Received 27 October 2022, accepted 11 November 2022, date of publication 14 November 2022, date of current version 18 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3222063

RESEARCH ARTICLE

HermesPy: An Open-Source Link-Level Evaluator for 6G

JAN ADLER, TOBIAS KRONAUER, AND ANDRÉ NOLL BARRETO, (Senior Member, IEEE)
Barkhausen Institut, 01187 Dresden, Germany

Corresponding author: Jan Adler (jan.adler@barkhauseninstitut.org)

This work supported by the Saxon State government out of the state budget approved by the Saxon State Parliament.

ABSTRACT In this paper we present the **He**terogeneous **R**adio **M**obil**e** **S**imulator in **Py**thon (HermesPy), an open-source evaluator for studies on the physical layer of 6-th generation (6G) wireless systems. The software framework combines a link-level simulator, featuring a comprehensive set of models and algorithms, with hardware in the loop verification using software-defined radios. Among other features, it supports the evaluation of multi-node joint communications and sensing scenarios, which are likely to be one of the key features in 6G. We benchmark the framework's feature set against existing simulators and introduce its architectural concept. Finally, extensive simulation results demonstrating core functionalities are presented and discussed.

INDEX TERMS 6G, joint communications and sensing, link-level simulation.

I. INTRODUCTION

While the current 5-th generation (5G) of wireless networks are being massively deployed throughout the world, researchers are already investigating novel methods to improve the network performance and extend the existing feature set, both for 5G-Advanced and upcoming 6G networks. Although a consensus on the details of 6G has yet to be reached, it will certainly introduce a straightforward evolution of 5G, providing higher data rates, lower latencies and increased capacity, but may also introduce several completely new features. Specifically in the physical layer (PHY) researchers currently consider: (i) the usage of sub-THz waves [1]; (ii) the application of machine learning (ML) techniques [2], replacing algorithmic approaches for several PHY functions; (iii) the enhancement of propagation channels by means of reflective intelligent surface (RIS) [3] and (iv) integrated sensing and communications (ISAC), introducing radar as a service (RaaS) in wireless networks [4], [5].

These emerging techniques must be thoroughly investigated and discussed before their adoption into upcoming standards. As an integral part of their investigation, predicting and verifying the performance of proposed algorithms for wireless systems is one of the most common challenges encountered in signal-processing research for both communication and sensing applications. However, the theoretical derivation of performance indicators for complex, non-linear systems usually proves to be challenging. On the other hand, the experimental approach of realizing multiple hardware iterations and performing the respective measurement campaigns for benchmark data collection is expensive and time-consuming. Both theoretical and experimental evaluation strategies are therefore typically infeasible in early stages of development. Instead, researchers routinely resort to numerical simulations based on precise models of the wireless system behaviours on the PHY, referred to as link-level simulations.

With the expected emergence of novel PHY features in 6G software tools providing environments to perform the respective link-level simulations are required as well. We propose HermesPy, the **He**terogeneous **R**adio **M**obil**e** **S**imulator in **Py**thon, for the investigation of algorithms within current and next-generation wireless systems, from now on referred to as Hermes. Hermes is a software framework implemented primarily in Python [6] and distributed under the Affero General Public License [7] as a package [8] listed in the Python package index [9]. Hermes aims to bridge the existing gap between physical layer simulations of sensing and communication applications by providing a flexible and easily extensible software architecture adaptable to any

The associate editor coordinating the review of this manuscript and approving it for publication was Jie Tang.

link-level wireless scenario. It may be configured to control software-defined radio (SDR) devices, allowing researchers to conduct measurement campaigns of novel algorithms in real-life multi-node multiple input multiple output (MIMO) scenarios with minimal implementation overhead. We firmly believe that open-source research software is essential to make research results reproducible and verifiable. Therefore, Hermes' source code is publicly accessible as a GitHub repository [10], with the project being explicitly open to any contribution. Hermes is available free of charge for application and extension to both the scientific community and commercial users.

The paper is structured the following way: In Section II Hermes' feature set will be benchmarked against other available link-level simulation tools, highlighting Hermes' unique contribution to the state of the art. Hermes' architecture and system model will be described in Section III, and, since it supports both communications and sensing, the operation for both services will be described in Section V and in Section VI, respectively. Simulation results obtained with the simulator will be shown and discussed in section VII. Finally, section VIII describes Hermes hardware in the loop (HiL) verification architecture and provides a minimal code example demonstrating the workflow using Hermes' application programming interface (API).

II. STATE OF THE ART

A. EXISTING SIMULATION TOOLS

In recent years, a diverse landscape of tools for link-level physical layer simulations of communication systems has emerged [11]. Focusing on various aspects of the communications signal-processing chain, each tool has unique strengths and weaknesses, resulting from both software architecture choices and the selection of implemented communication algorithms and models.

The Communications Toolbox [12] and 5G Toolbox [13] are part of the software suite by Mathworks and controlled by the MATLAB script language. They offer third generation partnership project (3GPP) standard-compliant waveforms and channel models as well as link- and system-level simulation routines. Both MATLAB and the aforementioned toolboxes are commercially licensed closed-source products. While the MATLAB suite is, in terms of modeling features for link-level simulation, the most complete among the compared tools, it is in and of itself not a simulator. The task of implementing a viable simulation pipeline is left to the user, resulting in high flexibility while at the same time demanding significant additional effort when building more intricate simulations.

The GTEC 5G simulator [14] combines testbed universal software radio peripheral (USRP) integration with link-level simulation functionalities. Implemented in C++ and MATLAB, it enables the transmission of orthogonal frequency-division multiplexing (OFDM) and filterbank multicarrier modulation (FBMC) waveforms over a singluar 2×2 MIMO antenna link using real hardware in addition to link-level simulations with several 3GPP standard-compliant channel models. Each simulation sweep over a specific parameter must be implemented manually, extended simulation features such as multidimensional parameter sweeping, stopping criteria and workload distribution are not available.

The Vienna 5G link-level simulator [15] is part of the 5G simulation suite developed by technical university of Vienna. It is implemented in MATLAB and released under an academic license. It provides a simulation framework for multi-node investigations of OFDM waveforms, channel codings and equalization schemes as well as channel and antenna characteristics models. The simulator is configurable to sweep over a wide range of parameters, the computational load may be distributed via the Matlab Parallel Computing Toolbox [16], but natively supports neither multidimensional parameter sweeps nor premature stopping.

Note that both the GTEC and the Vienna simulators require an additional acquisition of MATLAB licenses, regardless of their own licensing models.

The third version of Network Simulator [17] (ns-3) is a community-developed C++ library for the event-based simulation of multi-node computer network communication, considering the link- data-link and network layer with extensions for wireless link-level simulations [18], [19]. Distributed as a Git repository under a public license, it features a wide range of channel propagation models including antenna characteristics modeling but has limited configuration options regarding channel coding, equalization and radio-frequency chain modeling, since it focuses on network-level simulations, relying mostly on a simplified PHY model. The simulator itself only sweeps over the time domain, users are required to implement custom callbacks to vary simulation parameters or implement a multidimensional sweep. The workload may be distributed by ns-3's interface to OpenMPI [20].

The Simulator for Mobile Networks [21] (SiMoNe) is a link-level simulator specialising in carrier frequencies up to THz bands. Implemented in C# and not publicly available, it features Rayleigh, Rician and raytracing-based channel modeling as well as channel codes and single carrier waveforms for THz communications. Among the reviewed simulators, it is the only one to consider oscillator phase noise (PN). Users may investigate a single communication link by sweeping over a choice of several parameters, multidimensional sweeping, premature stopping or workload distribution are, to the best of our knowledge, not supported.

A Fast Forward Error Correction Toolbox (AFF3CT) is a set of channel coding evaluation tools implemented in C++ and distributed under the MIT public license [24]. It provides fast implementations of state-of-the-art error-correction algorithms and a complementing simulation helper to investigate bit error rate (BER) and frame error rate (FER) performances over a single input single output (SISO) additive white Gaussian noise (AWGN) channel. It should be noted that both SiMoNe and the proposed HermesPy simulator obtain parts

TABLE 1. Simulation tools general comparison.

Simulator	Language	License	Pipeline	Sweep Params	Multidim	Stopping	Distribution	Links	HiL
Matlab [12], [13]	MATLAB	Commercial	No	-	-	-	-	-	-
GTEC 5G [14]	MATLAB, C++	Public	Yes	Specific	No	No	No	Single	Yes
Vienna 5G [15]	MATLAB	Academic	Yes	All	No	No	Matlab [16]	Multiple	No
ns-3 [17]–[19]	C++, Python	Public	Yes	Time	No	No	OpenMPI [20]	Multiple	No
SiMoNe [21]	C#, C++	Closed	Yes	Specific	No	No	No	Single	No
Sionna [22]	Python	Public	Yes	SNR	No	No	Tensorflow [23]	Single	No
AFF3CT [24]	C++	Public	Yes	SNR	No	No	No	Single	No
HermesPy	Python, C++	Public	Yes	All	Yes	Yes	Ray [25]	Multiple	Yes

of their error correction abilities by wrapping the AFF3CT project in C# and Python, respectively.

Sionna [22] is a Python library for link-level simulations built around the Tensorflow [23] framework, published by Nvidia under the Apache 2.0 license. It is explicitly geared towards GPU-accelerated machine learning by implementing the whole communication signal processing chain in Tensorflow and provides 3GPP standard-compliant channel codings and channel models, as well as basic MIMO support and OFDM communication waveforms. While Sionna's programming interface is highly flexible and theoretically supports the simulation of multi-node scenarios, the provided pipelines only sweep over the receiver signal-to-noise ratio (SNR) of a single link.

Formally, the reviewed tools differ in terms of their chosen implementation languages and licensing models under which they are being distributed. Their core functionalities as link-level simulators can be characterized by whether a configurable simulation pipeline is provided, over which parameters this pipeline can be configured to sweep during simulation runtime, if this parameter sweep can be multidimensional, if a premature simulation stopping criterion can be defined, how the simulation workload is distributed and how many communication links may be considered within a single simulation. Table 1 displays a structured overview of these indicators, including a comparison to the proposed Hermes simulator. In order to further distinguish Hermes' novelty, the table indicates the support for HiL evaluations.

