Online Radio Environment Map Creation via UAV Vision for Aerial Networks

***Abstract*—Radio environment maps provide a comprehensive spatial view of the wireless channel and are especially useful in on-demand UAV wireless networks where operators are not afforded the typical time spent planning base station deployments (e.g. emergency response). Equipped with an accurate radio environment map, a mobile UAV can quickly locate to an optimal location to serve users on the ground. Machine learning has recently been proposed as a tool to create radio environment maps from satellite images of the target environment. However, the highly dynamic nature that precipitates most on-demand aerial network deployments likely renders the satellite image data available for the environment to be inaccurate. In this paper we present, *OREMAN*, a hybrid offline-online system for aerial radio environment map creation which leverages a common sensing modality present on most UAVs: visual cameras. *OREMAN* combines a suite of off-line trained neural network models with an adaptive trajectory planning algorithm to iteratively predict/refine the REM and estimate the most valuable trajectory locations. By using UAV vision, *OREMAN* arrives at a highly accurate map much faster and with fewer measurements than other approaches, and is very effective even in scenarios where no prior environmental knowledge is available.**

I. Introduction

The growing impact of climate change through increased frequency of natural disasters has stressed the importance of on-demand mobile connectivity for public safety missions, when terrestrial infrastructure is compromised or unavailable. Indeed, on-demand aerial mobile networks (non-terrestrial networks [1]), where the base stations are hosted on UAVs, is considered to be an integral component of 5G's broader vision under "safe and smarter cities" for providing communication and sensing services. While network operators spend significant time and resources in planning the deployment of terrestrial base stations (BS), this is not feasible for UAV-based deployments that are often on-demand. With UAVs having limited flight time, they must position themselves in an appropriate location as quickly as possible to provide optimized connectivity to clients (user equipment, UEs), potentially without the benefit of prior knowledge of the environment or the UE locations. Creating an aerial radio environment map (REM, e.g. Figure 1) on-demand that captures the channel quality (e.g. path loss) for various UE locations on the ground from various locations in the sky would be a game-changer for our connectivity vision. With an aerial REM, the problem of finding the optimal UAV location for connectivity becomes a straight-forward one.

Constructing REMs for terrestrial networks has been an important problem of the last decade [2]–[7] owing to its numerous applications in cellular network planning and deployment. Several approaches exist for REM creation: Traditional methods can be classified as either model-based or data-driven. The model-based methods include stochastic models and physics-based simulations such as ray-tracing. The input to such methods is a representation of the environment which can range from a statistical description (as is the case in stochastic models) or a full geometric model. Alternatively, data-driven methods such as Kriging [8] rely on interpolating a set of known values. The samples can be acquired passively, e.g. through normal network operation, or actively through a method such as drive-testing. Recently, machine learning (ML) has been proposed as a promising solution. In this approach, deep neural networks (DNN) are trained to predict the REM directly from a set of inputs that can include either an environmental representation, RF samples, or both.

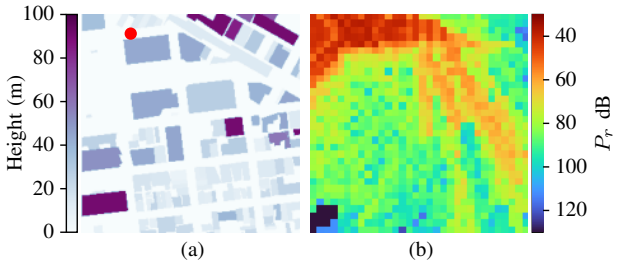

Fig. 1. a) Example building height map and UE location (red dot) and b) the corresponding REM showing the received power at a UAV located at various locations at a given altitude.

There are several challenges that are unique to creating aerial vs. terrestrial REMs: (i) *dimension:* The mobility of the UAV makes the aerial REM high dimensional. In the terrestrial case, in which the BS is fixed, the REM is only indexed by the UE location. In an aerial scenario, REM is indexed both by the UAV and UE locations. (ii) *Air-to-Ground channel:* Compared to terrestrial ground-to- ground (GtG) channels, the air-to-ground (AtG) channel has different properties; namely, the impact of buildings (particularly their height) on the channel is different [9]. (iii) *time-criticality*: Since the UAV has limited resources, any time spent collecting data to create an accurate REM rather than carrying out the connectivity mission must be minimized. While the UAV's mobility allows it to sample the environment, care must be taken to intelligently plan and manage an efficient measurement process. (iv) *insufficient/inaccurate data:* On-demand deployments, where UAVs are most likely to be utilized, must cope with missing environment data (e.g. from disasters), which significantly impacts the utility of REM constructed on inaccurate data.

Thus, existing approaches that work with complete, accurate data for offline prediction and/or focus only on terrestrial GtG REMs are unable to cater to aerial environments. To this end, we propose *OREMAN* (Online REM for Aerial Networks): a novel, first-of-its-kind hybrid approach that brings together both offline neural network models and online, adaptive, multi-modal algorithms for fast REM creation in on-demand aerial deployments. *OREMAN* is driven by the observation that neither offline nor online approaches are sufficient in isolation for our target environment: While offline satellite images can be used for REM creation [3], [10], its accuracy is heavily impacted by varying degrees of missing information (as much as 13 dB as we shown in Section V-B) in practical applications (e.g. public safety: a few downed tall buildings after a disaster; defense: lack of any environment information in tactical deployments, etc.). On the other hand, online algorithms that adapt to only RF measurements, will incur a high (flying) cost to visit a sufficient number of measurement locations, so as to deliver good REM accuracy, not to mention the need to continuously adapt to UE dynamics. Given the limited time-scales of UAV missions, this cost is often unacceptable.

OREMAN is the first online REM creation system which leverages the UAV's ability to "see" (through a widely available camera sensor) to augment its RF measurements and intelligently optimize both the offline and online processes. Incorporating vision allows *OREMAN* to significantly reduce the time needed to create an accurate aerial REM by: i) correcting any errors in the offline environmental data; and ii) providing a means to learn areas of the map without having to physically visit them to capture an RF sample.

Specifically, *OREMAN* utilizes encoder-decoder neural network models that leverages environment information (including the location and heights of the buildings, delivered by processing the camera images) along with the UE locations and a set of RF measurements to the UEs to predict not just the aerial REM from a given UAV altitude, but also the uncertainty in both the RF and corresponding building maps. A key differentiating aspect of *OREMAN* is that its online algorithm leverages *both* the uncertainty information of the RF *and* building maps output by the models to determine the important regions of the environment that need to be prioritized/sampled. This joint uncertainty is then used to devise an intelligent trajectory that maximizes the information utility (for RF and vision) while also incorporating an exploratory reward for unseen regions, so as to minimize the time needed to improve the REM. Online RF and camera measurements are used to update *OREMAN*'s knowledge of the physical environment and iteratively plan the next measurement trajectory. Note that *OREMAN*'s REM model at the end of the measurement period can be used to predict the REM for *any* UE location in the environment (even new UEs for which we have no RF samples), thus the measurement phase is a one-time overhead at the start of the mission and is not impacted by UE dynamics.

