Towards Massive MIMO Channel Emulation: Channel Accuracy Versus Implementation Resources

Pengda Huang, Matthew Jordan Tonnemacher, Yongjiu Du, *Member, IEEE*, Dinesh Rajan, *Senior Member, IEEE*, and Joseph Camp, *Member, IEEE*

***Abstract*—Channel emulation is widely used to control wireless channels and replay scenarios for repeatable experimentation and evaluation. Characterizing the propagation of massive Multiple-Input Multiple-Output (MIMO) systems, is challenging due to the rapid scaling of the number of channels needed to create representative scenarios as well as accurately modeling complex interdepedencies that exist within the channel matrix. Nonetheless, having the ability to emulate massive MIMO channels is critical for the development and testing of next-generation cellular protocols and algorithms. To achieve such scale for massive MIMO channel emulation solutions, in this paper, we study the tradeoff in channel emulation fidelity versus the hardware resources consumed using both analytical modeling and FPGA-based implementations. To reduce the memory footprint of our design, we optimize our channel emulation using an iterative structure to generate geometry-based channels at scale. In addition, the effects of varying the channel update rate and word length selection are evaluated in the paper and can significantly improve the implementation efficiency on an FPGA.**

***Index Terms*—Channel emulation, geometry-based channel, FPGA, massive MIMO, mmWave.**

I. Introduction

CHANNEL emulation is a widely-used tool in link and network performance evaluation since it offers repeatability and control of complex propagation environments for wireless hardware implementations. One common approach for accurately characterizing channels in simulation environments is geometry-based channel modeling [1], [2], which can be used for evaluating and analyzing the performance of mmWave systems in 5G and future networks [3]. To evaluate and analyze performance of wireless systems, accurate emulation of the geometry-based channels is important. Via emulation, we are able to investigate advanced algorithms, such as hybrid-precoding [4], multi-user (MU)-MIMO [5], and ultra-dense cell planning [6].

Manuscript received July 28, 2019; revised November 13, 2019 and January 9, 2020; accepted February 17, 2020. Date of publication March 13, 2020; date of current version May 14, 2020. This work was supported by the U.S. National Science Foundation under Grants CNS-0958436, CNS-1040429, CNS-1150215, CNS-1526269, and CNS-1823304. The review of this paper was coordinated by Prof. D. B. da Costa. *(Corresponding author: Pengda Huang.)*

Pengda Huang and Dinesh Rajan are with the Department of Electrical Engineering, Southern Methodist University, Dallas, TX 75206 USA (e-mail: p.huang.2013@ieee.org; rajand@smu.edu).

Matthew Jordan Tonnemacher is with the Samsung Research America, Mountain View, CA 94043 USA (e-mail: mtonnemach@smu.edu).

Yongjiu Du and Joseph Camp are with the Southern Methodist University, Dallas, TX 75205 USA (e-mail: ydu@smu.edu; camp@smu.edu).

Digital Object Identifier 10.1109/TVT.2020.2980583

Several geometry-based channel simulators have been reported in literature, for instance, [7] and [8]. These software-based simulators are able to simulate the geometry-based channels, which can be used to generate the 3 rd Generation Partnership Project (3GPP) mmWave channels [9] and also other mmWave channels. The simulator in [10], [11] specifically provides geometry-based channel models standardized by 3GPP and mm-Wave based Mobile Radio Access Networks for 5G Integrated Communications (mmMAGIC). As another example of the software based simulation, [12] considers channels where both the transmitter and receiver are mobile.

However, geometry-based channel emulation on hardware platforms has not been reported widely. For example, to the best of our knowledge, only *one* method has been used to emulate geometry-based stochastic channel on an FPGA [13], where discrete prolate spheroidal (DPS) sequences approximate the summation of sinusoids. This channel generation [13] is conducted on discrete-time segments, rather than continuously, due to the inherent feature of DPS sequences. The discontinuity implies that extra operations are needed to align the boundaries of the channel segments and the transmitted packets. However, the boundaries of the transmitted data packets are not exactly known in practice. Thus, the channel generation based on the DPS method causes additional error to the emulation of wireless systems. While geometry-based channel emulation is a new topic, conventional hardware-based channel generation has been widely studied. Due to their extensive use and low price, FPGAs are often selected as the hardware platform for fading channel emulation [14]–[16]. Rao *et al.* [14] developed an architecture for VLSI implementation of mobile wireless communication channels based on Xiao's improvement [17] to the Jakes' model. All of these works consider the emulation of a single link and do not focus on the scalability of their solutions, nor on the optimization of hardware resources.

Eslami *et al.* [18] presented a design and implementation of a frequency-based, scalable channel emulator based on an FFT/IFFT fading channel generation method where the FFT was implemented using the CORDIC algorithm. Such a fading channel emulation solution has two major drawbacks. First, the generated channel data length is fixed, which is determined by the size of the FFT. Thus, the scalability of the emulator is limited in terms of data generation flexibility. Second, although the CORDIC algorithm can save hardware resources, the computational speed is relatively low. Considering the large memory consumption of FFT-based generation, researchers proposed an

improved FFT-based method that shortens the FFT size and maintains the time correlation between channel samples [19].

In some scenarios, such as complex wireless environments and large-scale MIMO systems, a large number of channels might be needed. Thus, scability is an important factor in evaluating a channel emulator. Buscemi and Sass [20] described the feasibility of a scalable wireless channel emulator based on a 64-FPGA cluster, but hardware costs would preclude widespread usability. Others have addressed scalable network emulator design, but with a focus on emulator architecture without details on link-level channels [21], [22]. A wireless channel emulator that uses radio frequency front ends and a user interface but without baseband channel generation is proposed in [23].

An approach based on filtering a white Gaussian process generated 3×4 MIMO channels but did not use geometric emulation [24]. Filtering methods can be used only if the spectrum of the random process is precisely known. Indeed, the spectrum of the geometrical channels are typically diverse, precluding a filtering-only approach. For the same reason, the MIMO channel generation method proposed in Eslami [18] can not be applied in the geometry-based case. The generation of 4×4 MIMO channels on an FPGA was introduced in [25], where multiple lookup tables are required for the sinusoids used in channel generation. Compared with the FPGA-based generation of conventional statistical channels, geometry-based channel emulation requires significantly greater computation and hardware resources. Especially for the scale of massive MIMO systems, computationally-efficient emulation of geometry-based channel modeling becomes extremely challenging for the following three reasons.

First, as introduced above, the majority of existing studies on channel emulation mainly use statistical modeling for channel generation [25], [26]. In many MIMO systems, a very large number of antennas are used to implement beamforming to achieve propagation characteristics similar to today's cellular networks. The resulting high level of beamforming directionality makes geometrical models a more appropriate choice for characterizing the MIMO channel as opposed to statistical models [9] since the geometry-based models explicitly capture the features of single propagation rays rather the statistical features of a bundle of rays. Furthermore, the significantly-increased fading rate in mmWave bands (a motivating example of massive MIMO) reduces the number of the rays arriving at the receiver, which makes statistical description less meaningful.

Second, in a geometry-based channel model, the number of the required sinusoids becomes astonishingly large. For example, consider a MIMO system in which a base station is equipped with 32 antenna elements. There are 10 scattering clusters between the transmitter and the receiver. In each cluster, 20 rays exist. According to the 3GPP standard, $10 \times 20 \times 2$ sinusoids are needed for emulating the fading channel between one antenna element and the user equipment, where '$\times 2$' (in $10 \times 20 \times 2$ sinusoids) represents the sinusoids for In-phase and Quadrature components of a fading channel. For 32 antenna elements, 12,800 sinusoids are needed for a link between the base station and the user equipment.

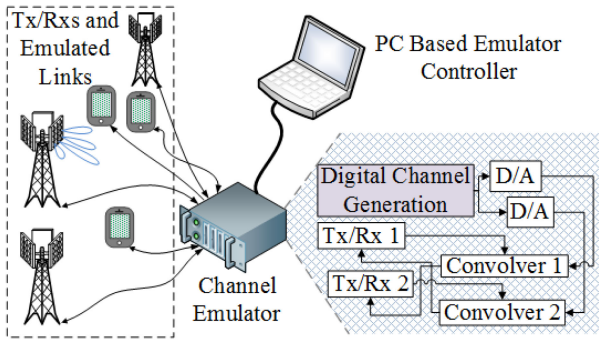

Fig. 1. Scalable wireless network emulation by exploiting knowledge of channel fidelity versus implementation resource costs.

Third, the emulators with the greatest channel fidelity, such as the Spirent VR5 HD spatial channel emulator [27] and Azimuth ACE-MX-4 channel fading emulator [28], often lack the ability to scale to the number of channels in a massive MIMO context. Our goal is to use geometry-based channel emulation techniques to save hardware resources without reducing the accuracy of the emulated channels. There have been some efforts to build low-fidelity emulators, often using FPGAs and/or DSPs (*e.g.*, up to 4×4 MIMO [25]). However, to enable massive MIMO emulation with a high number of dimensions at high fidelity, the relationship between emulated channel accuracy and hardware resource cost needs to be fully characterized and understood. As a result, the channel emulation fidelity-resources tradeoff guides the management of communication system evaluation, especially when the system scales to massive MIMO contexts.

In this paper, efficient emulation of geometry-based channels is investigated. To this end, we reduce the number of sinusoids without significantly sacrificing the fidelity of the generated channels and study the tradeoff in channel emulation accuracy versus the implementation resources consumed, using both analytical modeling and FPGA-based implementation. We optimize our channel emulation using an iterative structure to generate the sinusoids with a reduced memory footprint. The channel update rate and word-length selection are also analyzed to improve implementation efficiency. We then leverage these findings to efficiently emulate a single wireless path as a fundamental building block to scale to a massive MIMO geometry-based channel emulator, developing a metric for implementation complexity along the way.

To envision how the emulation would scale to massive MIMO links and networks, Fig. 1 presents a general framework for our system design. A large number of wireless transmitters and receivers are physically connected to the bank of hardware resources necessary to implement the geometry-based channel emulator. A control PC facilitates the design and control of the experiment. On the back end, FPGAs implement the channel emulation from the RF front end to the channel output. To enable these FPGAs to generate a large number of channels and know the amount of hardware required, the implementation's resource consumption must be understood and optimized according to the geometry-based channel emulation dimensions. We make the following contributions:

1) A geometry-based channel generator is proposed and built using FPGAs. In geometry-based channel modeling, hundreds of rays exist in a link connecting two antenna elements at the transmitter and receiver, respectively. To efficiently generate a large number of rays, we use a bank of orthogonal bases to represent the rays without sacrificing channel accuracy.
2) We design and implement an Autoregressive (AR) model based generator. Prior channel generation solutions employing Sum-Of-Sinusoids (SOS) demand large volumes of memory to store quantized cosine values. Since the memory resources available on FPGAs are limited, an AR-based fading channel generator using a second-order iterative structure is proposed and implemented. In the generation of one-path channels consisting of 20 sinusoid components, the iterative structure reduces the amount of memory by 95% compared to traditional lookup tables, thereby significantly enhancing the scalability of channel emulation.
3) We jointly analyze the effect of channel update rate and word length on channel emulation. The choice of word length and channel update rate affects hardware resource on an FPGA. We design an optimization algorithm to jointly determine word length and update rate to reduce hardware resource consumption. Experiments show that our algorithm generates channels with greater accuracy in terms of mean square error versus channels generated with fixed word length and update rate, an advantage that becomes even more significant with small hardware resource budgets.
4) Lastly, we study the scalability of the geometry-based channel emulator for massive MIMO systems. Based on our analysis and experimental evaluation of a single path, the capability of massive-MIMO dimensionality according to a desired level of channel emulation accuracy is demonstrated. Specifically, a Virtex-4 FPGA is able to generate 376 channel paths, which is 20 times the number of paths generated using lookup tables. In other words, when channel paths of the same scale are generated, the need for 19 out of 20 FPGAs is eliminated.

In the remainder of this paper, the basics of a geometry-based channel modeling is first introduced in Section II. In Section III, the projection of the original geometry-based channel to an orthogonal space is analyzed. In Section IV, the novel generation scheme is presented, which is based on a second-order AR model. Section V discusses the word length and channel status update rate optimization. Section VI includes the evaluation of our approach via FPGA implementation. Section VII discusses the application of our work to a massive MIMO geometry-based channel emulator. Finally, conclusions are presented in Section VIII. A summary of key notations used is given in Table I.

