AUT Journal of Modeling and Simulation

AUT J. Model. Simul., 53(2) (2021) 217-234
DOI: 10.22060/miscj.2022.20174.5250

Generative Adversarial Networks for Propagation Channel Modeling

Sh. Seyedsalehi, V. Pourahmadi*, H. Sheikhzadeh, A. H. Gharari Foumani

Electrical Engineering Department, Amirkabir University of Technology, Tehran, Iran.

ABSTRACT: Channel is one of the most important parts of a communication system as the medium of the propagation of electromagnetic waves. Being aware of how the channel affects the propagation waves is essential for the design, optimization, and performance analysis of a communication system. Along with conventional modeling schemes, in this paper, we present a novel propagation channel model. The proposed modeling strategy considers the 2-dimensional time-frequency response of the channel as an image. It models the distribution of these channel images using Deep Convolutional Generative Adversarial Networks (DCGANs). In addition, for the measurements with different user speeds, the user speed is considered as an auxiliary parameter for the model. StarGAN is used as an image-to-image translation technique to change the generated channel images with respect to the desired user speed. The performance of the proposed model is evaluated using a few existing evaluation metrics. Furthermore, as modeling the 2D time- frequency response is more general than the modeling of the channel only in time, the conventional metrics for evaluation of the channel models are not sufficient; therefore, a new metric is introduced in this paper. This metric is based on the Cepstral Distance Measure (CDM) between the mean autocorrelation of the generated samples and measurement data. Using this metric, the generated channels show significant statistical similarity to the measurement data.

Review History:

Received: Jun. 12, 2021
Revised: Jan. 09, 2022
Accepted: Feb. 14, 2022
Available Online: Mar. 10, 2022

Keywords:

Generative adversarial networks
Image-to-image translation
Propagation Channel modeling
Statistical model evaluation

1- Introduction

Propagation channel modeling is one of the most essential parts of the communication system design and simulation. When the channel state information is available, transfer characteristics such as signal constellation and allocated power can be adjusted in order to increase the system performance. A proper propagation model is needed for performance analysis of the system as well. Several methods have been developed for propagation modeling, which can be divided into deterministic and stochastic approaches [1]. In the first class of approaches, the propagation process is reconstructed by solving Maxwell equations in a particular medium with the given boundary conditions. Ray Tracing (RT) models are the most commonly employed models of this type [2], which are based on the geometrical optics. Deterministic approaches are accurate, but they need complex calculations and are heavily dependent on the exact description of the medium. On the other hand, stochastic approaches try to provide a statistical description of channel characteristics (e.g., Power Delay Profile (PDP)). They can be divided into two categories: Geometry-Based Stochastic Models (GBSMs) and Non-Geometry Based Stochastic Models (NGBSMs).

GBSM approaches use probability distributions to describe the properties of the effective scatterers, and the fundamental laws of wave propagation are then applied to model the propagation process. The 3rd Generation Partnership Project (3GPP) has introduced the Spatial Channel Model (SCM) as a geometry-based model [3]. Later it was extended to the Spatial Channel Model Extended (SCME) [4], where SCME supports wider bandwidth. In the Wireless World Initiative New Radio (WINNER) projects, the model was adapted to 15 different scenarios. Additionally, 3GPP has produced a 3-dimensional channel model 3D SCM for Long Term Evaluation advanced (LTE-advanced) [5].

NGSM approaches focus on paths between the transmitter and the receiver, and scatterers are not modeled explicitly. Tapped Delay Line (TDL) based models are the most known models of this type, where the propagation channel is considered as a number of delay taps. Each tap has a few characteristics, such as the average power, excess delay, and amplitude distribution function.

All the aforementioned traditional models have limitations. As mentioned before, other than the need for a large number of measurements for fitting the stochastic model, stochastic approaches are parametric and are based on mathematical expressions. Therefore, they impose some prior assumptions on the model, which are not necessarily correct. In addition, some complicated mediums such as underwater acoustic channels or in-body channels may introduce some complex distortion effects, which are impossible to be ex-

*Corresponding author's email: v.pourahmadi@aut.ac.ir.

pressed analytically. These observations signifies the need for a framework with the capability of learning complex models, without any presumption on the model parameters.

Recently, deep learning approaches have shown great performance in many areas. One of the most important advantages of deep learning over traditional statistical models is that Deep Neural Networks are able to learn some valuable features from the data, without the need for costly and time-consuming human effort for feature engineering. Moreover, when it comes to probabilistic distribution prediction from data, traditional statistical methods assume an analytical parametric formulation for the distribution and try to predict the parameters. These analytical formulations induce limitations to the model, while Deep Neural Networks are able to learn the statistical distributions, regardless of such limitations. Hence, they have proved astonishing performance in many areas at the cost of the need for a huge amount of data. Therefore, in case that data is available, it seems reasonable to take advantage of Deep Neural Networks.

Deep Neural Networks have been exploited largely in communications as well. Some examples are CSI (Channel State Information) feedback compression and reconstruction [6], joint source-channel coding [7], channel estimation and signal detection [8] and many other applications.

As far as the channel modeling is concerned, there are some research on employing Machine Learning (ML) and Deep Learning algorithms in this area. In the majority of cases, ML or Deep Learning frameworks have been exploited for channel estimation. For example, in [9] the authors have presented a model for estimating conditionally Gaussian random vectors with random covariance matrices, based on Convolutional Neural Networks (CNN). In [10], an online CSI prediction scheme is proposed that consists of a CNN for frequency representative vector extraction, a CNN for state representative vector extraction, and an LSTM (Long Short-Term with Memory) for state vector prediction.

ML algorithms and Deep Neural Networks (DNNs) have also been used in path loss models and prediction of the model parameters. However, to name a few, in [11] two ML algorithms, Random Forest and K Nearest Neighbor (KNN) are exploited to build a path loss model for the Unmanned Areal Vehicle (UAV) Air-to-Air (AA) scenario, based on the training data. In [12], Machine Learning methods are applied to high-speed channel modeling for signal integrity analysis. Linear, support vector, and Deep Neural Network (DNN) regressions are adopted to predict the eye- diagram metrics, taking advantage of the massive amounts of simulation data gathered from prior designs. In [13], the authors propose a Machine Learning based channel modeling technique for molecular MIMO communications, which consists of two processes: fitting the channel model parameters and learning the patterns in the input-output dataset by Artificial Neural Networks (ANNs). In [14], the authors have exploited Neural Networks to predict the AoD of the dominant propagation paths in the user channels. They use AoA dependent propagation-channel features for this purpose. In [15], Recurrent Neural Networks have been used to extract some unexplored structures (beyond linear correlations) from the Channel State Information (CSI). In [16], conditional GAN is used as a channel generator inside an end-to-end learning-based communication system. Similarly, a model which takes advantage of Generative Adversarial Networks (GAN) to model the distribution of one-dimensional channel responses is proposed in [17]. In addition, in [18], the authors have employed Conditional GANs to learn channel models at each beam-forming direction from the Air-to-Ground (A2G) channel in formation. In [19], the channel estimation problem is considered as modeling a conditional distribution of the received channel value, given the transmitted channel value and Variational Generative Adversarial Network are utilized to model this conditional distribution. It should be noted that these studies still consider the channel-response of the system (i.e., in time). An environment, however, may have different effects on the propagated signals in different frequencies. For a more complete modeling of the channel, we need to model the statistical behavior of the channel in both time and frequency.

In this paper, in the context of deep learning approaches for communication applications, a novel channel modeling technique is introduced. This model overcomes some of the limitations of the traditional methods and introduces a completely intelligent framework for modeling the statistics of the channel. The idea is based on using generative models as intelligent frameworks for modeling the statistical distribution of channel measurements. Generative models are able to learn the distribution of the training data and generate new samples having the same statistical properties as the training data. Generative Adversarial Networks (GANs), as one of the most important types of generative models, have shown very good performances in modeling the distribution of the images. If somehow the propagation channel can be considered as an image, GANs can be applied to model the distribution of channels as well. For this purpose and to have a more complete model of the channel, the 2D time-frequency responses of the channel are considered as images in this work, called channel images. The main idea of this paper is to model the distribution of these channel images using GANs. Furthermore, it is assumed that the user carrying the receiver is moving at various speeds. The effect of user speeds on channels is considered. We aim to propose a model that considers the user speed as an auxiliary parameter to generate channels matching the properties of a particular environment.

A two-phase procedure is followed for the modeling approach. First, the distribution of channel images having a reference user speed is modeled using Deep Convolutional GANs. In the second phase, we take advantage of StarGAN as an image-to-image translation technique to change the generated channel images with respect to the desired user speed. The model is applied to three different simulated channel types and real measurement data. We first evaluate the performance of the model using common metrics, such as Level Crossing Rate (LCR) and Average Fade Duration (AFD). These metrics have some shortcomings in evaluating the dissimilarity between different channel types (i.e., with these metrics, some channels of different types might show

high similarity). To obtain a more reliable criterion, a new metric based on the Cepstral Distance Measure (CDM) of the mean of the autocorrelation of the data is introduced. This metric captures the specific frequency patterns of the time-frequency response for each channel type. It is used to evaluate the statistical similarity between measurement and generated data of each type.

The main contributions of this paper can be summarized as follows:

1) Proposing a channel modeling technique which can generate channels with the desired user speed, having a similar distribution to the measurement data.

2) Introducing a metric for statistical evaluation of the proposed model (checking the statistical analogy between the generated channels and the measurement data.)

This paper is organized as follows: section 2 provides background on the propagation channel, GANs, and image-to-image translation techniques. In Section 3, problem definition is provided, in Section 4 the channel modeling procedure is described, in Section 5 experimental results are provided, and finally Section 6 concludes the paper.