We show that the impact of incomplete environment information on an offline-only system can be well over 10 dB of error. By using prior UAV REM creation systems that rely on just RF samples to improve the REM prediction, one can only reduce the error by 0.7 dB/min on average (assuming a UAV speed of 5 m/s). By contrast, *OREMAN*, which incorporates UAV vision into the prediction and planning phases of its online system, doubles this rate to 1.5 dB/min—an 8 dB improvement in just over 5 minutes of measurements compared to the initial offline prediction. The rate is amplified/tripled to 2.1 dB/min, a 12 dB overall gain, in completely unseen locations in which we have no prior environment information.

The contributions of this work are summarized as follows:

- We introduce the concept of using UAV vision as an additional sensor in online aerial REM creation systems. The camera data is used both to update the data used by the pre-trained REM prediction model and to drive the online measurement collection.
- To address environment data inaccuracy/incompleteness, *OREMAN* designs a DNN trained to predict the likelihood of missing buildings given the current seen buildings, the UE location, and the current set of RF samples.
- We propose using both model-predicted building and RF uncertainty, as well as an exploration incentive to plan efficient measurement trajectories designed to maximize the REM error reduction in the least amount of time.
- We have created a large dataset of over 25,000 examples of the AtG channel in multiple environments each with multiple UE locations used for training. The dataset and dataset creation pipeline will be made public.

II. Related Work

There are three traditional methods for creating a REM, which exhibit a trade-off between accuracy and time. Stochastic modeling (such as those provided by the 3GPP specification) is least accurate (or rather least specific) since it is a *model* of the radio environment rather than an explicit map. The benefit of using a model is that it produces results instantaneously. Physics based simulation, e.g. ray-tracing computes the wireless channel based on Maxwell's equations. In ray-tracing software, e.g. Remcom Wireless Insite[1], a 3D model of the desired environment, along with transmitter and reciever locations are specified. Then, the wireless channel is calculated by "shooting bouncing rays" from the transmitter that interact with the environment via reflection, diffraction, and diffusion before ultimately arriving at the receiving nodes. This method has a high compute complexity and takes a significant amount of time to compute the REM. Finally, drive testing is the practice of collecting measurements from a particular environment and is often paired with stochastic modeling as a means to fill in the gaps between the measurement locations and the remaining user locations. This is ultimately the most accurate for a specific site, but has, by far, the highest operational and cost complexity [11].

There have been many recent works utilizing ML to predict REMs [2]–[7], [12], [13], including a recent survey of the field [6] and even a ML style challenge [14]. Though the

[1]https://www.remcom.com/wireless-insite-em-propagation-applications

exact scenarios, datasets, and applications vary, most attempt to train a model to learn the relationship between the radio and physical environment (represented via images or parameters) in an offline terrestrial setting assuming complete information.

We will discuss three relevant and representative works: [4] Uses environment features and "expert-knowledge" (i.e. the calculated distance based path-loss) to predict the path loss between a fixed terrestrial transmitter location and a single point in the environment. This one-to-one approach is not scalable to generate full REMs quickly. [2]'s architecture employs two neural networks (back-to-back) to predict the device to device path loss over an entire area (i.e. one-to-many prediction), where the second network incorporates samples from the ground truth REM to refine the prediction. However, they do not consider the online version of the problem and how to acquire such measurements. [12], the closest to ours, develops two models: one which uses environment features and RF samples to do REM creation and another to predict the residual between the REM prediction and ground truth. The latter is then used to plan a trajectory to collect more measurements. However, environment features are assumed to be known completely and a priori. While *OREMAN* shares some similarities, it can be distinguished in several ways:

1) We specifically focus on the air-to-ground scenario, where the difference between the environment map (buildings) and the aerial REM is more pronounced (compared to terrestrial REM) and scalability is important because of the dimensionality of the problem.
2) We do not assume complete (or any) information about the environment as is the case in practice. We leverage the UAV's ability (equipped with a vision system)to "see" the true environment to improve both the prediction accuracy and reduce the time needed to achieve it online compared to RF measurements alone.
3) We consider the application of an aerial base-station in which the UAV must serve multiple UEs simultaneously. Our goal is thus "many-to-many" REM creation, and we demonstrate how our vision-aided trajectory planning serves this mission.

III. Problem Illustration and Formalization

Consider the following illustrating scenario: A natural disaster has damaged the existing wireless infrastructure in a small town. Emergency responders would like to set up an aerial 5G network to coordinate their response with a UAV acting as an aerial BS. The responders (UEs) are mobile, but their locations are known via GPS (or 5G positioning). Satellite images are available for the area, but due to the disaster, some of the buildings have been knocked down, rendering the images only partially accurate.

We argue that these constraints—prior building information being inaccurate or incomplete, and time-criticality—are realistic for many on-demand network scenarios such as the one described above. Furthermore, these scenarios are significantly different from terrestrial network deployment in which the network operator would have time to accurately map the environment, but would not be able to make online measurements in a manner a UAV-based system would.

The UAV must position itself, as quickly as possible, in a location that maximizes the received signal strength between itself and all of the UEs. It must do this fast both because the UAV has a limited flight-time budget and because of the time-sensitive nature of the mission. Because the UEs are mobile, and new UEs may appear in the area, the optimal UAV position is also dynamic, further emphasizing the need to position itself quickly. An accurate REM would make the solution to the positioning problem a straight-forward lookup.

Ultimately, our goal is to arrive at the most accurate REM, so that we may optimize the UAV position. However, because of the inaccuracies in the knowledge of the true environment, previously proposed offline REM prediction models are not enough to achieve the desired accuracy. To improve the accuracy we require an online process. Now the objective becomes to design a measurement collection process that maximizes the amount we can improve the REM with respect to the offline baseline, in a limited time frame. Because it would be impossible to design this optimal trajectory given the initial state of information, the measurement campaign is divided into multiple epochs, each of which must be designed inteligently to effectively use all UAV sensing modes: vision in addition to RF measurement. If, in each epoch, we maximize the REM improvement per unit time (dB/s), which we will refer to as the "time-to-accuracy" (TtA), we would accomplish our ultimate goal of maximizing the REM accuracy at the completion of all measurements.

Before providing the details of *OREMAN* let us formally define the various variables used and the objective functions serving our goal.