TABLE I
SUMMARY OF KEY NOTATIONS

Notation	Definition
$H_{u,s}$:	Channel between TX s and RX u antennas
f_k:	Freq. of k -th component of $H_{u,s}$
f_D:	Maximum Doppler frequency
$\tilde{H}_{u,s}$:	Chan. regenerated from projected L-dim space
$\mathcal{G}_L$:	Projected L-dimensional space
f_l^b:	Freq. of l-th component of $\mathcal{G}_L$
G_l:	Average gain of l-th component
Δt:	Channel generation time duration
$n_{\Delta t}$:	Number of the channel samples during Δt
$\delta\omega$:	Doppler frequency resolution
y_{Il}, y_{Ql}:	Cosine and sine sinusoid at l-th path
T_W:	Channel update period
T'_W:	Normalized update period, $T'_W = T_W f_D$
P:	Word length of a channel sample
$e_{\delta t}$:	Time discontinuity caused channel error
$e_{\delta y}$:	Finite word length caused channel error
β:	Error summation of $e_{\delta t}$ and $e_{\delta y}$
$\boldsymbol{C}, \tilde{\boldsymbol{C}}$:	Original and regenerated MIMO chan.
$\Delta\boldsymbol{C}$:	Channel difference between $\boldsymbol{C}$ and $\tilde{\boldsymbol{C}}$
$\lambda_{\Delta\boldsymbol{C}}$:	Eigenvalue of $\Delta\boldsymbol{C}$

II. BACKGROUND: MASSIVE-MIMO CHANNEL MODEL

According to the 3GPP standards [29], a geometry-based channel is described by (1) shown at the bottom of the next page, from which we now explain each term. The term (a) characterizes the radio field pattern of the receive antenna element u. The zenith arrival angle of the m-th ray in the cluster n is denoted by $\theta_{n,m,ZOA}$. The arrival angle variable $\phi_{n,m,AOA}$ is the azimuth arrival angle for the same ray. The zenith and azimuth departure angle variables are $\theta_{n,m,ZOD}$ and $\phi_{n,m,AOD}$ respectively. The variable P_n denotes the power of the n-th cluster, variable M_n denotes the number of all path scattering in the n-th cluster, and the $F(\cdot)$ represents the field components of the radiation pattern. The term (b) is for setting the initial phase for each ray in all clusters, which is uniformly distributed. The cross polarization power ratio in linear scale is $\kappa_{n,m}$. The phase variables $\Phi^{\theta\theta}$, $\Phi^{\theta\phi}$, $\Phi^{\phi\theta}$, and $\Phi^{\phi\phi}$ are the random initial phase for ray m of cluster n for four different polarization combinations: $\theta\theta$, $\theta\phi$, $\phi\theta$, and $\phi\phi$, respectively. Term (c) represents the radio field patterns of transmit antenna element u. Term (d) characterizes the phase delay caused by the propagation from cluster n to the receiver antenna element u. The wavelength of the carrier frequency is λ_0. The spherical unit vector with azimuth angle is $\hat{r}$. The location vector of receive antenna element u is $\bar{d}_{rx,u}$. Term (e) is the propagation phase delay between cluster n and transmit antenna element s. The Doppler shift on the m-th ray in cluster n is denoted by $\nu_{n,m}$. Additional details about geometry-based channel modeling can be found in the 3GPP standards documentation [29].

According to (1), a geometry-based channel model contains a large amount of rays. For example, to generate a channel model with 10 clusters containing 20 rays, 400 sinusoids need to be generated. The sum-of-sinusoids (SOS) method is widely used in channel generation on hardware platforms within the

exponential function of time in (1). To generate multiple sinusoids, we control the amplitude, frequency, and initial phase of each sinusoid to obtain a geometry-based channel. In the literature, the generation of sinusoids are typically implemented via sine/cosine lookup tables [30]–[32]. For the simulation of traditional fading channel models, the lookup table strategy is applicable since only a small number of sinusoids are needed. For implementing geometry-based channel generation, the use of lookup tables is challenging due to its high demand on memory resources.

III. Evolution of the Geometry-Based MIMO Channel Generation

As discussed in the previous section, the primary challenge in generating a geometry-based massive MIMO channel is that a large number of sinusoids need to be generated on the hardware platform. To reduce the number of hardware resources needed, we project the large number of sinusoids to a lower-dimensional Euclidean space. This lower-dimensional space is spanned by a small group of sinusoids with equal-frequency separation, which are orthogonal to each other. After the projection, the geometry-based channels can be generated within the projected space.

A. Projection of Original Geometry-Based Channel to Small-Dimensional Space

Notice that (1) represents a fading channel of the n-th cluster, where $n \in \{1, 2, \ldots, N\}$, and N represents the total number of clusters. For simplicity, we transform the channel $H_{u,s}$ by linking the s-th TX antenna element and u-th RX antenna element via the explicit computation of the matrix product in (1), resulting in:

$$\begin{aligned} H_{u,s}(t) &= \sum_{n=1}^{N} H_{u,s,n}(t) \\ &= \sum_{n=1}^{N} \sum_{m_n=1}^{M_n} A_{m_n} \cos\left(2\pi f_{m_n} t + \phi_{m_n}\right). \end{aligned} \tag{2}$$

Here, m_n is the m-th path scattering on the n-th cluster, and M_n denotes the number of all path scattering on the n-th cluster, which is denoted by f_{0_n}.

In (2), each sinusoid is fully determined by three elements: the amplitude A_{m_n}, the frequency f_{m_n}, and the initial phase ϕ_{m_n}. The three elements form a combination (A, f, ϕ), and each combination maps to a sinusoid. For example, the combination mapped to the m_n-th sinusoid is written as $(A_{m_n}, f_{m_n}, \phi_{m_n})$. We define the variable Υ, which is the set of all the combinations, $\Upsilon = \{(A_{m_n}, f_{m_n}, \phi_{m_n})\}$.

We reorder elements in Υ according to increasing value of f_{m_n}. After the reordering, (2) is rewritten as follows:

$$H_{u,s}(t) = \sum_{k=1}^{K} A_k \cos\left(2\pi f_k t + \phi_k\right), \tag{3}$$

where $K = \sum_{n=1}^{N} M_n$, and $f_{k_1} \leq f_{k_2}$ for all $k_1 < k_2$, $k_1, k_2 \in \{1, 2, \ldots, K\}$.

As analyzed in Section II, K is a large value for a geometry-based massive MIMO channel, which is a problem for FPGA implementation. To solve the problem, we project the K sinusoids to an L-dimensional space, where $L \ll K$. The L-dimensional space $\mathcal{G}_L$, is spanned by L bases:

$$\mathcal{G}_L = \underset{l \in \mathcal{L}}{span} \left\{\cos(2\pi f_l^b t + \psi_l)\right\} \tag{4}$$

Here, $l \in \mathcal{L}$, $\mathcal{L} = \{1, 2, \ldots, L\}$, $\Delta f = f_{l+1}^b - f_l^b = \frac{f_D}{L}$, where f_l^b denotes the frequency of the l-th basis in $\mathcal{G}_L$, and f_D is the maximum Doppler value, where ψ_l is randomly generated with a uniform distribution.

B. $K \rightarrow L$-Dimensional Space Projection

The K-to-L dimensional space projection is implemented via an inner product. We calculate the projection of $H_{u,s}(t)$ with each basis signal of the L-dimensional space to obtain

$$G_l = < H_{u,s}(t), \cos(2\pi f_l^b t + \psi_l) > |_{\Delta t}. \tag{5}$$

Here, $< \cdot, \cdot >$ denotes the inner product operation, and Δt represents the time duration over which the inner product is calculated. At the beginning of each segment of Δt, G_l is calculated.

We know that G_l will be the summation of sinusoids at the frequencies of $(f_k - f_l)$, $k \in \mathcal{K}$. Due to the periodicity of sinusoids, the time duration Δt needed to be selected such that each sinusoid member in $H_{u,s}(t)$ can contribute to G_l.

$$\begin{aligned} H_{u,s,n}(t) = \sqrt{\frac{P_n}{M_n}} \sum_{m=1}^{M_n} \Bigg\{ & \exp\left(j2\pi\nu_{n,m}t\right) \cdot \underbrace{\begin{bmatrix} F_{rx,u,\theta}\left(\theta_{n,m,ZOA}, \phi_{n,m,AOA}\right) \\ F_{rx,u,\phi}\left(\theta_{n,m,ZOA}, \phi_{n,m,AOA}\right) \end{bmatrix}^T}_{(a)} \underbrace{\begin{bmatrix} \exp(j\Phi_{n,m}^{\theta\theta}) & \sqrt{\kappa_{n,m}^{-1}}\exp(j\Phi_{n,m}^{\theta\phi}) \\ \sqrt{\kappa_{n,m}^{-1}}\exp(j\Phi_{n,m}^{\phi\theta}) & \exp(j\Phi_{n,m}^{\phi\phi}) \end{bmatrix}}_{(b)} \\ & \cdot \underbrace{\begin{bmatrix} F_{tx,u,\theta}\left(\theta_{n,m,ZOD}, \phi_{n,m,AOD}\right) \\ F_{tx,u,\phi}\left(\theta_{n,m,ZOD}, \phi_{n,m,AOD}\right) \end{bmatrix}}_{(c)} \underbrace{\exp\left(j2\pi\lambda_0^{-1}\left(\hat{r}_{tx,n,m}^T \bar{d}_{rx,u}\right)\right)}_{(d)} \underbrace{\exp\left(j2\pi\lambda_0^{-1}\left(\hat{r}_{rx,n,m}^T \bar{d}_{tx,s}\right)\right)}_{(e)} \Bigg\} \end{aligned} \tag{1}$$

The inner product between $\cos(2\pi f_l^b t + \psi_l)$ and $A_k \cos(2\pi f_k + \phi_k)$ over a Δt duration, for arbitrary $l \neq k$, denoted by $g_{k,l}$, is written as:

$$\begin{aligned} g_{k,l} &= \frac{1}{\Delta t}\int_{\Delta t} A_k \cos\left(2\pi f_k t + \phi_k\right)\cos(2\pi f_l^b t + \psi_l)dt \\ &= \frac{1}{\Delta t}\int_{\Delta t} \frac{A_k}{2}\begin{pmatrix}\cos\left(2\pi(f_k - f_l^b)t + \phi_k - \psi_l\right) \\ +\cos\left(2\pi(f_k + f_l^b)t + \phi_k + \psi_l\right)\end{pmatrix} dt \\ &\overset{(a)}{\cong} \frac{1}{\Delta t}\int_{\Delta t} \frac{A_k}{2}\cos\left(2\pi(f_k - f_l^b)t + \phi_k - \psi_l\right) dt. \end{aligned} \quad (6)$$

Here, (a) follows since

$$\frac{1}{\Delta t}\int_{\Delta t} \frac{A_k}{2}\cos\left(2\pi(f_k + f_l^b)t + \phi_k + \psi_l\right) dt \cong 0, \quad (7)$$

which holds under the condition of $\Delta t \gg 1/(f_k + f_l^b)$, and $g_{k,l}$ denotes the k-th component of $G_l = \sum_{k=1}^{K} g_{k,l}$.

When $f_k = f_l^b$, $g_{k,l}$ reaches the maximum value, $\frac{A_k}{2}$. When $f_k \neq f_l^b$, $g_{k,l}$ depends on Δt and ϕ_k. We proceed in our analysis along the following two directions:

$$2\pi(f_k - f_l^b)\Delta t > 2\pi, \quad (8)$$

and

$$2\pi(f_k - f_l^b)\Delta t \leq 2\pi. \quad (9)$$

For the case in (8), $g_{k,l}$ is approximately equal to zero, and the larger the value of Δt, the more accurate the approximation. We will discuss the valid range of Δt in Section III-B2.

For the case in (9), we have:

$$\begin{aligned} & g_{k,l} \\ &= \frac{1}{\Delta t}\int_{\Delta t} \frac{A_k}{2}\cos\left(2\pi(f_k - f_l^b)t + \phi_k - \psi_l\right) dt \\ &= \frac{1}{2\Delta t}\frac{A_k}{2\pi(f_k - f_l^b)} \\ &\quad \times \left(\sin(2\pi(f_k - f_l^b)\Delta t + \phi_l - \psi_l) - \sin(\phi_k - \psi_l)\right) \\ &= \frac{A_k 2\cos\left(\pi(f_k - f_l^b)\Delta t + \phi_k - \psi_l\right)\sin\left(\pi(f_k - f_l^b)\Delta t\right)}{4\pi\Delta t(f_k - f_l^b)} \\ &= \frac{A_k}{2}\cos\left(\pi(f_k - f_l^b)\Delta t + \phi_k - \psi_l\right) sinc\left((f_k - f_l^b)\Delta t\right), \end{aligned} \quad (10)$$

where $sinc(x) = \sin(\pi x)/\pi x$.