When considering the detailed simulator features, conducting a fair comparison between the existing simulation tools is challenging, since each tool focuses on different aspects of signal processing and channel modeling for communication links, providing unique features tailored towards its respective specialisation. However, in order to particularly highlight Hermes' contribution, we chose to characterize each tool by its respective natively supported waveforms, forward error correction (FEC), data precoding schemes, MIMO schemes, supported channel models, hardware impairment models and key performance indicators (KPIs). Table 2 displays a detailed feature comparison, checkmarks indicating feature support. The considered communication waveforms are OFDM, frequency-modulated continuous wave (FMCW), frequency shift keying (FSK), and single carrier (SC) with any pulse shape. The considered FEC operations are low-density parity-check (LDPC) [26], Polar [27], [28], Turbo [29], recursive systematic convolutional (RSC) [30], Reed-Solomon (RS) [31] and Bose-Chaudhuri-Hocquenghem (BCH) [32] codes, as well as bit interleaving and scrambling. The considered spatial MIMO codings are maximum ratio combining (MRC) [33], selection combining (SCEC) [34], space-time block code (STBC) and spatial multiplexing (SM). Additionally, we indicate whether any form of beamforming (BF) was implemented. The considered channel models are an ideal channel only introducing AWGN at the receiver side, the 3GPP tapped delay line (TDL) and cluster delay line (CDL) models and the Quadriga channel model implemented in MATLAB. Additionally, we indicate the support for any form of spatial radar channel modeling. The considered hardware models are antenna polarizations, mutual coupling between antenna elements, transmit-receive isolation in multi-antenna arrays, power amplifier (PA) and low-noise amplifier (LNA) non-linearities, in-phase and quadrature-phase imbalance (I/Q), analog-to-digital converter (ADC) quantization and phase noise (PN) [35]. The considered key performance indicators estimators are BER, FER, block error rate (BLER), data rate throughput (DRX) and receiver operation characteristics (ROC).

In summary, of the existing link-level simulation tools only few are distributed under a public license, release their source code and accept third-party contributions, which would be desirable in platforms for cooperative research and development, both in academia and industry. The reviewed tools usually offer only limited sweeping parameter options, sweeping over multiple parameters is either completely unsupported or has to be implemented by the user. None of the reviewed publicly available tools currently support the evaluation of sensing applications, such as providing implementations for radar waveforms or spatial radar channels and the respective KPI estimations. Apart from antenna polarization models, as suggested in the 3GPP standards, hardware effects are usually neglected during link-level simulations. However, simulations modeling only propagation channels and antenna polarizations will overestimate the performance of communication links, if the effects of radio frequency (RF) chain impairments are not considered (see Figure 16), as these can have a significant impact in the KPIs [36]. In addition, when designing ISAC systems for 6G, isolation between transmitting and receiving antennas is a crucial performance-limiting factor for the system's sensing capabilities [37], [38], which

TABLE 2. Simulation tools feature comparison.

Simulator	**Waveforms**				**FEC Codes**								**MIMO**					**Channels**					**Hardware**							**KPIs**				
	OFDM	FMCW	FSK	SC	LDPC	Polar	Turbo	RSC	RS	BCH	Interleave	Scramble	MRC	SCEC	STBC	SM	BF	AWGN	TDL	CDL	Quadriga	Radar	Polarity	Coupling	Isolation	I/Q	Amplifier	ADC	PN	BER	BLER	FER	DRX	ROC
Matlab [12], [13]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
GTEC 5G [14]	✓																		✓	✓										✓				
Vienna 5G [15]	✓				✓	✓	✓	✓			✓	✓	✓			✓	✓	✓	✓	✓			✓				✓			✓		✓	✓	
ns-3 [17]–[19]	✓																✓	✓	✓	✓	✓		✓							✓	✓			
SiMoNe [21]				✓	✓				✓									✓					✓				✓		✓	✓			✓	
Sionna [22]	✓				✓	✓	✓	✓			✓	✓						✓	✓	✓			✓							✓	✓			
AFF3CT [24]				✓	✓	✓	✓	✓	✓	✓	✓	✓						✓												✓		✓		
HermesPy	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓

needs to be taken into account in simulations. When moving from simulation-based evaluations to testbed-based evaluations, a switch in frameworks and the resulting adaptations to a new programming interface typically result in significant implementation overhead. Only few tools offer interfaces to SDRs, enabling the verification of algorithms implemented within the native simulation framework directly by conducting real-world measurement campaigns in testbed setups.

B. CONTRIBUTION

Hermes introduces a novel framework approach to link-level evaluations, enabling both the simulation and HiL verification of multi-node ISAC applications within a single unifying API. It contributes a unique combination of features only sparsely supported within the landscape of existing tools, namely

- Multidimensional sweeps over arbitrary parameters
- Online confidence estimation of KPIs
- Distributed computation of Monte Carlo simulations
- Comprehensive hardware effects modeling
- Extensible implementations of communication, sensing and ISAC waveforms
- Native SDR support for the verification of simulation results

Implemented in Python and C++, Hermes follows a strictly object-oriented, modular coding style, enabling the quick integration of novel waveforms and algorithms to be investigated within the framework. By default, Hermes provides implementations for OFDM, FMCW, FSK and SC communication waveforms, state of the art FEC codings, precodings, beamformers and standard-compliant TDL and CDL channel models. Within simulations, hardware effects such as antenna polarization, mutual coupling, transmit-receive isolation, I/Q imbalance, amplifier characteristics and ADC quantization are modeled. Interference scenarios featuring multiple nodes, possibly of different radio access technologys (RATs) operating in identical or neighbouring spectrum bands may be investigated. The resulting Monte-Carlo style simulation campaigns are not limited to specific parameters, such as SNR, and can instead be configured to sweep over arbitrary parameter combinations. Built around the Ray [25] framework, Hermes supports the distribution of simulation workloads on multi-core systems as well as high-performance computing (HPC) clusters. Hermes can be used as either a command-line tool configured by text-based parameter files or as a Python library within third-party projects, providing either its full simulation pipeline or individual modules for signal processing, PHY simulation and hardware control. Communications and radar techniques, such as IEEE 802.11 and LoRaWAN, can be modelled and simulated by adjusting the simulation parameters accordingly. Users may immediately verify simulation results of algorithms implemented within the framework over the air by configuring Hermes to transmit and receive waveforms over hardware testbeds. For this purpose, Hermes provides bindings to USRPs and audio interfaces, bindings to custom setups may be easily added.

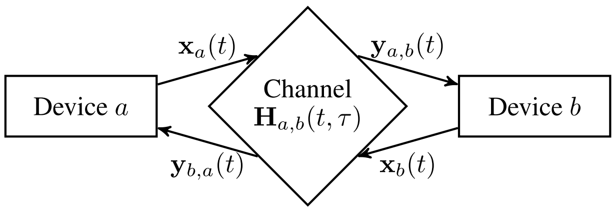


FIGURE 1. Hermes link architecture.

III. SIMULATOR ARCHITECTURE

Hermes aims to abstract the process of wireless communication and sensing signal processing within a strictly object-oriented software architecture. Signal processing steps and hardware models are represented by a composition of dedicated classes. Software users may simulate customized scenarios by adapting compositions and configuring exposed parameter attributes. Within the Hermes framework, any physical entity capable of emitting or receiving electromagnetic waves is represented by a device. The propagation characteristics of electromagnetic waves exchanged between two devices are represented by radio channels. Therefore, the model of a single communication link consists of the tuple of two devices and their respective connecting channel, as visualized in Figure 1. Each device with index a emits a complex-valued waveform $\mathbf{x}_a(t) \in \mathbb{C}^{M_{\mathrm{Tx}}^{(a)}}$, $M_{\mathrm{Tx}}^{(a)}$ being the number of active device antennas during transmission, and receives a complex-valued waveform $\mathbf{y}_{b,a}(t) \in \mathbb{C}^{M_{\mathrm{Rx}}^{(a)}}$ after channel propagation over a link with device b, $M_{\mathrm{Rx}}^{(a)}$ being the number of active device antennas during reception. The signal distortion resulting from channel propagation over a link between two devices may be characterized by a time-variant channel impulse response $\mathbf{H}_{a,b}(t,\tau) \in \mathbb{C}^{M_{\mathrm{Rx}}^{(b)} \times M_{\mathrm{Tx}}^{(a)}}$, leading to

$$\mathbf{y}_{a,b}(t) = \int_{0}^{\infty} \mathbf{H}_{a,b}(t,\tau)\mathbf{x}_a(t-\tau)\,d\tau \tag{1}$$

as a general time-continuous expression of the channel propagation, t denoting time and τ denoting delay in seconds, respectively. The actual design of $\mathbf{H}_{a,b}$ is defined by the selected channel model's specifications. By default, Hermes assumes channels to be reciprocal, therefore $\mathbf{H}_{b,a} = \mathbf{H}_{a,b}^{\mathsf{T}}$, with $\{\cdot\}^{\mathsf{T}}$ denoting the matrix transpose operator. It should be noted that this reciprocity assumption may be invalid in cases where two linked devices operate on widely differing carrier frequencies. Hermes does not distinguish between up- down- and side-link by defining a device as either base-station or terminal. Instead, the channel model choice implicitly characterizes the link. A full simulation scenario may consist of an arbitrary number of devices D, with every device being linked to all other devices and to itself by a channel instance. Therefore, a scenario contains $\sum_{d=1}^{D} d = \frac{D(D+1)}{2}$ links with the equal amount of channel instances, forming a symmetric matrix of links between devices, as visualized in Figure 2. In such an architecture, the interference among devices occuring in complex multi-node wireless communication scenarios can be addressed without the need for a unifying channel model over every device link. Instead, each link is configured with a specific channel model best fitting the scenario assumptions. For example, joint communications and sensing scenarios could contain radar channel model instances on the scenario self-interference diagonal and vehicle-to-vehicle side-link communication channels on the off-diagonals to investigate automotive applications. That being said, while each link is configured with an individual channel model instance, the instances may share mutual information. For example, considering a ray-tracing scenario where several devices are simulated, the channel impulse responses between the devices would be constructed from ray propagation paths reflected by the same objects, but illuminated from varying perspectives depending on the device orientations and positions.

