Radio Environment Maps through Spatial Interpolation: A Web-based Approach

A. Almazi Bipon, Md Shantanu Islam, A. Taufiq Asyhari, Adel Aneiba and Raouf Abozariba
College of Computing, Birmingham City University, United Kingdom
{abrarjahin.almazibipon, mdshantanu.islam, taufiq.asyhari, adel.aneiba, raouf.abozariba}@bcu.ac.uk

***Abstract*—The 5G era has seen the largest number of studies around Radio Environment Maps (REM) than in the previous three generations combined. Visualization of network coverage on interactive maps provides contextual information and numerous benefits to operators, regulators and to the public. In this context, spatial interpolation and extrapolation techniques are used to add synthetic data points between measurements to fill gaps in the data, where techniques such as machine learning, Ordinary Kriging (OK) and Inverse Distance Weighted (IDW) are used to enhance the quality of REMs. In this paper we present a state-of-the-art software package, which integrates a series of interpolation methods, augmented with polygon intersection queries functionality to control data used for estimation of coverage on the roads. The proposed web-based application is powered by a set of modular Python packages, making it future-proof and real-world ready, enabling efficient and precise network management.**

***Index Terms*—Interpolation, Network Analysis, REM software**

I. Introduction

Once were a solution for cognitive radio systems, Radio Environment Maps (REM) are now becoming indispensable not only for mobile operators as a tool for network optimization but for the telecommunications industry and spectrum regulators too [1]. A growing number of use cases now depend on reliable connectivity information in cities and rural areas, while governmental agencies use REMs to make informed decisions over spectrum policies and digital equality [2]. A recent application is mission planning for autonomous vehicles that require network connectivity, where coverage maps assist in minimizing the time and distance traversed without connectivity [3]. The contextual information provided by REMs can also aid in reducing the control signaling necessary for channel estimation, as shown in a platooning use case in [4]. Constructing direct radio maps can be achieved through displaying georeferenced data, acquired by deploying massive number of spatially dispersed sensor nodes, using crowdsourcing techniques such as MDT or through drive test campaigns [5], [6]. These data collection methods are known to present fundamental limitations, related to data privacy and costs, leading to an increased reliance on spatial interpolation methods to address sample data shortages, recovering the missing values to build local and global REMs. Machine learning-based techniques such as Inverse Distance Weighting (IDW), Nearest Neighbor (NN), Random Forest (RF) and Ordinary Krigging (OK) based interpolation approaches were all investigated in the literature under a variety of settings to improve the quality of REMs [7]–[10]. More advanced algorithms were proposed in the literature to further enhance the interpolation accuracy and to address the drawbacks of baseline methods, as shown in [11] and [12]. Other examples include [13], which uses feedforward neural network (FFNN) to improve the accuracy of Kriging in macro cells using data obtained through software defined radios (SDRs).

Despite the vast range of solutions, currently there is no conclusive evidence pointing to a single solution being universally superior across all network technologies, data types, cell sizes and interpolation tasks, as each technique has its own set of advantages and disadvantages [1]. Applying one blanket interpolation technique to large sets of data spread over large areas simply does not work in all cases and can lead to weak prediction performance and high computation time. For instance, IDW, which is based on weighted distance averages, was shown to work best for evenly distributed data points and is usually favored for its low computational complexity. However, it is only applicable for microcell environments, in the range of less than two kilometers wide [12]. On the other hand, although OK estimates both path loss and shadowing with acceptable accuracy, it requires estimating variogram models and solving large systems of equations, which can be computationally intensive, especially with a high ratio of sample data (number of traces/road lengths) [1], [4], [8]. Furthermore, the performance of OK in the presence of noisy data is quite weak, which limits its application in practice. Other techniques, such as Nearest Neighbor and bilinear interpolation, while they offer low complexity, lead to sharp transitions between the individual signal level zones, especially at the boundary of a given area [2].

Furthermore, the majority of the solutions in the literature were investigated under restricted data environments and scarcity conditions, considering relatively uniform sampling. For example, a single straight segment of a freeway was considered in [14], where synthetic dataset was used for training and tested over propagation data simulated in two distinct scenarios in Milan, Italy. The work in [8] on the other hand used walking measurement from a 1.0×1.3km area, while the results in [7] were based on deterministic training data modeled on a 200×200m grid.

