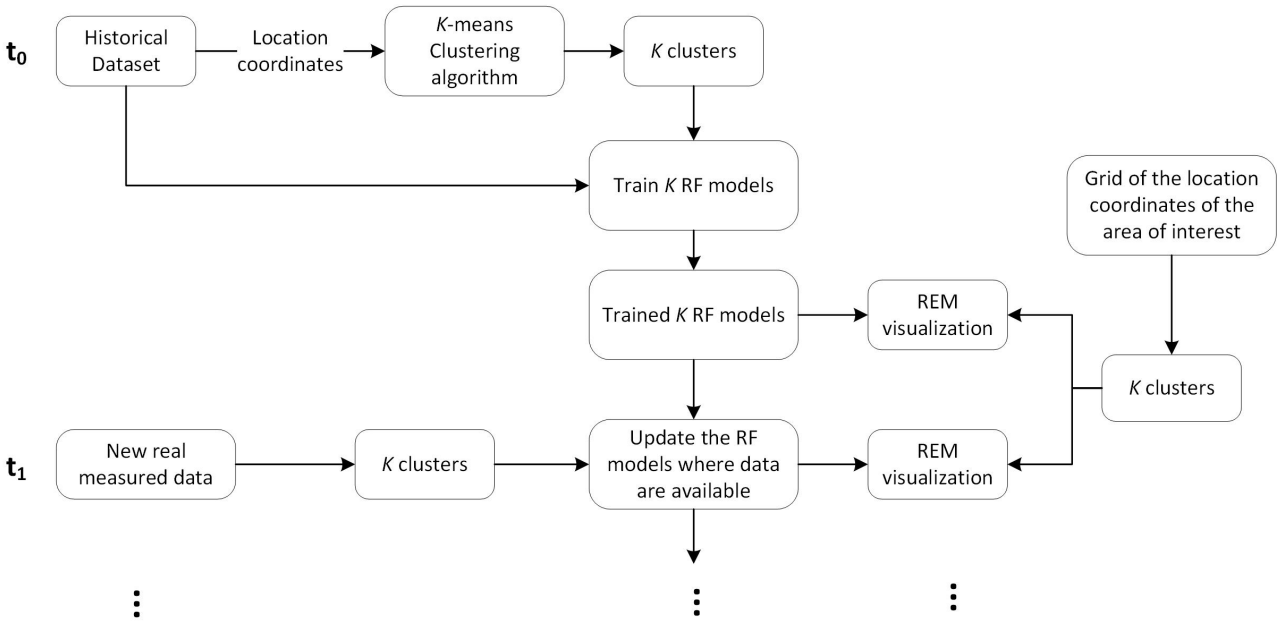


Figure 2. Proposed approach for REM update.

Next, we consider the REM update, denoted as $\mathbf{t}_1$ in Figure 2, based on newly collected measurements. Once we have trained the RF models based on the historical dataset, we apply the procedure to update the REM because the wireless environment constantly changes over time. For instance, in a smart warehouse, the positions of the products constantly change during the day. In general, the proposed scheme for the REM updates is based on training the RF models with newly collected data only in clusters with enough samples. We denote the newly collected dataset as $D_{\mathbf{t}_1} = \{\mathbf{z}_{1_{\mathbf{t}1}}, ..., \mathbf{z}_{n_{\mathbf{t}1}}, ..., \mathbf{z}_{N_{\mathbf{t}1}}\}$, where $N_{\mathbf{t}1}$ is the number of newly collected measurements at the time $\mathbf{t}_1$, and $\mathbf{z}_{n_{\mathbf{t}1}} = \{x_{n_{\mathbf{t}1}}, y_{n_{\mathbf{t}1}}, R_{n_{\mathbf{t}1}}\}$. First, we assign each $n_{\mathbf{t}1}$-th sample to one of the K clusters, which creates K possible datasets, each of them denoted $D_{k_{\mathbf{t}1}} = \left\{\mathbf{z}_1, ..., \mathbf{z}_{m_{k,\mathbf{t}1}}, ..., \mathbf{z}_{M_{k,\mathbf{t}1}}\right\}$. Since the newly collected measurements are not guaranteed to cover the whole area of interest, some datasets may contain no (or a very small number of) samples, which can lead to degradation when training the new RF model. Therefore, at the k-th cluster, to replace the k-th RF model with a new RF model trained with dataset $D_{k_{\mathbf{t}1}}$, we establish a condition calling for a minimum number of samples needed in the dataset to train a new k-th RF model: $\left|D_{k_{\mathbf{t}1}}\right| \geq N_{\min}$, where $N_{\min}$ is the minimum number of samples. If $\left|D_{k_{\mathbf{t}1}}\right| \geq N_{\min}$ is satisfied, the k-th RF model is trained based on the corresponding new dataset, $D_{k_{\mathbf{t}1}}$, replacing the old RF model in the k-th cluster. On the other hand, if dataset $D_{k_{\mathbf{t}1}}$ does not contain the minimum number of samples, $N_{\min}$, the previous k-th RF model is used to predict the points in the k-th cluster. Finally, we update the REM by using the current K RF models to predict the RSSI values of grid G.

3.2. Random Forest

An RF regressor is a type of ensemble learning method composed of W decision tree regressors, where the predicted RF value corresponds to the average value of the predictions for each independent decision tree regressor. Figure 3 shows the structure of a decision tree composed of a root node as the starting point, split nodes where a split rule is applied, and leaf nodes.

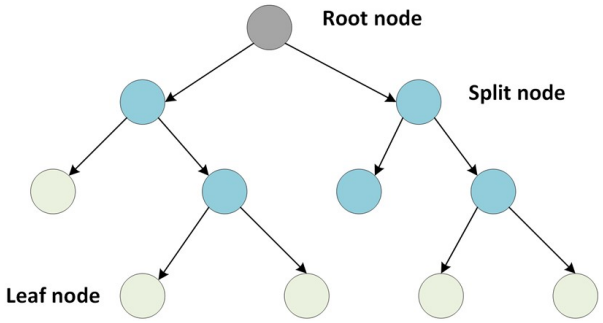


Figure 3. Example of a decision tree regressor.