2- Background

2- 1- Propagation Channel

Propagation channel is the medium between the transmitter and the receiver, which affects the amplitude and the phase of the transmitted signal. The complex channel gain of multi-path fading channels can be modeled as a stochastic process as:

$$\mu(t)=\sum_{n=1}^{N} c_n e^{j\left(2\pi f_n t+\theta_n\right)}, \tag{1}$$

where N is the number of paths, c_n, f_n and θ_n are the gain, Doppler frequency and phase of the n th path, respectively. The Short-Time Fourier Transform (STFT) of (1), is used for a time-frequency representation of the channel response.

2- 2- Generative Adversarial Networks

Machine Learning models can be divided into two general categories: discriminative models and generative models. Discriminative models try to find a relation between inputs and outputs and usually are used in regression, prediction, and classification applications. Generative models, on the other hand, have the ability to learn the distribution of the data and can generate samples having a similar distribution to the training data. One of the most popular generative models is GAN [20].

The idea behind GAN is to train two networks simultaneously. A generator network has to learn the distribution of training data and generate samples having the same statistical distribution as the training data from the input noise. On the other hand, a discriminator network has to find the probability that a sample comes from the training data.

At the first stages of training, nor the generator or the discriminator know the correct distribution (pattern) of the real data. The generator generates some samples from an input noise vector. The generated samples allocated with a 'fake' label and is fed to the discriminator along with the real data allocated with a 'real' label. The discriminator is actually a classifier between real and fake samples which examines the input sample and tries to determine whether they are real or fake. As the training goes on, the generator becomes stronger and generates samples more similar to the real data. Simultaneously, the discriminator evolves as well to detect the fake samples. During the training, both the generator and the discriminator evolve so much that it becomes difficult for the discriminator to distinguish real and fake samples. At this point, we say that the generator has learned the distribution of the real data and the network has converged.

GANs have been widely used in various applications and are extended for example to Deep Convolutional GANs (DCGANs) [21], Conditional GANs [22], Boundary Equilibrium GANs (BEGANs) [23], and Wasserstein GANs (WGANs) [24].

3- Problem Definition

3- 1- Channel as an Image

In Multiple-Input Multiple-Output (MIMO) systems, the transmitter/receiver antenna space is considered as the channel matrix H. Each element H_{ij} in the channel matrix represents the channel between the i^{th} transmitter antenna and the j^{th} receiver antenna. In the MIMO settings, H_{ij} usually is considered as a single complex value, and so H will be an $r\times t$ matrix (number of receive antennas $\times$ number of transmit antennas), and in some previous studies such as [25], H_{ij} is treated as an image.

For a more detailed look at the channel, the single value of H_{ij} can be substituted by a matrix that has $m\times n$ complex values, each value representing the effect of channel on a particular frequency and particular time (m represents the number of frequency sub-carriers, and n is the number of time slots).

In this work, we focus on the channel between one pair of transmitter and receiver antennas. The grid of the channel time-frequency response, which is actually the Short-Time Fourier Transform (STFT) of the channel transfer function, is translated into a $m\times n\times 2$ image by placing the real values in the first channel of image and the imaginary values in the second channel. Figure 1 shows how this translation is done for a grid of size $m\times n$. As mentioned in Section I, we denote these images by channel image.

3- 2- Channel Modeling

In data-driven channel modeling, which is followed in this paper, one has a set of collected channel measurements and is looking for a model that can find out the main statistics or features of the measurement data. With this model, new samples can be generated with the same statistics of the channel. In addition, there are a few parameters such as the user speed, which in some-form affect the statistics of the channel.

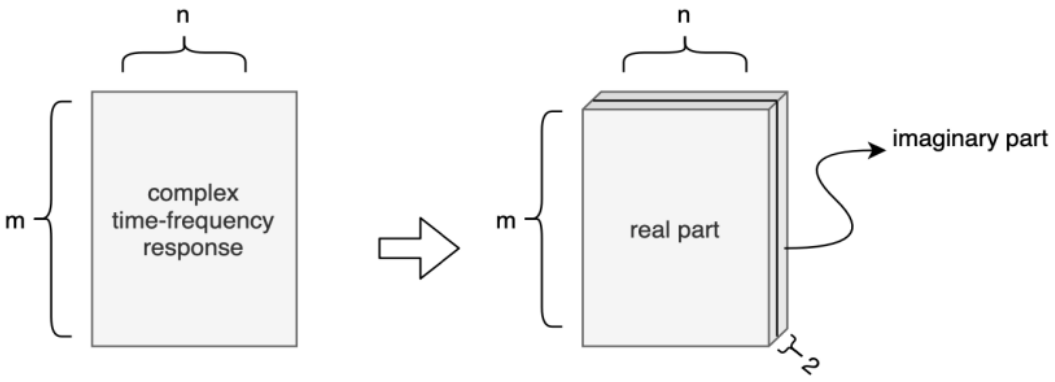


Fig. 1. Converting the channel time-frequency response into a channel image.

We intend to build a model that can incorporate such parameter (here we focus on user speed) when it generates a sample for that environment.

As for a formal problem definition, assume that many measurements of one particular channel type such as a specific environment (like an urban area) are given. Each channel measurement is performed at a particular user speed. For each user speed, a large number of samples must be collected so that there can be enough samples to be used as the dataset for training a network. Therefore, we have a set of channel measurements $X_{Urban}=\{X_1,X_2,...,X_n\}$ containing n samples of the channel measurements and a set of possible user speeds $V=\{v_1,v_2,...,v_m\}$, where m is the number of distinct user speeds. Each X_i is collected at a certain speed of the user. Note that when the environment changes (channel type changes), another set of measurements like W_{Rural} is obtained, where its elements are drawn from different statistics, and thus another model needs to be built for that environment. More accurately, the major effect of different environments is on the multi-path delay profiles of the channel. This is due to some physical characteristics of different environments. We note that the proposed network does not concern about the environment that the samples are representing, and just tries to capture the statistics of whatever samples provided to it.

The aim is to propose a model that can learn the statistical distribution of the measurement data of different channel types, and generate new samples for that environment. The set of generated samples ($\hat{X}$) should have the same distribution as the measurement data (X). The model should also capture the effects of different user speeds on channel measurements and generate channels concerning a specific desired user speed.

To evaluate the performance of such a model, a metric is needed to evaluate how the samples in $\hat{X}$ emulate the statistics of the samples in X. Since we are talking about statistics of two sets, measures like Euclidean distance of samples do not work, instead, one should employ metrics that: 1) can demonstrate that the samples of the model trained using dataset X are statistically similar to the samples in X (the actual measured data), 2) can evaluate the dissimilarity between the statistical distribution of the samples of the model trained using dataset X and the samples of another set like Y, measured for another channel type/user-speed.

For this purpose, there are a few of such metrics in literature, but as will be discussed later, they fail to simultaneously possess the two properties, especially because we are dealing with two-dimensional samples (like channel images). Thus, in this study, we have to introduce a proper comparison metric as well.

4- Data Driven Channel Modeling Procedure

4- 1- GAN-Based Channel Modeling

As discussed in Section 3.1, we can consider the channel measurements as channel images. With this idea, the problem of generating samples with the same distribution of channel measured samples can be posed as a problem, in which one is trying to generate images that have a similar shape to the images that are already in the training dataset. This view of the problem inspired us to try to use generative models (briefly discussed in Section 2.2) as a tool for propagation channel modeling.

The problem we are actually facing is that we have some channel images, each having a corresponding user-speed, and we aim to model the statistics of the images conditioned on their user-speeds. To do so, we first considered some conditional generative models such as Conditional GAN (cGAN). cGAN is a type of GAN that is used for a labeled categorical dataset. It considers the labels as the condition parameter and generates samples conditioned on this parameter. In our case, the dataset would be the channel images, and the labels would be their corresponding user-speeds.

We tried to train a cGAN with our channel images, but unfortunately, the network did not converge and the generated samples did not have similar desired statistics, probably due to the complexity of the dataset.

To manage the problem, we have proposed a two-phase solution. Especially, instead of trying to train a model that can directly give us a channel sample for a particular channel type and desired user-speed, we generate the desired sample using two networks. The first one models the distribution of the

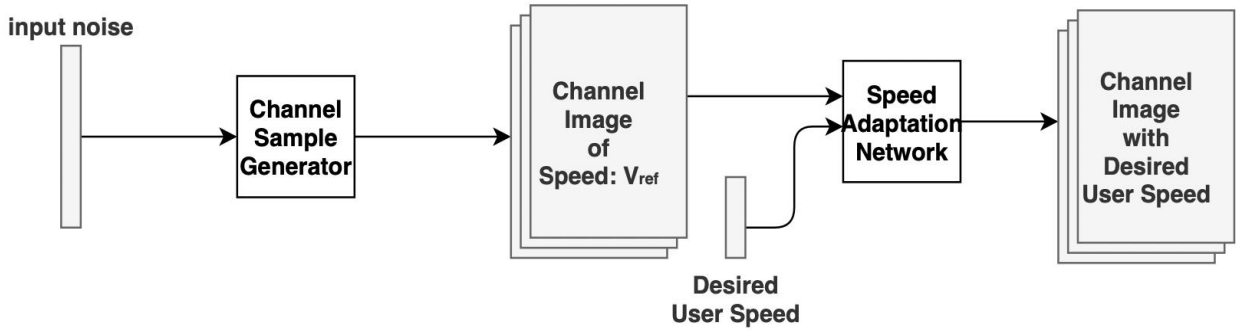


Fig. 2. The generation procedure of channel images with desired user speed.

channel images all having a reference user-speed. The second one adds the effect of the different user-speeds. For the second stage, we convert a generated channel with the reference user-speed to the desired target user-speed by changing the channel image with an image-to-image translation technique, namely STARGAN. This procedure is shown in Figure 2 pictorially, where:

- Channel sample generator: In this network, the generator does not need to deal with different user speeds. The goal of the first network is to generate samples that have the same distribution as the measured data, assuming that all are collected at a nominal speed, v_{ref} .
- Speed adaptation network: In the second step, the network modifies the generated sample such that its statistics match the statistics of the channel at the desired user speed, v .