To the best of our knowledge, there are no published interpolation tools, specifically designed to meet the requirements of cellular network management. Motivated by these observations and by the quest to combine tools into one application, our goal is to develop a software package capable of leveraging

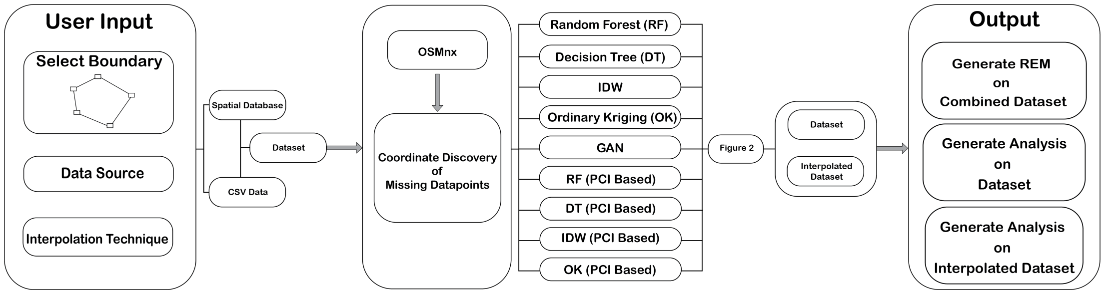

Fig. 1. System design and end-to-end workflow implementing customized polygon based area selection, a range of interpolation techniques, and REM.

a host of proven interpolation methods across different tasks that is more scalable and responsive, while enabling qualitative and quantitative comparisons of interpolators, allowing users to choose the best possible solver. We have made the source code available at https://github.com/nealmegh/srems2, hoping it's availability will foster more interest in further research.

The plan of the paper is as follows. Section II provides an overview of the application, focusing on how we enabled advanced capabilities. Section III highlights the challenges faced and how they were overcome. In Section IV, we provide a brief demonstration of the application and share practical insights. Section V concludes the paper and outlines future work.

II. High-level Architecture

We developed a Python-based web application to accelerate the visualization of REM, using a series of spatial interpolation methods built using the Django framework. The application interacts well with standard sklearn and TensorFlow functions to tightly integrate ML-based interpolation methods, which are increasingly popular in solving complex interpolation tasks. The application also incorporates statistical-based methods such as OK and deterministic algorithms such as IDW. A breakdown of the main components of the application is detailed below.

A. Data management:

The application is designed to handle large datasets, consisting of a series of essential radio frequency descriptors such as latitude, longitude, Reference Signal Received Power (RSRP), Physical Cell ID (PCI), Signal-to-Noise-Ratio (SNR) and 4G/5G bands, including new radio (NR) frequencies. Initially, we used MySQL, a relational database management system, due to its capabilities of fast execution and efficient large dataset handling [15]. However, MySQL's lack of advanced spatial querying capabilities led to inaccuracies in the query results, particularly for datasets requiring polygon-based queries, which aim to accumulate data available only inside the specified region. To overcome this issue, we migrated to PostgreSQL 16.1 and installed the PostGIS extension at the local storage layer, which adds support for geographic objects. PostgreSQL, a popular database management system for spatial data storage, is integrated with dynamic indexing mechanisms, such as R-trees, optimized for efficient querying of spatial data. While migrating the data to PostgreSQL, we created an extra column, named *location*, derived from the latitude and longitude pairs, providing a point object representing a single location in a coordinate system. We utilized *django.contrib.gis.geos* module in Django's Geographic Information Systems (GIS) framework to create the object. This column entry converts the dataset to a universally recognized geospatial format, enhancing its usability to process spatial queries and operations like area size calculation and proximity queries.

B. User input:

Spatial interpolation techniques provide the missing dataset derived from a subset of available measurements to reconstruct the REM of a given area [16]. It is paramount that the data used for training and calibrating interpolation tools be from within the relevant geographic area to minimize noise in the training data [17]. On the other hand, performing global interpolation on large areas with global datasets can demand substantial processing resources. To address this issue, we developed an interactive interface that enables users to specify geographical boundaries within their area of interest through a javascript library for interactive maps called Leaflet JS [18] and Leaflet Draw, an extension of the map library. Leaflet Draw enables users to outline polygons, selecting customized boundaries on a map, confining their targeted area for REM construction, minimizing processing time. The interface further enables users to select any interpolation method from the list of available options in the application, along with data source, operator, or network generation. The application triggers an event using *Draw.event* to capture the coordinates of the drawn polygon to create a shape. It then performs a query for data within this shape, utilizing the spatial functionalities as outlined in Fig. 1. The data retrieved from the query is subsequently utilized for further REM processing.