As a consequence of the assumed link model, the electromagnetic signal impinging onto each device is a superposition

$$\mathbf{y}_a(t) = \sum_{d=1}^{D} \mathbf{y}_{d,a}(t) \tag{2}$$

$$= \sum_{d=1}^{D} \int_{0}^{\infty} \mathbf{H}_{d,a}(t,\tau)\mathbf{x}_d(t-\tau)\,d\tau \tag{3}$$

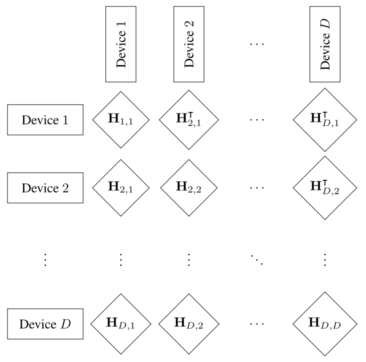


FIGURE 2. Hermes scenario architecture.

of all signals emerging from the device's respective links after channel propagation, in other words, a sum over all channel models within the a-th row of Figure 2.

Until now, Hermes' channel model has been described in time-continuous domain in order to properly motivate the implemented simulator architecture. Numeric simulations, however, are required to sample continuous models into discrete domains with a sufficient sampling rate. Hermes considers two dedicated sampling rate requirements per device, $f_{\mathrm{Tx,s}}^{(d)}$ and $f_{\mathrm{Rx,s}}^{(d)}$ in Hz, for transmit and receive processing, respectively. During simulation runtime, the signal models being generated and passed from one processing step to another are dynamically resampled in order to match the sampling rate requirements of the currently executed processing step. Therefore,

$$\begin{aligned}\mathbf{Y}_d &= \left[\mathbf{y}_d(0), \mathbf{y}_d\left(\frac{1}{f_{\mathrm{Rx,s}}^{(d)}}\right), \ldots, \mathbf{y}_d\left(\frac{N_{\mathrm{Rx}}^{(d)}-1}{f_{\mathrm{Rx,s}}^{(d)}}\right)\right] \\ &= \left[\mathbf{y}_d^{(1)}, \mathbf{y}_d^{(2)}, \ldots, \mathbf{y}_d^{(N_{\mathrm{Rx}}^{(d)})}\right] \in \mathbb{C}^{M_{\mathrm{Rx}}^{(d)} \times N_{\mathrm{Rx}}^{(d)}}\end{aligned} \tag{4}$$

is the matrix of $N_{\mathrm{Rx}}^{(d)}$ time-discrete sample streams impinging onto the d-th device's antennas, subsequently feeding into the device's receive signal processing chains. Equivalently,

$$\begin{aligned}\mathbf{X}_d &= \left[\mathbf{x}_d(0), \mathbf{x}_d\left(\frac{1}{f_{\mathrm{Tx,s}}^{(d)}}\right), \ldots, \mathbf{x}_d\left(\frac{N_{\mathrm{Tx}}^{(d)}-1}{f_{\mathrm{Tx,s}}^{(d)}}\right)\right] \\ &= \left[\mathbf{x}_d^{(1)}, \mathbf{x}_d^{(2)}, \ldots, \mathbf{x}_d^{(N_{\mathrm{Tx}}^{(d)})}\right] \in \mathbb{C}^{M_{\mathrm{Tx}}^{(d)} \times N_{\mathrm{Tx}}^{(d)}}\end{aligned} \tag{5}$$

is the matrix of $N_{\mathrm{Tx}}^{(d)}$ time-discrete antenna samples emerging from device transmit processing chains and feeding into channel links. The channel model is sampled into a time- and delay-discrete impulse response tensor

$$\mathbf{H}_{a,b}^{(n,\tau)} = \mathbf{H}_{a,b}\left(\frac{n-1}{f_{\mathrm{Rx,s}}^{(b)}}, \frac{\tau-1}{f_{\mathrm{Rx,s}}^{(b)}}\right) \quad \text{for} \quad n,\tau = 1 \ldots N_{\mathrm{Rx}}^{(b)} \tag{6}$$

according to the currently receiving device's sampling rate requirements $f_{\mathrm{Rx,s}}^{(d)}$. Computing the channel impulse response $\mathbf{H}_{a,b}^{(n,\tau)}$ for massive MIMO systems simulating large delay spreads can require significant processing and memory resources. That being said, most channel models consider discrete delay taps, leading to $\mathbf{H}_{a,b}^{(n,\tau)}$ being sparse in its delay dimension τ, which can be exploited to reduce memory requirements in practice.

A. RESAMPLING

Each MIMO signal $\mathbf{X}_d$ is assumed to be emitted at the device's central transmit carrier frequency $f_{\mathrm{Tx,c}}^{(d)}$, which is a freely configurable simulation parameter. Similarly, devices assume a central carrier frequency $f_{\mathrm{Rx,c}}^{(d)}$ during reception. Hermes does not consider the modulated RF pass-band samples during its simulation calculations. Instead, in order to minimize memory requirements, it internally handles MIMO signals as the tuple of their respective complex base-band samples, sampling rates and assumed carrier frequencies. The downside of this approach is the inability to generally compute the superposition of two signals directly as a sum of both sample matrices, since they may feature different centre frequencies and sampling rates. Instead, the signals have to be resampled and mixed to match the superposition's target carrier frequency and sampling rate. By applying the Whittaker-Kotelnikov-Shannon sampling theorem, the resampling of device a's transmission to the target specifications of device b can be expressed as a

$$\widetilde{\mathbf{X}}_{a,b} = \mathbf{X}_a \mathbf{W}_{\mathrm{f}}^{(a,b)} \mathbf{W}_{\mathrm{s}}^{(a,b)} \mathbf{W}_{\mathrm{c}}^{(a,b)} \tag{7}$$

$$= \left[\tilde{\mathbf{x}}_{a,b}^{(1)}, \tilde{\mathbf{x}}_{a,b}^{(2)}, \ldots, \tilde{\mathbf{x}}_{a,b}^{(N_{\mathrm{Rx}}^{(b)})} \right] \in \mathbb{C}^{M_{\mathrm{Tx}}^{(a)} \times N_{\mathrm{Rx}}^{(b)}} \tag{8}$$

consisting of a sequence of linear operations $\mathbf{W}$. Considering the target carrier frequency $f_{\mathrm{Rx,c}}^{(b)}$ and sampling rate $f_{\mathrm{Rx,s}}^{(b)}$ a superimposed signal with carrier frequency $f_{\mathrm{Tx,c}}^{(a)}$ and sampling rate $f_{\mathrm{Tx,s}}^{(a)}$ has to be mixed to a virtual carrier frequency

$$\Delta f_{\mathrm{c}}^{(a,b)} = f_{\mathrm{Tx,c}}^{(a)} - f_{\mathrm{Rx,c}}^{(b)} \tag{9}$$

perceived by the target. Depending on the sampling rates and carrier frequency distance, typically only a section of the signal spectra overlap, requiring a resampling bandpass filter of width

$$B_{\mathrm{Rs}}^{(a,b)} = \min\left(\frac{1}{2}\left(f_{\mathrm{Tx,s}}^{(a)} + f_{\mathrm{Rx,s}}^{(b)} \right) - \|\Delta f_{\mathrm{c}}^{(a,b)}\|,\, f_{\mathrm{Rx,s}}^{(b)} \right) \tag{10}$$

centered around the frequency

$$f_{\mathrm{Rs,c}}^{(a,b)} = \frac{1}{2}\left(f_{\mathrm{Tx,c}}^{(a)} - f_{\mathrm{Rx,c}}^{(b)} \right) \tag{11}$$

in order to avoid aliasing resulting from the mixing operations within the final superpositioned discrete samples. This relationship is visualized in Figure 3. $\mathbf{W}_{\mathrm{f}}^{(a,b)}$ is the convolution matrix representation of a bandpass filter impulse response for the respective bandwidth and center frequencies. Hermes considers superpositioned signals with non-overlapping bands $B_{\mathrm{Rs}}^{(a,b)} \leq 0$ to be completely orthogonal. Such a case may occur when simulating interference links with two devices transmitting at widely spaced carrier frequencies, where the interference power perceived by the respective receivers will subsequently be zero. After application of the initial anti-aliasing filter, the transmitted samples are converted to the target sampling rate $f_{\mathrm{Rx,s}}^{(b)}$ by applying a sinc interpolation kernel

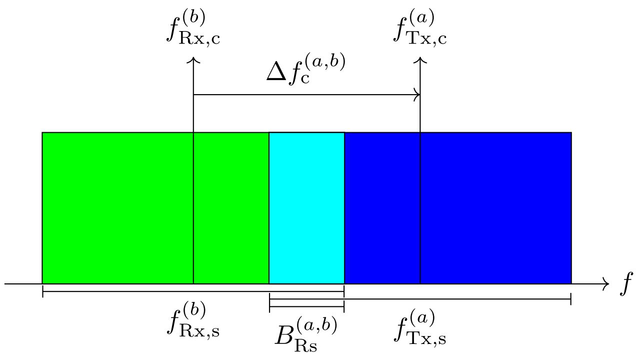

$f_{\mathrm{Rx,c}}^{(b)}$
$f_{\mathrm{Tx,c}}^{(a)}$
$\Delta f_{\mathrm{c}}^{(a,b)}$
f
$f_{\mathrm{Rx,s}}^{(b)}$
$B_{\mathrm{Rs}}^{(a,b)}$
$f_{\mathrm{Tx,s}}^{(a)}$

FIGURE 3. Signal resampling filter.

$$w_{\mathrm{s}}^{(a,b,n,\tilde{n})} = \mathrm{sinc}\left((\tilde{n} - 1)\frac{f_{\mathrm{Tx,s}}^{(a)}}{f_{\mathrm{Rx,s}}^{(b)}} - n + 1 \right) \tag{12}$$

to recalculate time-domain samples according to the target sampling rate sample instances. The whole resampling process

$$\mathbf{W}_{\mathrm{s}}^{(a,b)} = \begin{bmatrix} w_{\mathrm{s}}^{(a,b,1,1)} & \ldots & w_{\mathrm{s}}^{(a,b,1,N_{\mathrm{Rx}}^{(b)})} \\ \vdots & \ddots & \vdots \\ w_{\mathrm{s}}^{(a,b,N_{\mathrm{Tx}}^{(a)},1)} & \ldots & w_{\mathrm{s}}^{(a,b,N_{\mathrm{Tx}}^{(a)},N_{\mathrm{Rx}}^{(b)})} \end{bmatrix} \tag{13}$$

can be described a linear operation. In a final processing step, the resampled signal samples are modulated to the perceived target carrier frequency by multiplying with a sinusoid

$$\mathbf{W}_{\mathrm{c}}^{(a,b)} = \mathrm{Diag}\{1, \mathrm{e}^{2\mathrm{j}\pi \frac{\Delta f_{\mathrm{c}}^{(a,b)}}{f_{\mathrm{Rx,s}}^{(b)}}}, \ldots, \mathrm{e}^{2\mathrm{j}\pi (N_{\mathrm{Rx}}^{(b)}-1) \frac{\Delta f_{\mathrm{c}}^{(a,b)}}{f_{\mathrm{Rx,s}}^{(b)}}}\} \tag{14}$$

of a frequency equal to the carrier frequency distance (9). Note that, depending on the relation between the number of output samples $N_{\mathrm{Rx}}^{(b)}$ and number of input samples $N_{\mathrm{Tx}}^{(a)}$, the mixing operation (14) can be performed before or after re-sampling in order to optimize computational costs. Using the resampled signal, we can express the samples

$$\mathbf{y}_b^{(n)} = \sum_{a=1}^{D} \sum_{\tau=1}^{n} \mathbf{H}_{a,b}^{(n,\tau)} \tilde{\mathbf{x}}_{a,b}^{(n-\tau+1)} \quad \text{for} \quad n = 1 \ldots N_{\mathrm{Rx}}^{(b)} \tag{15}$$

impinging onto the b-th receiving device as a superposition of base-band signal samples transmitted by all D scenario devices after propagation over their respective linking channel models.