A. Formalization

Let the area of interest be square with side length L meters. Within the area of interest, there are N_B buildings of various shapes and heights. We represent this environment, via a height map B similar to [3]. B is an $M \times M$ image (with resolution L/M $\mathrm{m/pixel}$) representing a top-down aerial view of the area. The pixel values of B represent the height of any portion of the building at the corresponding location. One could think of B as a "2.5D" image since it encodes 3D information about the buildings through the pixel locations and their values. We represent the UE location by X, a 2D binary image the same dimensions as B that is all 0s except for the pixel containing the UE, which is set to 1. In our scenario, the UE is the transmitting node.

For a given environment and UE location, there is exactly one REM[2], P, an $N \times N$ image that represents the received power of the UAV at the corresponding aerial locations at a single height. While P represents the same geographical area as B and X, we do not require it to be the same size, i.e. P may have a different resolution (i.e. L/N $\mathrm{m/pixel}$) than B and

[2]This is not true in reverse: given the UE location, there are multiple environments that result in the same REM. This makes the task of inferring the building map from the REM much harder.

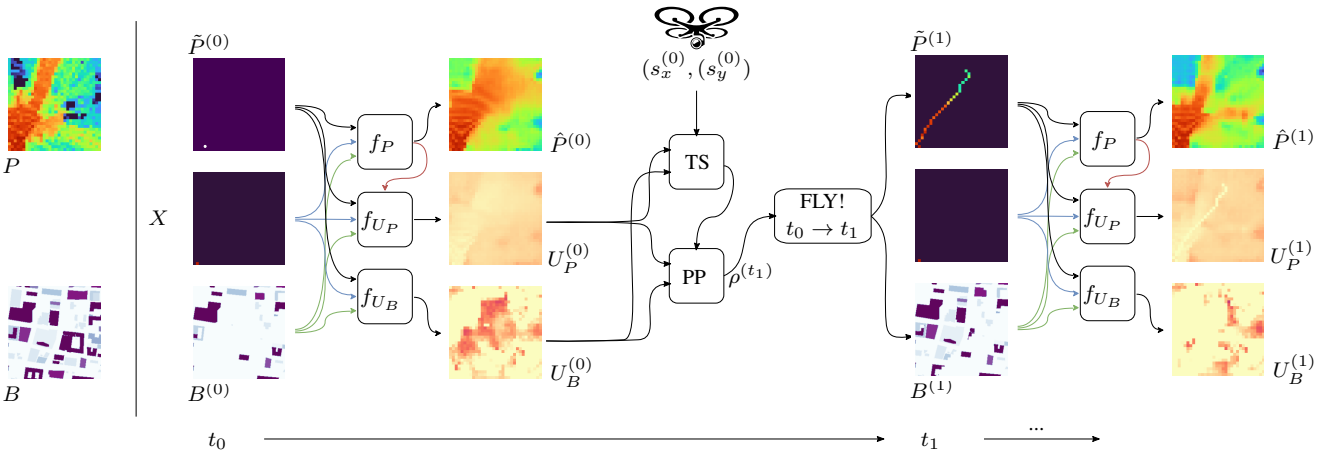

Fig. 2. *OREMAN* overview. The ground truth REM and building map are shown on the left. At time t_0 we begin with a single RF measurement, a known UE location, and a partially incorrect map. *OREMAN* then predicts the REM and an estimation of the uncertainty in both the RF prediction and the known buildings. It uses these to pick a measurement target (TS), plan a trajectory (PP). It then flies the trajectory, taking RF measurements and updating the building map as it flies. It uses this new information to make another set of predictions at time t_1 and repeats the process.

X. Finally, we define $\tilde{P}$ and $\tilde{B}$ as the measured or sampled (incomplete) versions of the RF and building environment respectively. They have the same shape as their respective ground truth versions.

The main objective function is to produce an accurate REM estimate, $\hat{P}$. The accuracy of a prediction is evaluated by the root-mean-squared error

$$\text{RMSE} = \sqrt{\frac{1}{N^2} \sum_{(x,y) \in P} \left([P]_{(x,y)} - [\hat{P}]_{(x,y)} \right)^2} \tag{1}$$

that has the same units as P, which throughout our paper is dBm, shortened to dB. The notation $[P]_{(x,y)}$ indicates the pixel value in the x-th column and y-th row of the image P.

Towards maximizing the utility, namely accuracy of $\tilde{P}$ at the end of the measurement period, the per-epoch objective is to maximize the marginal utility, namely time-to-accuracy (TtA), which is defined as the rate of error reduction in REM prediction per unit time period. Let a UAV trajectory, ρ, be a length-T sequence of steps along the pixels of P in an epoch.

$$\rho \triangleq \{(s_x^{(0)}, s_y^{(0)}), (s_x^{(1)}, s_y^{(1)}), \ldots, (s_x^{(T)}, s_y^{(T)})\} \tag{2}$$

where $(s_x^{(0)}, s_y^{(0)})$ is the location of the UAV at the start of the trajectory. We allow a step $(s_x^{(t+1)}, s_y^{(t+1)})$ to be any of the 8-adjacent pixels to the current location $(s_x^{(t)}, s_y^{(t)})$, thus the average UAV step length (accounting for diagonal movement) is

$$l = \frac{L}{2N}(1 + \sqrt{2}) \text{ m/step.} \tag{3}$$

If the UAV moves at a speed, ν m s^{-1}, then the time, τ, it takes to complete the average T-step ρ is given by

$$\tau = \frac{l}{\nu} T \text{ seconds.} \tag{4}$$

If after completing ρ the RMSE in the REM prediction is reduced by Δ dB, then the TtA is Δ/τ dB/s for the trajectory.

IV. System Design

We now present *OREMAN*, depicted in Figure 2, a first-of-its-kind on-line, adaptive aerial REM creation system for vision capable UAVs as a solution to the problem described in Section III. *OREMAN* consists of two iterative phases: *prediction* and *planning*. The outputs of each phase is used as the inputs of the other, iterating one after another to drive the error in the REM prediction as low as possible, and as quickly as possible.

In the *prediction* phase, at some time/epoch t, a suite of three DNN models use the previously taken RF samples till the epoch, current $\tilde{P}^{(t)}$, the currently known building map $\tilde{B}^{(t)}$, and the UE location X to predict/refine the REM, $\hat{P}^{(t)}$ as well as two uncertainty maps $U_P^{(t)}$ and $U_B^{(t)}$ which represent the uncertainty in the predicted RF map and the known building map, respectively. The three models used in the *planning* phase are pre-trained DNNs described in detail in Section IV-A.