We denote the training dataset of the RF algorithm as $D_{RF} = \{\mathbf{z}_1, ..., \mathbf{z}_m, ..., \mathbf{z}_M\}$, where M is the total number of training samples, and $\mathbf{z}_m = \{x_m, y_m, R_m\}$. The features correspond to position (x_m, y_m), and R_m is the target RSSI value at position (x_m, y_m). First, the RF algorithm creates W datasets, $\{D_{RF,w}\}$, from the original dataset, D_{RF}, with the bootstrap aggregation technique. Then, each w-th decision tree regressor is trained with corresponding dataset $D_{RF,w}$.

In each decision tree regressor, dataset $D_{RF,w}$ is used to select a split rule in each split node until reaching a leaf node. Let us define $D^b_{RF,w}$ as the subset of training samples at the b-th split node of the w-th decision tree regressor. Then, the best-split rule to be used at the b-th split node is determined with dataset $D^b_{RF,w}$. A candidate split rule is defined as follows:

$$s^{b,w}_{f,\alpha}(z_i) = \begin{cases} 1, & \text{if } f_{z_i} > \alpha \\ 0, & otherwise, \end{cases} \quad (1)$$

where f_{z_i} is the value of feature $f \in \{x_i, y_i\}$ in sample z_i, with $z_i \in D^b_{RF,w}$ during the training procedure and α representing a threshold. In the training process, at the b-th split node, a small pool of random features is selected, and a set of possible thresholds for each feature is evaluated to select the best-split rule based on the lowest mean squared error (MSE). Split rule (1) divides the samples in dataset $D^b_{RF,w}$ into two groups: $DR^b_{RF,w}$ contains the training samples satisfying the b-th split rule, and $DL^b_{RF,w}$ contains the rest. The MSE of each candidate split rule is evaluated as follows [3,20]:

$$\text{MSE}\left(s^{b,w}_{f,\alpha}\right) = \frac{1}{\left|DR^b_{RF,w}\right|}\sum_{i \in DR^b_{RF,w}}\left(R_i - \hat{R}^{DR}_b\right)^2 + \frac{1}{\left|DL^b_{RF,w}\right|}\sum_{i \in DL^b_{RF,w}}\left(R_i - \hat{R}^{DL}_b\right)^2, \quad (2)$$

where $\left|DR^b_{RF,w}\right|$ represents the number of samples in dataset $DR^b_{RF,w}$, $\left|DL^b_{RF,w}\right|$ represents the number of samples in dataset $DL^b_{RF,w}$, R_i is the true target value of the i-th sample, while $\hat{R}^{DR}_b$ and $\hat{R}^{DL}_b$ are the predicted values based on the average RSSI of the training samples in datasets $DR^b_{RF,w}$ and $DL^b_{RF,w}$, respectively, when the candidate split test, $s^{b,w}_{f,\alpha}$, is applied. Then, the candidate split test with the lowest MSE is selected as the best-split rule for the b-th split node. The aforementioned procedure is repeated in the next split node until reaching a leaf node, which is determined by the minimum number of training samples required to split a node, m_{min}. Finally, the RSSI value associated with the leaf node corresponds to the average of the RSSI values of the training samples in that leaf node.

Once the RF model is successfully trained, the test sample goes to each w-th decision tree regressor and evaluates the split rule at each split node to continue to the next split node. The process finishes when reaching the leaf node, where the associated value of that leaf node determines the predicted RSSI value for the sample in the w-th decision tree regressor. The final prediction of the RF model is the average of the RSSI values predicted by all the W decision tree regressors.

3.3. REM Construction

Given K cluster centers, the K-trained RF models, and grid G, we construct the REM for the area of interest. First, each $\mathbf{f}_h$ sample from grid G is assigned to one cluster among the K clusters available. Next, the border samples in the area of the intersection of the clusters are determined to avoid a hard change in the REM between clusters. Then, P neighbor samples are obtained for each sample in the border area based on Euclidean distance. Next, we group all the P neighbor samples of the border areas, and we remove duplicate points to create grid G_{border}. Finally, we remove the samples of G_{border} from original grid G to obtain grid G_{in}. Therefore, we have divided G into grids of the border areas, G_{border}, and a grid with the internal points in each cluster, G_{in}.

To obtain the RSSI value for the points of grid G_{in}, we assign each sample from G_{in} to one of the K clusters, and we use the corresponding RF model to predict the RSSI value. In the samples in grid G_{border}, we determine the two nearest cluster centers for each sample, and the predicted RSSI value is the weighted average of the corresponding two RF models. In detail, let us consider sample $\mathbf{f}_{h,border}$ from grid G_{border}. First, we evaluate the distance from $\mathbf{f}_{h,border}$ to all K cluster centers, selecting two clusters with the nearest Euclidean distance, denoted as $d_{\mathbf{f}_{h,boder},C_a}$ and $d_{\mathbf{f}_{h,boder},C_b}$, where $d_{\mathbf{f}_{h,boder},C_a} < d_{\mathbf{f}_{h,boder},C_b}$ and where C_a represents one of the K available clusters. Then, we use the RF model of cluster C_a to predict the first RSSI value of $\mathbf{f}_{h,border}$, denoted as $R_{h,border,C_a}$, and the RF model of cluster C_b predicts the second RSSI value of $\mathbf{f}_{h,border}$, denoted as $R_{h,border,C_b}$. Finally, the RSSI value for $\mathbf{f}_{h,border}$ is determined with the following weighted average:

$$R_{h,border} = \frac{\left(\frac{d_{\mathbf{f}_{h,boder},C_b}}{d_{\mathbf{f}_{h,boder},C_a}}\right)^5 R_{h,border,C_a} + \left(\frac{d_{\mathbf{f}_{h,boder},C_a}}{d_{\mathbf{f}_{h,boder},C_b}}\right)^5 R_{h,border,C_b}}{\left(\frac{d_{\mathbf{f}_{h,boder},C_b}}{d_{\mathbf{f}_{h,boder},C_a}}\right)^5 + \left(\frac{d_{\mathbf{f}_{h,boder},C_a}}{d_{\mathbf{f}_{h,boder},C_b}}\right)^5}. \quad (3)$$

Once all RSSI values are obtained for all samples in grids G_{in} and G_{border}, we use Matplotlib in Python to create the REM based on the color map. Moreover, we superpose the SLAM map of the room by carefully adjusting the opacity of the REM with the alpha parameter from Matplotlib.