B. DEVICE OPERATION

While the scenario and its respective links specify how electromagnetic waves are distorted during propagation from and to devices, the actual waveforms being transmitted depend on the specific device operator configurations. Operators represent Hermes' abstraction for all digital signal processing operations taking place before samples are converted to the analog domain during transmission, and, inversely, all signal processing operations taking place after signals are converted to digital domain during reception. They may implement different types of digital communication modems, radar signal processing chains or any other entity capable of generating or receiving base-band representations of electromagnetic waves from and to RF hardware. Transmitting device operators may submit a set of N_d base-band signal samples $\mathbf{S}_{\mathrm{Tx}}^{(o,d)} \in \mathbb{C}^{M_{\mathrm{Tx}}^{(d)} \times N_d}$ to the device's digital-to-analog converter (DAC) to be transmitted. The actual signal being transmitted over the device's hardware chain is a superposition

$$\mathbf{S}_{\mathrm{Tx}}^{(d)} = \sum_{o=1}^{O_{\mathrm{Tx}}^{(d)}} \mathbf{S}_{\mathrm{Tx}}^{(o,d)} \tag{16}$$

of all $O_{\mathrm{Tx}}^{(d)}$ operator base-band signal samples registered at the $d-$th device. Receive operators on the other hand are being fed a copy of their device's ADC buffers, so that the received base-band samples

$$\mathbf{S}_{\mathrm{Rx}}^{(o,d)} = \mathbf{S}_{\mathrm{Rx}}^{(d)} \quad \text{for} \quad o = 1 \ldots O_{\mathrm{Rx}}^{(d)} \tag{17}$$

are identical over all $O_{\mathrm{Rx}}^{(d)}$ receive operators of a single device. The information extracted from the base-band samples depends on its specific receive operator's implementation and parameterization. Hermes currently implements two duplex operators, duplex referring to operators having both transmit and receive functionalities, namely communication and sensing operators. They may be configured to only execute either transmit or receive functionalities in order to minimize simulation workload. The respective signal processing chains will be introduced in dedicated sections V and VI.

IV. HARDWARE MODELING

When running Hermes simulation campaigns, each virtual device placed within the simulation scenario features a customizable pipeline to model hardware effects during waveform transmission and reception. The implemented device model is visualized in Figure 4. Naturally, all models are optional and often provide multiple implementations from which users may chose the most appropriate one for their simulation assumptions. During signal transmission simulation, the samples submitted by transmit operators are sequentially processed by a DAC model introducing quantization noise, an I/Q imbalance model, PA modeling, mutual coupling (MC) and an transmit antenna array topology model (ANT). During reception most of the processing steps will be applied in reverse order, with the received samples after channel propagation being initially processed by the receive antenna array topology model, followed the MC model. Now, signal power leaking from the transmit to receive chains, simulated by the isolation model (ISO) is added. The combined leakage and received signal samples are processed by an LNA model and IQ imbalance, after which quantization noise introduced by the ADC is introduced. Finally, additive hardware noise is added. In the following subsections, each hardware model is described in more detail.

A. ANALOG-DIGITAL CONVERSION

Currently the ADC model considers only the quantization step. A uniform quantizer with either mid-riser or mid-tread behaviour is implemented, configurable to an arbitrary bit resolution. The quantizer range can be fixed or automatically adjusted based on the input signal power or amplitude. The model will only introduce quantization noise, though, as its output is still in floating point format and fixed-point operations are not yet supported.

B. I/Q IMBALANCE

During modulation and demodulation of complex-valued signals, the signal's real and imaginary parts are multiplied with a sinusoid and a $\frac{\pi}{2}$ phase-shifted copy of the sinusoid, respectively. Variations in the phase shift lead to an effect known as IQ Imbalance (I/Q). Hermes' radio-frequency chain model offers the option to configure a phase offset ϵ_θ and an amplitude offset ϵ_{A} so that the imbalance is described by

$$\mathbf{S}_{\mathrm{Tx,IQ}}^{(d)} = \eta_\alpha \mathbf{S}_{\mathrm{Tx,DAC}}^{(d)} + \eta_\beta \mathbf{S}_{\mathrm{Tx,DAC}}^{(d)*} \tag{18}$$

during signal transmission and

$$\mathbf{S}_{\mathrm{Rx,IQ}}^{(d)} = \eta_\alpha \mathbf{S}_{\mathrm{Rx,LNA}}^{(d)} + \eta_\beta \mathbf{S}_{\mathrm{Rx,LNA}}^{(d)*} \tag{19}$$

during signal reception with imbalance coefficients

$$\eta_\alpha = \cos\frac{\epsilon_\theta}{2} + \mathrm{j}\epsilon_{\mathrm{A}} \sin\frac{\epsilon_\theta}{2} \tag{20}$$

$$\eta_\beta = \epsilon_{\mathrm{A}} \cos\frac{\epsilon_\theta}{2} - \mathrm{j}\sin\frac{\epsilon_\theta}{2} \tag{21}$$

statically distorting the signal's amplitude and phase as suggested in [39].

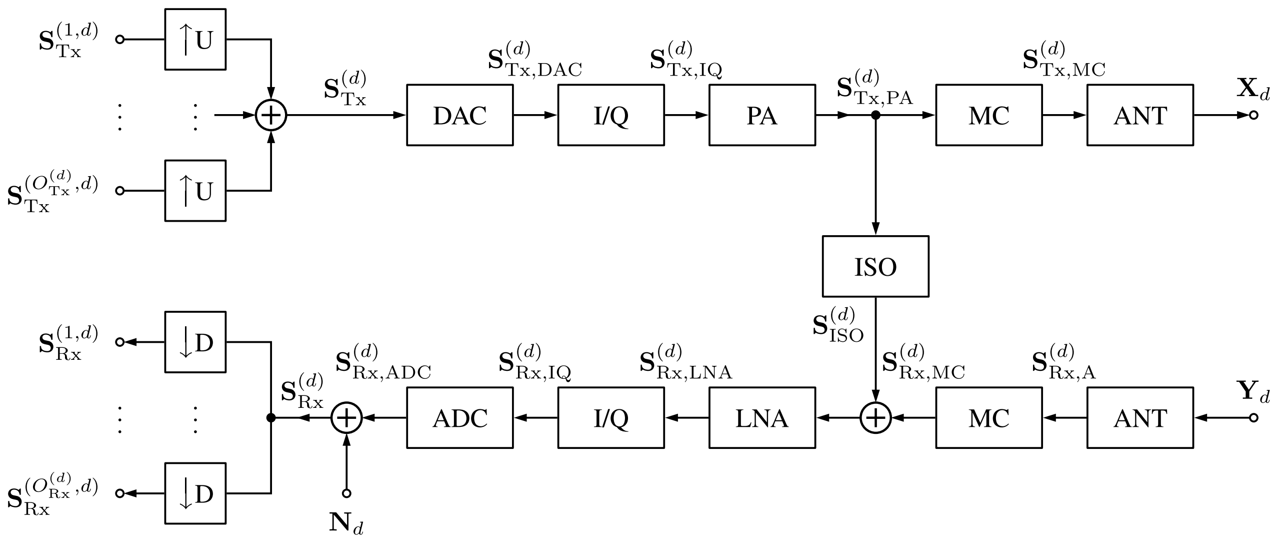
C. AMPLIFICATION

During signal transmission, the PA section amplifies the signal for transmission over the air. Inversely, during signal reception, LNAs amplify signals received by antennas after propagation over the air for further processing. While an ideal amplifier is usually assumed to amplify the signal according to a fixed gain, which can be modeled by a scalar factor, hardware amplifiers typically display a non-linear saturation characteristic, which is only approximately linear around an expected working point. Driving the amplification into saturation leads to a non-linear signal distortion, which may result in out-of-band emissions and impacts data recovery during reception. Hermes currently implements several memoryless amplifier models

$$\mathbf{S}_{\mathrm{Tx,PA}}^{(d)} = \mathcal{F}_{\mathrm{Amp}}\{\mathbf{S}_{\mathrm{Tx,IQ}}^{(d)}\} \tag{22}$$

$$\mathbf{S}_{\mathrm{Rx,LNA}}^{(d)} = \mathcal{F}_{\mathrm{Amp}}\{\mathbf{S}_{\mathrm{Rx,MC}}^{(d)} + \mathbf{S}_{\mathrm{ISO}}^{(d)}\} \tag{23}$$

as functions of the amplifier input, namely ideal distortionless amplification, a nonlinear clipping model, Saleh's



$\mathbf{S}_{\mathrm{Tx}}^{(1,d)}$
$\mathbf{S}_{\mathrm{Tx}}^{(O_{\mathrm{Tx}}^{(d)},d)}$
↑U
$\mathbf{S}_{\mathrm{Tx}}^{(d)}$
DAC
$\mathbf{S}_{\mathrm{Tx,DAC}}^{(d)}$
I/Q
$\mathbf{S}_{\mathrm{Tx,IQ}}^{(d)}$
PA
$\mathbf{S}_{\mathrm{Tx,PA}}^{(d)}$
MC
$\mathbf{S}_{\mathrm{Tx,MC}}^{(d)}$
ANT
$\mathbf{X}_d$
ISO
$\mathbf{S}_{\mathrm{ISO}}^{(d)}$
$\mathbf{S}_{\mathrm{Rx}}^{(1,d)}$
$\mathbf{S}_{\mathrm{Rx}}^{(O_{\mathrm{Rx}}^{(d)},d)}$
↓D
$\mathbf{S}_{\mathrm{Rx}}^{(d)}$
ADC
$\mathbf{S}_{\mathrm{Rx,ADC}}^{(d)}$
$\mathbf{S}_{\mathrm{Rx,IQ}}^{(d)}$
LNA
$\mathbf{S}_{\mathrm{Rx,LNA}}^{(d)}$
$\mathbf{S}_{\mathrm{Rx,MC}}^{(d)}$
$\mathbf{S}_{\mathrm{Rx,A}}^{(d)}$
$\mathbf{Y}_d$
$\mathbf{N}_d$

FIGURE 4. Hermes device model.