In the following two sections, the structures of the Channel sample generator and Speed adaptation network are described. Afterwards, in Section IV-C, the metrics employed to evaluate the similarity between the generated and actual channel samples are discussed.

4- 2- Channel Sample Generator

As for the first step, we need to train a network that learns the distribution of channel images (collected at v_{ref}) and generates similar samples. In this study, a DCGAN (briefly reviewed in Section 2.2) operates as the channel sample generator.

The training procedure is shown in Figure 3. For each environment, the set of collected channel measurements (e.g., X_{Rural} or X_{Urban}) is considered as the training data. The generator tries to generate channel images having the same distribution as the training data (i.e., it tries to find a mapping $G\left(z;\theta_g\right)$ from the noise space to the data space, where θ_g is the set of parameters for G). The input noise is sampled from a prior distribution $P_Z\left(z\right)$ that is usually considered a uniform distribution on [-1,1]. The discriminator network, denoted by $D\left(X;\theta_D\right)$, tries to distinguish real channel images from the ones generated by the generator network.

Both the generator and the discriminator are deep convolutional networks. In the training of the networks, labels 0 (representing fake images) are allocated to the generated channel images and labels 1 are allocated to the real channel images. The discriminator performs a classification between real and generated channel images. The loss function is:

$$min_G max_D V\left(G,D\right)=E_{x\sim P_{data}(x)}[\log\left(D\left(x\right)\right]+ E_{z\sim P_z(z)}[log\left(1-D\left(G\left(z\right)\right)\right] \quad (2)$$

where the discriminator's output $D\left(x\right)$ is a single scalar representing the probability of x coming from the measured dataset. During the training, the discriminator tries to maximize the loss function and classify its input images as real or fake (generated by the generator). On the other hand, the generator aims to fool the discriminator by minimizing $V\left(G,D\right)$.

Obviously, in the first iterations of training, the generated images are almost noise and it is easy for the discriminator to distinguish them from real channel images. As the training goes on, the generator improves in competition with the discriminator until the channel images generated by the generator are so similar to the training data that the discriminator cannot distinguish them. At this point, the generator learns the distribution of channel images and the network converges.

4- 3- Speed Adaptation Network

User speed mainly affects the Doppler frequency, and its impact is seen in the time domain of the time-frequency response of the channel. Figure 4 shows a sample Extended Typical Urban (ETU) channel with four different user speeds: $25,25,75,$ and 100 km/h. As can be seen, when the user speed increases, the variations in time axis are increased.

In the second step of the network, we aim to adjust the channel images such that the network learns how different user speeds affect the channel image and incorporates the effect of the user speed on the generated sample images.

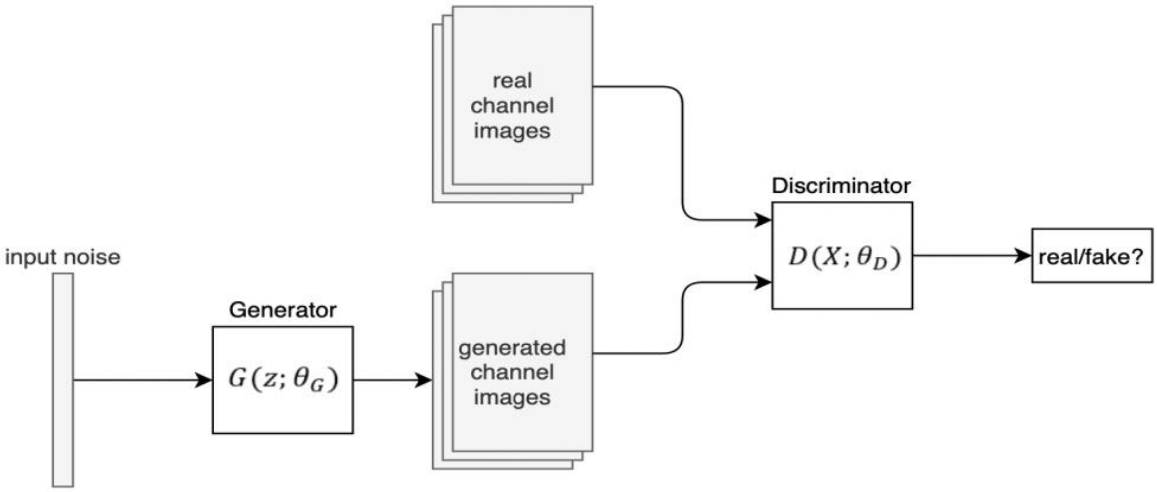
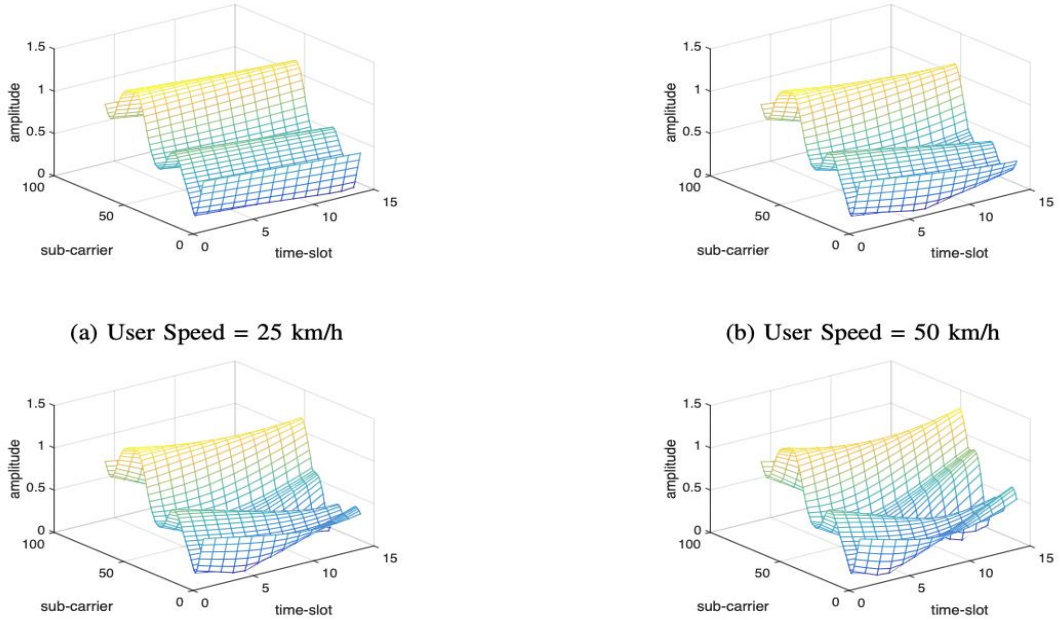


Fig. 3. Training procedure of DCGAN for learning the distribution of the channel images.



(a) User Speed = 25 km/h

(b) User Speed = 50 km/h

(c) User Speed = 75 km/h

(d) User Speed = 100 km/h

Fig. 4. A sample of ETU channel with four different User Speeds: (a) 25 km/h, (b) 50 km/h, (c) 75 km/h, (d) 100 km/h.

One immediate idea to build such a model is to train a supervised image-to-image translation network, in which corresponding pairs of images in two domains are available. Such network would get channel images of v_{ref} as input, and the corresponding channel image of the same sample when the user speed is, for example, v_1 as output. Having trained the network with many of these samples, one can hope that it will do the correct modification on the generated sample to convert them from v_{ref} to the desired speed.

The main challenge for this method is that it is not possible to gather such dataset. We can definitely measure the channel when the user is moving with speed v_{ref} and v_1, but it is very hard to keep all the elements of the environment the same between the two experiments. In other words, it is possible to have two sets of training data, one collected at v_{ref} and the other collected at v_1, but it is impossible to determine which element of the first set is associated with which element of the second set.

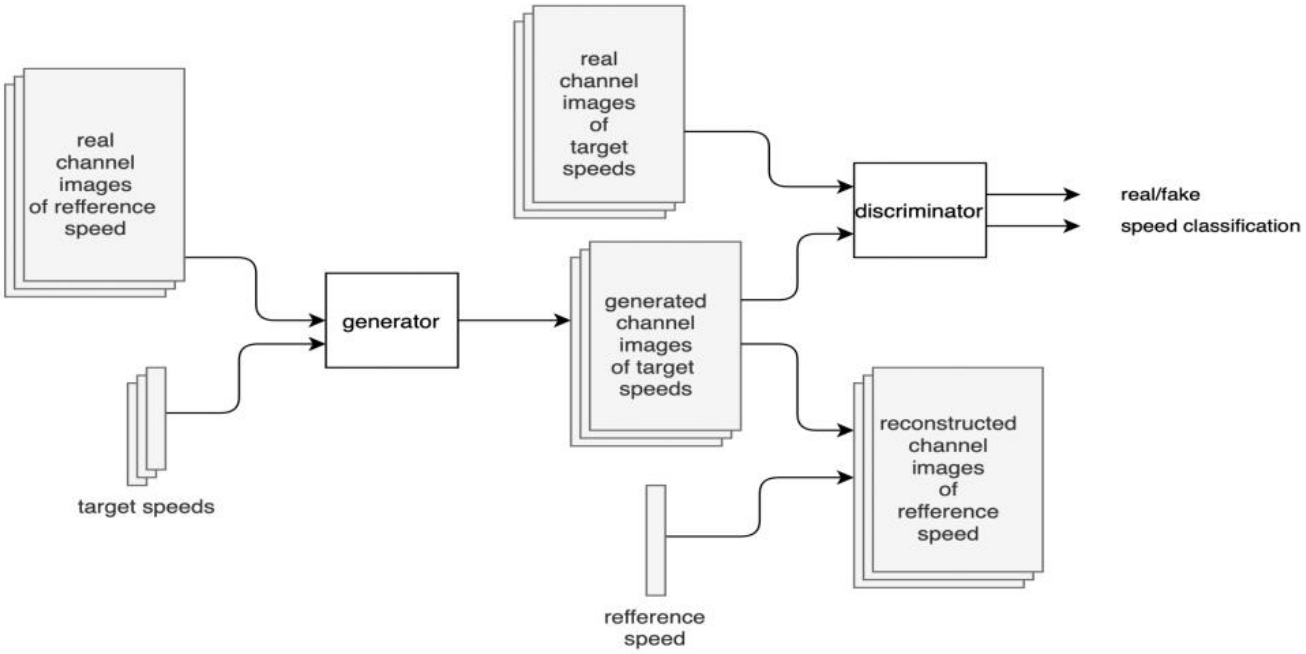


Fig. 5. Training procedure of StarGAN for converting channel images with user speed v_ref to channel images with user speeds v_1,v_2,…,v_n.