The purpose of f_P is to predict the most accurate REM possible given the data, which is one of the objectives of *OREMAN*. The uncertainty models contribute to the per-epoch objective, TtA, by providing valuable information to the next phase. While employing a model to predict RF uncertainty and using it to plan a trajectory has been considered recently [12], this is not sufficient for practical deployments, where building information may be incorrect, partially available or even completely unavailable. Hence, we have developed an additional model f_{U_B} to predict the uncertainty in the current building map, i.e. missing buildings, and incorporate it into trajectory planning as well.

These uncertainty maps, as well as the current location of the UAV $(s_x^{(t)}, s_y^{(t)})$ are fed into the next phase: *planning*. *Planning* itself has two components, Target Selection (TS) and Path Planning (PP). The output of planning is a trajectory for the next epoch, $\rho^{(t+1)}$. The TS and PP components are designed to maximize the amount of information gained over ρ while ensuring that information is collected as quickly and

efficiently as possible. We detail the path planning components in Section IV-B.

The RF samples taken and buildings seen by the UAV camera along $\rho^{(t+1)}$ are used to update the values of $\tilde{P}^{(t+1)}$ and $\tilde{B}^{(t+1)}$. This data is then used to predict the REM and uncertainty estimates for the next epoch, $\hat{P}^{(t+1)}, U_P^{(t+1)}, U_B^{(t+1)}$, and so on.

At the end of the measurement period, once the UAV has predicted and refined a REM for each of the UEs (in parallel) it wishes to serve, it can then aggregate the REMs by adding them point-wise, and fly to the location that provides optimized service and coverage to all UEs as determined by a desired connectivity objective. As we will show, this is a *one-time* online process. If the UEs move, or if new UEs enter the area, *OREMAN* can predict accurate REM for the new UE location *without needing to collect new measurements* and include this REM in its UAV positioning calculation.

We now give a detailed description of the various components of *OREMAN* mentioned above.

A. REM Prediction and Uncertainty Models

There are three DNN models involved in *OREMAN*: the REM creator f_P, the RF uncertainty estimator f_{U_P}, and the missing building predictor f_{U_B}. With the exception of their input layers and hyperparameters such as the overall size (number of stages), depth (number of layers per stage) and width (number of weights per layer), each of the models is implemented as a U-Net style encoder-decoder DNN [15] with ConvNext blocks [16].

From a ML perspective, we are attempting an image-to-image task, namely (buildings, UE location, RF samples) $\rightarrow$ REM. Thus, we have chosen to use two recent ML techniques shown to be well suited to the task. The U-Net architecture, originally developed for image segmentation, is a modified encoder-decoder model, which is commonly used in many image-to-image translation tasks. The "U" refers to the fact that early layer output activations are concatenated directly to those of later layers, which supports training. ConvNext, the so called "[fully convolutional network] for the 2020s", is a fully convolutional architecture style inspired by the performance and behavior of vision transformers. It has been shown to be state-of-the art on a number of image-to-image tasks. A ConvNext block contains three convolutional layers, a depth-wise convolutional layer using a large kernel size (e.g. 7), an expanding 1x1 convolutional layer, and a 1x1 bottleneck layer. Layer Normalization follows the initial depth-wise convolution, and the Gaussian Linear Unit (GeLU) activation is used after the last layer. A skip connection adds the input of the ConvNext to its output.

In the decoder of each model, the many inputs accepted by a particular model are concatenated together and immediately passed through a "stem", which down samples the inputs by 4. Then the data passes through the network in stages. Each stage consists of one or more ConvNext blocks. The output of each stage is fed forward via a skip connection to the corresponding layer in the encoder. Between each stage is a downsampling layer, which reduces the dimension by 2. Following the decoder, the data moves onto the encoder, which is very similar to the decoder with upsampling blocks replacing the downsampling.

The REM prediction model f_P attempts to predict the most accurate REM, $\hat{P}$ given the current values of $\tilde{B}$ and $\tilde{P}$. The model is trained to minimize the mean-squared-error loss (MSE) between the estimate and the ground truth:

$$\hat{P} = f_P(\tilde{P}, \tilde{B}, X), \qquad \mathcal{L}_{f_P} = \text{MSE}(P, \hat{P})$$

The RF uncertainty model f_{U_P} attempts to estimate the *difference* between current predicted REM $\hat{P}$ and the ground truth. This type of model is an approximation of the posterior variance in REM prediction [12]. It is also trained using the MSE loss:

$$U_P = f_{U_P}(\tilde{P}, \tilde{B}, X, \hat{P}), \qquad \mathcal{L}_{f_{U_P}} = \text{MSE}(|P - \hat{P}|, U_P)$$

A novel building uncertainty model f_{U_B} is unique to *OREMAN*. It leverages the UAV's online camera input, and is trained to predict the difference between $\tilde{B}$, the buildings known to the UAV (either *a priori* or through measurements), and the true building map B. In other words it predicts the location and height of the missing buildings given the current state, namely U_B and will play a critical role in online trajectory planning along with U_P.

$$U_B = f_{U_B}(\tilde{P}, \tilde{B}, X), \qquad \mathcal{L}_{f_{U_B}} = \text{MSE}(B - \tilde{B}, U_B)$$

Specific training performance is described in Section V-B.

B. Measurment Trajectory Planning

After *OREMAN* has predicted the REM and uncertainties, the second phase of *OREMAN* is to plan a trajectory for measurement collection in the next epoch. The trajectory planning takes place in two steps: target selection (TS) and path planning (PP). The points with the highest uncertainties tend to be the most valuable in terms of their ability to reduce the error in our REM creation. Indeed, this is the case since the uncertainties approximate the residual in our RF prediction and the building map. Thus, visiting locations having the highest uncertainty will eliminate the largest sources of error affecting our prediction.

1) Target Selection: We select a target point which maximizes a weighted sum of the uncertainties and an exploration incentive d_a. The latter is defined to be the distance to the target point from the current UAV location. The exploration incentive is aimed to reward visiting unseen locations, which is especially critical in scenarios with incomplete maps. By selecting a target further away, we are guaranteed to "see" more of the environment, thus learning more information. The target selection is given by:

$$(t_x^*, t_y^*) = \arg\max_{(t_x, t_y)} \{\beta[U_P]_{x,y} + (1-\beta)[U_b]_{x,y} + \delta[d_a]_{x,y}\} \tag{5}$$

Figure 3 illustrates five potential targets at calculated via different values of β. The building map, RF uncertainty, and

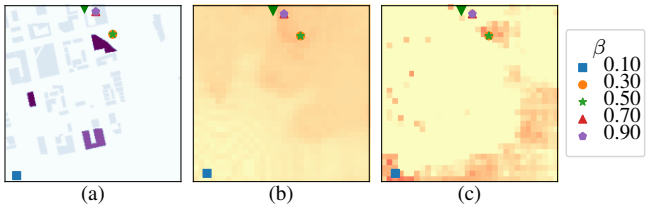

Fig. 3. Potential trajectory targets for different values of β when $\delta = 0.1$ a) building height map b) RF uncertainty c) building uncertainty. The current UAV location is shown by the green triangle.

building uncertainty are shown in a), b), and c), respectively. By increasing β we incentivize planning a trajectory to areas of high RF uncertainty as opposed to building uncertainty. For example, $\beta = 0.1$ picks the location in the lower left corner which has high estimated building uncertainty. A value of $\beta = 0.9$ chooses the location in the top of the map of maximum U_P (RF uncertainty) but minimal U_B, while $\beta = 0.5$ picks a point that balances the two.