C. Interpolation process:

Once the network dataset is filtered by spatial query, the application identifies the roads and captures a list of coordinates within the polygon in 20-meter intervals along the roads that are not included in the source data utilizing a Python library, OSMnx [19]. To realize this, we used two different methods: *graph from polygon* and *graph to gdfs*. The first method requires a geometric shape, which, in our case, is a polygon

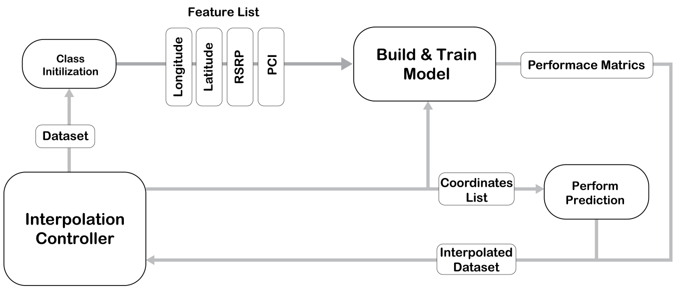

Fig. 2. Interpolation controller structure for building, training and prediction.

```
distances = self.calculate_distances(self.df['latitude'].values, self.df['longitude'].values, target_coord)

distances = [1 if d == 0 else d for d in distances]

weights = [1 / (d ** self.power) for d in distances]
weighted_values = [self.df['signalStrength'].iloc[i] * weights[i] for i in range(len(weights))]
```

Fig. 3. Implemented IDW snippet, where distance (d) is calculated through Euclidian formula, implemented in Python.

and a network-type attribute. OSMnx allows different levels of road networks to be analyzed, ranging from pedestrian pathways to drivable roads. We used *drive* as a default network type, focusing on street-level analysis. This method generates a directed graph consisting of the street network within the defined polygon area, where each node in the graph represents a road intersection. The second method converts the graph to GeoDataFrame contains the edge data, representing the street segments. An *edge* is a fundamental component that represents a connection or relationship between two nodes in a graph. Each row in this DataFrame corresponds to an edge in the graph, and columns include information, such as the geometry of the street.

Django framework follows the Model View Controller (MVC) architecture, a popular structure for frameworks built with object oriented programming (OOP) language. The application is designed to leverage this structure to form individual classes/models for individual interpolation techniques further controlled by the interpolation controller. The controller uses a three-step process to initiate, train, and predict the RSRP values. Figure 2 illustrates the steps, starting with initializing the interpolation class selected by the user with the network source dataset. In the next step, the controller calls the class to build and train the model with the given dataset. The application leverages a series of open-source Python based interpolation packages as listed in Table I. We plan to keep upgrading the application in the future when more advanced and developed interpolation techniques become available. In the final step, the coordinates list from the OMSnx, passes by the controller to the prediction function and receives a list of RSRP values tagged with associated latitude and longitude.

D. Mapping and analysis:

Qualitative analysis: REM can be built to illustrate received signal strength, which are determined by the aggregate effects of the channel upon the signals transmitted by all active sources. REM can also be built based on the signal-to-noise-power ratio (SNR) or the signal-to-interference-plus-noise-power ratio (SINR) [3]. In this paper the focus is beamed towards signal strength based REMs through interpolation, which is of interest to a wider range of user groups. It is essential to provide statistics and information related to the quality of empirical estimates, to determine the best interpolator for a given task. The application appends the predicted and source data to form a unified list. It passes this list to Python's Folium library to construct street-level REM. Folium interfaces with the Leaflet library, offering mapping capabilities through a Python API. To enhance the visual analysis quality of the REM, the application also uses *branca.colormap* library to generate a gradient-based heatmap, which depicts the power distribution of signal strength along the captured streets. This heatmap utilizes a color gradient to represent signal strength quality across the area, ranging from -80 dBm (green) to -120 dBm (red). The heatmap also features interceptions in consecutive data points that are more than 200 meters apart, highlighting the network's continuity.

Quantitative analysis: The prediction algorithms come with an error in their design. Interpolation of network signal strength is no exception. More importantly, if interpolation has been introduced for REM construction, it is crucial to consider the error metrics before using the map to manage, analyze, or make decisions on the network infrastructure. The application uses *sklearn.metrics* to generate local error metrics such as RMSE, MAE, R^2, and inference latency as shown in Table II to analyze the prediction performance of the models, enabling users to solicit the best-possible interpolator. The results can be more relevant to the area size and volume of the dataset provided for training. All these parameters are auto-displayed as an output with any completed interpolation tasks.

TABLE II
APPLICATION'S BUILT-IN MEASUREMENTS METRICS AND STATISTICS.