In essence, we are dealing with two datasets, and we intend to learn how to translate from the first domain to the second domain (without knowing the exact match between the dataset members). Therefore, we must take advantage of an unsupervised image-to-image translation technique, in which there are two sets of images, one consisting of the input images and the other consisting of the target images. No paired images are available to know how one image is translated to its corresponding output.

StarGAN is a structure that has been suggested for such settings to translate between multiple domains. StarGAN accepts as inputs a reference channel image and a label (of the desired target domain), and generates an image associated with the reference domain but in the target domain. The training procedure, which is basically similar to the adversarial training of GANs, is shown in Figure 5.

For our goal, starGAN is trained by feeding the generator with reference channel images with user speed v_{ref}, and labels of the target user speeds $v_1, v_2, ..., v_m$. The generator tries to generate channel images of target user speeds. The generated channel images, along with real target channel images, are fed into the discriminator. The discriminator has two tasks: first, to distinguish fake channel images from the real ones, and second to classify the channel images based on their labels (user speeds). To make sure the procedure preserves the correspondence between the reference and the target images, StarGAN should also train the generator to reconstruct the reference channel image. This time, the generated channel image (at the target speed) and the reference speed are given to the generator as inputs, and it has to generate an image that should be as close as possible to the original reference channel images (at v_{ref}).

When training is complete, we use the generator of the StarGAN; the reference channel image with user speed v_{ref} and the desired user speed are given to the generator. Afterwards, it generates a channel with adjusted user speed based on the desired speed.

4- 4- Evaluation Metric

As we mentioned in section 3.2, commonly used metrics for evaluation of the performance of the channel models are looking at the 1D channel response (either in frequency or in time).

In this work, we have also utilized these measures. Since the emphasis in this work is on generating 2-dimensional time-frequency responses of the channel, we have additionally presented a new metric to evaluate the pattern of the generated 2D images. The 2-dimensional ($m \times n$) time-frequency response provides frequency responses in consecutive time-slots. By putting along all the n frequency responses horizontally, we actually obtain a 1-dimensional quasi-periodic signal. Its first m elements are corresponding to the first time-slot, its second m elements are corresponding to the second time slot, and so on. This operation is repeated for all of the time-frequency responses. The resulting 1D sequences are denoted as $x_i[n]$ ($i = 1, 2, ..., N$), where N is the number of samples (time-frequency responses). We take the autocorrelation of this quasi-periodic signal($x_i[n]$) as:

$$R_{xx}(m) = E\left(x_i(n)\, x_i(n+m)\right) \tag{3}$$

To evaluate the similarity of the generated samples to the real measurements, one can consider the mean of the resulting autocorrelation functions over the samples. In Section 4.1, few of these mean autocorrelation functions are depicted, where we can verify a good match between the generated and measured samples.

Next we would like to obtain a quantitative comparison of the autocorrelation function results. To achieve this goal, we employ the real Cepstrom ($c[n]$) of the mean autocorrelation functions as:

$$c[n] = real[IFFT[log[abs[FFT[mean[R_{xx}(m)]]]]]] \quad (4)$$

The reason we choose Cepstrum is that we are looking for repeated features. Moreover, Cepstrum does two operations on $c[n]$: 1) compression, which makes the comparison tractable, and 2) spectral smoothing. Autocorrelation holds spectral information with respect to the Wiener-Kinchin theorem [26] that provides the relation between the power spectrum and the autocorrelation function. By taking Cepstrom of the autocorrelation, we smooth the spectrum, skip the redundant information, and keep the envelope which plays the main role for our comparison.

The similarity measure now can be defined as the Mean Squared Error (MSE) between the lower Cepstral coefficients for generated and measurement data.

5- Experimental Results

In this section, we describe the exact structure of the deep networks used for Channel Sample Generator and Speed Adaptation Network. We also discuss the training procedure and the results obtained from each network. The quality of the generated samples is then compared based on commonly used criteria as well as the introduced metric based on the Cepstrum of the mean autocorrelations.

5- 1- Channel Sample Generator

1) Network Structure: The Channel Sample Generator uses deep convolutional networks for both the discriminator and the generator of DCGAN. The structure of the generator and the discriminator networks are depicted in Figure 6 and Figure 7, respectively. The network is trained for 10 epochs for the total of 40000 channel images with bach size of 64. The size of the noise vector is 100, the learning rate is 0.0002 , and Adam optimizer with $\beta_1 = 0.5$ is used for optimization. The training parameters are listed in Table 1.

Table 1. Parameters Used During Training

(a) DCGAN network

Parameters	Value
Batch Size	64
Learning Rate	0.0002
Number of Epochs	10
Noise Vector Size	100
Optimizer	Adam ($\beta_1 = 0.5$)

(b) Parameters of Vienna LTE link simulator

Parameters	Value
Channel Type	ETU/EVA/PedA
Band Width	1.4 MHz
Simulation Type	SUMIMO
Carrier Frequency	2.1 GHz
Filtering	Fast fading
User Speed	ETU:50,EVA:80,PedA: 3 km/h

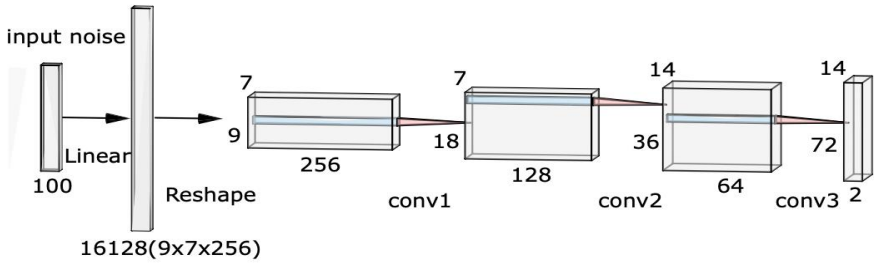


Fig. 6. Structure of DCGAN's generator network for channel modeling.

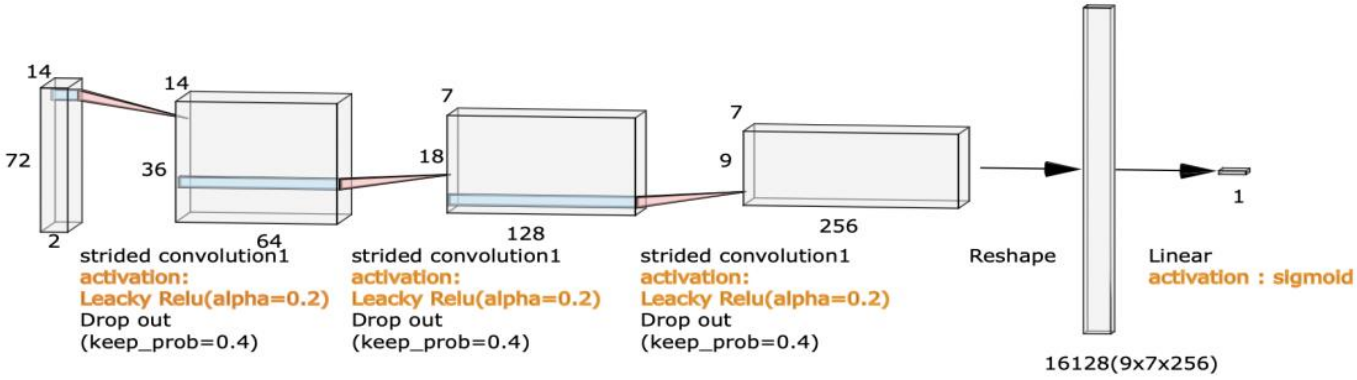


Fig. 7. Structure of DCGAN's discriminator network for channel modeling.

2) Training Data: To train the model, we should collect some channel measurements from the environment. The model should provide us samples which have similar statistics.

To evaluate the performance of our method, we have collected the training data in two different experiments.

- In the first experiment (which is mainly for verification purposes), instead of the actual channel measurements, we employ a simulator to generate channel samples. The samples generated by this simulator are then considered as the actual channel measurements, and our model aims to generate samples similar to these (virtually) measured samples. As we do not have the burden of actual channel measurements, we can easily simulate different environments and evaluate the performance of our model with this method.
- In the second experiment, and for a realistic scenario, we collect some channel samples from the environment and then model our environment using the proposed scheme.

For the first experiment, we use the Vienna link simulator [27] to generate our training data (time-frequency responses) for three types of channels: Extended Typical Urban model (ETU), Extended Vehicular A model (EVA), and Pedestrian A model (PedA) as the representatives of environments with low, medium, and high delay spread, respectively. In terms of channel complexity, as shown in Figure 8, ETU provides a more complex channel than EVA, and EVA is more complicated than PedA (i.e., the variations in the frequency axis are more significant). We trained our channel sample generator to model each of these distributions separately.