1) Effect of Initial Phase of ϕ_k on $g_{l,k}$: From (10), the initial phase ϕ_k also affects $g_{l,k}$. According to the 3GPP standard [29], the initial phase stems from diverse factors, such as antenna element arrangement and antenna polarization. Thus, the explicit analytical expression for calculating ϕ_k is complicated to derive. Hence, a numerical method is used to analyze the effect of ϕ_k on $g_{l,k}$.

The first step of the numerical method is to calculate the empirical distribution of the initial phase ϕ based on the description in the 3GPP standard [29]. We consider that a base station is equipped with 40 antennas, and a UE has only one antenna. There are 10 clusters in the channel between the BS and the UE. Then, the initial phase of each ray is obtained.

Second, the *Kolmogorov Smirnov* (KS) test is used to investigate the similarity of the empirical distribution of ϕ_k. For this purpose, a set of uniformly-distributed data is generated by MATLAB. Then, the KS test is employed to investigate the similarity between the distribution of the MATLAB-generated uniform data and that of $\{\phi_k\}$. In the empirically-determined KS test, the resulting p value is equal to 0.619. This resulting p is far larger than the widely-used significance level of 0.05, which agrees with the hypothesis that ϕ_k is uniformly distributed. Since there are a large number of rays in a real massive MIMO system and there are no constraints on the propagation channel, the initial phases of the rays tend to distribute uniformly, matching the result in [33].

Knowing the distribution of ϕ_k, we calculate the mean value of G_l with respect to ϕ_k as follows:

$$\begin{aligned} G_l &= \sum_{k=1}^{K} g_{k,l} \cong E\left[g_{k,l}\right] \\ &= E\left[\frac{A_k}{2}\cos\left(\pi(f_k - f_l^b)\Delta t + \phi_k - \psi_l\right) sinc\left((f_k - f_l^b)\Delta t\right)\right] \\ &= sinc\left((f_k - f_l^b)\Delta t\right) \\ &\quad \times \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \frac{1}{2\pi} A_k \cos\left(\pi(f_k - f_l^b)\Delta t + \phi_k - \psi_l\right) d\phi_k \\ &= \frac{1}{\pi} A_k \cos\left(\pi(f_k - f_l^b)\Delta t - \psi_l\right) sinc\left((f_k - f_l^b)\Delta t\right). \end{aligned} \quad (11)$$

From (11), G_l is affected by the frequency offset $(f_k - f_l^b)$, and the length of the data for integration is Δt. For a geometry-based channel, each f_k is predefined, and each f_l^b is fixed in the L-dimensional space. Thus, Δt is the only variable affecting G_l.

2) Selection of Δt: The frequency difference, $\delta\omega$ between the basis of the L-dimensional space, is calculated as:

$$\delta\omega = 2\pi\left(f_D/L\right). \quad (12)$$

With the Doppler bin interval of $\frac{f_D}{L}$, the largest Doppler ambiguity is equal to $\frac{f_D}{2L}$. Thus, within Δt, the largest phase change of $g_{l,k}$ during Δt is equal to $2\pi\frac{f_D}{2L}\Delta t$. As analyzed in the previous part, to guarantee G_l is not approximately equal to zero, the phase change during Δt should be smaller than 2π. Combining (9) and (12), we set the requirement on the selection of Δt as follows:

$$\Delta t < 2\ L/f_D. \quad (13)$$

To demonstrate the accuracy of the analysis, we perform simulations of the $K \rightarrow L$ dimension projection and plot results in Fig. 2. Recall that K is determined by the summation of the sinusoid components M_n, i.e. $K = \sum_{n=1}^{N} M_n$. Here, $K = 400$ and $L = 20$. Detailed evaluation on the above method will be presented in Section VI.

In Fig. 2, a blue vertical line signifies a sinusoid component at the frequency indicated by the x-axis of the blue line. The density of the blue lines is proportional to the number of sinusoids in

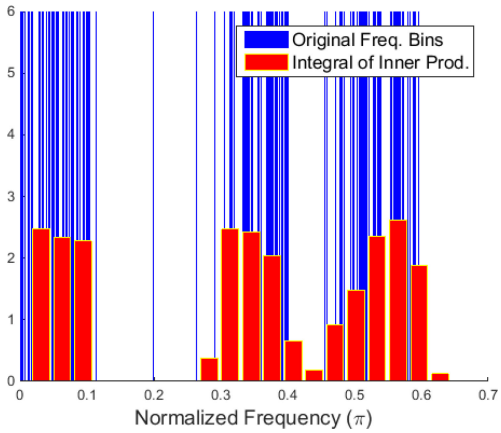

Fig. 2. Demonstration of the frequency bins reduction.

$H_{u,s}(t)$ within a frequency range. The heights of the red bars are proportional to the absolute values of $G_l, l \in \mathcal{L}$. Fig. 2 shows the amplitude of a projected component G_l reflecting the density of the rays in $H_{u,s}(t)$.

We generate $\tilde{H}_{u,s}(t)$ by adding the projections of $H_{u,s}(t)$ to the space spanned by the L sinusoids as,

$$\tilde{H}_{u,s}(t) = \sum_{l=1}^{L} G_l \cos(2\pi f_l^b t + \psi_l) = \sum_{l=1}^{L} G_l \cos(\omega_l t + \psi_l). \tag{14}$$

Compared with $H_{u,s}(t)$, the number of the sinusoids in $\tilde{H}_{u,s}(t)$ is significantly reduced.

IV. Geometry-Based Massive MIMO Channel Generation on FPGAs

Frequently, lookup tables are used to generate sinusoids. However, the lookup table strategy induces a high demand on the memory [34], [35]. Such a strategy makes it infeasible to generate a geometry-based massive MIMO channel on a single FPGA chip. In this paper, a second-order AR model is proposed to generate the geometry-based channel.

A. Preprocessing for the Channel Generation

The geometry-based channel model as presented in (1) comprises a large number of parameters. Details about calculating the parameters are given in [29] and references therein. A general-purpose computer is used for initializing the parameters in an offline manner. The initialized parameters are then fed to an FPGA chip, which generates the channel in an online manner. FPGAs are not used to initialize the parameters due to their limited computational resources.

First, to configure a MIMO system, we determine parameters including distance, path loss, number of antennas, mechanical orientation of the antennas, and polarization pattern of the antennas. According to the settings, the parameters, P_n, $\nu_{n,m}$, $\theta_{n,m,ZOA}$, and $\phi_{n,m,AOA}$ needed in (1) are calculated via a general-purpose computer.

After the MIMO channel is generated, the sinusoids are sorted based on their frequency. Following the reordering, the channel $H_{u,s}(t)$ is written in the form of (3). Next, $H_{u,s}(t)$ is converted into $\tilde{H}_{u,s}(t)$ via projection using inner products. After the projection step, the geometry-based MIMO channel can be generated on an FPGA.

B. Geometry-Based Channel Generation for Massive MIMO

1) Disadvantage of Conventional Generation Method: As previously mentioned, the most popular method of cosine implementation is based on a lookup table for quantized cosine values stored in RAM. However, there are two main drawbacks to channel emulation based on lookup tables. First, RAM resources are occupied throughout the entire procedure of channel generation. While often computationally powerful, the FPGA chip is often limited by memory resources [36].

Second, the lookup operation has a non-zero delay. For each channel, the number of multipath components is usually selected within the range of 8 to 12 [37] to ensure desirable statistical characteristics of the resulting channels. In most cases, there are a large number of transmitter-receiver antenna pairs. Thus, the simultaneous lookup operations would cause excessive memory-lookup delays for FPGA-based channel generation. If we ultimately seek to emulate geometry-based channels in massive-MIMO contexts, memory consumption and latency with a large amount of lookup operations preclude the use of SOS-based generation.

2) Iterative Solution to Channel Generation: We now propose a simple, second-order AR model to generate the sinusoids needed for channel emulation. This AR model for the sinusoid generation is written as:

$$y[i] = 2\cos(\omega) \cdot y[i-1] - y[i-2]. \tag{15}$$

Algorithm 1 presents the AR-based iteration structure for generating geometry-based channels. There are two main phases: initialization and iterative calculation. In the PC initialization phase, geometry-based MIMO channel parameters are computed, and the channel is projected onto an L-dimensional space. With FPGA initialization, the initial two-iteration values are obtained by using a cosine lookup table. Then, the algorithm moves to the iterative-computation phase, where the resulting sinusoid values $y_l[i], i \geq 2$ are generated iteratively, and the envelope of the weighted I/Q components ($y_{Il}[i], y_{Ql}[i], l \in \mathcal{L}$) is the channel gain at the i-th time instant.

Algorithm 1 shows that the cosine lookup table is used in the FPGA initialization phase. Thus, only one cosine table is needed in the iterative generation. During the channel emulation phase, subsequent computations require only one addition and one multiplication at each iteration. Since an FPGA is powerful in terms of computation, the addition and multiplication can be processed with minimal latency.

C. Block Diagram of the Proposed Generation System

A block diagram of the proposed channel generation algorithm is given in Fig. 3. The cosine value initialization is based on a lookup table, which is straightforward to implement. Fig. 4 illustrates the architecture of the iterative structure of the proposed channel generation. Fig. 4 shows that only two registers are needed for each sinusoid component. As introduced before, let L denote the number of sinusoids in each geometry-based channel

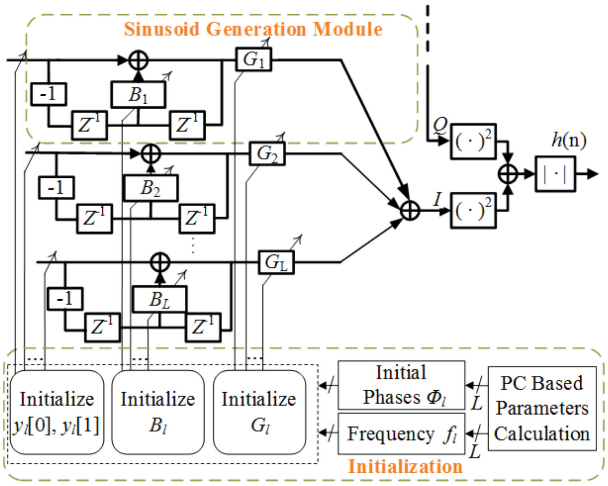

Fig. 3. Proposed fading channel generation simulator.

Algorithm 1: Massive MIMO Channel Generation.

Phase 1 - Initialization

1: Configure the massive MIMO scenario on a PC
2: Compute the parameters used in (1) according to the massive MIMO setting
3: Expand (1) on the orthogonal basis set $\{y_l, l \in \mathcal{L}\}$
4: Initialize the sinusoid generation

• generate $2L$ uniform random variable ψ_l,

$$\psi_l \sim \mathcal{U}[0, \frac{\pi}{2}), l \in \mathcal{L}$$

• look up $5L$ cosine values $y_{Il}[0] = \cos(\psi_l)$,

$$\begin{aligned} y_{Ql}[0] &= \sin(\psi_l), \\ y_{Il}[1] &= \cos\left(2\pi \tfrac{l}{L} T_W f_D + \psi_l\right), \\ y_{Ql}[1] &= \sin\left(2\pi \tfrac{l}{L} T_W f_D + \psi_l\right), \\ B_l &= 2\cos\left(2\pi \tfrac{l}{L} T_W f_D\right); \end{aligned}$$

Phase 2 - Iterative Computation of FPGA

5: **for** $i < \lfloor T_{sim}/T_W \rfloor$ **do**; ▷ T_{sim} and T_W are simulation duration and sampling interval, respectively
6: $y_{Il}[i] = B_l y_{Il}[i-1] - y_{Il}[i-2]$, $y_{Ql}[i] = B_l y_{Ql}[i-1] - y_{Ql}[i-2]$,
7: $c(i) = \frac{1}{L}\sqrt{\left(\sum_{l=0}^{L-1} G_{Il} y_{Il}[i]\right)^2 + \left(\sum_{l=0}^{L-1} G_{Ql} y_{Ql}[i]\right)^2}$, c denotes the envelope of the channel.
end for

and N_{ch} represent the desired number of geometry-based MIMO channels to be generated. Assume the volume of RAM used for building the *sinusoid* lookup table is V_{LUT}. Thus, the AR-based channel generation scheme can save RAM resources up to V_D and quantified as:

$$V_D = N_{ch}\left(2L(V_{LUT} - 2) - V_{LUT}\right). \tag{16}$$

Assume a *sinusoid* lookup table consists of 1024 RAM words of two bytes each. The number is used for theoretical analysis, which is not equal to the amount of the memory needed for building a sin/cos lookup table in a chipset. Due to the memory usage in the theoretical analysis being proportional to that in the real chipset, the RAM resource saving percentage in the theoretical estimation is the same with that in the real chip based estimation. For instance, consider ten ($N_{ch} = 10$) geometry-based MIMO channels and twenty components for a channel ($L = 20$). The amount of RAM savings can be calculated according to $V_D \cong 398$ kB, which saves 97.3% of the RAM resources as compared to the LUT-based solution. This result holds not only for the theoretical analysis but also applies to the estimation in real chipsets.