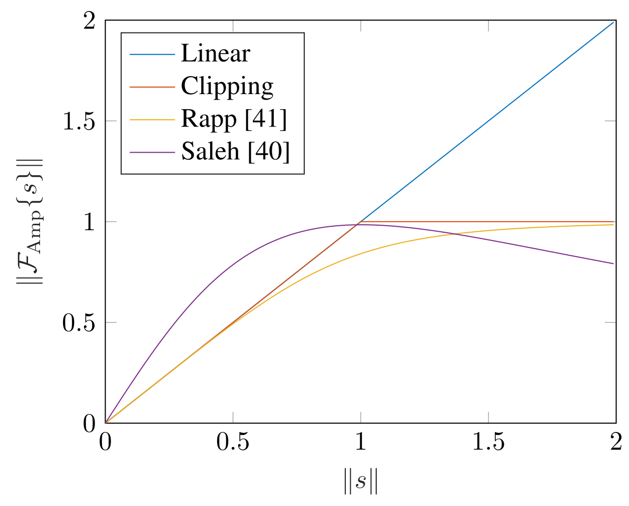


FIGURE 5. Amplifier model characteristics.

model [40], Rapp's model [41] and an option to model custom amplifier AM-AM and AM-PM responses derived from a measurement data set. The LNA's input is a superposition of the received signal after mutual coupling simulation and the crosstalk device self-interference due to imperfect self-isolation $\mathbf{S}_{\mathrm{ISO}}^{(d)}$ as defined in (35). Figure 5 displays a comparison of the implemented power amplifiers' amplitude characteristics around an assumed saturation point of 1, with Rapp's model being parameterized with a smoothness factor $p_{\mathrm{Rapp}} = 2$ and Saleh's model being parameterized by amplitude factors $\alpha_{\mathrm{a}} = 1.9638$, $\beta_{\mathrm{a}} = 0.9945$ and phase factors $\alpha_{\Phi} = 2.5293$, $\beta_{\Phi} = 2.8168$, respectively.

D. ANTENNA MODELING

An individual antenna within Hermes is characterized by its polarization pattern, frequency response characteristics and impedance at vertical and horizontal polarization components

$$\mathbf{F}(\phi,\theta)^{(d,n)} = \left[F_{\mathrm{V}}^{(d,n)}(\phi,\theta),\ F_{\mathrm{H}}^{(d,n)}(\phi,\theta)\right]^{\mathsf{T}} \in \mathbb{R}^2, \quad (24)$$

which specify the polarization of an emitted wave towards horizontal coordinates azimuth ϕ and elevation θ, respectively. Note that the normalized antenna gain characteristics towards a certain angle of departure during transmission, or angle of arrival during reception,

$$G(\phi,\theta) = \|\mathbf{F}(\phi,\theta)\|_2 = \sqrt{F_{\mathrm{V}}(\phi,\theta)^2 + F_{\mathrm{H}}(\phi,\theta)^2} \quad (25)$$

depend on the sum of squared polarization components. Therefore, the polarization model implicitly describes the antenna gain characteristics. Currently, Hermes provides implementations for omnidirectional and patch antennas with circular polarization, as well as half-wavelength dipole antennas with horizontal polarization. Spatial channel models considering a ray-like electromagnetic wave propagation between antennas, such as [42] and [43], may apply the antenna polarization information to generate more realistic channels: For a single path between two antennas a and b, the polarization loss

$$\mathbf{F}^{(a)\mathsf{T}}\Phi\mathbf{F}^{(b)} \quad (26)$$

depends on the path cross-polarization $\Phi \in \mathbb{C}^{2\times 2}$. For line-of-sight paths the cross-polarization can be assumed to be identity, $\Phi = \mathbf{I}$. In this case it is straightforward to see that orthogonally polarized antennas may actually exchange zero power, given unfavorable antenna orientations. A device is considered to feature an array of M_d individual antenna elements, with each antenna element being located at cartesian position

$$\mathbf{p}_{\mathrm{A}}^{(d,m)} \in \mathbb{R}^3 \quad (27)$$

within the device's local coordinate grid, oriented according to an azimuth-offset $\phi_{\mathrm{A}}^{(d,m)}$ and elevation-offset $\theta_{\mathrm{A}}^{(d,m)}$.

Antennas within the array may be considered to be simplex antennas, which are only active during either transmission or reception, or duplex antennas, which transmit and receive waveforms simultaneously. The respective antenna matrix

$$\mathbf{G}_{\mathrm{A}}^{(d)} = \mathrm{Diag}\{G^{(d,1)}, G^{(d,2)}, \dots, G^{(d,M^{(d)})}\} \tag{28}$$

contains the linear gain of each antenna element within the considered array on its diagonal. Therefore, the antenna signals are amplified according to

$$\mathbf{X}_d = \mathbf{G}_{\mathrm{Tx,A}}^{(d)} \mathbf{S}_{\mathrm{Tx,MC}}^{(d)} \tag{29}$$
$$\mathbf{S}_{\mathrm{Rx,A}}^{(d)} = \mathbf{G}_{\mathrm{Rx,A}}^{(d)} \mathbf{Y}_d \tag{30}$$

during transmission and reception over the the configured subarrays, $\mathbf{G}_{\mathrm{Tx,A}}^{(d)}$ being the gains of the transmit subarray and $\mathbf{S}_{\mathrm{Rx,A}}^{(d)}$ being the gains of the receive subarray, respectively.

E. MUTUAL COUPLING AND ISOLATION

Within MIMO antenna arrays, transmitting electromagnetic waves over an individual antenna results in an excitation of neighbouring antenna circuits within the array. Considering every antenna within the array transmitting, the cross-talk between the individual antennas creates a feedback-loop referred to as mutual coupling [44]. The cross-talk's effects on signals can be modeled as a complex symmetric correlation matrix $\Psi_d \in \mathbb{C}^{M_d \times M_d}$. By default, Hermes assumes ideal antenna arrays with no cross-correlation, so $\Psi_d = \mathbf{I}$. If non-ideal antenna array cross-correlations are specified, signals are being distorted according to

$$\mathbf{S}_{\mathrm{Tx,MC}}^{(d)} = \Re\left\{\mathbf{Z}_{\mathrm{Tx,A}}^{(d)}\right\}^{-\frac{1}{2}} \Psi_{\mathrm{Tx}}^{(d)\frac{1}{2}} \mathbf{S}_{\mathrm{Tx,PA}}^{(d)} \tag{31}$$

during transmission. A detailed model of the coupling behaviour between a transmitting and a receiving antenna array requires the consideration of antenna impedances and adjacent matching circuits [45], [46]. We may interpret an antenna array as a multiport network with symmetric impedance matrix $\mathbf{Z}_{\mathrm{A}}^{(d)} \in \mathbb{C}^{M_d \times M_d}$, the impedance matrix diagonal elements

$$\mathrm{Diag}\{\mathbf{Z}_{\mathrm{A}}^{(d)}\} = [z_{\mathrm{A}}^{(d,1,1)}, \dots, z_{\mathrm{A}}^{(d,M_d,M_d)}]^{\mathsf{T}} \tag{32}$$

representing the actual antenna element impedances and the off-diagonal elements representing the mutual impedances between two antenna circuits. Impedances matrices $\mathbf{Z}_{\mathrm{Tx,A}}^{(d)}$ and $\mathbf{Z}_{\mathrm{Rx,A}}^{(d)}$ represent the subarrays of elements transmitting and receiving, respectively. Additionally, Hermes considers a matching network $\mathbf{Z}_{\mathrm{M}}^{(d)} \in \mathbb{C}^{M_d \times M_d}$ in order to mitigate mutual coupling effects, as suggested in [45]. The signal received after channel propagation

$$\mathbf{S}_{\mathrm{Rx,MC}}^{(d)} = \mathbf{Z}_{\mathrm{Rx}}^{(d)} \Psi_{\mathrm{Rx}}^{(d)\frac{1}{2}} \mathbf{S}_{\mathrm{Rx,ANT}}^{(d)} \tag{33}$$

depends on the overall multiport receiver resistance

$$\mathbf{Z}_{\mathrm{Rx}}^{(d)} = 2\Re\left\{z_{\mathrm{A}}^{(d,1,1)}\right\} \Re\left\{\mathbf{Z}_{\mathrm{M}}^{(d)}\right\}^{\frac{1}{2}} \left(\mathbf{Z}_{\mathrm{M}}^{(d)} + \mathbf{Z}_{\mathrm{Rx,A}}^{(d)}\right)^{-1} \tag{34}$$

normalized with respect to the first antenna within the receiving device's antenna array. The implemented mutual coupling model is visualized in Figure 6. An identical hardware model as deployed for mutual coupling considering the transmit and receive antenna arrays for two spatially separated devices is assumed to model the isolation between the transmit and receive chains within a single device. Defining the transmit-receive antenna array subarray correlation matrix $\Psi_{\mathrm{ISO}}^{(d)} \in \mathbb{C}^{M_{\mathrm{Rx}}^{(d)} \times M_{\mathrm{Tx}}^{(d)}}$ modeling the coupling between the transmit and receive chains of a single device, Hermes implements

$$\mathbf{S}_{\mathrm{ISO}}^{(d)} = \mathbf{Z}_{\mathrm{Rx}}^{(d)} \Psi_{\mathrm{ISO}} \Re\left\{\mathbf{Z}_{\mathrm{A}}^{(d)}\right\}^{-\frac{1}{2}} \mathbf{S}_{\mathrm{Tx,PA}}^{(d)} \tag{35}$$

as the isolation model considering both antenna impedances and cross-correlations, describing the leakage of signal power from transmit to receive chains within devices.