Learning information	Interpolation quality	REM metrics
Training duration (sec)	MAE	Operator Name
Inference duration (sec)	RMSE	SA/NSA/4G coverage ratio
Total Data Points	R^2	Channel bands
Polygon area size (km^2)	MBE	RSRP

III. Integration Challenges of Interpolation Methods

A. Generalization of training data:

A primary challenge in developing the application was managing different data structures required by various interpolation models during the training/calibration, testing, and prediction phases. For instance, IDW yields distinct predictive values for each specific point in the dataset, whereas OK collectively computes these values, generating all predictions simultaneously. In contrast, methods such as RF and DT deliver their predictive outputs in a batch format. In addition, incorporating user-uploaded data within the application, which might include varying formats of location, cell information and signal strength data, requires the data structure to be unified across all employed interpolators. This aspect was critical to ensure the interoperability of diverse analytical methods. We developed a protocol for data management, encompassing detailed labeling (such as latitude and longitude,

TABLE I

INTERPOLATION METHODS AND ASSOCIATED PYTHON LIBRARIES.

Interpolation method:	Random Forest	Decision Tree	Ordinary Kriging	GAN	IDW
Python library:	RandomForestRegressor() SKlearn	DecisionTreeRegressor() SKlearn	Pykrige OrdinaryKriging: 2D	Tensorflow Keras	See Fig. 3 [20]

signal strength values, and PCI identifiers) and systematic restructuring of datasets. This protocol was applied at each model's data entry and retrieval stages, enabling data format uniformity and facilitating seamless integration across varied interpolation techniques.

B. Measured and uncharted road lists:

The mechanism behind interpolation methods for building REMs is to treat existing values as input to estimate the RSRP at arbitrary points on a map as the output. Our aim is to automatically place the interpolated values on the roads, within the selected geographic boundary, which were not previously mapped. This requires obtaining an explicit list of roads/streets and their associated GNSS coordinates. However, depending on the data collection method, a subset of roads may only be partially measured, which can influence how this part of the area should be interpolated. Initially, we used the OpenStreetMap's Overpass API to get the road and street names in an area of interest. After fetching the names, we used OSM API to get all the GNSS coordinates and compare and remove the ones that already have data [21]. However, while effective, this method fails to capture all the required roads, leaving a subset of the roads uninterpolated. In addition, the required time for generating the coordinates list was significantly prolonged due to utilizing two APIs in the system under sequential processing.

To overcome these obstacles, we adopted the use of OSMnx instead of Overpass API. This Python package is designed to simplify the process of downloading, modeling, analyzing, and visualizing spatial data from OpenStreetMap. Unlike the OSM API, OSMnx focuses more on network analysis and is particularly useful for working with street networks with the ability to save street networks to disk as shapefiles, GraphML, or SVG files, useful functions in the application. It also utilizes caching to minimize repetitive data download. Our evaluation shows that it captures more roads than the first method while significantly reducing execution time. Table III shows an example of processing time using the above-mentioned APIs across three tasks of varied area sizes.

TABLE III

API PROCESSING TIME FOR OBTAINING STREET NAMES AND FILTERING OUT MEASURED REGIONS.

Area ID	Area Size (km^2)	OSM	OSMnx	OSMnx w/o Overlapping
1	5.28	32.12 sec	1.97 sec	1622.94 sec
2	3.17	31.32 sec	1.86 sec	942.47 sec
3	1.5	17.01 sec	1.15 sec	56.07 sec

C. Data aggregation:

REM combines the source data and the estimated values through interpolation methods into an unmarked map. It is fair

Fig. 4. Combining interpolated data with source data. (a) area selected for interpolation, (b) source data used for training (c) Random Forest interpolation, excluding measured areas (d) collinear problem.

to say that the values obtained using physical measurement are more accurate than the interpolated values. Due to the nature of RSRP heatmaps, when interpolated values and real data overlap, the colour of lines on the map misrepresents the signal strength and interpolation errors will contaminate areas with real measurement values as shown in Fig. 4. This is caused by the so-called collinear points when both measured and interpolated values fall along a road or street in the vicinity of each other [22]. This issue arises from the inherent accuracy limitations of GPS in user devices, typically around 5 meters, which often results in measurement data being placed outside of actual roads [3]. In contrast, OSMnx provides a precise road coordinates list. This discrepancy leads to a situation where collinear points emerge despite the efforts to eliminate similar location information from the coordinates list by comparing these two sets of data to avoid overlaps. An algorithm was developed to compute the distance between each OSMnx coordinate and the measurement data. The coordinate is excluded from the list if this distance is less than twenty meters. This threshold was chosen based on the interval of coordinate collection. While effective, this approach adds complexity and extends the time required for generating the final coordinates list for interpolation, as shown in Table III.