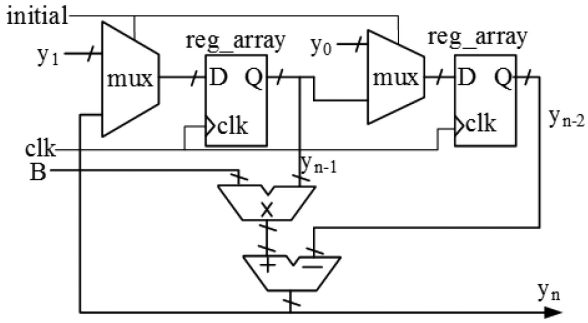

Fig. 4. Iteration architecture of one ray component.

V. Update Rate and Word Length Optimization

In this section, we study the effect of varying the channel update rate and FPGA-operating word length at which channel coefficients are regenerated. The word-length selection is important in balancing the generated channel accuracy and hardware cost. We jointly employ analytical and numerical methods to study the optimal parameter selection. The parameters are optimized by searching for the combination of update rate and word length at which the channel is generated at the optimum accuracy. In the channel generation, a group of sinusoids are the basic elements of a geometry-based channel. Thus, the error of a generated sinusoid is taken as the metric for evaluating the channel accuracy. The joint optimization on the update rate and word length is only performed in the initial stage (Algorithm 1) of the channel generation procedure. Thus, the complexity and latency induced by the joint optimization is negligible.

A. Channel Generation Error in Discrete-Time Update

Let T_W denote the time duration between updates, which should be smaller than the coherence time of the channel. Hence, the channel update interval T_W is selected to be smaller than 0.423 times the maximum Doppler inverse, $\frac{0.423}{f_D}$ [38]. In a conservative approximation, T_W is smaller than 0.25 times of the maximum Doppler inverse, $T_W \leq \frac{1}{4f_D}$. Let I denote the number of update points in the quarter-period of a sinusoid (Fig. 5). As shown in this figure, at a given frequency f_M, $f_M \leq f_D$, I can be determined as $I = \lfloor 1/4 f_M T_W \rfloor$. Furthermore, to avoid large channel emulation error, T_W is selected such that $I > 1$.

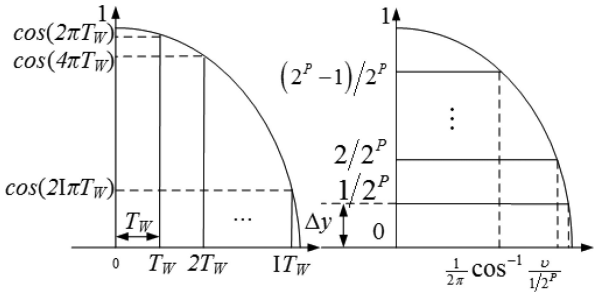

Fig. 5. Channel update rate effect and word length optimization illustration.

In the quarter-period of a *cosine* curve, the error $e_{\delta t}$ in the discrete generation of a channel is calculated by:

$$e_{\delta t} = \sum_{i=0}^{I-1} \int_{iT_W}^{(i+1)T_W} \left(\cos(2\pi i f_M T_W) - \cos(2\pi f_M t)\right) dt$$
$$= \sum_{i=0}^{I-1} \cos(2\pi i f_M T_W) T_W - \frac{1}{2\pi}. \tag{17}$$

Due to the sampling criteria, the sampling frequency needs to be larger than twice the Doppler frequency, f_M. Thus, within $\frac{1}{4}$ of a period, as shown in Fig. 5, we have the limits of T_W, $0 < T_W \leq 1/8 f_M$.

B. Word-Length Effect on Channel Generation

Besides the channel update rate consideration, the word length of each point in the fading channel generation will affect the hardware resources consumed. In this section, the approximation error for a quantized cosine waveform at a single carrier is first discussed.

We assume the cosine function is quantized by P bits, where P is a positive integer. Thus, the distance between two quantization levels is $\Delta d = 1/M$, which is illustrated in Fig. 5. In the quarter-period of a cosine function, the inverse function is calculated by:

$$t = 1/2\pi \arccos(y). \tag{18}$$

Thus, the error $e_{\delta y}$ in the quantization of the quarter-period of a cosine function is calculated by:

$$e_{\delta y} = \sum_{\nu=0}^{2^P-1} \int_{\frac{\nu}{2^P}}^{\frac{\nu+1}{2^P}} \frac{1}{2\pi} \left(\arccos\left(\frac{\nu}{2^P}\right) - \arccos(y)\right) dy$$
$$= \frac{1}{2\pi 2^P} \sum_{\nu=0}^{2^P-1} \arccos\left(\frac{\nu}{2^P}\right) - \frac{1}{2\pi}. \tag{19}$$

Numerically evaluating (19) for various values of P, we can realize $e_{\delta y} \geq 1.0056$, which is larger than the power of the sinusoid over the quarter-period at $P < 3$. In the case of $P > 21$, the hardware cost becomes extremely large, which is extremely challenging to be implemented in hardware. Thus, the number of bits used for quantizing the sinusoids is limited in the range of $[3, 20]$, $3 \leq P \leq 20$.

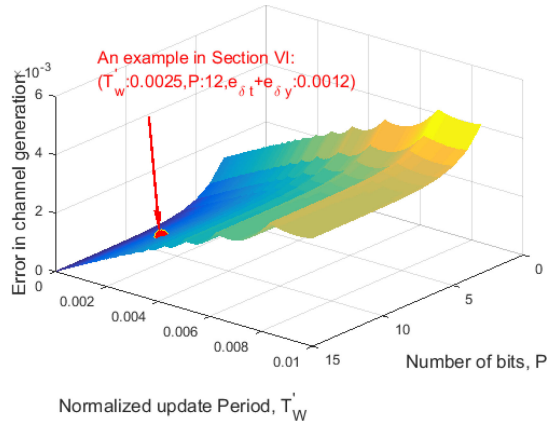

Fig. 6. Massive MIMO channel generation error.

C. Joint Optimization on Updated Rate and Word Length

In this subsection, we jointly consider the emulation accuracy with respect to two factors: finite word length and update period. For mathematical tractability, the summation of $e_{\delta t}$ and $e_{\delta y}$, denoted by $\beta(P, T_W)$, is taken as the total error:

$$\beta(P, T_W) = e_{\delta t} + e_{\delta y}. \tag{20}$$

In general, more accurate channel generation needs greater hardware resources with longer word lengths and shorter update rates. In this paper, the number of bits per time unit used for generating the channel is used as the metric for resource usage. Based on this idea, we define variable a as:

$$a(P, T_W) = P/T_W. \tag{21}$$

With the defined metric of the channel construction accuracy $\beta(P, T_W)$ and the consumed resource size a, we build an optimization problem in which the construction error is minimized under the constraint of the resource budget, $a \leq A$, where A is a constant. The optimization problem is written as follows:

$$\begin{aligned} \min_{T_W, P} \quad & \beta(P, T_W) = e_{\delta t} + e_{\delta y} \\ \text{subject to: } & a(P, T_W) \leq A, 0 < T_W \leq \frac{1}{8 f_M} \\ & 3 \leq P \leq 20 \end{aligned} \tag{22}$$

where

$$\beta = \sum_{i=0}^{I-1} \cos(2\pi i f_M T_W) T_W + \frac{1}{2\pi 2^P} \sum_{\nu=0}^{2^P-1} \arccos\left(\frac{\nu}{2^P}\right) - \frac{2}{2\pi}. \tag{23}$$

From (23), the error $\beta(P, T_W)$ not only depends on the variables (word length P and update period T_W), but also the parameter f_M. To simplify the analysis, we define the normalized update period $T'_W = T_W f_M$. Fig. 6 plots the error versus T'_W and P. As indicated in Appendix B, the error β is not convex with respect to P and T'_W. Thus, we solve it by numerically trying all combinations of P and T'_W and find the optimum pair of (T'_W, P), denoted by (T'^*_W, P^*), at which the minimum $\beta(\cdot)$ is achieved.

Up to this point, we have presented a method for optimizing the update rate and word length by minimizing $e_{\delta t} + e_{\delta y}$. At

a selected update rate and word length, the spectrum of the generated channel, denoted by $S_{\hat{H}}(f)$, is a function of the spectrum of the real channel $S'_H(f)$, the update interval T_W and the word length P, given by:

$$S_{\hat{H}}(f) = \begin{pmatrix} S'_H(2\pi f T_W) \\ +\frac{1}{12}2^{\frac{1}{2}\log 2\pi e\sigma_H^2}2^{-2P} \end{pmatrix} \left(\frac{\sin(\pi T_W f)}{\pi f}\right)^2. \quad (24)$$

Additional details are given in Appendix A.

From (24), as T_W becomes smaller, the shape of $\frac{\sin(\pi T_W f)}{\pi f}$ becomes more flat. Thus, the shape of $S_{\hat{H}}(f)$ is closer to the shape of $S'_H(2\pi f T_W)$. When more bits are used, P is larger, and thus the distortion caused by the quantization on $S'_H(2\pi f T_W)$ is smaller.

The PDF of the generated channel, denoted by $q_{\hat{H}}$, is the function of the PDF of the real channel $q_H(h)$ and the word length P, as given by:

$$q_{\hat{H}}(h) = q_H(h) \otimes u_q(h), \quad (25)$$

Here, $\otimes$ denotes convolution operation. Lastly, u_q is defined as follows:

$$u_q(h) = \begin{cases} \frac{1}{12}\frac{2^P}{D}, & -\frac{D}{2\cdot 2^P} \le h \le \frac{D}{2\cdot 2^P} \\ 0, & \text{otherwise.} \end{cases} \quad (26)$$

Here, D denotes the dynamic range of the fading channel. From (41), when P becomes larger, $u_q(h)$ tends to be a $\delta(\cdot)$ function. Thus, $q_{\hat{H}}(h)$ approaches $q_H(h)$.

D. Continuity of the Generated Fading Channel

As discussed in Section III-B2, in the $K \to L$-dimensional space projection, the channel generation duration Δt satisfies, $\Delta t < 2\,L/f_D$. In this subsection, the continuity of the generated channel between two adjacent Δt blocks is discussed. In the analysis, we consider the maximum Doppler shift f_D. At the channel update period of T_W, assuming there are $n_{\Delta t}$ updates within the time duration of Δt, and $n_{\Delta t}$ is calculated by:

$$n_{\Delta t} \triangleq \left\lfloor \frac{\Delta t}{T_W} \right\rfloor = \frac{2\,L}{f_D T_W} - \epsilon_l, \quad (27)$$

where ϵ_l uniformly distributes within [0, 1).