F. NOISE

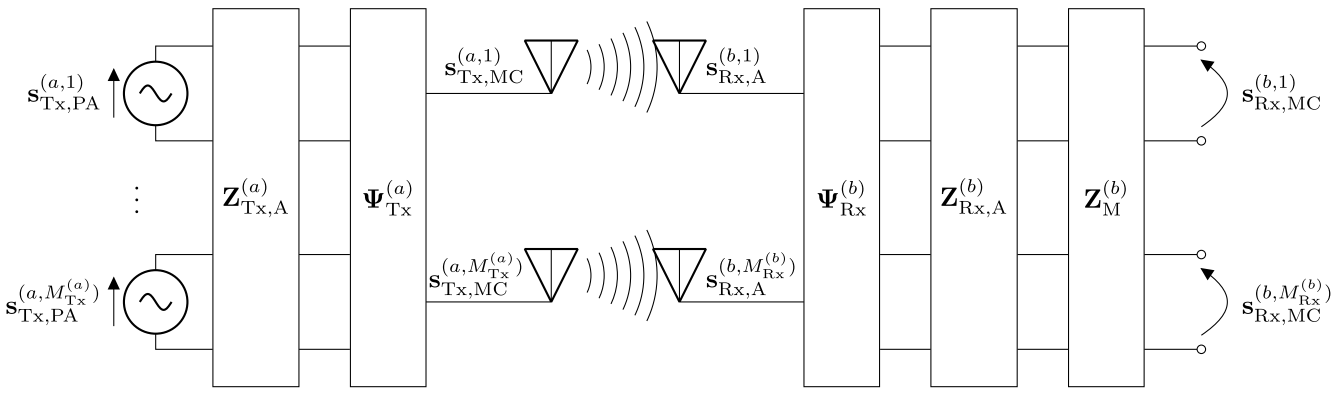
Hardware noise is one of the primary factors limiting the performance of wireless systems in both communication [47] and sensing applications. While any processing step introduces noise to some extent, Hermes assumes receiving front-ends to be the primary source of noise power added to any incoming signal. In Hermes' hardware model, additive noise $\mathbf{N}_d \in \mathbb{C}^{M_d \times N_{\mathrm{Rx}}^{(d)}}$ is introduced at the last stage of device receive modeling according to

$$\mathbf{S}_{\mathrm{Rx}}^{(d)} = \mathbf{S}_{\mathrm{Rx,ADC}}^{(d)} + \mathbf{N}_d. \tag{36}$$

For circular invariant additive white Gaussian noise, the noise samples $\mathbf{N}_d \sim \mathcal{CN}(0, \sigma_{\mathrm{N}}^{(d)}\mathbf{I})$ are independently distributed with variance $\sigma_{\mathrm{N}}^{(d)}$, drawn from the complex normal distribution.

V. COMMUNICATION OPERATION

For the operation of communication devices, Hermes offers the modem module. It implements a highly customizable signal-processing chain for information transmission in form of bits, as well as evaluation tools for the estimation of BERs, BLERs, FERs and DRXs. A flowchart of the processing steps is visualized in Figure 7. Users may configure the source of bits to be transmitted over the operated device as either randomly generated or deterministic, the latter being any file system stream such as images, text or audio files. Bits are being generated on a frame-by-frame basis and encoded with a block-based FEC scheme. Afterwards, the encoded bit stream is split into substreams of equal length. The number of generated substreams depends on the MIMO configuration consisting of a symbol and stream coding stage, and, implicitly, on the number of device antennas the configured modem operates on. For instance, consider a device featuring an array of multiple transmit and receive antennas. The MIMO coding scheme could either be a delay-and-sum beamformer, which would require a single input stream of base-band samples, or a spatial multiplexing scheme, which would require multiple independent streams to be transmitted over the air simultaneously, encoded into the streams fed into the transmit antennas.



$\mathbf{s}_{\mathrm{Tx,PA}}^{(a,1)}$
$\mathbf{s}_{\mathrm{Tx,PA}}^{(a,M_{\mathrm{Tx}}^{(a)})}$
$\mathbf{Z}_{\mathrm{Tx,A}}^{(a)}$
$\mathbf{\Psi}_{\mathrm{Tx}}^{(a)}$
$\mathbf{s}_{\mathrm{Tx,MC}}^{(a,1)}$
$\mathbf{s}_{\mathrm{Tx,MC}}^{(a,M_{\mathrm{Tx}}^{(a)})}$
$\mathbf{s}_{\mathrm{Rx,A}}^{(b,1)}$
$\mathbf{s}_{\mathrm{Rx,A}}^{(b,M_{\mathrm{Rx}}^{(b)})}$
$\mathbf{\Psi}_{\mathrm{Rx}}^{(b)}$
$\mathbf{Z}_{\mathrm{Rx,A}}^{(b)}$
$\mathbf{Z}_{\mathrm{M}}^{(b)}$
$\mathbf{s}_{\mathrm{Rx,MC}}^{(b,1)}$
$\mathbf{s}_{\mathrm{Rx,MC}}^{(b,M_{\mathrm{Rx}}^{(b)})}$

FIGURE 6. Antenna array link impedance model.

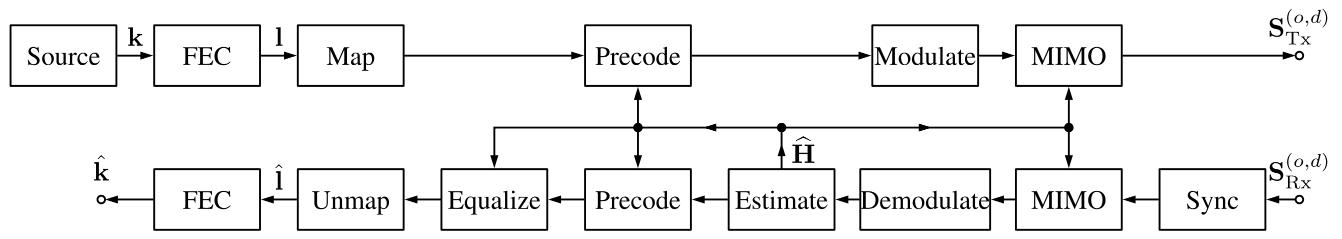


FIGURE 7. Communication signal processing.

The substreams of bits are independently mapped and modulated to base-band sample streams of data frames according to the configured waveform. In between the mapping and modulation stage, codings operating on the symbol level may be configured. After modulation, the resulting frames are encoded by the configured MIMO stream coding configuration and transferred to the operated device for transmission over the air as $\mathbf{S}_{\mathrm{Tx}}^{(o,d)}$.

Data reception requires processing steps for synchronization, channel estimation and channel equalization in addition to the inversion of encoding steps executed during transmission. After device reception, the modem initially divides the receive baseband samples $\mathbf{S}_{\mathrm{Rx}}^{(o,d)}$ into frame sections of equal length through a configured synchronization routine. After synchronization, each resulting frame section is processed in serial independently. Initially, the MIMO stream coding configuration is decoded. Afterwards, the signal is demodulated to communication symbols. A channel estimation routine is deployed, estimating the channel impulse response $\widehat{\mathbf{H}}$ representing the channel state of the received frame during propagation. The estimated impulse response may be accessed by MIMO coding algorithms as well as waveform equalization routines. Channel estimation is followed by a symbol MIMO decoding routine and an equalization stage. Similarly to the transmission signal processing, decoding may result in several, ideally orthogonal, signal substreams which are unmapped in parallel and concatenated, resulting in a bitstream to be decoded by the configured FEC routines.

A. FORWARD ERROR CORRECTION

The FEC configuration is realized as a successive chain of independent coding steps, which enables the easy

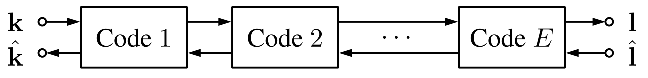


FIGURE 8. Forward error correction chain.

combination of error correcting and error detecting codes to evaluate custom configurations, as well as complementary operations such as bit scrambling and interleaving. Each encoding step e is assumed to achieve a rate $R_e = \frac{K_e}{L_e}$ with an input data block size of K_e bit and an output code block size after encoding of L_e bit. The resulting error correction code chain is characterized by its overall input block size K and its output block size L. The chain is visualized in Figure 8. During the generation of each communication frame, the modem computes the number of bits $\tilde{L}$ required to modulate a single frame and generates them by querying

$$B = \lfloor \frac{K\tilde{L}}{L} \rfloor \tag{37}$$

blocks of data bits at the configured bit source, $\lfloor \cdot \rfloor$ denoting the floor operator to the next integer. The resulting vectors

$$\mathbf{k} = [\mathbf{k}_1^\mathsf{T}, \dots, \mathbf{k}_B^\mathsf{T}]^\mathsf{T}, \quad \hat{\mathbf{k}} = [\hat{\mathbf{k}}_1^\mathsf{T}, \dots, \hat{\mathbf{k}}_B^\mathsf{T}]^\mathsf{T} \in \{0, 1\}^{BK} \tag{38}$$

represent the transmitted and estimated streams of a single data frame consisting of B data blocks $\mathbf{k}_b$ of length K, respectively. The blocks are encoded independently over the FEC chain, with the resulting code blocks being concatenated to a communication frame payload $\mathbf{l} \in \{0, 1\}^L$. In case of block size mismatches between adjacent coding steps, the chain may be configured to run coding steps in parallel in order to increase block sizes, padding overheads resulting from non-integer block size relations with random bits. Decoding

TABLE 3. Hermes communication waveforms.

Waveform	Sync	Estimation	Equalizer
OFDM	Sch-Co, Corr	LS	ZF, MMSE
FSK	Corr	-	-
SC	Corr	LS	ZF, MMSE

of a received payload $\hat{\mathbf{l}} \in \{0, 1\}^L$ is conducted in the reverse order with respect to encoding, resulting in a data estimation $\hat{\mathbf{k}}$. During the decoding, random bits added to padded blocks will be removed. Mismatches between estimated and transmitted bits define the communication performance indicators computed by Hermes, which may be expressed as

$$\mathrm{BER} = \mathbb{E}\{\|\mathbf{k} - \hat{\mathbf{k}}\|_2^2\} \tag{39}$$

$$\mathrm{BLER} = \mathbb{E}\{\|\mathbf{k}_b - \hat{\mathbf{k}}_b\|_2^2 > 0\} \tag{40}$$

$$\mathrm{FER} = \mathbb{E}\{\|\mathbf{k} - \hat{\mathbf{k}}\|_2^2 > 0\} \tag{41}$$

$$\mathrm{DRX} = \mathbb{E}\{\|\mathbf{k} - \hat{\mathbf{k}}\|_2^2 > 0\}\frac{\mathrm{bit}}{T_\mathrm{F}} \tag{42}$$

for BER, BLER FER, and DRX, respectively, T_F being the time interval of frame transmission in seconds. Naturally, for configurations not considering any FEC $\mathbf{k} = \mathbf{l}$ and $\hat{\mathbf{k}} = \hat{\mathbf{l}}$ is assumed, so that the error correction stage is bypassed within the communication signal processing chain.