2) Path Planning: Once the target location is selected, we wish to maximize the information gained from the measurement trajectory, but we want to do so efficiently to minimize the TtA. In order to come up with the optimal trajectory, we utilize efficient graph-based shortest path algorithms. We first form a graph with nodes corresponding to the pixels in our uncertainty maps. Nodes share a directed edge if their corresponding pixels are 8-way adjacent, meaning nodes are neighbors if their pixels share an edge or vertex. Each edge e is weighted by a weighted sum of the inverse uncertainties of the incoming node and the true distance between the locations represented by the nodes, d_e. The inverse operation allows a shortest path algorithm (e.g. Dijkstra's), which minimizes the total edge weight along the path, maximizes the uncertainty, while the distance-based weight ensures that the path chosen is not too long, balancing between exploration and TtA. The selected path is given by:

$$\rho^* = \arg\min_{\rho} \sum_{(x,y)\in\rho} \alpha \frac{1}{[U_P]_{x,y}} + (1-\alpha)\frac{1}{[U_B]_{x,y}} + \varepsilon d_e \quad (6)$$

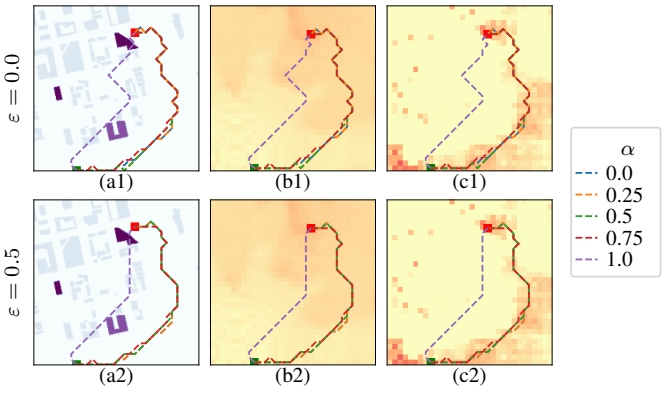

Fig. 4. Planned trajectories with different values of α and $\varepsilon = 0$ and 0.5. a) building height map b) RF uncertainty c) Building uncertainty

Figure 4 shows how changing α and ε affect the overall trajectory relative to the two uncertainty maps. The value of ε is 0 in the top row and 0.5 in the bottom. Observe how lower values of α take the UAV through higher values of RF uncertainty compared to that of the buildings. By increasing ε we encourage straighter, shorter paths. On the other hand, reducing ε encourages exploration, allowing the UAV to see more at the cost of a longer path.

3) Measurement Update: Upon the initial deployment of a UAV to a new environment at some time t_0, we assume that each element of $\tilde{P}$ is set to its default value of -1 indicating that the UAV has not taken a measurement at that location. We also assume that in general $\tilde{B} \neq B$–some areas of $\tilde{B}$ may be correct while others are not. In the worst case $\tilde{B}$ is entirely 0 (indicating no building information).

After the trajectory for t-th epoch, $\rho^{(t)}$, the measurement images $\tilde{P}$ and $\tilde{B}$ are updated. The values of $[\tilde{P}]_{x,y}$ are populated with samples from the true REM $[P]_{x,y}$ if the index (x, y) is in the trajectory.

$$[\tilde{P}^{(t)}]_{x,y} \leftarrow [P]_{x,y} \text{ if } (x,y) \in \rho^{(t)} \quad (7)$$

We assume the UAV is equipped with a downwards facing camera with a field of view θ meters, i.e. the camera exposes all buildings $\pm\frac{\theta}{2}$m in either direction from the UAV's location. We also assume the UAV's vision system is capable of determining the heights of buildings. This is possible using either a traditional quasi-stereographic imaging method identifying key points in the image sequence as the UAV moves or by using DNNs [17]–[19]. To update $\tilde{B}$, we define $\Pi_\theta(x, y)$ to be all of the points the camera exposes, when the location of the UAV is at the point (x, y).

$$\Pi_\theta(x,y) \triangleq \left[x - \frac{\theta}{2}, x + \frac{\theta}{2}\right] \times \left[y - \frac{\theta}{2}, y + \frac{\theta}{2}\right]$$

Then the sampled building image becomes:

$$[\tilde{B}]_{x,y} \leftarrow [B]_{x,y} \text{ if } (x,y) \in \Pi_\theta(s_x, s_y),\ \forall (s_x, s_y) \in \rho^{(t)} \quad (8)$$

Clearly from (7) and (8) we learn more about the environment from the camera than we do from an RF measurement. This motivates the critical use of f_{U_B} and U_B in *OREMAN*'s online process.

C. Handling Multiple UEs

A compelling reason for building REMs is to enable the UAV to position itself in the best location to provide coverage and service to multiple UEs on the ground in an on-demand network scenario. *OREMAN* is equipped with features to handle multiple UEs in both prediction and planning. First, the fully convolutional architecture of the models naturally allow them to predict multiple REMs by concatenating the inputs along the batch dimension. During training, each batch contains a random set of environments and UEs; however, we can take advantage of this batch processing to produce REMs and uncertainties for multiple UEs within the *same* environment in parallel. Secondly, the trajectory planning is done on a single set of RF and building uncertainty maps,

i.e. on a per UE basis. To aggregate planning with respect to each of the UEs, we do so over time. This is possible since the overall measurement phase is broken into multiple epochs, each consisting of a target selection and path planning components. Thus, for each epoch, we rotate the "anchor UE" whose RF and building uncertainties are used to compute the trajectory, allowing *OREMAN* to leverage the diversity of UE locations in mapping the environment faster.

V. Evaluation

A. Air-to-Ground Ground Truth Dataset

In order to train and evaluate *OREMAN*, we have created a large dataset of air-to-ground channel data generated using open source mapping tools and the ray-tracing software WirelessInSite (WI) from RemCom. The dataset consists of 504 unique building maps of various types (e.g. urban, campus, park, etc.) and 50 UE locations per map, for a total of 25,200 total unique REMs.