At the ending update point $n_{\Delta t}$, the sinusoid value is $\cos(2\pi n_{\Delta t} T_W f_l^b + \psi_l)$, where f_l^b and ψ_l are defined in (4) and Algorithm 1, respectively. We assume there is no constraint of Δt. Thus, the sinusoid value at the next update is $\cos(2\pi(n_{\Delta t}+1)T_W f_l^b + \psi_l)$. Between the two closest updating points, the difference, denoted by $\delta\bar{c}$, is called as *reference difference* and calculated by:

$$\begin{aligned} \delta\bar{c} &= \cos(2\pi(n_{\Delta t}+1)T_W f_l^b + \psi_l) - \cos(2\pi n_{\Delta t} T_W f_l^b + \psi_l) \\ &= -2\sin(2\pi n_{\Delta t} T_W f_l^b + \pi T_W f_l^b + \psi_l)\sin(\pi T_W f_l^b) \\ &\overset{(a)}{=} -2\sin(2\pi\left(\frac{2\,L}{f_D T_W} - \epsilon_l\right) T_W f_l^b \\ &\quad + \pi T_W f_l^b + \psi_l)\sin(\pi T_W f_l^b) \\ &\overset{(b)}{=} 2\sin(2\pi\epsilon_l T_W f_l^b - \pi T_W f_l^b - \psi_l)\sin(\pi T_W f_l^b). \end{aligned} \quad (28)$$

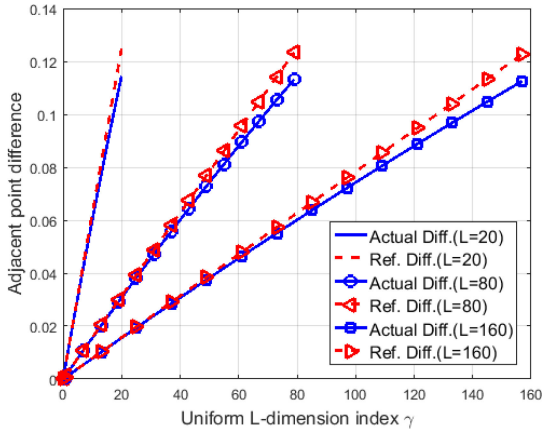

Fig. 7. Adjacent sinusoid difference comparison between continuous generation and the proposed method.

Here, (a) follows from (27); and (b) follows from the definition of f_l^b in (4) by which the equation $f_l^b = \gamma\frac{f_D}{L}$, $\gamma \in Z$ and $0 \le \gamma \le L$, holds, and then, $2\pi\frac{2L}{f_D T_W}T_W f_l^b = 4\gamma\pi$.

In the proposed channel generation method, after the $n_{\Delta t}$-th update, the following sinusoid value is $\cos(\psi_l)$, rather than $\cos(2\pi n_{\Delta t} T_W f_l^b + \psi_l)$. Therefore, the true difference between the two closest updating points, denoted by δc, is called the *actual difference*, which is calculated by:

$$\begin{aligned} \delta c &= \cos(\psi_l) - \cos(2\pi n_{\Delta t} T_W f_l^b + \psi_l) \\ &= \cos(\psi_l) - \cos(2\pi\epsilon_l T_W f_l^b - \psi_l) \end{aligned} \quad (29)$$

Next, the mean value of $\delta\bar{c}$ and δc is calculated. By comparing the mean differences, $E[\delta\bar{c}]$ and $E[\delta c]$, we can understand the continuity of the channel generated by the proposed method.

$$\begin{aligned} E[\delta\bar{c}] &= E[E_{\psi_l}[\delta\bar{c}|\epsilon_l]] \\ &= E\left[\int_0^{\frac{\pi}{2}} 2\sin(2\pi\epsilon_l T_W f_l^b - \pi T_W f_l^b - \psi_l)\sin(\pi T_W f_l^b)d\psi_l\right] \\ &= \int_0^1 2\begin{pmatrix} \sin(2\pi\epsilon_l T_W f_l^b - \pi T_W f_l^b) \\ -\cos(2\pi\epsilon_l T_W f_l^b - \pi T_W f_l^b) \end{pmatrix}\sin(\pi T_W f_l^b)d\epsilon_l \\ &= \frac{\sin^2(\pi T_W f_l^b)}{\pi T_W f_l^b} \end{aligned} \quad (30)$$

$$\begin{aligned} E[\delta c] &= E[E_{\psi_l}[\delta c|\epsilon_l]] \\ &= E\left[\int_0^{\frac{\pi}{2}} \cos(\psi_l) - \cos(2\pi\epsilon_l T_W f_l^b - \psi_l)d\psi_l\right] \\ &= E\left[\int_0^1 \left(1 - \cos(2\pi\epsilon_l T_W f_l^b) - \sin(2\pi\epsilon_l T_W f_l^b)\right)\right] \\ &= 1 - \frac{\sin(\pi T_W f_l^b)}{\pi T_W f_l^b} + \frac{\cos(\pi T_W f_l^b) - 1}{\pi T_W f_l^b}. \end{aligned} \quad (31)$$

To quantify this result, we plot the values of $E[\delta\bar{c}]$ and $E[\delta c]$ at different l's in Fig. 7. In the plot, $T_W f_D = 0.04$ and the dimension of the projected space, L, is set to be 20, 80, and 160. From Fig. 7, the mean of the actual difference is no larger

than that of the reference difference. Therefore, the sinusoid generated by the proposed method is regarded to be continuous in terms of Nyquist sampling.

VI. Experimental Evaluation

In this section, we evaluate the performance of the proposed channel generation using analytical evaluation, numerical simulations, and experimentation with an FPGA platform. For comparison, a channel generation method using the conventional structure is also evaluated using an FPGA. Further, an offline channel generation method using a general-purpose computer is also considered in the experiments.

In the experiment, a scenario with 20 clusters is considered. At each cluster, there are 20 rays. Therefore, 800 sinusoids are needed if I/Q components of the geometry-based massive MIMO channel are directly generated. We project the 800 sinusoids to a bank of 20 sinusoids with equally-spaced frequencies in each element. The maximum Doppler in the scenario is 500 Hz.

A. Iterative-Based Channel Generation

1) Verification of Iterative Structure: We perform fixed-point experiments on the method of generating the geometry-based MIMO channel in the Xilinx Integrated Synthesis Environment (ISE). The FPGA chip used is a Virtex-4 VFX100FFG1517. In the fixed-point experiment, we generate a geometry-based MIMO channel, which was described in the previous paragraph. As a reference, a channel with the same parameter settings is also generated using floating-point arithmetic in MATLAB. In this part of the simulation, the channel samples are generated at a period of 0.005 ms ($T_w = 0.005$ ms). Thus, the normalized update period is 2.5×10^{-3}, $T'_W = 2.5 \times 10^{-3}$. Each channel sample is expressed using 8 bits ($P = 12$).

Furthermore, the performance of the iterative method and lookup table based method is directly compared. To evaluate the accuracy of generated channels, we leverage two widely-used metrics: level crossing rate (LCR) and average duration of fading (ADF) [33], which jointly characterize the severity of the fading over time.

First, a floating-point channel is generated in MATLAB. The channel is obtained by adding up the 400 sinusoids, which are the ones before being projected to the bank of 20 sinusoids. We label this as an original channel generation, which corresponds to the red curves in Fig. 8(a) and Fig. 8(b).

For each of I/Q component, after projecting 400 sinusoids onto the space spanned by 20 sinusoids, the two methods of proposed channel generation (AR model based) and conventional channel generation (lookup-table based) are implemented in the FPGA. The LCR and ADF of the proposed method are plotted with a blue dashed curve with upright triangular markers, and the conventional method generation results are plotted by green curves with circular markers.

In Fig. 8(a) and Fig. 8(b), the matching curves indicate the projection of 400 sinusoids to the bank of 20 sinusoids is valid for channel generation without sacrificing fidelity. Furthermore, the curve based on the AR model also matches the one based on the lookup tables. Next, the resulting hardware resource consumption is investigated.

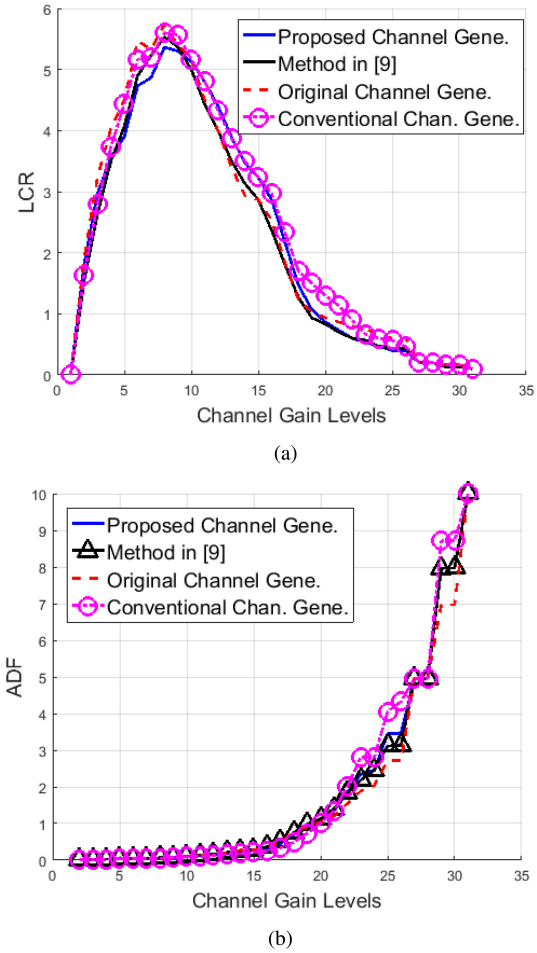

Fig. 8. Statistical probability of generated data. In the generation, the maximum Doppler is set to be 500 Hz. The channel update period is 0.005 ms. The original channel generation needs 800 sinusoids. For the proposed generation method, the 800 sinusoids are projects to 20 sinusoids. (a) LCR. (b) ADF.

2) MIMO Channel Covariance Matrix Analysis: The covariance matrix is a critical component to the representation of MIMO channels. Thus, the impact of the proposed method on the covariance matrix is investigated. Assume the number of the antennas per UE is equal to 2. The number of antennas, η, at the base station ranges from 2 to 16 in our investigation. Let random vector X_i of size $\eta \times 1$ denote the transmission symbols at the i-th time slot. Each element of X_i has a Gaussian distribution. On the UE side, received symbols are denoted by Y_i, which is a random vector of size 2×1. The MIMO channel is denoted by H, which is of size of $2 \times \eta$. The realization of the channel at the i-th time slot is H_i. The (u, s)-th element of H_i is determined by $H_{u,s}(i \cdot T_W)$, $u \in \{1, 2\}$, and $s \in \{1, 2, \ldots, \eta\}$.

Recall that the fading channel to calculate is written in (3), and the parameters in (3) are determined by (2). As noted before, the parameter K in (3) is very large for geometrical channels. We leverage the proposed method to project the channel denoted by (3) to the channel denoted by (14). Let $\tilde{H}_i$ denote the projection

TABLE II
NUMBER OF BS ANTENNA

MIMO Dimension	2	4	8	16
$\|\Delta \boldsymbol{C}\|_2$ [15] : $\times 10^{-6}$	1.2	1.9	2.9	1.9
$\|\Delta \boldsymbol{C}\|_2$: $\times 10^{-6}$	1.1	1.9	3.1	1.6

of the original channel H_i. $\tilde{H}_i$ also represents the output of the proposed channel generator.

At discrete time slots and using the two generated channels, H_i and $\tilde{H}_i$, the channel outputs Y_i and $\tilde{Y}_i$ are calculated according to:

$$Y_i = H_i X_i + \Omega_i, \text{and } \tilde{Y}_i = \tilde{H}_i X_i + \Omega_i. \tag{32}$$

Here, $i \in \{1, 2, \ldots N_T\}$ where N_T is the number of time slots over which the channel is generated and Ω_i denotes additive white Gaussian noise.

Next, we utilize the simulated channel input and output, X_i and Y_i, respectively, and $\tilde{Y}_i$ to estimate the MIMO channel covariance. Let C denote the channel covariance matrix generated from the original channel model (3), and let $\tilde{C}$ represent the proposed channel generation method. The estimation of C and $\tilde{C}$ are calculated by:

$$\boldsymbol{C} = \boldsymbol{X}\boldsymbol{Y}' \left(\boldsymbol{Y}\boldsymbol{Y}'\right)^{-1}, \tilde{\boldsymbol{C}} = \boldsymbol{X}\tilde{\boldsymbol{Y}}' \left(\tilde{\boldsymbol{Y}}\tilde{\boldsymbol{Y}}'\right)^{-1}, \tag{33}$$

where

$$\boldsymbol{X} = [X_1, X_2, \ldots, X_{N_T}]$$
$$\boldsymbol{Y} = [Y_1, Y_2, \ldots, Y_{N_T}], \tilde{\boldsymbol{Y}} = \left[\tilde{Y}_1, \tilde{Y}_2, \ldots, \tilde{Y}_{N_T}\right]. \tag{34}$$

We use l_2-norm to measure the distance between $\boldsymbol{C}$ and $\tilde{\boldsymbol{C}}$. Let $\Delta \boldsymbol{C}$ denote the difference between $\boldsymbol{C}$ and $\tilde{\boldsymbol{C}}$,

$$\Delta \boldsymbol{C} = \boldsymbol{C} - \tilde{\boldsymbol{C}}. \tag{35}$$

Let $\boldsymbol{\lambda}_{\Delta \boldsymbol{C}}$ denote the set of the eigenvalues of $\Delta \boldsymbol{C}$. By definition, the l_2 norm of the matrix difference is equal to the maximum eigenvalue of $\Delta \boldsymbol{C}$,

$$\|\Delta \boldsymbol{C}\|_2 = \max(\boldsymbol{\lambda}_{\Delta \boldsymbol{C}}). \tag{36}$$

Table II lists the l_2 norm of the difference matrix with the different numbers of base station antennas.