4. Evaluation

4.1. Historical Dataset and Comparative Models

Initial dataset D_H is composed of measurements collected inside Room 302, illustrated in Figure 1, with a total of $N = 1160$ samples covering the whole room. We considered three different error metrics to evaluate the performance of the proposed scheme: mean absolute percentage error (MAPE), root mean square error (RMSE), and R2 score. As comparative schemes, we considered the support vector regression (SVR) algorithm [21], multilayer perceptron (MLP) [22], and the AdaBoost regressor [23]. Moreover, we obtained error results by considering different numbers of clusters. The RF model and the AdaBoost regressor were trained with 200 decision tree regressors; the comparative SVR algorithm considered the amount of regularization, $C = 1000$, and the radial basis function (RBF) kernel; and MLP had three hidden layers with 100 hidden units per layer from using a rectified linear unit (ReLU) activation function. The computer used for the simulations had an AMD Ryzen 9 5900X 12-Core processor and 48 GB of RAM.

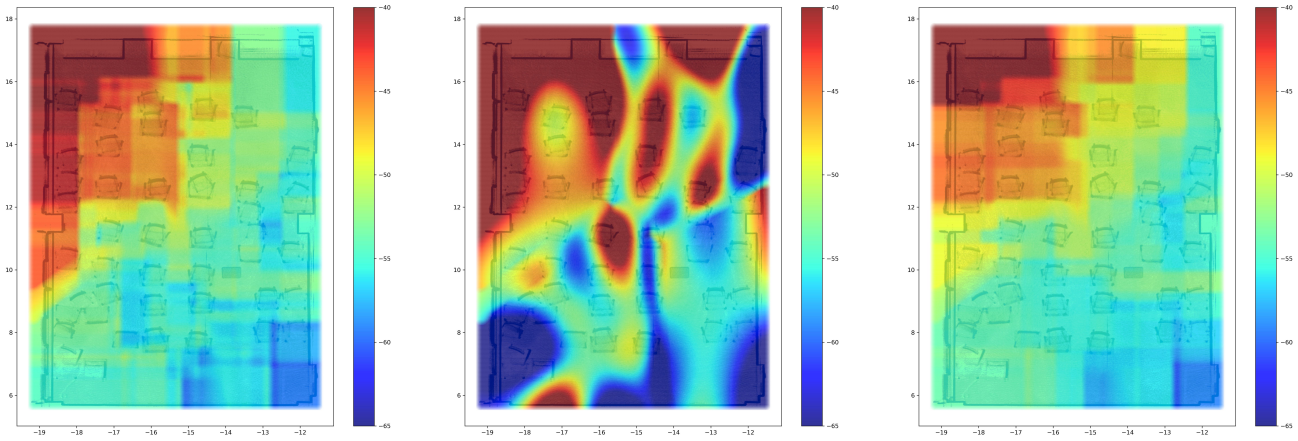
Table 1 presents the aforementioned error metrics of the proposed scheme and several comparative approaches by using 5-fold cross-validation with historical dataset D_H, where the presented results are averaged over several independent simulations. We can see that the RF-based scheme achieved the fewest errors among the comparative methods, and the AdaBoost algorithm was the second-best scheme. In addition, the impact from the number of clusters in the RF algorithm was very small since the same error rate can be obtained by using one cluster or four clusters. Note that Table 1 analyzes the error results on dataset D_H, where measurements for the whole area of interest are available. However,

in Section 4.2, we analyze the performance from different numbers of clusters in a realistic case when only partial data from the area of interest are available.

Table 1. Error evaluation of the dataset D_H.

MAPE				
Model for Prediction	**4 Clusters**	**3 Clusters**	**2 Clusters**	**1 Cluster**
RF	1.511%	1.513%	1.507%	1.515%
SVR	3.565%	3.718%	3.828%	4.504%
MLP	4.325%	4.174%	4.254%	4.835%
AdaBoost	3.001%	3.178%	3.603%	4.064%
RMSE				
Model for Prediction	**4 Clusters**	**3 Clusters**	**2 Clusters**	**1 Cluster**
RF	1.295	1.290	1.290	1.285
SVR	2.418	2.486	2.537	2.793
MLP	2.648	2.556	2.602	2.977
AdaBoost	1.810	1.912	2.150	2.409
R2 Score				
Model for Prediction	**4 Clusters**	**3 Clusters**	**2 Clusters**	**1 Cluster**
RF	0.948	0.949	0.949	0.949
SVR	0.819	0.809	0.801	0.760
MLP	0.783	0.799	0.791	0.727
AdaBoost	0.899	0.887	0.857	0.821

Figure 4 illustrates the REMs obtained for the historical dataset by using 4 clusters with the RF model, SVR, and AdaBoost. We can see that the REMs obtained with the RF algorithm and AdaBoost have similar patterns since both algorithms use the decision tree regressor as the base estimator. However, the REM obtained with RF is considered the most realistic because it has the lowest error, as we can observe in Table 1.



(**a**) (**b**) (**c**)

Figure 4. REM for the initial measurements by considering four clusters. (**a**) RF; (**b**) SVR; (**c**) AdaBoost.

4.2. REM Update Evaluation

In this subsection, we present the performance of the proposed scheme for REM updates. In particular, we considered a practical scenario where obstacles are added around the AP, which degrades coverage of the area of interest. Figure 5 shows the REM obtained by using RF in the ideal case where we can collect data measurements of the whole room after adding the obstacles. The data collected for this ideal case were used

as testing data to analyze the error in the proposed scheme compared with the baseline method. By comparing Figures 4a and 5, we can see that adding obstacles around the AP degraded the coverage, particularly in the quality of the received signal in the bottom area of the room, compared with no obstacles.