Each building map covers an area of L=$512\mathrm{m}^2$. The building footprints and height information comes from OpenStreetMaps[3] (OSM). Building heights are clipped to $95\mathrm{m}$. The OSM data can be converted into the ESRI shapefile format which is directly importable into WI. The OSM data is also rasterized at a resolution of $2\mathrm{m/px}$ ($M = 256$) resulting in the true building maps B like the one shown in Figure 2.

Once a building map is defined, UEs place randomly within the map area outside of the building footprints at a height of $2\mathrm{m}$. Similarly, the potential UAV locations are defined over the entire map area on a uniform $N \times N = 32 \times 32$ grid with a spacing of $16\mathrm{m}$ at a height of $100\mathrm{m}$ above the ground. This location information is imported into WI and the UE information is similarly rasterized into X to be used by the model.[4] The building and location data are imported into WI and ray-tracing is done using the parameters in Table I. WI outputs many channel metrics for every Tx-Rx (UE-UAV) pair; we are interested in the received power P_r. The power is calculated by summing (taking into account both amplitude and phase) all the multi-path components of the signal received at the Rx. For a given UE there are $32 \times 32 = 1024$ power values corresponding to each of the UAV locations forming the ground-truth REM P.

This dataset, and the automated pipeline used to create it (environment specification, OSM import, generating Tx-Rx locations, creating WI setup files, and running WI) will be published along with the paper.

Before the data is consumed by the model, each of B, X, and P must be preprocessed. The first step is to scale them between 0 and 1. The height values in B are simply divided by the maximum height of 95. P is first clipped from its full range $[-250, 0]$ dB to $[-130, -30]$ dB before being shifted and scaled. This approach is similar to that adopted in [2].

To train the models to cope with missing building data and incorporate RF sample data, we develop sampling layers as part of the preprocessing step. The RF sampling layer uniformly selects a number of points from P to include in $\tilde{P}$. The number can either be specified directly or chosen uniformly at random in a specified range. Similarly for the buildings, the sampling layer can remove a specified percentage of the buildings, producing $\tilde{B}$. The sampling process acts as a natural form of data augmentation. Note that these sampling layers are only used during the offline training and testing of the various DNNs. During online evaluation of *OREMAN*, $\tilde{P}$ and $\tilde{B}$ are calculated based on the UAV trajectory and online measurements as described by (7) and (8).

The dataset is split 90/10% for training and testing at the environment level. This is done because within an environment similar UE locations may (and should) produce similar REMs. During evaluation, *OREMAN* is evaluated on environments it has never seen before, which is the most realistic scenario from a generalization standpoint, albeit missing in several prior works.

TABLE I
Wireless Insite Ray-Tracing Parameters

f_C	2484 MHz
P_{TX}	30 dBm
Rx Noise Floor	-250 dBm
Antenna Pattern	Isotropic
Ray spacing	$0.5°$
Maximum Reflections	4
Maximum Diffractions	1

B. Model Training and Evaluation

As mentioned in Section IV, each of the models is implemented as a fully convolutional network using ConvNext blocks and based on the U-Net architecture. The exact size and shape of the network was fine-tuned through a heuristic hyperparameter search. The size of each model as well as the training parameters of each model are given in Table II.

Importance of building height for AtG models: We first present results indicating that height maps are necessary for predicting AtG REMs. Figure 5a shows the training and test performance when providing two types of building maps. A "binary" building map does not contain the height information; the pixel values are either 0 or 1 depending on whether or not a building occupies that location. Note that this is how many previous models [2], [4], [12] designed for GtG channels represent the environment. This is appropriate in terrestrial scenarios because the *presence* of the building affects the channel shadowing much more than its height. However, in the AtG channel we have a much higher probability of LOS [9] because of the extreme height difference between the nodes. Thus shadowing is a function of both the buildings location and its height relative to the locations of the Tx and Rx. We find that both the overall performance, and the gap between training and testing improves considerably when height information is embedded into B.

Higher utility of building data: The primary insight of *OREMAN* is that while both RF and building information

[3]https://www.openstreetmap.org/

[4]Note that since our model predicts the channel from one UE to all possible UAV locations, the UAV raster is not included as a model input.

TABLE II
MODEL TRAINING PARAMETERS

Model	f_P	f_{U_P}	f_{U_B}
Total Parameters	10M	10M	5M
Optimizer	Adam		
Learning Rate	1E-4		
LR Schedule	linear increase, cosine decay		
LR warmpup	20	10	
epochs	100	50	

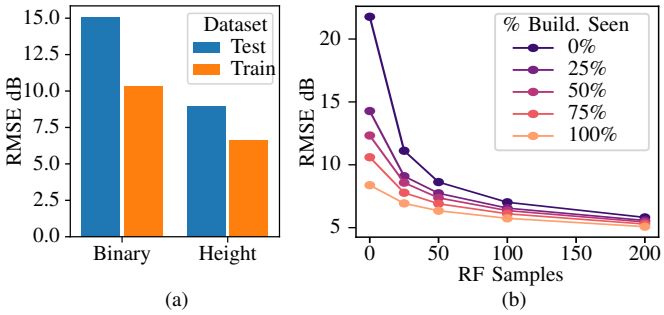

Fig. 5. a) Building height information is necessary for predicting AtG rems. b) The impact of the number of given RF samples and seen buildings on the performance of the REM prediction model.

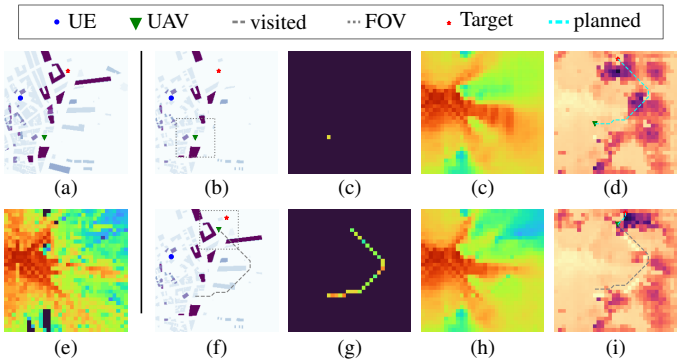

Fig. 6. *OREMAN* simulation example. a) and e) show ground truth building map and REM. b-d) on the top row show the known map, RF samples, the predicted REM and the weighted uncertainties at time $t = 0$, respectively, along with the location of the UE, UAV, and planned trajectory. f-i) on the bottom row show the same after completing the planned trajectory at $t = 1$. The completing the 25-step trajectory the REM RMSE drops from 15.1 $\rightarrow$ 10.8 dB.

are valuable in predicting the REM, the marginal benefit of increasing the amount of building data is *higher* than that of acquiring more RF samples. This is illustrated in Figure 5b, which shows the REM prediction error for different levels of building and RF knowledge as quantified by the percentage of buildings seen and the number RF samples taken. Clearly, increasing the number of RF samples reduces the REM prediction error, regardless of how many buildings are missing. However, when the number of RF samples is low, increasing the percentage of the buildings seen has a significant effect.