For each channel type, 40000 time-frequency responses of size 72×14 are sampled as the training data. The simulation parameters are listed in Table 1b.

For the second experiment, we use a commercially available Wi-Fi card equipped with Atheros AR9380 chip-set [28] to collect data. Atheros AR9380 is able to report back the Channel State Information (CSI), so it can be used as a cheap spectrum analyzer.

For this experiment, the transmitter and the receiver are placed about 3m apart. We walked between them, moved the transmitter and the receiver to change the channel between them, and generate distinct measurements so that they can be used as training data. The channel bandwidth was 20 MHz. With this bandwidth, Atheros chip-set reports the channel state for 56 sub-carriers. Thus, we will obtain 56 complex numbers in one CSI measurement for each transmission pair. We measured the CSI for a long time and selected 40000 time-frequency grids of size 56×14 from it.

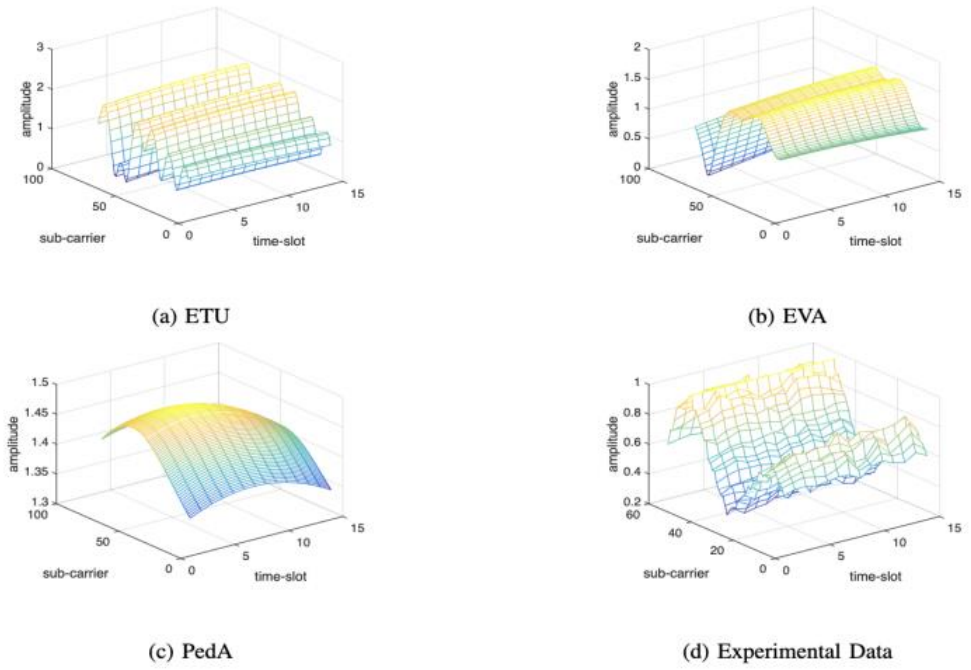


Fig. 8. A random sample of training data for (a) ETU, (b) EVA, (c) PedA simulated channels, and (d) experimental data.

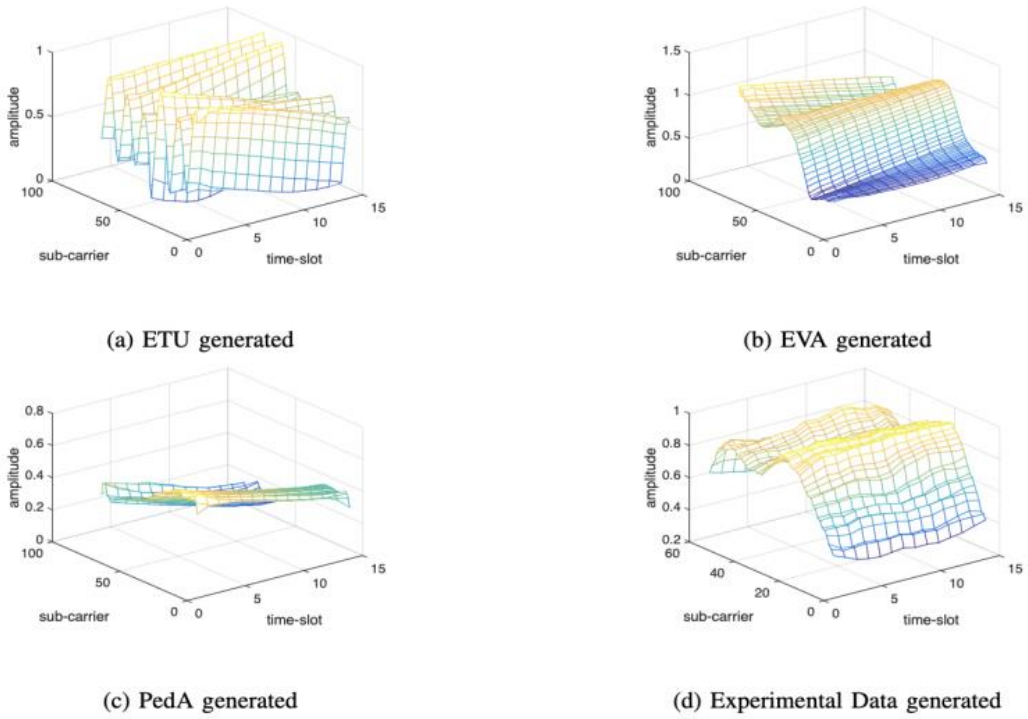


Fig. 9. A random sample of generated channels for (a) ETU, (b) EVA, (c) PedA, and (d) experimental data.

3) Performance Analysis: In this section, we first present a few samples of the generated time-frequency responses, and then verify the statistical similarity of them to the measurement data. We train a model for each of the datasets (i.e., ETU, EVA, PedA) and real experimental data. Afterwards, the model is used to generate channel images from the learned distribution of the training data. The generated channel images are then used to obtain the complex time-frequency responses of the channel. The absolute value of one sample of generated time-frequency responses for each case is shown in Figure 9. By comparing Figure 9 with Figure 8, visually, we can observe high analogy between the measured and generated data which verifies that they both come from a common distribution.

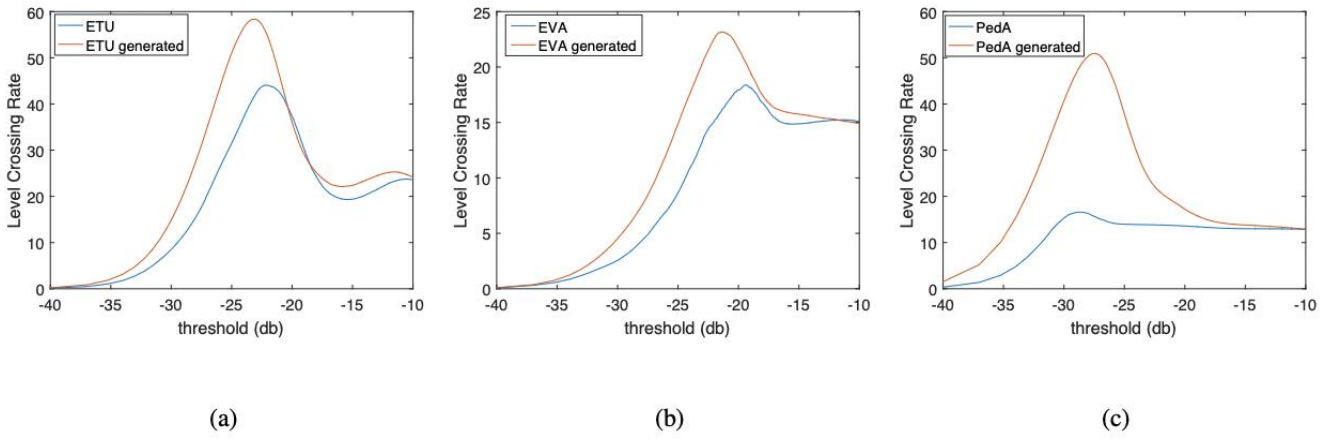


Fig. 10. Level Crossing Rate (LCR) of measurement and generated channels for three channel types:(a) ETU, (b) EVA, and (c) PedA.

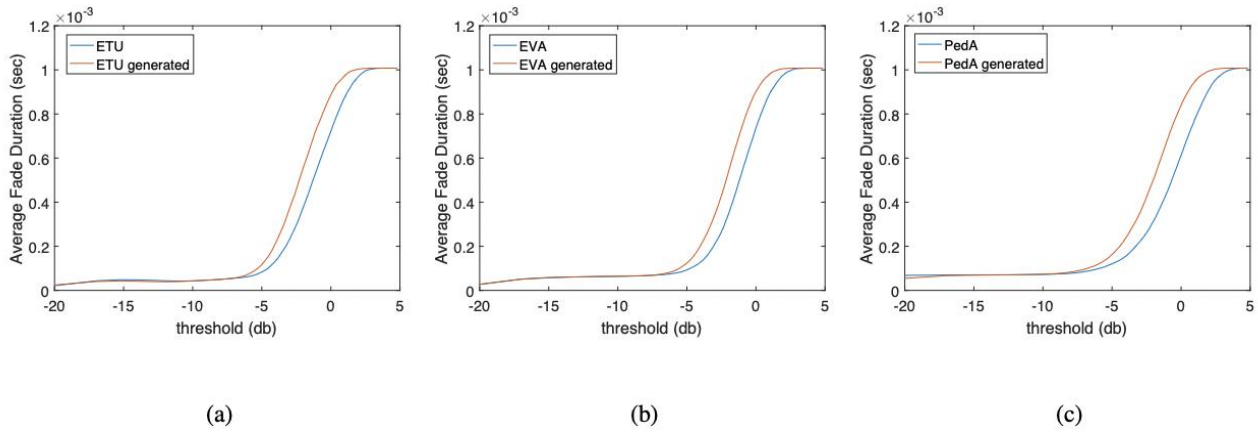


Fig. 11. Average Fade Duration (AFD) of measurement and generated channels for three channel types:(a) ETU, (b) EVA, and (c) PedA.