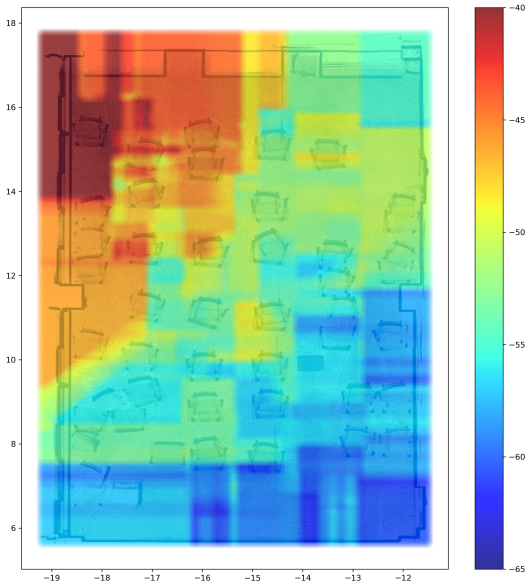
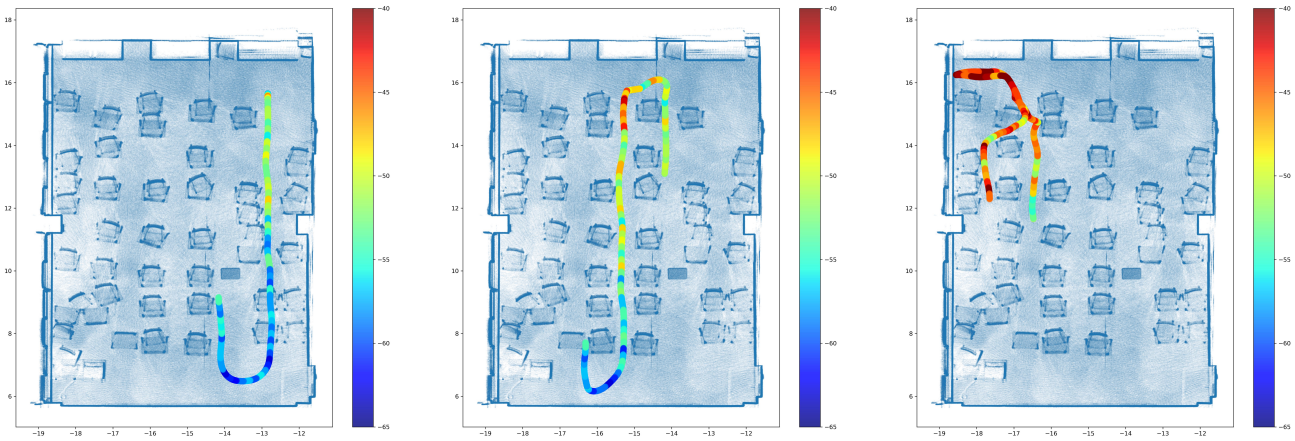


Figure 5. Target REM after changing the physical environment of the room by using RF.

Next, we analyzed a scenario where newly collected data were obtained only from a partial area of the room, which is a realistic assumption since it is not always possible to measure the whole area of interest each time. We considered three sets of newly collected data (at times $\mathbf{t}_1$, $\mathbf{t}_2$, and $\mathbf{t}_3$). It was assumed that during $\mathbf{t}_1$, $\mathbf{t}_2$, and $\mathbf{t}_3$, coverage by the Wi-Fi signal was stable in the whole room. Figure 6 shows the newly collected measurements at the three different times.

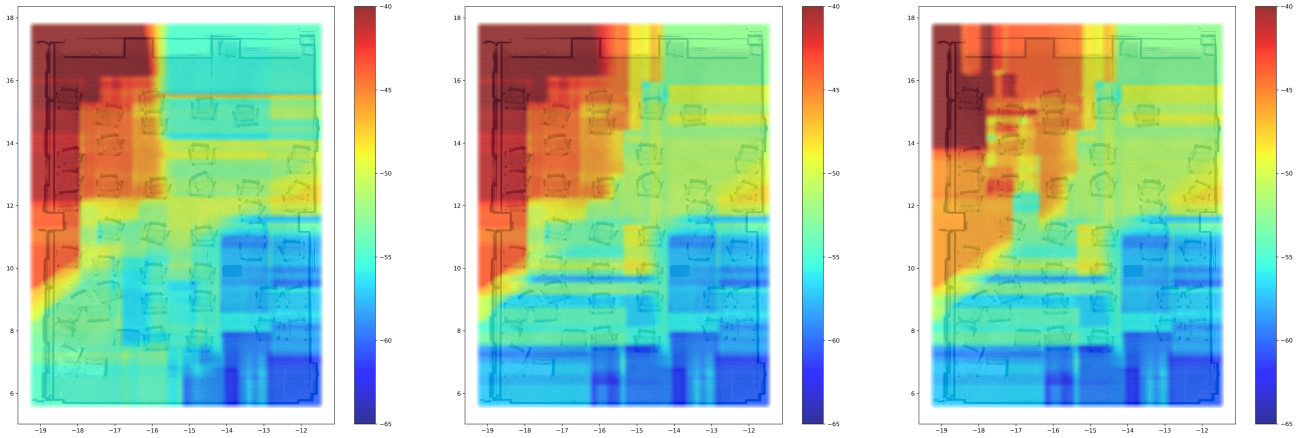


(**a**) (**b**) (**c**)

Figure 6. The three areas considered for newly collected measurements. (**a**) Data collected at time $\mathbf{t}_1$; (**b**) Data collected at time $\mathbf{t}_2$; (**c**) Data collected at time $\mathbf{t}_3$.

Figure 7 illustrates the proposed REM update mechanism for the three sets of newly collected data that followed the scheme in Figure 2. In detail, data collected at time $\mathbf{t}_1$ were used to train the RF model only for the corresponding clusters, while the remaining clusters used the previous RF model. We observe in Figure 7a that only the right area of the room was updated to the new scenario, which can be confirmed from Figure 5. On the other hand, the left part of the room was still predicted as having the historical measurements.