Consider a scenario in which only 25% of the buildings are known. This is not unlikely depending on the deployment scenario. At initialization the RMSE is around 15 dB. Without updating the map we need ∼100 RF samples to reduce the error to ∼6 dB. If we assume one sample per UAV step at a UAV speed of $5\mathrm{m/s}$, 100 steps would take around 5 minutes. On the other hand, if we can increase the number of buildings seen from 25% to 75%, we could reduce the error by the same amount with only 25 RF samples (1.25 minutes). Since building discovery can be done in fewer steps than RF sampling, we could improve the TtA by including a building discovery incentive when planning the measurement trajectory. The next section illustrates how *OREMAN* does this in practice.

C. Online Multi-Sensor Trajectory Planning

***OREMAN* in Action:** An example of *OREMAN* in action is shown in Figure 6 showing the initial state and the state after an epoch. In the considered example, we assume that *OREMAN* starts out with a map containing only 50% of the buildings in the true map. The uncertainty map shows the weighted sum of the uncertainty matrices and exploration incentive described in Eqn. 6 as well as the trajectory planned according to IV-B2. After one epoch, $\tilde{B}$ and $\tilde{P}$ have been filled with the collected data, and the predicted REM has improved significantly from 15 to 10 dB RMSE.

Algorithm Comparison: We performed numerous simulations like the one shown to compare the performance of *OREMAN* with several baselines. The closest existing work is [12], which does not update the perceived building map $\tilde{B}$ during flight and plans trajectories only using U_P. The remaining approaches (variants of *OREMAN*) do update $\tilde{B}$ and use the various combinations of U_P, U_B, and d_a (exploration incentive), in their path planning algorithms. Our proposed system *OREMAN* uses all three. We simulated the behavior of each algorithm over various environments, UE locations, initial UAV locations, and initial map states.

Unless otherwise stated, the weights $\alpha, \beta, \varepsilon, \delta$ from the target selection (5) and path planning (6) algorithms used by *OREMAN* in the following results were 0.2, 0.2, 0.5, 0.1, respectively. We assume that the UAV is equipped with a camera with a 45° FOV; for a UAV at an altitude of 100 m, this exposes an area of 81 m^2, which corresponds to roughly a 40×40 pixel area in our building map.

Figure 7 shows the average TtA of the different REM creation systems for a) partially known maps (where 50% of the buildings are missing from the map) and b) initially blank maps. The x-axis shows the number of UAV steps, i.e. trajectory points, over time. There are several main takeaways from these results: First, in a deployment with with inaccurate *a priori* building data, utilizing UAV vision is critical for producing the most accurate REM. This matches the offline results shown in Figure 5b). Without updating the building map with UAV vision data, the RF samples are unable to overcome the discrepancy between the true building map, and the one we have, initially. Second, RF-only planning (note that $\tilde{B}$ is still updated) results in a higher TtA than the other

combinations, especially in the blank map scenario. In the blank map scenario, after 100 steps, the RF only system has an average RMSE of 12.8 dB; including the building uncertainty as a criteria in the trajectory planning lowers the error to 9.9 in the same amount of time. *OREMAN* which includes both building uncertainty and the distance-based exploration incentive has reduced the error to 8.4 dB. That corresponds to 2.1 dB/minute improvement in roughly 5 minutes of flight time over prior art (no building update). The RF only plan is nearly only half of that at 1.3 dB/minute.

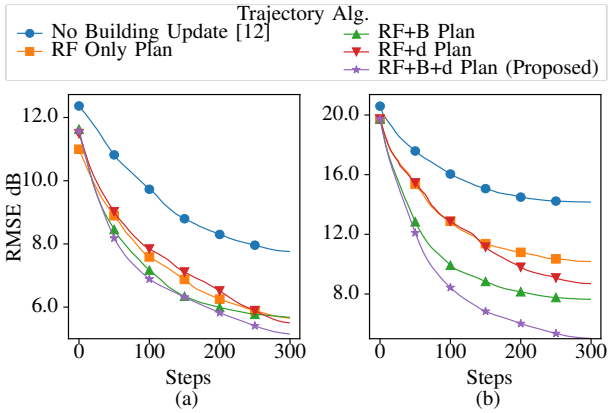

Fig. 7. Comparison of different online REM creation systems when starting with a) a partially known map (50% of buildings missing) and b) a completely blank map. Our proposed *OREMAN* reduces the time to any desired level of accuracy.

D. Multi-UE Coverage and Dynamics

To evaluate the multi-UE scenario, we run a simulation in Figure 8, where at initialization, t_0, there are four active UEs (numbers 1-4, shown as blue dots) and 75% of the true buildings are missing (e.g. due to outdated satellite data). The true building map is shown in a) while the initially known map is shown in b). Two, separate 100-step trajectories taken by the UAVs equipped with the baseline system from [12] and *OREMAN* are shown in c) and d), respectively along with the final building map. Notice how *OREMAN* explores much more of the area, uncovering nearly the entirety of the map (it has seen 99% of the area), while the other system flies a very compact trajectory in comparison since it is only incentivized to reduce the RF uncertainty.

A key aspect of providing multi-UE coverage is being able to respond appropriately to UE dynamics. Hence, at some later time t_1, after the UAV has completed its (100 step) measurement campaign and established its position based on the created REMs, four new UEs (5-8) arrive to the area at the locations marked by red squares. The UAV has to update its position quickly to optimize its coverage for the new larger set. To do this it predicts 4 new REMs based on the known environment and the locations of the new arrivals.

The RMSE at times t_0 and t_1 for the four initially active, measured UEs as well as new, unmeasured UEs in the REMs predicted by [12] and *OREMAN* are shown in Figure 9 a) and b), respectively. Both systems are able improve the REM

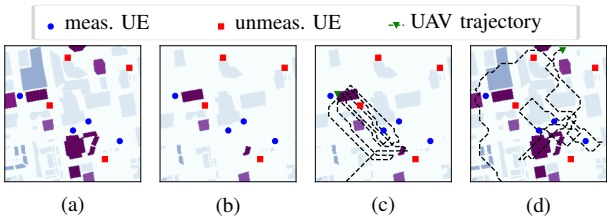

Fig. 8. a) True building map. b) Initial known map, 75% of buildings missing. c) Overall trajectory for [12] which does not update the building map, plan a trajectory based on building uncertainty. d) The overall trajectory planned by *OREMAN* which uncovers 99% of the map via the UAV's camera.

accuracy by roughly the same amount for the initial set of UEs after flying their measurement campaigns. However, once the new set of UEs arrive, *OREMAN*, which obtained an updated/accurate map for the entire area is able to predict more accurate REMs for the new UEs despite having *no measurements for them*. The average RMSE in the REMs predicted by *OREMAN* is 10.1 dB vs. 16.3 dB for the other system, showcasing the effectiveness of its one-time measurement campaign in handling UE dynamics.