D. IDW calculations process:

Network data collection is prone to collect data in irregular spatial distribution for reasons such as vehicle speed and network configuration. As mentioned earlier, IDW operates by assigning weights to data points inversely proportional to their distance from the estimation point. However, when the estimation point is in close vicinity to one or more data points, it produces negligible or zero value in distance, which in turn creates computational overhead. This scenario is defined as the IDW zero distance problem [23], which can potentially interrupt the algorithm's conventional computation flow. A filtration is used in response to this issue where distances amounting to zero are excluded from the weight computation

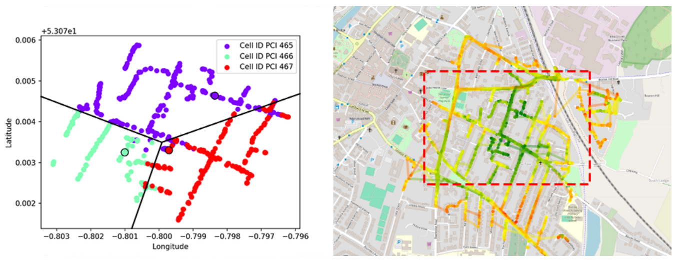

Fig. 5. Data segregation by PCI, revealing similarities with cell patterns.

process. This adjustment is vital in ensuring the best possible mapping of the radio network conditions while the IDW methodology remains consistent and effective, especially when dealing with spatially dispersed data.

E. Exploiting PCI information:

Several papers assumed the base station coordinates are known to the interpolation models [24]. However, network data collected using consumer devices through drive tests lack BS coordinates as well as CellId data. We used PCI (physical cell identifier) values as an alternative to cell information, based on the principle that PCI values are not repeated in smaller areas to prevent collision and confusion [25]. PCI values can identify the different cells of a 4G/5G system, which can help build models based on each cell. This approach can improve the quality of the RSRP interpolation as shown in [10]. Fig. 5 (left) shows an example from our dataset where a BS coverage area and the PCIs associated with each sector. Fig. 5 (right) shows the corresponding heatmap.

The application builds and trains individual models for each unique PCI in the dataset. It initially groups the data by PCI number, which enables drawing approximate cell boundaries, determined by the extremities of latitude and longitude within the group. The prediction methods work on this type of model by finding the right PCI model for a given coordinate. However, this methodology often results in insufficient data for some PCIs, adversely affecting the performance of deep learning models. We observed that models can adversely impact the interpolation quality when the number of training/calibration samples is below 60. To address this issue, we implemented a filter to exclude any PCI model with a dataset of 60 or fewer. Based on the above, we created a new variant for every interpolation technique considered with PCI as a feature, with the goal of improving the predictive performance.

IV. Example Usage and Evaluation

To evaluate the performance of the application, we employed a dataset (size: 2 GB, samples: 5.6 million) obtained from a network survey we conducted across Nottinghamshire County, UK, spanning rural and urban areas of around 2100 km^2. The data collection method and representation of various metrics is discussed in our previous paper [26].

A. Visualizing interpolated heatmaps:

Fig. 6 shows REM heatmaps of RSRP for an urban area, considering 9 interpolation methods. The heatmap in the upper left image highlights that a significant portion of the selected area lacks source data points on several roads.

The coverage of the source data is bordered on three sides, with the upper edge exhibiting a greater concentration of data points. This uneven data distribution is mirrored in the variance observed across the heatmaps by different interpolation techniques. Analyzing the uncharted roads in the upper left, we observe that a consistent pattern among all 9 interpolation outcomes, predicting the signal strength of these roads from average to poor quality. On the other hand, the projections of other regions, especially the roads at the lower center, exhibit greater variance, with different models estimating coverage from high to low signal strength. These findings further strengthen the need for a system capable of carrying out multiple interpolation methods, given the inconsistency in the prediction accuracy of these interpolation methods.

B. Statistics of interpolation quality:

Table IV shows the RMSE and R^2 of 6 carefully selected areas, varying in size from approximately 1 to 15 km^2, encompassing rural and urban regions. The RMSE values illustrate a deterioration of prediction accuracy in larger areas, with PCI based models achieving higher accuracy than baseline models. This improved performance could be attributed to the localized modeling approach of PCI based interpolation, which selectively uses sample data, effectively dividing the target area into several smaller regions. However, this method reduces the available sample size for each PCI sector, which could explain the relatively smaller increase in RMSE. It is important to mention that we considered the weighted average of RMSE and R^2 for PCI based models.

Although OK is widely used for REM construction, our analysis shows it is rarely the best interpolator when applied to the street level REM. We encountered a few out of context RSRP values from the source data, which impacted only OK while other methods remained unaffected. Further looking at the R^2 values in the table, it is safe to say that GAN suffers from underfitting and can not perform to its' potential. This leads us to avoid implementing a PCI based GAN model. DT and IDW also presented limited interpolation accuracy across several regions.