B. WAVEFORMS

Communication waveforms are Hermes' abstraction to generate and process base-band complex-valued communication signal sample frames. Each modem is required to specify its waveform, with the waveform providing routines for mapping and unmapping of bits to data symbols and the following modulation and demodulation algorithms. Additionally, each waveform implementation provides its own choice of equalization, channel estimation and synchronization routines to be configured by the user. As summarised in Table 3, Hermes currently provides implementations for OFDM, chirp FSK and SC wavefoms, with reference-symbol based Least-Squares (LS) channel estimation and Zero-Forcing (ZF) and minimum mean square error (MMSE) equalization routines.

For SC modulations, raised cosine, root raised cosine, rectangular and FMCW filters are available. Time-domain synchronization of communication frames is implemented by a correlation-based detection [48] (Corr) of a customizable frame preamble. OFDM waveforms may additionally configure Schmidl-Cox (Sch-Co) [49] pilot sections.

C. MIMO- AND PRE-CODING

Hermes offers the option to configure MIMO codings and precodings in two dedicated stages of its communication signal processing chain: On the symbol level in between the mapping of bits to data symbols and their modulation to base-band signal representations and on the antenna stream level after modulation and before transmission over the RF chain. To highlight this distinction, consider the transmitter side of two 2×1 MIMO communication configurations, one applying Alamouti STBC, the other ZF precoding. Both configurations generate two streams to be transmitted over their respective antenna pairs. But while Alamouti precoding generates two sets of modulation symbols to be transmitted independently over each antenna, ZF precoding can be applied to the already modulated signal stream, even for incoherent modulation schemes such as FSK. Therefore, the optimal implementation location for MIMO algorithms differs within the signal processing chain. Considering the general case of communication modems operating on a MIMO device d featuring multiple transmit and receive antennas $M_\mathrm{Tx}^{(d)}$ and $M_\mathrm{Rx}^{(d)}$, the MIMO codings specify how individual base-band signal streams $\mathbf{S}_\mathrm{Tx}^{(o,d)}$ feeding into each antenna element during transmission are being generated, and, inversely, how the base-band signal streams $\mathbf{S}_\mathrm{Rx}^{(o,d)}$ emerging from antenna RF chains during reception are being processed. Therefore, Hermes requires each modem operating a device featuring more than one antenna to specify a MIMO scheme configuring how the individual antenna streams are generated. Depending on the configured MIMO scheme, multiple parallel streams of symbols might be generated during signal transmission, so that the waveform modulation will be conducted on multiple symbol streams in parallel, resulting in an equal amount of base-band signal streams, each representing a communication frame in time domain, encoding individual data symbols. During reception, symbol streams of demodulated base-band signals after equalization are fed to the decoding block, and are processed and combined towards a singular stream. The resulting stream is the input of the waveform's unmapping routine. Table 4 displays an overview of the codings operating on the symbol level currently implemented in Hermes, comparing the number of required and generated symbol streams during transmission and reception, respectively.

TABLE 4. Symbol codings.

Scheme	Tx Streams		Rx Streams	
	In	Out	In	Out
DFT [50]	1	1	1	1
Alamouti [51]	1	2	1	1
MRC [33]	1	1	$M_\mathrm{Rx}^{(d)}$	1
SCEC [34]	1	1	$M_\mathrm{Rx}^{(d)}$	1

Stream coding is implemented identically to symbol coding, but operates on MIMO streams representing base-band signal samples instead of modulation symbols. During transmission, the base-band signal sample streams resulting from the waveform's modulation routine are processed by the configured stream coding's encoding routines, which again may result in an increase or a reduction of the number of streams. The resulting streams represent the communication modem's output to the device. During reception, synchronized base-band sample streams are the input of the stream decoding stage. The resulting decoded streams are fed to the waveform's demodulator and independently demodulated in parallel. Table 4 displays an overview of coding schemes

TABLE 5. Stream codings.

Scheme	Tx Streams		Rx Streams	
	In	Out	In	Out
Conventional BF [52]	1	$M_{\text{Tx}}^{(d)}$	$M_{\text{Rx}}^{(d)}$	1
Capon BF [53]	1	$M_{\text{Tx}}^{(d)}$	$M_{\text{Rx}}^{(d)}$	1
ZF [54]	$M_{\text{Tx}}^{(d)}$ -1	$M_{\text{Tx}}^{(d)}$	1	1

operating on the stream level currently implemented in Hermes, comparing the number of required and generated signal streams during transmission and reception, respectively.

D. SYNCHRONIZATION

Synchronization in Hermes refers to the process of detecting preamble sections of communication frames in time domain. It is the first processing step during signal reception. In terms of the channel model introduced in section III, synchronization can be interpreted as an optimization problem

$$\underset{\tau}{\text{maximize}} \ \|\mathbf{H}_{a,b}(t,\tau)\|_{\text{F}} \quad \text{for} \quad T_{\min} \le t \le T_{\max} \tag{43}$$

estimating the primary delay component τ of the channel model $\mathbf{H}_{a,b}(t,\tau)$ within specific interval between $T_{\min}$ and $T_{\max}$. The estimated delay is equivalent to a perceived timing offset between the transmitter's and receiver's respective clocks. Most statistical channel models for link-level simulations ignore the minimum free-space propagation delays of waveforms traveling from one device to another and instead model only the delay spread of multipath components. As a result, the first signal sample after channel propagation contains the first sample of the propagated signal's line of sight component, or, in non line of sight cases, the first sample of the shortest path propagation. Therefore, link-level simulations tend to ignore synchronization and only focus on processing the multipath components. When considering spatial channel models with realistic propagation delays or, more importantly, transmitting waveforms over real hardware, estimating the introduced propagation delays becomes vital for error-free information transmission. The Hermes API supports the integration of custom synchronization algorithms for communication waveforms. Currently, the library provides a correlation-based synchronization approach [48] applicable to any waveform featuring a dedicated pilot section, as well as the Schmidl-Cox algorithm [49] for OFDM.

E. CHANNEL ESTIMATION

Channel estimation is the process of estimating the channel state $\widehat{\mathbf{H}}$ during signal reception by observing the channel's effect on known reference symbols distributed over the communication frame. Within the Hermes communication processing pipeline, channel estimation is directly following the demodulation stage. Several other processing steps rely on the channel estimation, namely symbol and stream coding during both transmission and reception, as well as the final equalization step before unmapping. Hermes implements LS channel estimation for SC and OFDM waveforms. During simulation runtime the ideal channel state information used to propagate the transmitted and received signals is available, so that users may configure an ideal channel state estimator leading to $\widehat{\mathbf{H}} = \mathbf{H}$. When using Hermes in hardware loop mode operating SDRs or when configuring non-ideal channel state estimators during simulations, transmit processing blocks only have access to the channel state estimated from the most recent reception.

VI. SENSING OPERATION

For the operation of devices as sensors, Hermes provides the radar operator class. It implements a customizable signal processing chain for sensing operations based on electromagnetic waves. During transmission, a sensing waveform is being generated and, in case of a device featuring multiple transmit antennas, precoded according to a configured transmit beamforming algorithm. During reception, in case of a device featuring multiple receive antennas, the signal is steered towards a predefined set of angles of interest by a configured receive beamforming algorithm. For each steered signal line, resulting from the beamformer focusing towards specific angles of arrival, an estimation of the power distribution with respect to target range and velocity is performed. The results are combined to a four-dimensional radar image cube, where the first two dimensions denote discrete azimuth and zenith angles of arrival in a spherical coordinate system, and the last two dimensions denote discrete ranges and velocities, respectively. The radar cube typically displays a constant noise floor over all discrete estimation bins, with clusters of high-power bins indicating target candidates, which may be both real targets or ghosts resulting from higher order multipath propagations or beamforming aliasing. Therefore, a detection routine is applied to the image cube in order to extract an, ideally sparse, point cloud representing the cartesian positions of target detections with respect to the device's location and the detection's estimated velocity. The full processing chain is visualized in Figure 9. Compared to communication operators, the processing chain for sensing operators is rather simplistic, since it lacks the sophisticated steps for symbol coding and error correction. On a more formal level, the resulting point cloud can be considered the output of sensing operators, as opposed to transmitted and received bits of communication operators. From the point cloud, Hermes can be configured to estimate the ROC [55] as a general sensing performance indicator. Hermes currently provides implementations of FMCW waveforms as well as an ISAC module extending modems by sensing capabilities by estimating delays between transmitted and received communication waveforms in duplex devices.

VII. SIMULATION

This section contains a selection of simulation results generated by Hermes, highlighting the core functionalities of the simulator.

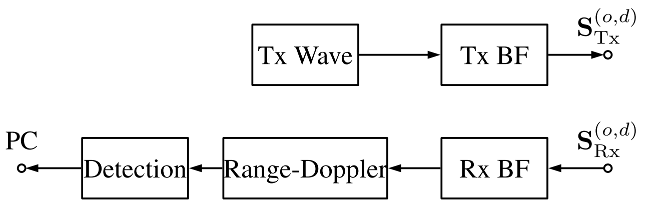


FIGURE 9. Radar signal processing.

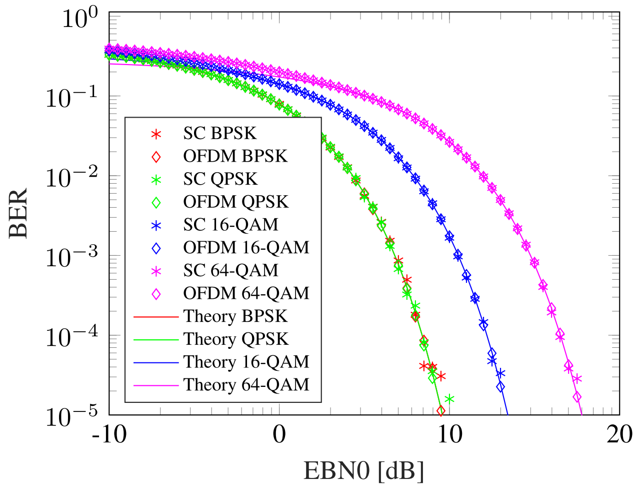


FIGURE 10. Validation SC and OFDM over AWGN channel.