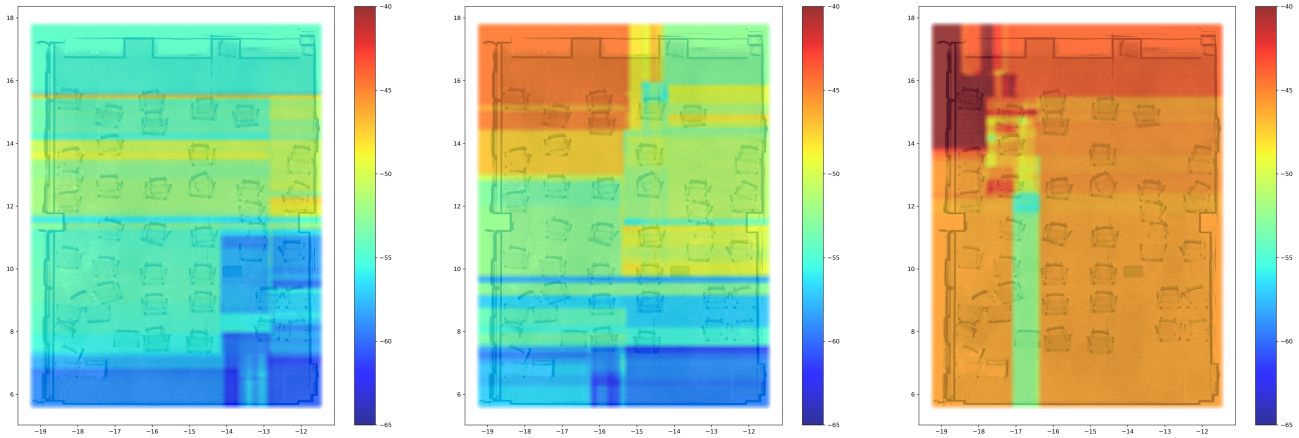
The reason is that, at time $\mathbf{t}_1$, there were no data available to update the left part of the REM. Next, at time $\mathbf{t}_2$, we could update the bottom area of the room while the upper left area was still not updated due to the lack of measurements in that area. Finally, at time $\mathbf{t}_3$, we updated the upper left area, leading to a complete update of the whole room, compared with Figure 5.



(**a**) (**b**) (**c**)

Figure 7. The proposed REM update mechanism considering four clusters. (**a**) The REM at time $\mathbf{t}_1$; (**b**) The REM at time $\mathbf{t}_2$; (**c**) The REM at time $\mathbf{t}_3$.

Figure 8 shows a baseline REM update mechanism without clustering for the 3 newly collected datasets. In this baseline scheme, the newly collected data were used to train an RF model for the whole room without using clustering. First, at time $\mathbf{t}_1$, the newly collected measurements were not enough to show the position of the AP; i.e., the upper left area was poorly predicted, leading to a high error. Next, at time $\mathbf{t}_2$, the collected measurements at the center of the room provided some small insight into coverage of the room, but the obtained REM is not ideal when we compare it with Figure 5. Finally, at time $\mathbf{t}_3$, the measurements collected in the area close to the AP make the RF model trained with those data overestimate the coverage of the room, leading to high error and an absence of shadow areas.



(**a**) (**b**) (**c**)

Figure 8. The baseline REM update mechanism without clustering. (**a**) The REM at time $\mathbf{t}_1$; (**b**) The REM at time $\mathbf{t}_2$; (**c**) The REM at time $\mathbf{t}_3$.

Table 2 shows error evaluations for the 3 times when data were newly collected. In the four clusters, we can see that the error lessened as time passed because new clusters could be updated as new measurement data were collected. As presented in Figure 7, at time $\mathbf{t}_3$, we could update the whole room, leading to a significant reduction in the error

metrics. In the case of the baseline method without clustering, the error was not reduced as time passed, and the errors obtained at each time significantly depended on the area where the new data were collected. For instance, we observe that the baseline method without clustering obtained the lowest error at time $\mathbf{t}_2$ for most of the compared methods because the data were collected in the middle of the room, which can provide a small insight into the coverage of the area of interest. The above-mentioned observation is consistent with the results shown in Figure 8 where, among the 3 time cases, the REM at time $\mathbf{t}_2$ is the closest representation to the target REM illustrated in Figure 5.

Table 2. Error evaluations from newly collected measurements.

MAPE	**4 Clusters**			**Without Clustering**		
Model for Prediction	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
RF	6.10%	5.03%	1.92%	10.52%	6.60%	12.40%
SVR	7.90%	6.01%	4.90%	15.89%	8.74%	15.54%
MLP	12.98%	5.76%	4.55%	27.27%	18.56%	7.97%
AdaBoost	6.40%	5.49%	3.82%	13.37%	6.78%	9.03%
RMSE	**4 Clusters**			**Without Clustering**		
Model for Prediction	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
RF	3.731	3.401	2.069	6.100	4.019	8.167
SVR	4.952	3.823	3.439	8.996	5.174	9.514
MLP	9.959	3.538	3.018	15.811	11.235	5.193
AdaBoost	3.952	3.422	2.527	7.513	3.960	6.064
R2 Score	**4 Clusters**			**Without Clustering**		
Model for prediction	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
RF	0.646	0.735	0.842	0.055	0.630	−1.463
SVR	0.563	0.598	0.440	−1.056	0.387	−2.342
MLP	−1.520	0.714	0.664	−5.351	-1.888	0.004
AdaBoost	0.603	0.732	0.764	−0.434	0.641	−0.358

Table 3 presents the computational time for the training and prediction phases of the historical dataset and the newly collected data using the proposed approach and the baseline methods. The computational time of the *K*-means algorithm to select the best centroids based on the historical dataset D_H (consisting of 1160 samples) is 0.097 s, and the computational time to predict the nearest cluster for the total number of samples is 0.013 s. The number of samples in the newly collected datasets at times $\mathbf{t}_1$, $\mathbf{t}_2$, and $\mathbf{t}_3$ are 1800, 1800, and 2400, respectively. In Table 3, we observe that the computational time of the historical dataset is higher than the time when updating the model at time $\mathbf{t}_1$ because the historical dataset is used to train all the ML models for each cluster, while the data at time $\mathbf{t}_1$ are only used to update the ML model of the corresponding cluster. Moreover, we see that the computational time in the training phase increased as the number of samples increased because, in general, more samples mean more computations and larger memory requirements, leading to longer processing times. The higher computational time of the RF with four clusters compared to the RF without clustering is due to the need to split the samples into their respective clusters and the requirement of training four different instances of RF.