C. Runtime:

Table V presents the training and inference duration of all interpolation methods across the regions, as detailed in table IV. We observe that the training duration is the longest in GAN and shortest in DT, except IDW, which is a deterministic model and does not require training. Also, the training duration is directly proportional to the sample size. The extended training time for PCI based models could result from the additional building duration of multiple PCI variants. On the other hand, the inference time depends on a few factors, such as uncharted area size and road network density. Amongst the interpolators, PCI IDW inference time is reduced significantly compared to its baseline model since IDW considers all sample points for each prediction.

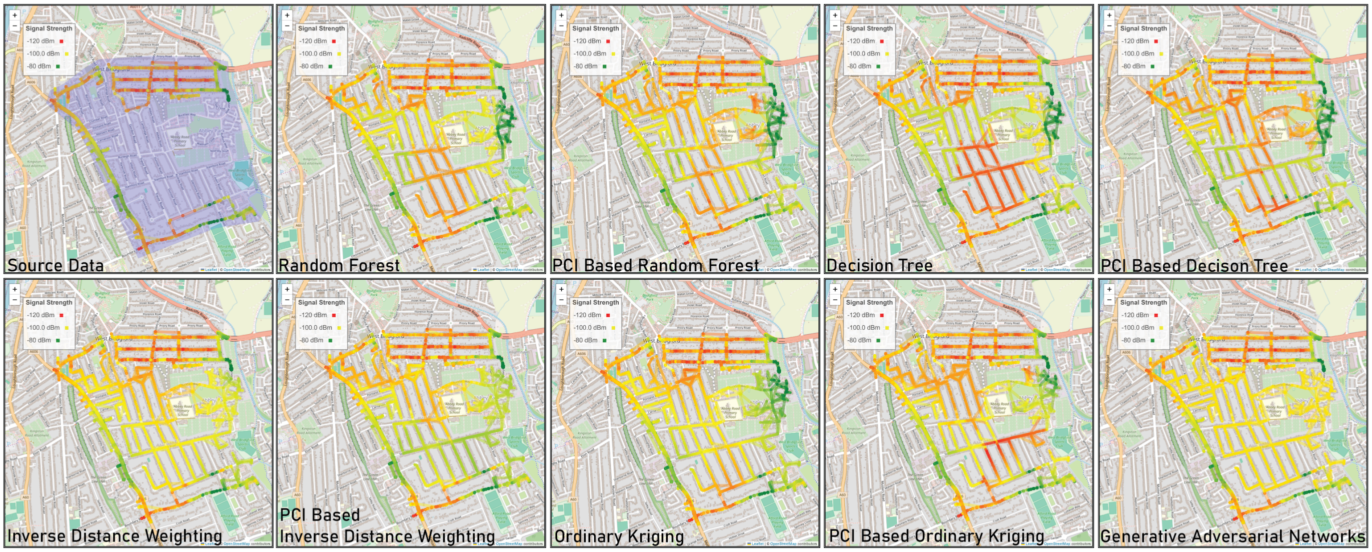

Fig. 6. Heatmaps generated from source and interpolated dataset.

TABLE IV
RMSE & R^2 COMPARISON OF A RANGE OF AREA SIZES AND TYPES FOR DIFFERENT INTERPOLATORS.

ID	Area type	Samples	Area Size	RF		GAN		DT		IDW		OK		RF(PCI)		DT(PCI)		IDW(PCI)		OK(PCI)	
			km^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2
1	Urban	1944	1	**2.84**	0.86	8.31	-0.02	3.25	0.82	3.27	0.83	3.91	0.74	**2.60**	0.86	2.95	0.82	2.65	0.86	9×10^5	-5×10^2
2	Rural	2215	1	**2.5**	0.85	6.45	-0.02	2.79	0.82	2.66	0.81	4.39	0.55	**2.15**	0.86	2.61	0.80	2.21	0.86	116.98	-883.8
3	Urban	11519	5	3.35	0.69	5.99	-0.01	4.12	0.54	**3.2**	0.72	36.72	36.19	3.09	0.66	3.84	0.48	**3.02**	0.68	4.17	0.35
4	Rural	855	5	**4.36**	0.92	16.11	-0.02	5.37	0.88	4.49	0.92	7.33	0.78	**4.86**	0.86	5.20	0.83	5.19	0.86	9.10	0.24
5	Rural	2316	10	3.87	0.85	10.46	-0.01	4.37	0.80	**3.83**	0.87	7.34	0.45	3.25	0.81	3.73	0.76	**3.21**	0.84	836.9	-7×10^4
6	Urban	4638	15	**3.75**	0.79	8.37	0	4.42	0.71	3.87	0.78	2×10^4	-7×10^6	**3.35**	0.73	3.96	0.64	3.45	0.68	10.30	-15.17