From Table II, the l_2 norm of the difference of the MIMO channel covariance matrix is smaller than 1×10^{-5}. Therefore, the proposed channel generation method ensures there is minimal distortion in the covariance matrix of the original MIMO channel.

3) Complexity Analysis:

a) Computation complexity: We now perform a complexity analysis for the proposed iterative method and the lookup-based method. Table III presents a summary of the resources used as reported by Xilinx ISE for the two channel generation methods.

From Table III, the advantages in terms of RAM is evident due to the fewer *cosine* value tables. The calculation of the product between the constant B and $y[i-1]$ is implemented via bit shifting. Thus, DSP48 s are not consumed in the iterative-based channel generator.

TABLE III
20 SINUSOIDS GENERATED NEEDED HARDWARE CONSUMPTION COMPARISON

	Flip-Flops	RAM16S	DSP48S
Iterative Str.	824	1	0
Table Lookup	783	20	0
Total Resources	84352	376	160

TABLE IV
TIME CONSUMPTION COMPARISON WITH CONVENTIONAL METHOD

	20 sinusoid gen.	Massive MIMO (3)
Iterative Str.	9.6 ns	9.6 ns
Table Lookup	0.8ns	16 ns
Update period	5000ns	5000 ns

b) Time consumption comparison: Besides the analysis in complexity, comparison between the time consumption between the table lookup method and the iterative method is also performed. In the comparison, we estimate the number of FPGA clock cycles required to generate a fading channel sample by the two methods.

For the table lookup method, the clock cycles are required to calculate the address index of the *sine/cosine* table, and read values from the table. First, the calculation of address index needs one clock cycle. Afterwards, according to the calculated index, reading the value from the table to the registers uses another clock cycle. Thus, there are two total cycles in the generation of a single sinusoid value.

For the proposed iterative structure, clock cycles are consumed for the bit shifting and addition operations. In the analysis, the word length is assumed to be 12 bits. To calculate the first term of the right-hand side of (15), the previous sinusoid value $y[i-1]$ is shifted by each bit of the 12-bit number $\cos(\omega)$. Then, the shifted values are summed up. In this step, there are 23 clock cycles consumed. Second, the first-step output is added to the sinusoid value at two previous updates $y[i-2]$, which needs one clock cycle. Thus, there are 24 clock cycles needed by the iterative-structure method.

For an FPGA chip, a typical value of one clock cycle is approximately 0.02 ns. For conservative estimation, the table lookup method needs 0.04 ns to generate a sinusoid value, while the iterative method needs 0.48 ns. If both methods need to generate 20 sinusoids for a channel sample, the table lookup method needs 0.8 ns, and the iterative method needs 9.6 ns. To generate a massive MIMO channel for a TX-RX antenna pair (3), 400 sinusoids ($K = 400$) are needed. Regardless of the memory limits, the lookup table method needs 16 ns for generating one channel sample, and the proposed method (after the projection operation introduced in Section III-A) still needs 9.6 ns. Considering a channel update period of 5 μs, we have the comparison table listed in Table IV.

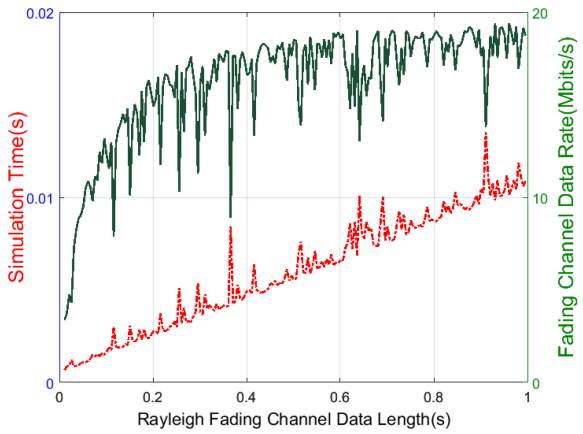

Fig. 9. Channel data generation throughput of offline (PC based) schemes.

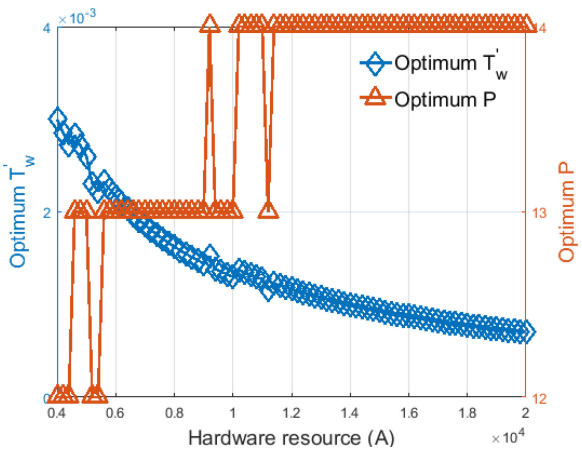

Fig. 10. The optimum bit width P and the normalized update period T'_w ($3 \leq P \leq 15$).

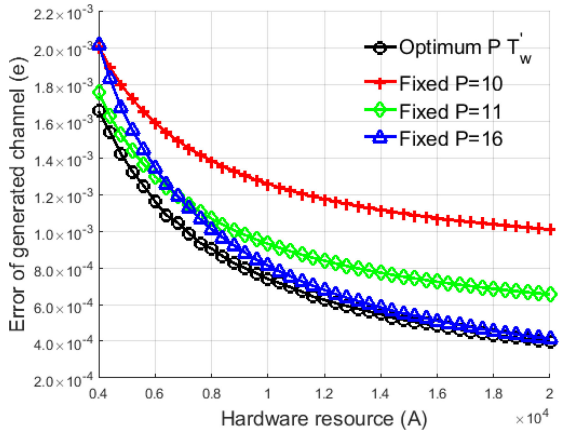

Fig. 11. Error of the generated geometry-based MIMO channel versus hardware resource consumption in terms of bit rate (bits/sec) ($3 \leq P \leq 15$).

B. Channel Generation Data Throughput Analysis

Another solution to channel emulation is to use an offline general-purpose computer to generate the fading channel coefficients and periodically transfer these to the FPGA to implement the emulation. The FPGA uses the stored channel data to modify the signal passing through the emulator. We call such a solution an offline generation scheme, as opposed to an online generation scheme, on the FPGA.

In the experiment, the PC used for generating the channel is equipped with an Intel i3 CPU at 2.26 GHz with 4 GB of RAM. In Fig. 9, the red dot-dashed curve represents the time consumed for generating a certain duration of the channel. The green solid curve denotes the corresponding channel parameter data rate, which represents the effective rate at which data needs to be transferred between the PC and the FPGA to enable emulation. From Fig. 9, we see that this rate is on the order of 20 Mbits/s for each channel.

From the result of experiments on the Virtex-4 VFX100, the FPGA can generate up to 3.3 Gbits/s of channel coefficients for emulation. Clearly, the proposed channel generation scheme based on an FPGA outperforms the PC-based solution in terms of channel data throughput. Further, the offline PC-based generation requires additional RAM and buffers, especially for large-scale networks. On the other hand, the PC-based solution would reduce the computational burden on the FPGA. The evaluation of a single-link system at low data rates, such as voice calls on sub-6 GHz bands, is an example on which the PC-based solution may be appropriate.

C. Performance Across Update Rates and Word Lengths

In Section V, we analyze the optimization of channel accuracy in term of $e_{\delta t} + e_{\delta y}$ with respect to word length and update interval. In this section, the optimization will be numerically verified. Given a hardware resource A in terms of 'bits/sec', the minimax error using the method presented is calculated in Section V.

Using the method presented in the previous section, we calculate the optimum word length and normalized update period pair (P^*, T'^*_W) at which the minimax error in generation is achieved. The calculated (P^*, T'^*_W) are plotted in Fig. 10 versus the hardware resource in terms of bit rates. In Fig. 10, both the optimum $T'^*_w \ vs \ A$ and $P \ vs \ A$ curves are not smooth. The granularity is the result of the discontinuity of P.

With (P^*, T'^*_W), the minimax error of the generated geometry-based channel is calculated in a straightforward manner. The variation of the error versus hardware resources are plotted in Fig. 11. At (P^*, T'^*_W), the geometry-based channel is generated on an FPGA. The FPGA-generated channel is compared with the channel generated using MATLAB to calculate the error. The optimal channel generation error is plotted in Fig. 11 as the black circles. For comparison, the channel generation with a widely-used fixed bit width is also implemented and plotted.

In Fig. 11, all error-resource (E-A) curves decrease, which means more accurate channels can be generated but at the cost of additional hardware resources. Compared with the fixed word length, the optimum combination of the update period T_W and the word length P generates the smaller error. Furthermore, when the hardware resource is small, the error is relatively closer than that with large resource size. The three fixed-word length curves cross each other within the small resource segment,

TABLE V
COMPUTATION COMPLEXITY COMPARISON

Num. of Operations	Mul.	Add	Bit-shift	Div.	Lookup Table
FFT based method [19] (1024-Size)	10760	30728	0	0	3
Filtering white Gaussian [39]	84	176	0	11	12
DPS based method [42]	240	600	0	0	40
Proposed method	0	40	40	0	0

which implies the update period T_W and the word length P have different weights on determining the error. When allowed to have greater hardware resources, higher P values have lower error than lower P values. This phenomena means the word length is the dominating factor affecting the error for large resource consumption regions. Consider an extreme case in which there are infinite hardware resources, that is, $A \to \infty$. In this extreme case, P can be fixed at an infinitely large value, and T_W can be fixed at an infinitely small value. There will be no loss in the channel generation. Thus, the optimization does not help. The consideration of the extreme case explain why the higher word length $(P = 16)$ curve approaches the optimum curve as A increases to large values. With Fig. 11, using the joint optimization method, we generate the channels with the greatest fidelity.

VII. GEOMETRY-BASED MIMO CHANNEL EMULATION SCALABILITY

Geometry-based MIMO channel emulation using off-the-shelf channel emulators requires significantly large hardware resources and is consequently very expensive. In this section, the scalability of the proposed emulation framework is discussed. With regard to the computational complexity and hardware consumption, the proposed framework is compared with the channel generation methods in [39]–[42]. We first estimate the computational burden in terms of the operations, such as multiplication, addition, and bit shifting. Afterward, the hardware resource consumption in the implementation of the proposed framework on an FPGA is evaluated.

A. Computational-Complexity Analysis

We investigate the computational complexity of the proposed channel emulation method compared to prior state-of-the-art works [39], [41], [42]. The implementation of the reference methods varies depending on different hardware platforms. Hence, to make a fair comparison, we investigate the computational complexity in terms of the number of mathematical operations used per scheme.

Table V lists the numbers of the operations needed per channel linking a TX-RX antenna element pair per time sample. For fair comparison, an arbitrary number of 20 path components per tap is chosen for all the methods. The number of operations for a entire massive MIMO channel can be extrapolated by multiplying the numbers in Table V by the number of TX-RX antenna element pairs.

As a reference, the FFT-based channel generation method proposed in [19] is included in the comparison. The fading channel generation method in [39] is based on filtering white Gaussian noise, which is called the 'correlation method'. A geometric method is explored in [42], where the sinusoids are projected on the space spanned by a discrete prolate spheroidal (DPS) basis.