First, we evaluate our model with the metrics of the Level Crossing Rate (LCR) and Average Fade Duration (AFD). LCR measures the number of times a signal exceeds a certain level, and it shows how fast the fading is for that channel. AFD measures the average time in which a signal is below a specific power. These metrics are used for 1-dimensional temporal data while the time-frequency responses are two-dimensional. Therefore, our two-dimensional time-frequency responses have to be converted into 1-dimensional temporal data. For this purpose, we take the inverse fast Fourier transform from each column of time slots and put along all the n columns horizontally to have a 1D sequence.

The charts of LCR and AFD for three different channel types are illustrated in Figure 10 and Figure 11, respectively. The LCR for ETU and EVA channels demonstrate a good match in the location and amplitude of the pick. For the PedA channel, the location of picks is the same, but the amplitudes do not match. AFD for all channel types shows high similarity between measurement and generated channels.

Despite this pictorial similarity, which to some extend shows the performance of the proposed scheme, it is hard to prove the statistical similarity this way, especially because these metrics are 1-dimensional metrics. The 1-dimensional metrics are not capable of capturing and comparing the variations in the frequency domain of the channel responses, which are the main distinguishing feature between the different channel types that are represented as 2-dimensional time-frequency responses. It is one of the reasons that we did not base our performance evaluation on them and we have used the CDM metric. More importantly, although the LCR and AFD metrics show a good match between the generated and the actual samples, they are not able to discriminate between different environments, even for the actual samples. For example, in Figure 11, the AFD of the actual ETU and EVA samples are similar.

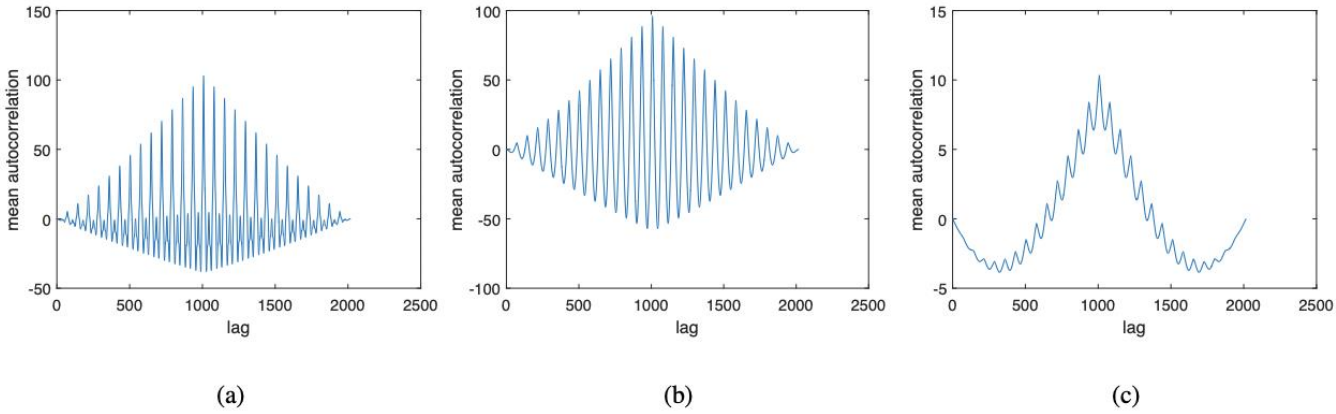


Fig. 12. The mean autocorrelation of the training data for three different channel types: (a) ETU, (b) EVA, and (c) PedA.

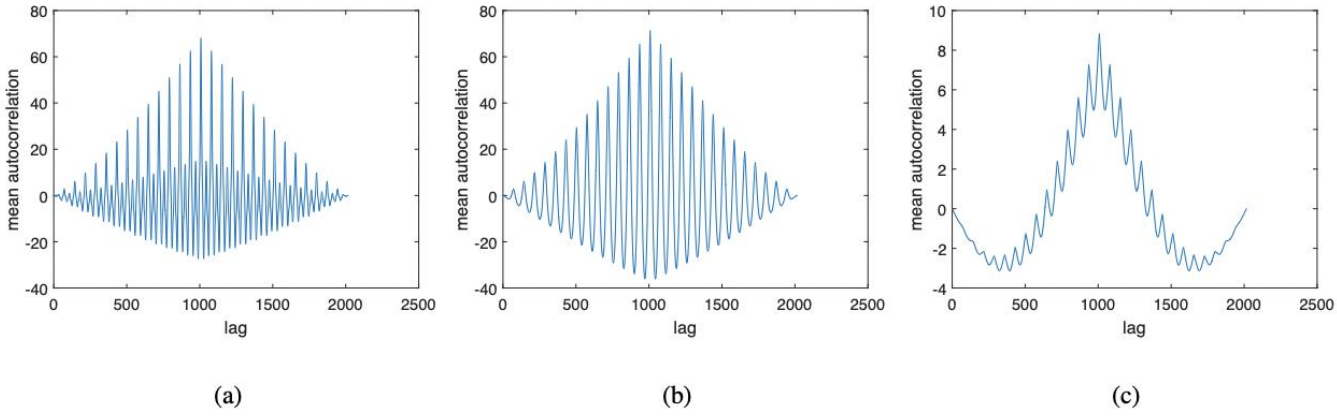


Fig. 13.The mean autocorrelation of the generated channels for three different channel types: (a) ETU, (b) EVA, and (c) PedA.

To show the good statistical closeness between the generated and actual samples, we use some commonly used metrics as well as the metric we introduced in Section IV-C.

The CDM in Section 4.3, is able to differentiate between different environments. As discussed before, we first compute the mean autocorrelation function of the actual and generated channel samples for this metric. Figure 12 shows the results for ETU, EVA, and PedA actual channel samples. In Figure 13 on the other hand, the mean autocorrelation of the generated channels is depicted. By comparing the two figures, it can be visually verified that the autocorrelation function of the generated channels of each type demonstrates a very high similarity to their corresponding autocorrelation function of the measured data. It can also be observed that different channel types resulted in different autocorrelation functions.

As mentioned in Section 4.3, to analyze beyond the visual comparison, we have used CDM for comparing the autocorrelation functions. The CDM for different channel types is listed in Table 2a. It is clear from the table that the distance between generated channels and their corresponding measurement is an order of magnitude less than the distance between generated channels of one type and measurements of different types.

We note that it would be interesting to compare the performance of the proposed scheme with the results of the previous studies. However, to the best of our knowledge, the 2-dimensional modeling approach which learns the statistical distribution of channel measurements in various frequencies is not studied before this work. In fact, other generative methods including [17], [18], [19] apply GANs to model 1-dimensional channel repossess. Due to this restriction, previous results cannot be used for 2D channel response modeling; thus, we are not able to compare our results with them.

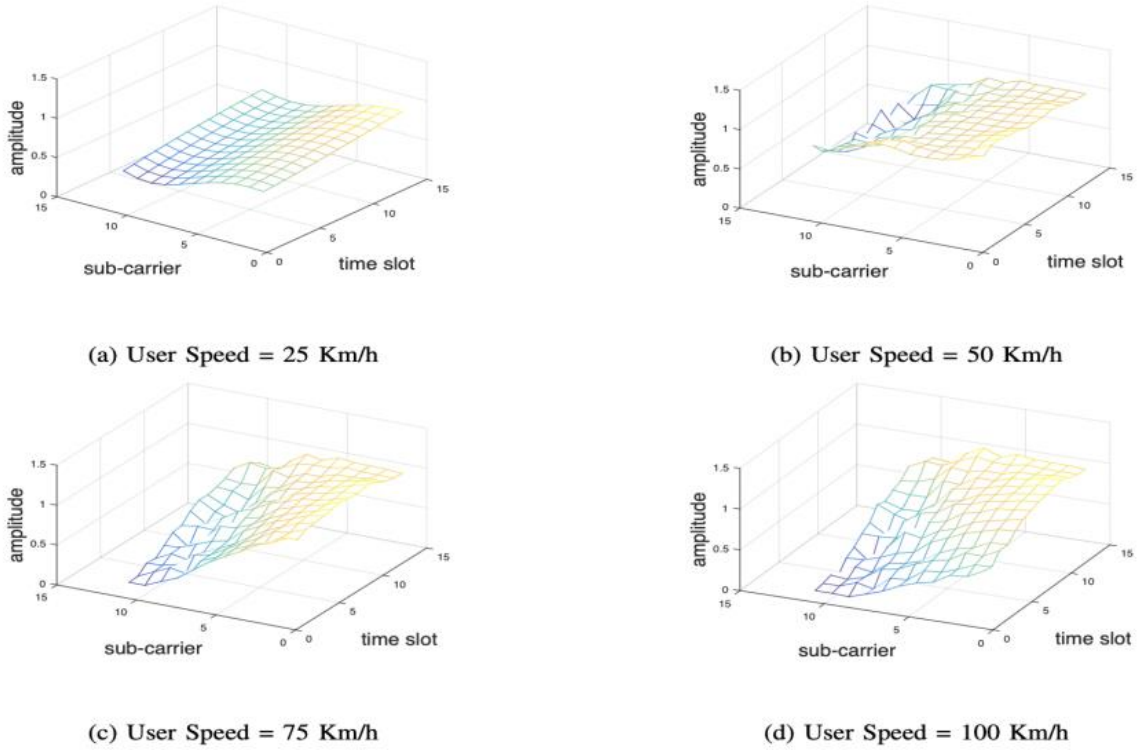


Fig. 14. A random resulting sample of the Speed Adaptation Network. (a) The input ETU channel with User Speed of 50 km/h. (b) The adapted ETU channel to the speed of 25 km/h. (c) The adapted ETU channel to the speed of 75 km/h. (d) The adapted ETU channel to the speed of 100 km/h.