TABLE V
TRAINING AND INFERENCE TIME COMPARISON OF A RANGE OF AREA SIZES AND TYPES FOR DIFFERENT INTERPOLATORS.
(HARDWARE DETAILS: INTEL CORE I7-10750H, 16 GB DDR4 (2933 MHZ) RAM, NVIDIA QUADRO P1000)

ID	RF		GAN		DT		IDW		OK		RF(PCI)		DT(PCI)		IDW(PCI)		OK(PCI)	
	Training	Inference	Training	Inference	Training	Inference	Training	Inference	Training	Inference	Training	Inference	Training	Inference	Training	Inference	Training	Inference
1	0.66s	0.51s	4.69s	5.69s	0.01s	0.01s	0s	1.67s	0.42s	0.62s	0.72s	0.42s	0.01s	0.01s	0s	0.2s	0.32s	4.55s
2	0.71s	1.1s	5.15s	13.47s	0.01s	0.02s	0s	4.66s	0.59s	0.77s	0.77s	1.11	0.01s	0.02s	0s	0.57s	0.31s	18.59s
3	4.15s	4.73s	25.05s	67.6s	0.05s	0.09s	0s	123.61s	39.32s	56.88s	4.47s	5.19s	0.05s	0.11s	0s	7.63s	5.9s	927.63s
4	0.38s	5.14s	2.67s	108.79s	0.01s	0.12s	0s	9.5s	0.07s	0.14s	0.49s	5.24s	0.01s	0.1s	0s	0.88s	0.07s	10.49s
5	1.34s	13.53s	5.23s	110.57s	0.01s	0.17s	0s	38.14s	0.59s	1.11s	1.04s	8.74s	0.01s	0.17s	0s	2.53s	0.57s	41.22s
6	1.81s	19.12s	9.81s	251.22s	0.02s	0.34s	0s	184.45s	3.39s	6.83s	1.96	19.11s	0.03s	0.47s	0s	4.3s	0.68s	65.39s

It is possible to program our software package to automatically display the best interpolator quantitively, using numerical values and training the models asynchronously. However, visual inspection and comparative analysis are often required in the context of REMs, which is why we did not attempt to add this function to our solution.

V. CONCLUSION AND FUTURE WORK

In this work, we presented a REM software toolkit, including (*i*) various baseline spatial interpolation methods with automatic model training and dynamic learning (*ii*) polygon area selection to determine source and target areas with high precision (*iii*) and utilized a series of GIS techniques to accelerate the generation of radio maps and provide high interactivity. Spatial interpolations are developed to enable performance comparisons of different interpolations methods, through a set of system KPIs such as RMSE and R^2. The polygon-search style enables more precise and flexible radio map reconstruction. Based on our results, it can be concluded that it is difficult to determine the optimal spatial interpolation method for a selected target area that minimizes the risk. Future work should focus on enabling auto-model selection as a function of the training data by analysing samples before applying it to an interpolator. Several indicators from the source data can be used to suggest a safe choice of interpolation, minimizing RMSE and prediction time.

ACKNOWLEDGEMENT

The financial support by the Local Government Association (LGA), UK, through its Digital Pathfinders Programme and the Department for Digital, Culture, Media & Sport (DCMS), UK, is gratefully acknowledged.

References

[1] H. N. Qureshi, U. Masood, M. Manalastas, S. M. A. Zaidi, H. Farooq, J. Forgeat, M. Bouton, S. Bothe, P. Karlsson, A. Rizwan *et al.*, "Towards addressing training data scarcity challenge in emerging radio access networks: A survey and framework," *IEEE Communications Surveys & Tutorials*, 2023.

[2] M. Pesko, T. Javornik, A. Kosir, M. Stular, and M. Mohorcic, "Radio environment maps: The survey of construction methods," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 8, no. 11, pp. 3789–3809, 2014.

[3] D. Romero and S.-J. Kim, "Radio map estimation: A data-driven approach to spectrum cartography," *IEEE Signal Processing Magazine*, vol. 39, no. 6, pp. 53–72, 2022.

[4] S. Roger, C. Botella, J. J. Pérez-Solano, and J. Perez, "Application of radio environment map reconstruction techniques to platoon-based cellular V2X communications," *Sensors*, vol. 20, no. 9, p. 2440, 2020.