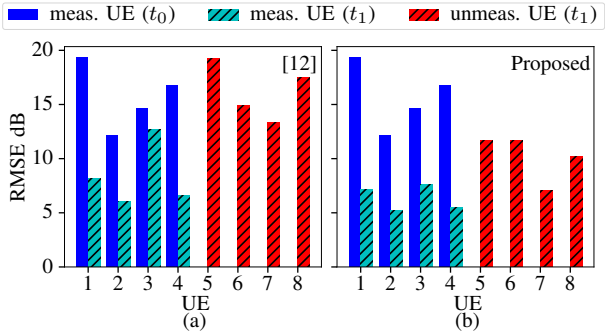

Fig. 9. The RMSE in the REMs predicted in a multi-UE scenario for a blank initial starting map when using a) the [12]'s system and b) *OREMAN*. *OREMAN* is able to improve the starting error for all, even unmeasured UEs.

VI. Conclusion

UAV based networks are a quickly growing part of next generation networks, especially in dynamic environments such as disaster recovery or defense applications. REMs are an immensely useful tool for optimizing the location of the UAV BS in these networks; however, because of the nature of the deployment scenario, the complete, trustworthy environmental information needed to create an accurate REM is not often available. In this work we proposed *OREMAN* which allows a camera-equipped UAV to use 1) offline-trained models to predict the REM and 2) an online, adaptive algorithm to plan the most useful measurement trajectory to maximize improvement over offline prediction quickly for deployment. We have shown that *OREMAN* out performs existing REM creation systems in accuracy and TtA because of its unique ability to leverage online vision updates of the system to improve both REM prediction and trajectory planning. Future work will consist of integrating *OREMAN* into a real system to evaluate in practical deployments.

REFERENCES

[1] A. Chakraborty, E. Chai, K. Sundaresan, A. Khojastepour, and S. Rangarajan, "SkyRAN: a self-organizing LTE RAN in the sky," in *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*, ser. CoNEXT '18. Heraklion, Greece: Association for Computing Machinery, Dec. 2018, pp. 280–292. [Online]. Available: https://doi.org/10.1145/3281411.3281437

[2] R. Levie, 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, Jun. 2021, conference Name: IEEE Transactions on Wireless Communications.

[3] A. Marey, M. Bal, H. F. Ates, and B. K. Gunturk, "PL-GAN: Path Loss Prediction Using Generative Adversarial Networks," *IEEE Access*, vol. 10, pp. 90 474–90 480, 2022, conference Name: IEEE Access.

[4] J. Thrane, B. Sliwa, C. Wietfeld, and H. L. Christiansen, "Deep Learning-based Signal Strength Prediction Using Geographical Images and Expert Knowledge," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, Dec. 2020, pp. 1–6, iSSN: 2576-6813.

[5] D. Romero, S.-J. Kim, G. B. Giannakis, and R. Lopez-Valcarce, "Learning Power Spectrum Maps From Quantized Power Measurements," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2547–2560, May 2017. [Online]. Available: https://doi.org/10.1109/TSP.2017.2666775

[6] 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, Nov. 2022, conference Name: IEEE Signal Processing Magazine.

[7] O. Ahmadien, H. F. Ates, T. Baykas, and B. K. Gunturk, "Predicting Path Loss Distribution of an Area From Satellite Images Using Deep Learning," *IEEE Access*, vol. 8, pp. 64 982–64 991, 2020, conference Name: IEEE Access.

[8] 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, Mar. 2011, conference Name: IEEE Transactions on Vehicular Technology.

[9] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *2014 IEEE Global Communications Conference*. Austin, TX, USA: IEEE, Dec. 2014, pp. 2898–2904. [Online]. Available: http://ieeexplore.ieee.org/document/7037248/

[10] L. Wang, R. Li, C. Zhang, S. Fang, C. Duan, X. Meng, and P. M. Atkinson, "UNetFormer: A UNet-like Transformer for Efficient Semantic Segmentation of Remote Sensing Urban Scene Imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 190, pp. 196–214, Aug. 2022, arXiv:2109.08937 [cs]. [Online]. Available: http://arxiv.org/abs/2109.08937

[11] N. A. M. DANO, E. Director, 5G, and M. Strategies 8/19/2020, "Verizon, T-Mobile, AT&T balk at drive testing their networks." [Online]. Available: https://www.lightreading.com/test-and-measurement/verizon-t-mobile-atandt-balk-at-drive-testing-their-networks/d/d-id/763329

[12] R. Shrestha, D. Romero, and S. P. Chepuri, "Spectrum Surveying: Active Radio Map Estimation With Autonomous UAVs," *IEEE Transactions on Wireless Communications*, vol. 22, no. 1, pp. 627–641, Jan. 2023, conference Name: IEEE Transactions on Wireless Communications.

[13] D. Gesbert, O. Esrafilian, J. Chen, R. Gangula, and U. Mitra, "UAV-aided RF Mapping for Sensing and Connectivity in Wireless Networks," *IEEE Wireless Communications*, pp. 1–7, 2022, conference Name: IEEE Wireless Communications.

[14] J.-H. Lee, J. Lee, S.-H. Lee, and A. F. Molisch, "PMNet: Large-Scale Channel Prediction System for ICASSP 2023 First Pathloss Radio Map Prediction Challenge," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2023, pp. 1–2.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015, arXiv:1505.04597 [cs]. [Online]. Available: http://arxiv.org/abs/1505.04597

[16] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A ConvNet for the 2020s," Mar. 2022, arXiv:2201.03545 [cs]. [Online]. Available: http://arxiv.org/abs/2201.03545

[17] Y. Lyu, G. Vosselman, G.-S. Xia, A. Yilmaz, and M. Y. Yang, "UAVid: A semantic segmentation dataset for UAV imagery," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 165, pp. 108–119, Jul. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0924271620301295

[18] A. S. Chakravarthy, S. Sinha, P. Narang, M. Mandal, V. Chamola, and F. R. Yu, "DroneSegNet: Robust Aerial Semantic Segmentation for UAV-Based IoT Applications," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4277–4286, Apr. 2022, conference Name: IEEE Transactions on Vehicular Technology.

[19] L. Madhuanand, F. Nex, and M. Y. Yang, "Self-supervised monocular depth estimation from oblique UAV videos," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 176, pp. 1–14, Jun. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0924271621000952