In the grid prediction phase, the computational time corresponds to the time to split the samples of the grid into their respective clusters and the time to predict the total number of points in the grid with their respective ML algorithm. For instance, in the case of RF with 4 clusters and a 100 × 100 grid, a total of 10,000 samples were assigned to their respective clusters, and the corresponding RF model predicted the RSSI value for each sample, resulting in a total time of 0.112 s. Moreover, we see that the proposed RF algorithm

achieved the second lowest computational time among the compared ML methods with clustering. Note that MLP has the lowest computational time for the grid prediction with clustering but also has the highest error in the prediction as presented in Table 2.

Table 3. Computational time for the training and prediction phases.

Training Time (s)				
Model for Prediction	**Historical Data D_H**	**Data at t_1**	**Data at t_2**	**Data at t_3**
RF 4 clusters	0.539	0.366	0.375	0.418
RF without clusters	0.213	0.215	0.234	0.341
SVR 4 clusters	0.111	0.205	0.231	0.374
MLP 4 clusters	1.14	0.669	1.48	1.13
AdaBoost 4 clusters	0.502	0.234	0.369	0.301
Grid Prediction Time (s)				
Model for Prediction	**100 × 100 grid**		**300 × 300 grid**	
RF 4 clusters	0.112		0.285	
RF without clusters	0.04		0.251	
SVR 4 clusters	0.195		1.25	
MLP 4 clusters	0.038		0.218	
AdaBoost 4 clusters	0.134		0.625	

Next, we consider a second scenario by moving the location of the AP. In particular, the AP is relocated to the other corner of the room, which significantly changes the coverage condition in the area of interest. Figure 9 illustrates the REM that was obtained with RF under ideal conditions when data measurements for the entire room were collected after the relocation of the AP. The colormap limits for the current axes have been adjusted for better data visualization. By comparing Figures 4a and 9, we can observe the significant changes in the wireless conditions of the area of interest and the importance of timely REM updates.

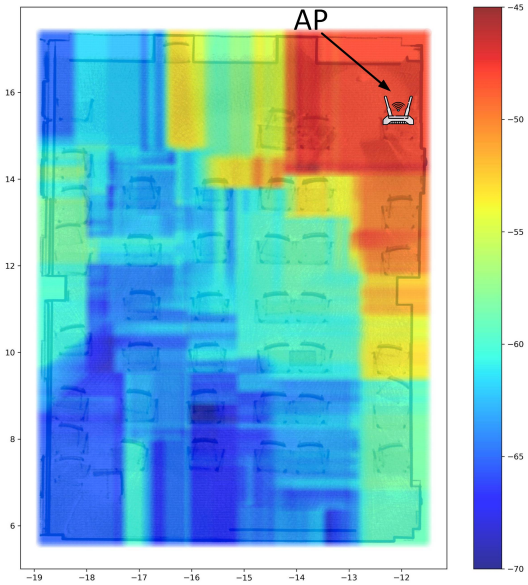
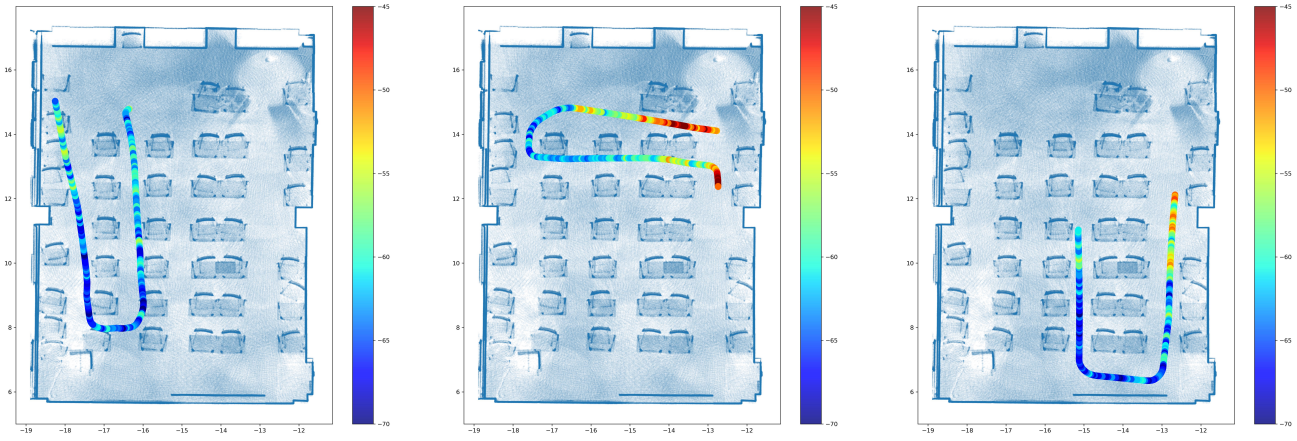


Figure 9. Target REM after the relocation of the AP by using RF.

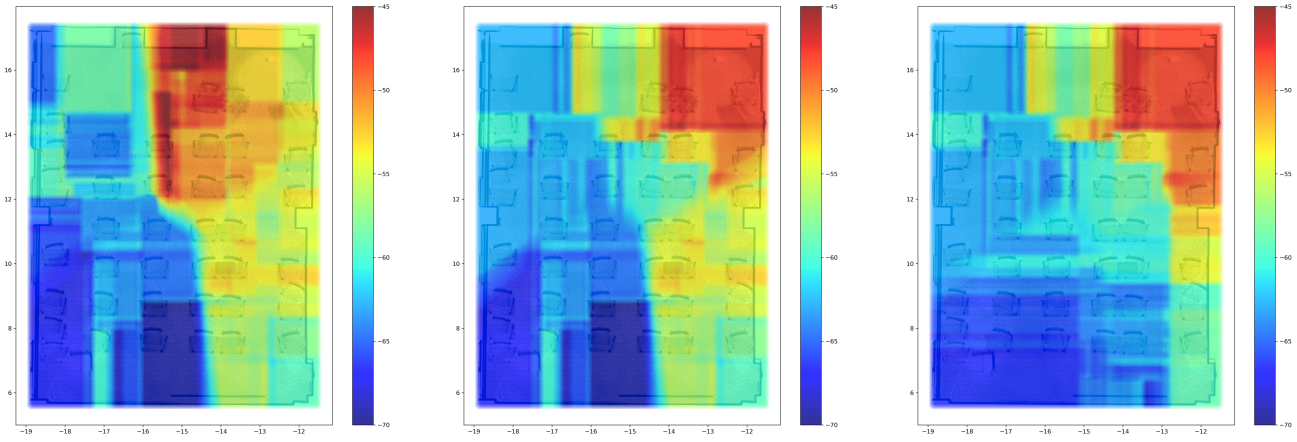
Similar to the previously considered scenario, new data were acquired at different times as illustrated in Figure 10. Then, the proposed update methodology was performed based on the three datasets collected at times $\mathbf{t}_1$, $\mathbf{t}_2$ and $\mathbf{t}_3$. Figure 11 shows the results of the proposed update methodology for the three sets of newly collected data. In Figure 11a,