5- 2- Speed Adaptation Network

Having the generated channel samples at the reference speed, the speed adaptation network will generate an equivalent sample for the desired user speed. The structure of the generator, discriminator, and loss functions are selected the same as the architecture of StarGAN [29].

To train and evaluate the results of the Speed Adaptation Network, we consider the ETU channel model. More specifically, StarGAN is trained considering ETU channel samples with the user speed of $50\ \mathrm{km/h}$ as the reference input, and ETU channel samples with the user speeds of $25,75,100\,km/h$ as the target channel images. The desired user speed is fed to the network as a One-hot vector.

Figure 14 illustrates the results of the Speed Adaptation Network when it got the channel image of 50 km/s as the input, and was instructed to generate samples with different user speeds. Figure 14(a) shows the input channel image for user speed of 50 km/s. Figure 14(b) is the resulting channel with the User Speed of 25 km/h. As can be seen from the figure, the variations in the time-axis have been reduced. Figures 14(c) and 14(d) are the resulting channels with the User Speeds of 75 km/h and 100 km/h, respectively. The increase in the variations of the time-axis is obvious. All the time-frequency responses are plotted for only 10 consecutive frequency sub-carriers so that the changes in the time axis can be seen more clearly.

We also note that the samples presented in Figure 14 are provided just for visual verification. The more accurate and precise evaluation which statistically evaluates the performance of the speed adaptation network based on the CDM metric, is presented in Table 2b.

To test the statistical similarity of the resulting 2D channel images, after obtaining the complex time-frequency responses of the channel from the channel images, first the 2D time-frequency responses are converted to one-dimensional sequences. Since the main effect of the different user speeds is on the time axis, this time we put along all the time slots of the subcarriers for this purpose (note that for evaluation of the Channel Sample Generator, all sub-carriers of the time-slots are put along). Afterwards, we use the mean autocorrelation and CDM. The mean autocorrelation of the actual and generated ETU channels for three different user speeds are shown in Figure 15 and Figure 16, respectively. By comparing these two figures, the similarity between the mean autocorrelation function of the generated channels of each speed with its corresponding mean, autocorrelation function of the actual channel samples can be verified. This shows the ability of the network in modeling the channels with different user speeds. We have also computed the CDM between the resulting mean autocorrelation functions. The results are listed in Table 2b. It is clear that the distance between generated channels and their corresponding measurement is on average an order of magni-

Table 2. Similarity Results

(a) Cepstral Distance Measure for different channel types.

	ETU Generated	EVA Generated	PedA Generated
ETU	4.12×10^{-5}	3.64×10^{-4}	1.00×10^{-3}
EVA	5.95×10^{-4}	7.98×10^{-6}	6.52×10^{-4}
PedA	13.00×10^{-3}	8.36×10^{-4}	5.37×10^{-7}

(b) Cepstral Distance Measure for different User Speeds.

	ETU Generated 25 km/h	EVA Generated 75 km/h	PedA Generated 100 km/h
ETU-25	1.56×10^{-5}	4.08×10^{-4}	8.89×10^{-4}
ETU-75	1.20×10^{-4}	2.98×10^{-5}	2.28×10^{-4}
ETU-100	4.99×10^{-4}	3.79×10^{-5}	1.43×10^{-5}

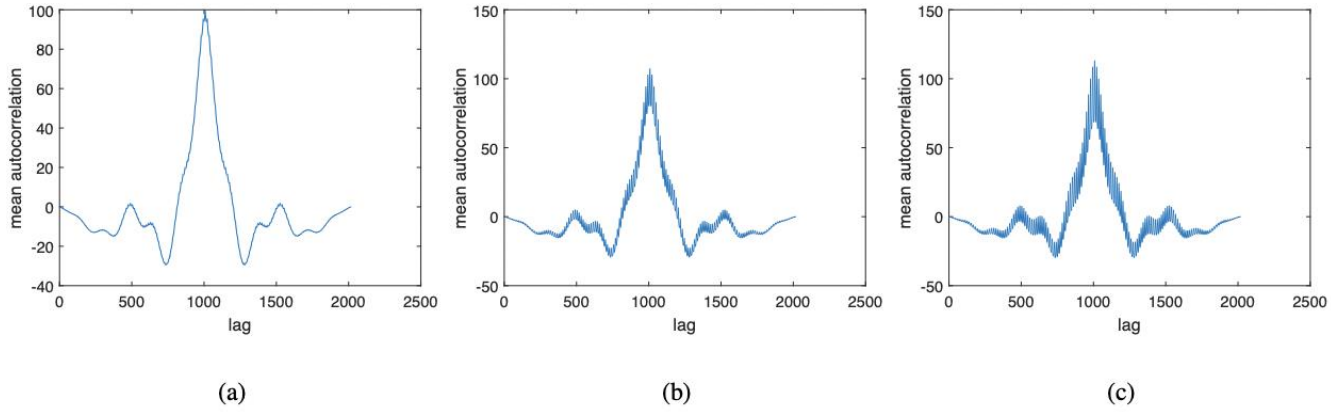


Fig. 15. The mean autocorrelation of the actual ETU channels with three User Speeds: (a) 25 km/h, (b) 75 km/h, (c) 100 km/h.

tude less than the distance between generated channels of one speed and measurements of different speeds. Note that for the speed of $100\,\mathrm{km/h}$, the CDM shows the best match between the generated and actual samples of $100\,\mathrm{km/h}$, however, the value is only 0.4 times the value for the speed of $75\,\mathrm{km/h}$. This might show that the channel statistics are not very different at high speeds.

6- Multiple-Input Multiple-Output (MIMO) Configuration

As mentioned earlier, the focus of this work is on modeling the statistics of the channel time-frequency grid between one transmit-receive antenna pair. In MIMO configurations, based on the assumption that the distance between antennas is large enough, they are considered independent so they do not affect each other. In addition, since all the antennas are in the same environment, the statistics of the channel between any two antennas are the same. Therefore, our model can be used for the MIMO setting and generate channels between any two pairs of antennas. For example, consider a 2×2 MIMO configuration, in which the channel matrix in antenna space is as:

$$H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \tag{5}$$

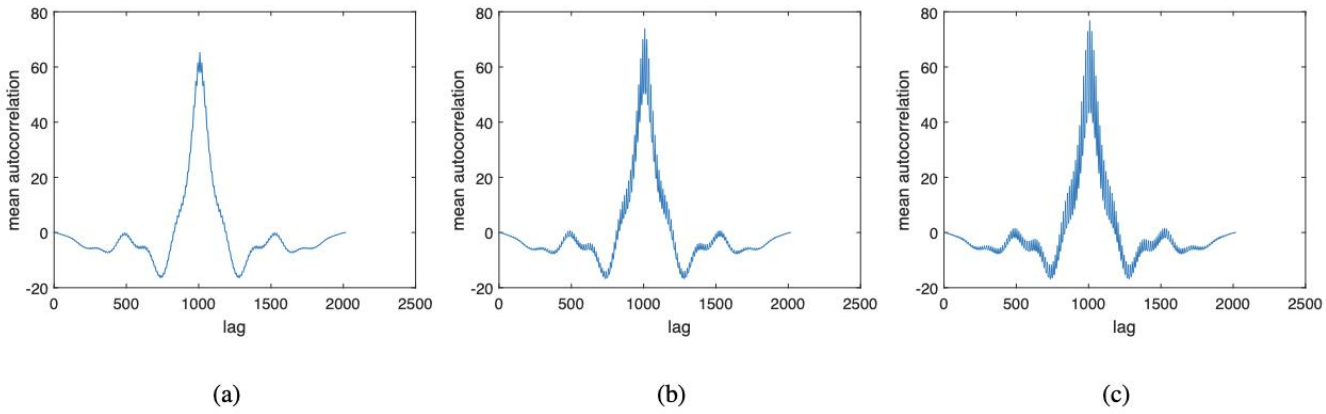


Fig. 16. The mean autocorrelation of the generated ETU channels with three User Speeds: (a) 25 km/h, (b) 75 km/h, (c) 100 km/h.