For the FFT-based channel generation method, a 32,768-size FFT is considered in the original analysis [19], which is too challenging to implement on a hardware platform. For fair comparison, a 1024-size FFT is considered in our analysis. The computation used in the FFT-based method consist of two parts: the FFT computation and the white Gaussian sequence generation. For example, a 1024-size FFT needs 10248 real-number multiplications and the Gaussian generation needs 512 multiplications. For the correlation method based on filtering white Gaussian noise, multiplications and additions are used to filter the Gaussian variable. In parallel, the generation of the Gaussian variable is completed by other additions and lookup table operations. For the DPS-based method, an eigenvalue decomposition is needed to generate the DPS sequence, which consumes a large number of operations. We consider the DPS sequence generation as the operation for initialization, and the corresponding operations do not count towards the computational-complexity analysis. From Table V, the DPS-based method requires a large number of the multiplications and additions, which are used to calculate the product between the channel coefficients and the DPS sequences. Lookup table operations are used to read the values from pre-stored DPS sequences. The coefficients $\cos(\omega)$ can be considered constant during the channel generation process. Therefore, the bit-shift operations are used to implement the product with the coefficients $\cos(\omega)$, and further multiplications are not needed.

B. Geometry-Based MIMO Channel Dimension Scalability

We now estimate the achievable scale on the selected FPGA chip: Virtex-4 VFX100. Two key novelties in the proposed emulation approach allow for better scalability. First, the proposed iterative structure results in significant memory savings for the FPGA. According to the experimental results, one geometry-based MIMO channel consumes one RAMB16. The total memory resources of the aforementioned FPGA chip equals 376 units of RAMB16. Therefore, using the proposed iterative structure, we can support the generation of 376 geometry-based MIMO channels. Second, from the experimental results in Fig. 11, the optimization can effectively save hardware resources at the same channel generation fidelity.

With the same update rate, the top of Table VI shows the consumption of RAM, flip-flops, and LUTs versus word lengths for one channel. The bottom of Table VI presents the expenditure

TABLE VI
HARDWARE RESOURCE CONSUMPTION

Word Length (Bits)	8	9	10	11	12	13	14	15	16	Total Resource
Flip-Flops	810	818	820	824	824	824	826	826	826	84352
LUTs	2905	3056	3640	3664	3704	3744	3788	3826	3834	84352
Slices	1891	1844	2163	2190	2210	2230	2250	2270	2269	42176
18Kb Block RAM	1	1	1	1	1	1	1	1	1	376
Num. of *cos*	20	21	22	23	24	25	26	27	28	Total Resource
Flip-Flops	826	866	890	932	972	1024	1060	1100	1138	84352
LUTs	3704	3900	3758	3880	4254	4918	4302	5200	4284	84352
Slices	2210	2325	2245	2331	2538	2905	2579	3066	2620	42176
18Kb Block RAM	1	1	1	1	1	1	1	1	1	376

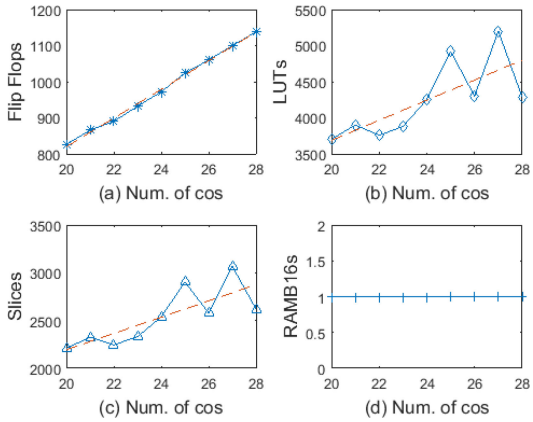

Fig. 12. Consumption of hardware resources versus number of sinusoids.

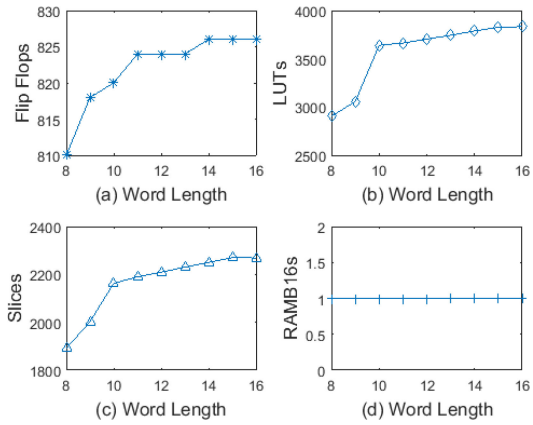

Fig. 13. Consumption of hardware resources versus word lengths.

versus the number of sinusoids with a fixed word length. Since the mean-squared error (MSE) achieved at 12 bits is close to the minimum MSE, the hardware expenditure at 12 bits per data sample is evaluated. The consumption versus the number of sinusoids is plotted in Fig. 12. We plot the consumption of flip-flops, LUTs, Slices, and RAMB16 s versus word lengths for one channel in Fig. 13. In Fig. 12, the blue curves are the hardware resource expenditure versus the number of sinusoids, and the red dash curves are the linear fitting of the blue ones. From Fig. 12, the hardware resource consumption linearly increases with the number of sinusoids. However, the linear relationship does not exist between hardware expenditure and word length in terms of flip-flops, LUTs, and Slices in Fig. 13.

We now analytically compute the number of channels that can be emulated in a single FPGA. Let variables P, L, and N_{ch} denote, respectively, the word length, the number of sinusoids per channel, and the number of channels. Let N_P^{FF} represent the number of flip-flops consumed in the generation of 20 sinusoids at the word length of P, where $8 \leq P \leq 16$. The range of $8 \leq P \leq 16$ is selected for two reasons. First, the optimum bit width is fully confined to this range, as shown in previous section. Second, the channel generation method is validated within this range of P on an FPGA.

Within the range of $8 \leq P \leq 16$, N_P^{FF} is determined by the upper part of Table VI. For simplicity, $N_P^{FF}|_{P=12}$ is taken as the reference. Let q_P^{FF} denote the ratio between N_P^{FF} and N_{12}^{FF}, $q_P^{FF} = N_P^{FF}/N_{12}^{FF}$. Similarly, q_P^{LUT} and q_P^{Slice} are defined. Based on the values shown in the bottom part of Table VI, the total RAM required equals N_{ch}. The number of flip-flops required is of the form $\lceil 39.7L + 25.9 \rceil N_{ch}$ for $P = 12$. When P is not required to be equal to 12, the number of flip-flops required is of the form of $\lceil q_P^{FF}(39.7L + 25.9) \rceil N_{ch}$. Similarly, the total number of LUTs and slices are in the forms of $\lceil q_P^{LUT}(139.1L + 906.0) \rceil N_{ch}$ and $\lceil q_P^{Slice}(85.1L + 493.4) \rceil N_{ch}$, respectively. The selected Virtex-4 FPGA has 84352 flip-flops, 84352 LUTs, 42176 slices, and 376 RAMB16 s. Thus, the selected geometry-based massive MIMO channel can be emulated if the following conditions are satisfied:

$$\begin{cases} \lceil q_P^{FF}(39.7L + 25.9) \rceil N_{ch} \leq 84532 \\ \lceil q_P^{LUT}(139.1L + 906.0) \rceil N_{ch} \leq 84532 \\ \lceil q_P^{Slice}(85.1L + 493.4) \rceil N_{ch} \leq 42176 \\ N_{ch} \leq 376 \end{cases} \tag{37}$$

Based on the constraint equations, the maximum number of geometry-based MIMO dimensions can be estimated. Assume the word length is 16 bits ($P = 16$), and each channel consists of 30 components ($L = 30$). According to (37), the maximum N_{ch} is 13. If the word length is reduced to 12 bits and there are 20 components per channel, the achievable number is 19 for a single FPGA.

VIII. Conclusion

This paper discussed the design of a geometry-based massive MIMO channel emulator by analytically and experimentally studying the tradeoff in channel accuracy and implementation resource consumption. To do so, we presented and analyzed an iterative-based method for generating geometry-based MIMO channels. For additional hardware resource efficiency, we used minimax optimization with respect to the update interval and word length. Extensive experimentation was performed and compared against idealized models generated by a simulator. We leveraged our single channel analysis to understand the implementation resources necessary to build a geometry-based MIMO channel emulator. Using our methods, a single FPGA could emulate the geometry-based MIMO channels up to 19 TX-RX antenna element pairs in real-time. While our analysis is based on the Virtex 4 VFX100, there exists a Virtex-7 VH870 FPGA with 2,820 RAM, which would allow channels linking 141 TX-RX antenna element pairs. In the future, we plan to employ multiple FPGA chips that work together to emulate a network of massive MIMO systems.

Appendix

A. Spectrum of the Generated Channel

Let $H_{u,s}[n]$ denote the generated channel samples at $t = nT_W$. With the samples, the fading channel can be reconstructed. Let $\hat{H}_{u,s}(t)$ denote reconstructed channel,

$$\hat{H}_{u,s}(t) = \sum_{n=-\infty}^{\infty} H^Q_{u,s}[n]u(t - nT_W). \tag{38}$$

Here, $H^Q_{u,s}[n]$ is the quantized channel sample, and the quantization noise is denoted by ξ_n,

$$H_{u,s}[n] = H^Q_{u,s}[n] + \xi_n. \tag{39}$$

The standard deviation of ξ_n, σ_ξ, is determined by:

$$\sigma^2_\xi = 1/12 \cdot 2^{\frac{1}{2}\log 2\pi e\sigma^2_H} 2^{-2P}. \tag{40}$$

Also, σ^2_H is the variance of a channel sample, and $u(t)$ is a rectangular function defined as follows:

$$u(t) = \begin{cases} 1, & 0 \le t \le T_W \\ 0, & \text{otherwise.} \end{cases} \tag{41}$$

The spectrum of a fading channel is denoted by $S_H(f)$, and the spectrum of the reconstructed channel is $S_{\hat{H}}(f)$. The spectrum of the reconstructed channel is affected by word length and update rate. The explicit expression of $S_{\hat{H}}(f)$ is:

$$\begin{aligned} S_{\hat{H}}(f) &= \mathcal{FT}\{\hat{H}_{u,s}(t)\} \\ &= \mathcal{FT}\left\{\sum_{n=-\infty}^{\infty} H^Q_{u,s}[n]u(t - nT_W)\right\} \\ &= \mathcal{FT}\{H^Q_{u,s}[n]\}\mathcal{FT}\{u(t)\} \\ &= \left(S'_H(2\pi f T_W) + \sigma^2_\xi\right) S_u(f), \end{aligned} \tag{42}$$

where $\mathcal{FT}\{\cdot\}$ denotes the Fourier transform operation. Here, $S'_H(\omega)$ denotes the spectrum of $H_{u,s}[n]$, and $S_u(f)$ is calculated by $S_u(f) = (\sin(\pi T_W f)/\pi f)$.

Thus, the proof is complete.

B. Non-Convexity of $\beta(P, T_W)$

In this section, the non-convexity of $\beta(P, T_W)$ is proved. For an analytical expression for the convexity, $\beta(P, T_W)$ is approximated as follows:

$$\begin{aligned} &\beta(P, T_W) \\ &\cong \sum_{i=0}^{I-1} \cos(2\pi i f_M T_W)T_W + \int_0^{1-2^{-P}} \arccos(x)dx - \frac{2}{2\pi} \\ &= \sum_{i=0}^{I-1} \cos(2\pi i f_M T_W)T_W \\ &\quad + \left(x\arccos(x) - \sqrt{1-x^2}\right)\Big|_0^{1-2^{-P}} - \frac{2}{2\pi}. \end{aligned} \tag{43}$$

To show the non-convexity of $\beta(P, T_W)$, the Hessian matrix is first calculated. The partial derivative of β with respect to P and T_W is presented as follows:

$$\frac{\partial^2\beta}{\partial P^2} = \begin{pmatrix} 2^{-(P+1)}\left(1 - \frac{P(P+1)}{2}\right)\left(\arccos(1-2^{-P}) - \frac{\pi}{2}\right) \\ -P^2 2^{-2(P+1)}\left(\left(1-\left(1-2^{-P}\right)^2\right)^{-\frac{1}{2}} - 1\right) \end{pmatrix} \tag{44}$$

$$\frac{\partial^2\beta}{\partial T_W^2} = \sum_{i=0}^{I-1} 2\pi i f_M \begin{pmatrix} -2\sin(2\pi i f_M T_W) \\ -\cos(2\pi i f_M T_W)2\pi i f_M T_W \end{pmatrix} \tag{45}$$

$$\frac{\partial\beta}{\partial P}\frac{\partial\beta}{\partial T_W} = \frac{\partial\beta}{\partial T_W}\frac{\partial\beta}{\partial P} = 0. \tag{46}$$

From (44), (45), and (46), the determinant of the Hessian matrix of $\beta(P, T_W)$,

$$\begin{vmatrix} \frac{\partial^2\beta}{\partial T_W^2} & \frac{\partial\beta}{\partial T_W}\frac{\partial\beta}{\partial P} \\ \frac{\partial\beta}{\partial P}\frac{\partial\beta}{\partial T_W} & \frac{\partial^2\beta}{\partial P^2} \end{vmatrix}$$

is either not always positive or negative. Thus, $\beta(P, T_W)$ is neither convex nor concave.