we can see that only the left area of the room was updated to the actual wireless coverage condition because the newly collected data at time $\mathbf{t}_1$ do not have information about the remaining areas of the room. Next, by using the newly collected data at time $\mathbf{t}_2$, the upper area of the room was updated, while the bottom right area remained unaltered because of insufficient measurements. Finally, the entire room was updated with the newly collected data at time $\mathbf{t}_3$, as evidenced by comparing it to Figure 9.



(**a**) (**b**) (**c**)

Figure 10. The three areas considered for newly collected measurements after the relocation of the AP. (**a**) Data collected at time $\mathbf{t}_1$; (**b**) Data collected at time $\mathbf{t}_2$; (**c**) Data collected at time $\mathbf{t}_3$.



(**a**) (**b**) (**c**)

Figure 11. REMs obtained with the proposed update mechanism after the relocation of the AP. (**a**) The REM at time $\mathbf{t}_1$; (**b**) The REM at time $\mathbf{t}_2$; (**c**) The REM at time $\mathbf{t}_3$.

Table 4 presents the error evaluation for the newly collected data after the relocation of the AP at three different times by using the RF algorithm. Similar to the results obtained in Table 2, the error in the prediction decreased as time passed because new clusters could be updated as new measurement data were collected. For instance, the low error observed at time $\mathbf{t}_3$ is in concordance with the REM presented in Figure 11c, which is very similar to the ideal REM of Figure 9. In the case of the baseline scheme without clustering, the error remains consistent over time, and the errors at each time are greatly influenced by the location of the newly collected data.

Table 4. Error evaluations from newly collected measurements after the relocation of the AP.

	4 Clusters			Without Clustering		
Error metric	$\mathbf{t_1}$	$\mathbf{t_2}$	$\mathbf{t_3}$	$\mathbf{t_1}$	$\mathbf{t_2}$	$\mathbf{t_3}$
MAPE	7.11%	4.24%	2.19%	8.61%	7.45%	5.96%
RMSE	5.806	4.018	2.346	6.317	6.422	4.749
R2 score	0.044	0.255	0.826	−0.132	−0.902	0.288

Recently, reconfigurable intelligent surfaces (RIS) have been explored as a potential solution to enable a smart radio environment that can be dynamically configured using software. RIS is a type of metasurface that can be used to manipulate radio waves in order to control wireless communication [24]. An RIS is essentially a two-dimensional array of small, controllable elements that can adjust the phase and amplitude of incident electromagnetic waves to create a desired wavefront. Using RIS can enable the creation of smart environments that can adapt to changing electromagnetic conditions in real time. Field-programmable gate array can be used to control the operation of the RIS, or it can be manually controlled, as proposed in [25], by using touch controls. Therefore, analyzing the information in the REM can enable the optimization of the placement and reflection properties of an RIS to maximize the performance of wireless communication systems. Moreover, the REM can facilitate the dynamic adjustment of the reflection coefficients of RIS elements to adapt to changes in the wireless environment. For example, a REM can be used to identify areas with poor wireless coverage, such as areas with high levels of interference or signal attenuation. Then, an RIS can be strategically deployed in these areas to improve wireless coverage by reflecting and redirecting signals in the desired direction. Moreover, by using the information contained in a REM, we can optimize the design of the reflecting elements in the RIS to avoid interference with other wireless signals in the environment.

5. Conclusions

In this paper, we proposed a REM update methodology based on clustering and the RF algorithm. The proposed approach divides the area of interest into several clusters by using the *K*-means algorithm, and it trains one RF model per cluster based on real data measurements assigned to the corresponding clusters. A mobile robot was used to collect the RSSI measurements, which can reduce the risk of human error while improving the accuracy and reliability of the measurements. Next, only the RF models for clusters with enough measurement samples were updated when newly collected measurements became available. As time passed, the proposed scheme could update the whole REM by sector while reducing the error each time. Simulation results proved the superior performance of the proposed scheme compared to several well-known ML models, as well as the conventional case without clustering, in various scenarios, including the presence of obstacles and AP relocation. Subsequently, the proposed framework will be very useful for REM management in wireless scenarios where the physical element distribution constantly changes. For future research directions, an exciting topic is the utilization of information from the REM to optimize the allocation of resources in wireless networks, including spectrum, power, and antennas. This can improve overall system performance by leveraging the knowledge of the wireless propagation environment provided by the REM. For instance, the development of schemes to optimize the placement and the reflection properties of an RIS based on the information contained in a REM can help achieve desired wireless communication performance.