[5] W. A. Hapsari *et al.*, "Minimization of drive tests solution in 3GPP," *IEEE Communications Magazine*, vol. 50, no. 6, pp. 28–36, 2012.

[6] C. K. Anjinappa *et al.*, "Coverage hole detection for mmWave networks: An unsupervised learning approach," *IEEE Comms. Letters*, vol. 25, no. 11, pp. 3580–3584, 2021.

[7] E. Dall'Anese, S.-J. Kim, and G. B. Giannakis, "Channel gain map tracking via distributed kriging," *IEEE transactions on vehicular technology*, vol. 60, no. 3, pp. 1205–1211, 2011.

[8] K. Sato, K. Suto, K. Inage, K. Adachi, and T. Fujii, "Space-frequency-interpolated radio map," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 1, pp. 714–725, 2021.

[9] P. Zhen, B. Zhang, Y.-Q. Xu, Z. Chen, H. Wang, and D. Guo, "Radio environment map construction based on Gaussian process with positional uncertainty," *IEEE Wireless Communications Letters*, vol. 11, no. 8, pp. 1639–1643, 2022.

[10] C. E. Garcia and I. Koo, "Coverage prediction and REM construction for 5G networks in band n78," in *2023 15th International Conference on Computer and Automation Engineering (ICCAE)*. IEEE, 2023, pp. 125–129.

[11] Y. Du, H. Wang, and J. Liu, "Radio environment map construction based on random forest regression," in *2022 IEEE 22nd International Conference on Communication Technology (ICCT)*. IEEE, 2022, pp. 551–556.

[12] H. Xia, S. Zha, J. Huang, and J. Liu, "Radio environment map construction by adaptive ordinary kriging algorithm based on affinity propagation clustering," *International journal of distributed sensor networks*, vol. 16, no. 5, p. 1550147720922484, 2020.

[13] K. Sato, K. Inage, and T. Fujii, "On the performance of neural network residual kriging in radio environment mapping," *IEEE Access*, vol. 7, pp. 94 557–94 568, 2019.

[14] S. Roger, M. Brambilla, B. C. Tedeschini, C. Botella-Mascarell, M. Cobos, and M. Nicoli, "Deep-learning-based radio map reconstruction for V2X communications," *IEEE Transactions on Vehicular Technology*, 2023.

[15] M. Reichardt, M. Gundall, and H. D. Schotten, "Benchmarking the operation times of NoSQL and MySQL databases for python clients," in *IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2021, pp. 1–8.

[16] K. Suto, S. Bannai, K. Sato, K. Inage, K. Adachi, and T. Fujii, "Image-driven spatial interpolation with deep learning for radio map construction," *IEEE Wireless Communications Letters*, vol. 10, no. 6, pp. 1222–1226, 2021.

[17] A. Kuznetsova, A. Talati, Y. Luo, K. Simmons, and V. Ferrari, "Efficient video annotation with visual interpolation and frame selection guidance," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 3070–3079.

[18] P. Crickard III, *Leaflet. js essentials*. Packt Publishing Ltd, 2014.

[19] G. Boeing, "Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks," *Computers, Environment and Urban Systems*, vol. 65, pp. 126–139, 2017.

[20] E. Oktavia, I. W. Mustika *et al.*, "Inverse distance weighting and kriging spatial interpolation for data center thermal monitoring," in *2016 1st International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*. IEEE, 2016, pp. 69–74.

[21] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive computing*, vol. 7, no. 4, pp. 12–18, 2008.

[22] R. Cao, Y. Zhang, X. Liu, and Z. Zhao, "3D building roof reconstruction from airborne LiDAR point clouds: A framework based on a spatial database," *International Journal of Geographical Information Science*, vol. 31, no. 7, pp. 1359–1380, 2017.

[23] L. De Mesnard, "Pollution models and inverse distance weighting: Some critical remarks," *Computers & Geosciences*, vol. 52, pp. 459–469, 2013.

[24] R. Levie, Ç. Yapar, G. Kutyniok, and G. Caire, "RadioUNet: fast radio map estimation with convolutional neural networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 4001–4015, 2021.

[25] A. Zakrzewska, D. Lopez-Perez, L. Ho, H. Claussen, and H. Gacanin, "Cell ID management in Multi-Vendor and Multi-RAT heterogeneous networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 417–429, 2019.

[26] A. A. Bipon, A. Osman, M. S. Islam, A. T. Asyhari, and R. Abozariba, "Pathfinder: End-to-end automation of coverage mapping of 4G/5G networks at street level," in *2023 20th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2023, pp. 375–377.