References

[1] J. Yang *et al.*, "A geometry-based stochastic channel model for the millimeter-Wave band in a 3GPP high-speed train scenario," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 3853–3865, May 2018.

[2] C. A. Gutiérrez, J. T. Gutiérrez-Mena, J. M. Luna-Rivera, D. U. Campos-Delgado, R. Velázquez, and M. Pätzold, "Geometry-based statistical modeling of non-WSSUS mobile-to-mobile Rayleigh fading channels," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 362–377, Jan. 2018.

[3] M. Xiao *et al.*, "Millimeter wave communications for future mobile networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 9, pp. 1909–1935, Sep. 2017.

[4] J. Zhang, Y. Huang, T. Yu, J. Wang, and M. Xiao, "Hybrid precoding for multi-subarray millimeter-wave communication systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 440–443, Jun. 2018.

[5] S. Sun, T. S. Rappaport, M. Shafi, P. Tang, J. Zhang, and P. J. Smith, "Propagation models and performance evaluation for 5G millimeter-wave bands," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8422–8439, Sep. 2018.

[6] W. Xu, Y. Huang, and M. Xiao, *Millimeter Wave Massive MIMO*. Berlin, Germany: Springer, 2018, pp. 1–4.
[7] S. Sun, G. R. MacCartney, and T. S. Rappaport, "A novel millimeter-wave channel simulator and applications for 5G wireless communications," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–7.
[8] Y. X. S. Ju, O. Kanhere, and T. S. Rappaport, "A millimeter-wave channel simulator NYUSIM with spatial consistency and human blockage," in *Proc. IEEE Global Commun. Conf.*, Dec. 2019, pp. 1–6.
[9] "Study on channel model for frequency spectrum above 6 GHz," 3rd Generation Partnership Project, Sophia Antipolis Technology Park, France, TR-36.900, Mar. 2018.
[10] S. Jaeckel, L. Raschkowski, S. Wu, L. Thiele, and W. Keusgen, "An explicit ground reflection model for mm-Wave channels," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops*, Mar. 2017, pp. 1–5.
[11] S. Jaeckel, L. Raschkowski, F. Burkhardt, and L. Thiele, "Efficient sum-of-sinusoids-based spatial consistency for the 3GPP new-radio channel model," in *Proc. IEEE Global Commun. Conf. Workshops*, Dec. 2018, pp. 1–7.
[12] A. M. Pessoa *et al.*, "A stochastic channel model with dual mobility for 5G massive networks," *IEEE Access*, vol. 7, pp. 149 971–149 987, 2019.
[13] M. Hofer *et al.*, "Real-time geometry-based wireless channel emulation," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1631–1645, Feb. 2019.
[14] G. Rao, R. Bhattacharjee, and S. Nandi, "VLSI architecture for Rayleigh and Rician fading generators," *IEEE Region 10 Conf. TENCON*, Nov. 2004, pp. 121–124, vol. 3.
[15] P. Huang, M. J. Tonnemacher, Y. Du, D. Rajan, and J. Camp, "Towards scalable network emulation: Channel accuracy versus implementation resources," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1959–1967.
[16] P. Huang, D. Rajan, and J. Camp, "Weibull and Suzuki fading channel generator design to reduce hardware resources," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2013, pp. 3443–3448.
[17] C. Xiao, Y. R. Zheng, and N. C. Beaulieu, "Novel sum-of-sinusoids simulation models for Rayleigh and Rician fading channels," *IEEE Trans. Wireless Commun.*, vol. 5, no. 12, pp. 3667–3679, Dec. 2006.
[18] H. Eslami, S. V. Tran, and A. M. Eltawil, "Design and implementation of a scalable channel emulator for wideband MIMO systems," *IEEE Trans. Veh. Technol.*, vol. 58, no. 9, pp. 4698–4709, Nov. 2009.
[19] J. Yang, C. A. Nour, and C. Langlais, "Correlated fading channel simulator based on the overlap-save method," *IEEE Trans. Wireless Commun.*, vol. 12, no. 6, pp. 3060–3071, Jun. 2013.
[20] S. Buscemi and R. Sass, "Design of a scalable digital wireless channel emulator for networking radios," in *Proc. Military Commun. Conf.*, Nov. 2011, pp. 1858–1863.
[21] M. Koizumi, T. Ebata, T. Tsutsumi, K. Ohshima, and M. Terada, "Design and implementation of scalable distributed wireless network emulator for high-speed mobility," in *Proc. Int. Conf. Inf. Netw.*, Feb. 2012, pp. 302–307.
[22] P. Zheng and L. Ni, "Empower: A network emulator for wireline and wireless networks," in *Proc. 22nd Annu. Joint Conf. IEEE Comput. Commun. IEEE Societies*, vol. 3, Mar./Apr. 2003, pp. 1933–1942.
[23] J. Beshay *et al.*, "Wireless networking testbed and emulator (WiNeTestEr)," *Comput. Commun.*, vol. 73, pp. 99–107, 2016.
[24] O. Nasr and B. Daneshrad, "Design and FPGA implementation an accurate real time 3X4 MIMO channel emulator," in *Proc. Conf. Record 43rd Asilomar Conf. Signals, Syst. Comput.*, Nov. 2009, pp. 764–768.
[25] F. Ren and Y. Zheng, "A novel emulator for discrete-time MIMO triply selective fading channels," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 57, no. 9, pp. 2542–2551, Sep. 2010.
[26] A. Alimohammad and S. F. Fard, "A compact architecture for simulation of spatio-temporally correlated MIMO fading channels," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 61, no. 4, pp. 1280–1288, Apr. 2014.
[27] "Spirent VR5 HD spatial channel emulator," Dec. 2016. [Online]. Available: https://www.spirent.com/Products/VR5-Channel-Emulator
[28] "Azimuth ACE - MIMO channel emulator for broadband wireless testing," Jul. 2012. [Online]. Available: http://www.azimuthsystems.com/platforms_channel_mx.htm
[29] "Study on 3D channel model for LTE," , 3rd Generation Partnership Project, Sophia Antipolis Technology Park, France, TR-36.873, 2015.
[30] M. Patzold, R. Garcia, and F. Laue, "Design of high-speed simulation models for mobile fading channels by using table look-up techniques," *IEEE Trans. Veh. Technol.*, vol. 49, no. 4, pp. 1178–1190, Jul. 2000.
[31] D. De Caro, N. Petra, and A. G. M. Strollo, "Reducing lookup-table size in direct digital frequency synthesizers using optimized multipartite table method," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 55, no. 7, pp. 2116–2127, Aug. 2008.
[32] L. Yan *et al.*, "A 13 μa analog signal processing IC for accurate recognition of multiple intra-cardiac signals," *IEEE Trans. Biomed. Circuits Syst.*, vol. 7, no. 6, pp. 785–795, Dec. 2013.
[33] W. Jake, *Microwave Mobile Communication*. Piscataway, NJ, USA: IEEE Press, 1974.
[34] A. Bellaouar, M. S. O'brecht, A. M. Fahim, and M. I. Elmasry, "Low-power direct digital frequency synthesis for wireless communications," *IEEE J. Solid-State Circuits*, vol. 35, no. 3, pp. 385–390, Mar. 2000.
[35] S. Hong and D. Jee, "A 0.052 mm^2, $<$0.4% THD, sinusoidal current generator for bio-impedance measurement using a recursive digital oscillator and current-domain FIR filter," *IEEE Trans. Circuits Syst. II: Express Briefs*, vol. 66, no. 6, pp. 894–898, Jun. 2019.
[36] G. Wu, Y. Dou, J. Sun, and G. D. Peterson, "A high performance and memory efficient LU decomposer on FPGAs," *IEEE Trans. Comput.*, vol. 61, no. 3, pp. 366–378, Mar. 2012.
[37] A. Alimohammad and B. Cockburn, "Modeling and hardware implementation aspects of fading channel simulators," *IEEE Trans. Veh. Technol.*, vol. 57, no. 4, pp. 2055–2069, Jul. 2008.
[38] B. Sklar, "Rayleigh fading channels in mobile digital communication systems .I. Characterization," *IEEE Commun. Mag.*, vol. 35, no. 7, pp. 90–100, Jul. 1997.
[39] P. Kyosti and T. Jamsa, "Complexity comparison of MIMO channel modelling methods," in *Proc. 4th Int. Symp. Wireless Commun. Syst.*, Oct. 2007, pp. 219–223.
[40] F. Kaltenberger, G. Steinbock, G. Humer, and T. Zemen, "Low-complexity geometry based MIMO channel emulation," in *Proc. 1st Eur. Conf. Antennas Propag.*, Nov. 2006, pp. 1–8.
[41] F. Kaltenberger, T. Zemen, and C. W. Ueberhuber, "Low-complexity geometry-based MIMO channel simulation," *EURASIP J. Adv. Signal Process.*, vol. 2007, no. 1, Jul. 2007, Art. no. 095281.
[42] M. Hofer, Z. Xu, and T. Zemen, "Real-time channel emulation of a geometry-based stochastic channel model on a SDR platform," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun.*, Jul. 2017, pp. 1–5.

Pengda Huang received the Ph.D. degree in electrical engineering from Southern Methodist University, Dallas, TX, USA. He is currently with Marvell Semiconductors, Santa Clara, CA, USA. His research interests fall in the areas of signal processing in spectrum sharing networks and source coding.

Matthew Jordan Tonnemacher received the B.S., M.S., and Ph.D. degrees in electrical engineering from Southern Methodist University, Dallas, TX, USA, in 2011, 2013, and 2019, respectively. He has been with Samsung Research America, Plano, TX, USA, since 2016, as a Wireless Systems Engineer. His current research interests include spectrum sharing, software-defined radio, and next-generation wireless technologies.

Yongjiu Du (Member, IEEE) received the B.S. and M.S. degrees from the Beijing University of Aeronautics and Astronautics, Nanjing, China (renamed as Beihang University) in 2004 and 2007, respectively, and the Ph.D. degree from Southern Methodist University, Dallas, TX, USA, in 2014. He was a Research Engineer with THOMSON (renamed as Technicolor now) Corporate Research Department from 2006 to 2009. His research interest includes physical-layer and cross-layer design and optimization of wireless communications, multiuser MIMO, OFDM, channel modeling and emulation, hardware architecture design, and implementation of digital signal processing.

Dinesh Rajan (Senior Member, IEEE) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology (IIT), Madras, India, and the M.S. and Ph.D. degrees in electrical and computer engineering from Rice University, Houston, TX, USA. He joined the Electrical Engineering Department with Southern Methodist University (SMU), Dallas, TX, USA in August 2002, as an Assistant Professor. He is currently the Department Chair and Cecil and IDA Green Professor with the Electrical and Computer Engineering Department, Southern Methodist University. His current research interests include communications theory, wireless networks, information theory, and computational imaging.

He was the recipient of the NSF CAREER Award for his work on applying information theory to the design of mobile wireless networks. He is also a recipient of the Golden Mustang Outstanding Faculty Award and the Senior Ford Research Fellowship from SMU.

Joseph Camp (Member, IEEE) received the B.S. (Hons.) degree in electrical and computer engineering from the UT-Austin, Austin, TX, USA, and the M.S. and Ph.D. degrees in electrical and computer engineering from Rice University, Houston, TX, USA. He is currently an Associate Professor of Electrical and Computer Engineering with Southern Methodist University, Dallas, TX, USA. He joined the SMU Faculty in 2009. His research team has performed more than 200 million in-field wireless measurements around the world via Android deployment and local characterization via drones, campus buses, vehicles, and buildings. His research interests are wireless communications and networking, crowdsourcing, and drones, specifically focused on the deployment, measurement, and analysis of large-scale systems and development of embedded protocols. He was the Chief Network Architect for the Technology For All (TFA) Network, a mesh deployment in Houston, TX, which served thousands of users in an under-resourced community. He was the recipient of the Ralph Budd Award for the Best Engineering Thesis at Rice University in 2010, the National Science Foundation CAREER Award in 2012, and the Golden Mustang Teaching Award in 2014.