Author Contributions: Conceptualization, M.R.C., C.E.G. and I.K.; methodology, M.R.C.; software, M.R.C., C.E.G. and T.H.; validation, C.E.G., T.H. and I.K.; formal analysis, M.R.C.; investigation, M.R.C. and C.E.G.; resources, T.H. and I.K.; data curation, M.R.C. and T.H.; writing—original draft preparation, M.R.C.; writing—review and editing, C.E.G., T.H. and I.K.; visualization, M.R.C. and T.H.; supervision, I.K.; project administration, I.K.; funding acquisition, I.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Research Foundation of Korea (NRF) through the Korean Government's Ministry of Science and ICT (MSIT) under Grant NRF-2021R1A2B5 B01001721, and in part by the Regional Innovation Strategy (RIS) through the NRF funded by the Ministry of Education (MOE) under Grant 2021RIS-003.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chou, S.-F.; Yen, H.-W.; Pang, A.-C. A REM-Enabled Diagnostic Framework in Cellular-Based IoT Networks. *IEEE Internet Things J.* **2019**, *6*, 5273–5284. [CrossRef]
2. Bi, S.; Lyu, J.; Ding, Z.; Zhang, R. Engineering Radio Maps for Wireless Resource Management. *IEEE Wirel. Commun.* **2019**, *26*, 133–141. [CrossRef]
3. Garcia, C.E.; Camana, M.R.; Koo, I. Prediction of Digital Terrestrial Television Coverage Using Machine Learning Regression. *IEEE Trans. Broadcast.* **2019**, *65*, 702–712.
4. Han, T.; Bozorgi, S.; Orang, A.; Hosseinabadi, A.; Sangaiah, A.; Chen, M.-Y. A Hybrid Unequal Clustering Based on Density with Energy Conservation in Wireless Nodes. *Sustainability* **2019**, *11*, 746. [CrossRef]
5. Gallagher, T.; Li, B.; Dempster, A.G.; Rizos, C. Database updating through user feedback in fingerprint-based Wi-Fi location systems. In Proceedings of the 2010 Ubiquitous Positioning Indoor Navigation and Location Based Service, Kirkkonummi, Finland, 14–15 October 2010.
6. Lim, J.-S.; Jang, W.-H.; Yoon, G.-W.; Han, D.-S. Radio Map Update Automation for WiFi Positioning Systems. *IEEE Commun. Lett.* **2013**, *17*, 693–696. [CrossRef]
7. Luo, C.; Hong, H.; Chan, M.C.; Li, J.; Zhang, X.; Ming, Z. MPiLoc: Self-Calibrating Multi-Floor Indoor Localization Exploiting Participatory Sensing. *IEEE Trans. Mob. Comput.* **2018**, *17*, 141–154. [CrossRef]
8. Liu, X.; Cen, J.; Zhan, Y.; Tang, C. An Adaptive Fingerprint Database Updating Method for Room Localization. *IEEE Access* **2019**, *7*, 42626–42638. [CrossRef]
9. Wu, C.; Yang, Z.; Xiao, C. Automatic Radio Map Adaptation for Indoor Localization Using Smartphones. *IEEE Trans. Mob. Comput.* **2018**, *17*, 517–528. [CrossRef]
10. Wu, C.; Yang, Z.; Xiao, C.; Yang, C.; Liu, Y.; Liu, M. Static power of mobile devices: Self-updating radio maps for wireless indoor localization. In Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM), Hong Kong, China, 26 April–1 May 2015.
11. Liu, X.; Cen, J.; Hu, H.; Yu, Z.; Huang, Y.; A radio map self-updating algorithm based on mobile crowd sensing. *J. Netw. Comput. Appl.* **2021**, *194*, 103225. [CrossRef]
12. Yang, B.; He, S.; Chan, S.-H.G. Updating Wireless Signal Map with Bayesian Compressive Sensing. In Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, Malta, 13–17 November 2016.
13. Katagiri, K.; Fujii, T. Radio Environment Map Updating Procedure Considering Change of Surrounding Environment. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Seoul, Republic of Korea, 6–9 April 2020.
14. Zhen, P.; Zhang B.; Xie, C.; Guo, D. A Radio Environment Map Updating Mechanism Based on an Attention Mechanism and Siamese Neural Networks. *Sensors* **2022**, *22*, 6797. [CrossRef] [PubMed]
15. Zhao, J.; Gao, X.; Wang, X.; Li, C.; Song, M.; Sun, Q. An Efficient Radio Map Updating Algorithm based on K-Means and Gaussian Process Regression. *J. Navig.* **2018**, *71*, 1055–1068. [CrossRef]
16. TurtleBot3 Manual, Open-Source Robotics Foundation, Mountain View, CA, USA. Available online: https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/ (accessed on 20 February 2023).
17. Khan, M.U.; Zaidi, S.A.A.; Ishtiaq, A.; Bukhari, S.U.R.; Samer, S.; Farman, A. A Comparative Survey of LiDAR-SLAM and LiDAR based Sensor Technologies. In Proceedings of the 2021 Mohammad Ali Jinnah University International Conference on Computing (MAJICC), Karachi, Pakistan, 15–17 July 2021.
18. *Hovermap Mapping and Autonomy Payload User Manual*; Emesent (PTY) LTD: Milton, QLD , Australia, April 2021.

19. Na, S.; Xumin, L.; Yong, G. Research on k-means Clustering Algorithm: An Improved k-means Clustering Algorithm. In Proceedings of the 2010 Third International Symposium on Intelligent Information Technology and Security Informatics, Jian, China, 2–4 April 2010.
20. Phan, H.; Maaß, M.; Mazu, R.; Mertins, A. Random regression forests for acoustic event detection and classification. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2015**, *23*, 20–31. [CrossRef]
21. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222. [CrossRef]
22. Aggarwal, C.C. *Neural Networks and Deep Learning: A Textbook*, 1st ed.; Springer: Cham, Switzerland, 2018.
23. Drucker, H. Improving regressors using boosting techniques. In Proceedings of the Fourteenth International Conference on Machine Learning, San Francisco, CA, USA, 8–12 July 1997.
24. Dajer, M.; Ma, Z.; Piazzi, L.; Prasad, N.; Qi, X.-F.; Sheen, B.; Yang, J.; Yue, G. Reconfigurable intelligent surface: Design the channel—A new opportunity for future wireless networks. *Digit. Commun. Netw.* **2022**, *8*, 87–104. [CrossRef]
25. Chen, L.; Ma, Q.; Luo, S.S.; Ye, F.J.; Cui, H.Y.; Cui, T.J. Touch-Programmable Metasurface for Various Electromagnetic Manipulations and Encryptions. *Small* **2022**, *18*, 2203871. [CrossRef] [PubMed]