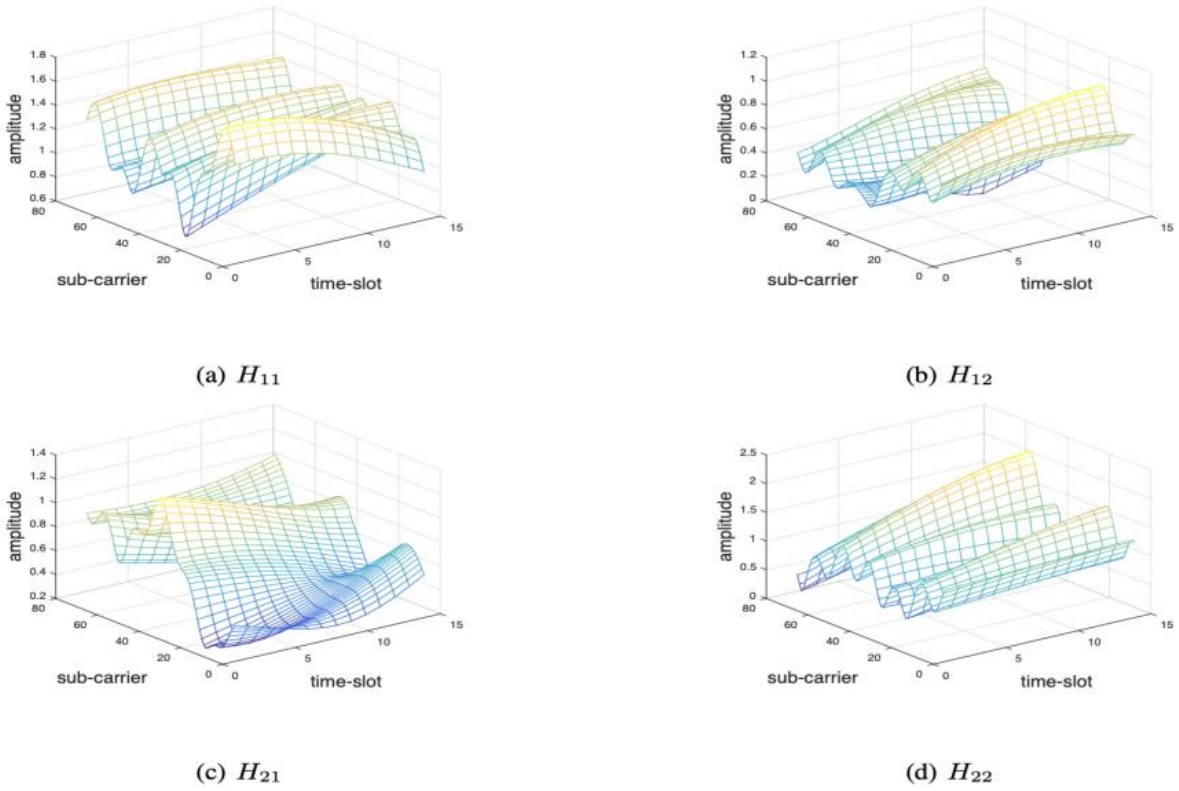


Fig. 17. One sample of generated channels for a 2 × 2 MIMO setting for an ETU environment. (a) the channel between the first receiver and the first transmitter, (b) the channel between the first receiver and the second transmitter, (c) the channel between the second receiver and the first transmitter, (d) the channel between the second receiver and the second transmitter.

where H_{11} is the channel between the first receiver and the first transmitter, H_{12} is the channel between the first receiver and the second transmitter, H_{21} is the channel between the second receiver and the first transmitter, and H_{22} is the channel between the second receiver and the second transmitter. As these channels are independent, a sample of the MIMO channel matrix can be generated by sampling from the proposed model four times independently. One sample of generated channels for such a MIMO setting for an ETU environment is illustrated in Figure 17.

Note that if the distance between the antennas is small that the antennas cannot be considered independent, the model has to be trained with the measurement data of channels between every antenna pair (i.e. a higher dimensional channel image as the training data); although, this case is not studied in this paper.

7- Conclusion

In this paper, a novel propagation channel modeling method based on Deep Learning techniques is presented. The time-frequency response of the propagation channel is considered as an image, and the distribution of channel images is modeled using DCGANs. Moreover, the model is extended for measurements having different user speeds. A speed adaptation network is trained to learn the effect of different user speeds on channel images. We take advantage of StarGAN as an image-to-image translation technique for this purpose. The statistical similarity between the generated and the actual samples are examined with some commonly used metrics such as LCR and AFD, as well as a newly introduced metric based on the Cepstral distance between the mean of the autocorrelation functions.

References

[1] Wang, C.X, Bian, J., Sun, J., *et al*: `A survey of 5g channel measurements and models', in *IEEE Communications Surveys and Tutorials*, 2018, **20**, pp.~3142--3168.

[2] Degli-Esposti, V., Fuschini, F., Vitucci., V.M., *et al*, "Speed-Up Techniques for Ray Tracing Field Prediction Models", *IEEE Transactions on Antennas and Propagation*, 2009, **57**, pp.~ 1469-1480.

[3] 3GPP. " Spatial Channel Model for MIMO simulations," 2003, [online] available: https://www.3gpp.org/

[4] Baum, D.S., Hansen, J., Salo, J., *et al*, "An interim channel model for beyond-3G systems: extending the 3GPP Spatial Channel Model (SCM)," *2005 IEEE 61st Vehicular Technology Conference*, Stockholm, 2005, **5**, pp.~3132-3136.

[5] Thomas, T.A., Vook, F.W., Mellios, E., *et al*, "3D Extension of the 3GPP/ITU Channel Model", *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, Dresden, 2013, pp.~1-5.

[6] Wen, C., Shih, W., and Jin, S., "Deep Learning for Massive MIMO CSI Feedback," *IEEE Wireless Communications Letters*, Oct. 2018, **7**, (5), pp.~748-751.

[7] N. Farsad, M. Rao and A. Goldsmith, "Deep Learning for Joint Source-Channel Coding of Text," 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, 2018, pp. 2326-2330.

[8] Ye, H., Li, G.Y., and Juang, B., "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems," *IEEE Wireless Communications Letters*, Feb. 2018, **7**, (1), pp. 114-117.

[9] Neumann, D., Wiese, T., Utschick, W., "Learning the MMSE channel estimator", *IEEE Transactions on Signal Processing*, 2018, **66**, (11), pp. 2905-2917.

[10] Luo, C., Ji, J., Wang, Q.,*et al*, "Channel state information prediction for 5G wireless communications: A deep learning approach. *IEEE Transactions on Network Science and Engineering.*, 2018.

[11] Zhang, Y., Wen, J., Yang, G., *et al*, " Air-to-Air path loss prediction based on Machine Learning methods in urban environments", *Wireless Communications and Mobile Computing*, 2018, **2018**, pp. 1-9.

[12] Lu, T., Sun, J., Wu, K., *et al* "High-Speed Channel Modeling With Machine Learning Methods for Signal Integrity Analysis," *IEEE Transactions on Electromagnetic Compatibility*, 2018, **60**, (6), pp. 1957-1964.

[13] Lee, C., Yilmaz, H.B., Chae, C., *et al*, " Machine Learning based channel modeling for molecular MIMO communications," *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Sapporo, 2017, pp. 1-5.

[14] Navabi, S., Wang, C., Bursalioglu, O.Y. and Papadopoulos, H., "Predicting wireless channel features using neural networks." In 2018 IEEE international conference on communications (ICC), May 2018, pp. 1-6.

[15] Jiang, Z., Chen, S., Molisch, A.F., Vannithamby, R., Zhou, S. and Niu, Z., "Exploiting wireless channel state information structures beyond linear correlations: A deep learning approach", *IEEE Communications Magazine*, **57**, (3), pp. 28-34.

[16] Ye, H., Li, G.Y., Juang, B.H.F. and Sivanesan, K., "Channel agnostic end-to-end learning based communication systems with conditional GAN", In 2018 IEEE Globecom Workshops (GC Wkshps), December 2018, pp. 1-5.

[17] Yang Yang, Yang Li, Wuxiong Zhang, *et al*, Generative-Adversarial-Network-Based Wireless Channel Modeling: Challenges and Opportunities. *IEEE Commun. Mag. 57(3): 22-27 (2019)*

[18] Qianqian Zhang, Aidin Ferdowsi, Walid Saad, *et al*, Distributed Conditional Generative Adversarial Networks (GANs) for Data-Driven Millimeter Wave Communications in UAV Networks. *CoRR abs/2102.01751 (2021)*

[19] Timothy J. O'Shea, Tamoghna Roy, Nathan E. West, Approximating the Void: Learning Stochastic Channel Models from Observation with Variational Generative Adversarial Networks. *ICNC 2019: 681-686*

[20] Goodfellow, Ian and Pouget-Abadie, Jean and Mirza, *et al*, "Generative Adversarial Nets," in Ghahramani, Z., Welling, M., Cortes, C., *et al*, "Advances in Neural Information Processing Systems 27," (Curran Associates, Inc., 2014), pp. 2672-2680.

[21] Radford, A., Metz, L., Chintala, S., "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," arXive preprint arXiv:1511.06434, 2015.

[22] Mirza, M., Osindero, S., "conditional generative adversarial nets," arXie preprint arXiv:1411.1784, 2014.

[23] Berthelot, D., Schumm, T., Metz, L., "BEGAN: Boundary Equilibrium Generative Adversarial

Networks," arXive preprint arXiv:1703.10717, 2017.

[24] Gulrajani, Ishaan and Ahmed, Faruk and Arjovsky, *et al*, "Improved Training of Wasserstein GANs," in Guyon, I., Luxburg, U.V., Bengio, S., *et al*, "Advances in Neural Information Processing Systems 30," (Curran Associates, Inc., 2017), pp. 5767-5777.

[25] He, H., Wen, C., Jin, S., *et al*, "Deep Learning-Based Channel Estimation for Beamspace mmWave Massive MIMO Systems,"*IEEE Wireless Communications Letters*, 2018, **7**, (5), pp. 852-855.

[26] Yates, R.D., Goodman, D.J., "Probability and Stochastic Processes: A Friendly Introduction for Electrical and Computer Engineers, 3rd Edition," (Wiley, J., Sons, 2014)

[27] Rupp, M., Schwarz, S., Tarantez, M., "The Vienna LTE-advanced simulators." (Springer, 2016)

[28] Xie, Y., Li, Z., Li, M., "Precise Power Delay Profiling with Commodity Wi-Fi,"*IEEE Transactions on Mobile Computing*, 2019, **18**, (6), pp. 1342-1355.

[29] Choi, Y., Choi, M., Kim, M., *et al*, "StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 2018, pp. 8789-8797.

HOW TO CITE THIS ARTICLE

Sh. Seyedsalehi, Generative Adversarial Networks for Propagation Channel Modeling, AUT J. Model. Simul., 53 (2) (2021) 217-234.

DOI: 10.22060/miscj.2022.20174.5250