In order to validate Hermes' implementations of SC, OFDM and FSK waveforms as well as AWGN and Rayleigh channels as a special case of TDL channels, Monte Carlo simulations iterating over an SNR range between -10 dB and 20 dB collecting the mean BER over up to 10^6 modulation symbols are performed. For each modulation scheme, modulation orders featuring two, four, 16 and 64 unique communication symbols are considered. The SC waveform is unfiltered, each frame consisting of 100 data symbols transmitted at a rate of 100 MHz, without any additional pilot or reference symbols. The OFDM waveform consists of a single data section featuring 1200 subcarriers, considering neither cyclic prefixes nor reference symbols. The simulated FSK waveform consists of a frame of 100 data chirps, each chirp sweeping over a bandwidth of 200 MHz in a duration of 4 us. The results are compared to theoretic derivations of the expected BERs. Figure 10 displays the results for SC and OFDM communication links over an AWGN channel, Figure 11 displays the performance of the identical waveforms over a Rayleigh channel, and Figure 12 displays the performance of an FSK waveform over an AWGN channel, respectively.

Applying FEC algorithms encoding the transmitted data bits before modulation and decoding them after demodulation can increase the error robustness in noisier domains at the cost of maximal information throughput. Considering the identical $64-$QAM modulated OFDM waveform as in the previous example, we configure a Turbo coding with a data block size

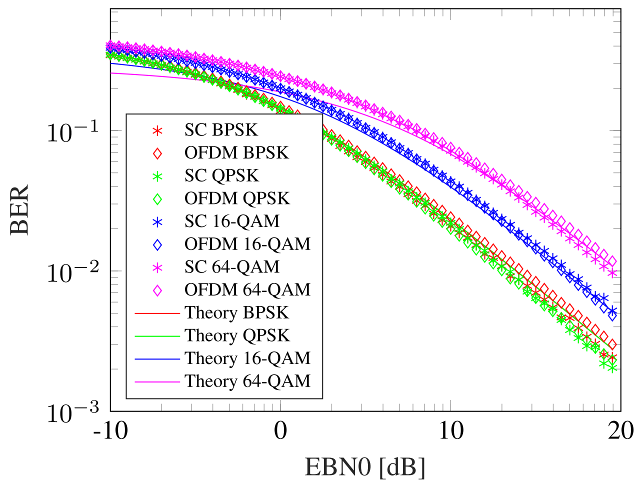


FIGURE 11. Validation SC and OFDM over Rayleigh channel.

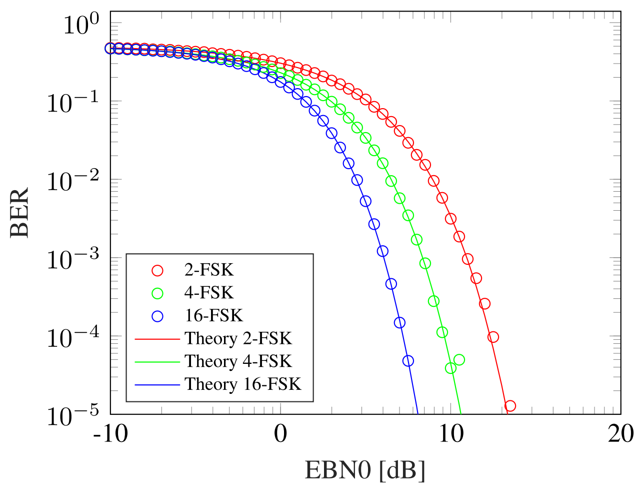


FIGURE 12. Validation FSK over AWGN channel.

of $K_e = 40$ Bit and code block size of $L_e = 132$ Bit, achieving a rate of $R_e \approx 0.303$. Figure 13 displays the result of a simulation estimating the BER, BLER, FER and DRX over an AWGN channel with an SNR between -10 dB and 10 dB. We can conclude that the Turbo coding is able to provide a stable data throughput for SNRs greater than 3 dB, at which point the FER is rapidly approaching zero and the DRX is approaching its maximum.

Depending on the scenario configuration, executing Hermes simulation campaigns may require significant processing power and memory resources. Especially on consumer-grade machines such as notebooks and desktops, high runtimes are the result. In order to minimize these expected runtimes, Hermes implements stopping criteria for Monte Carlo simulations as suggested in [56], enabling users to specify a confidence (Conf) and tolerance (Tol) requirement for each estimated performance indicator. Once the stopping criteria suggest the error of all estimated performance indicators to be smaller than Tol with a probability greater or equal to Conf, the simulation moves to the next parameter combina-

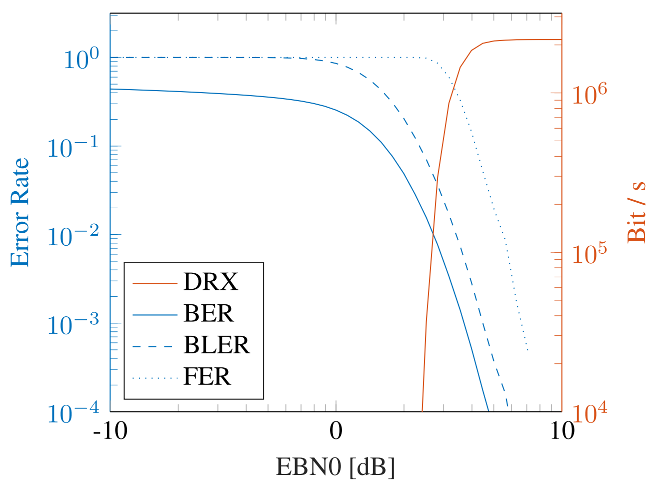


FIGURE 13. Turbo forward error correction.

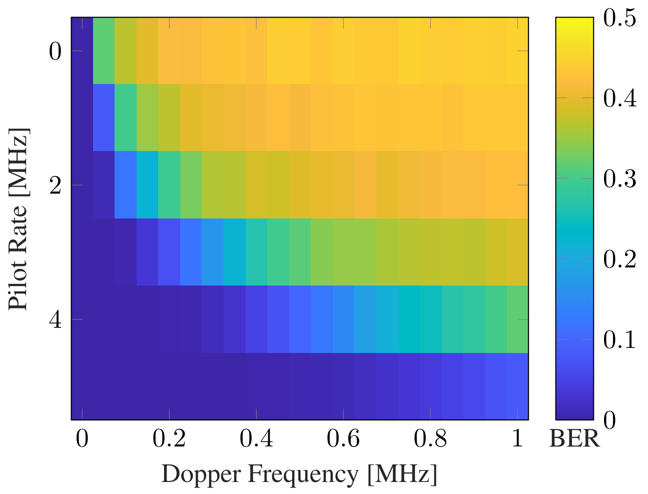


FIGURE 15. Multidimensional parameter sweep.

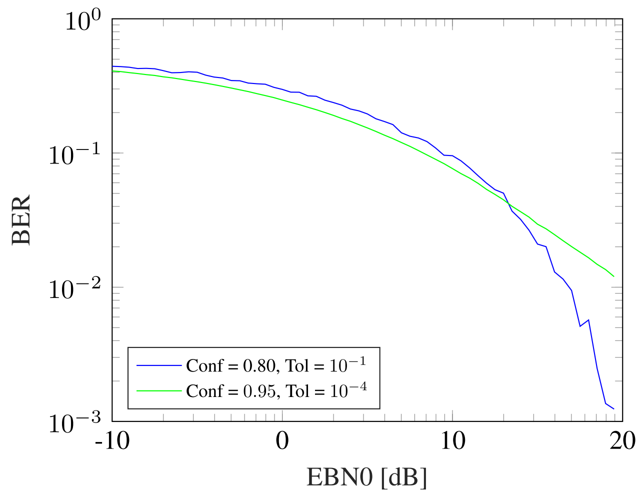


FIGURE 14. Simulator stopping criteria.

tion within the configured parameter sweep to be evaluated. Figure 14 displays the evaluation results of an OFDM communication link transmitting a single section of 1200 64-QAM modulated data subcarriers over a 3GPP TDL-A channel with perfect channel state information at the receiver and subsequent zero-forcing equalization. The results of a BER evaluation with relaxed (blue) and strict (green) stopping criteria is compared. While the evaluation with strict criteria generates a much smoother result curve than its relaxed counterpart, it should be noted that computing the relaxed curve required approximately 23 seconds of simulation runtime, while the strict curve required 1200 seconds. Therefore, if only a rough performance trend estimate is required, relaxing the stopping criteria can save significant computation time.

Although most of the displayed simulation results iterate over the SNR and estimate the resulting BER, it should be noted that Hermes can be configured to sweep over virtually any combination of simulation parameters. To highlight this functionality, a simulation attempting to numerically estimate the ideal pilot symbol rate with respect to the channel doppler frequency shift is performed. A Root-Raised-Cosine filtered SC waveform transmits frames consisting of 10^5 symbols at a rate of 100 MHz over a 3GPP TDL-A channel model, with the simulation estimating the BER over a parameter grid spanned by pilot rate candidates and channel model doppler frequency candidates ranging from zero to 1 MHz. The receiver is assumed to be perfect and noiseless and estimates the channel by a LS routine followed by a ZF equalization. The results are displayed in Figure 15 and clearly highlight the trade-off between pilot symbol rate and channel equalization success for increasing channel doppler frequencies.

Figure 16 displays the performance impact of hardware imperfections on a single-carrier communication link by estimating the BER over an SNR between -10 dB and 20 dB. The simulation considers 64-QAM modulated Root-Raised-Cosine pulses, within frames consisting of 16 pilot symbols and 100 data symbols, mixed to 3.7 GHz carrier frequency and transmitted over a TDL channel model, which is equalized by a ZF equalization routine based on LS channel estimation. An ADC with 8 bit quantization, I/Q imbalance with phase offset $\epsilon_\theta = 1^\circ$ and amplitude imbalance $\epsilon_\mathrm{A} = 0.05$, power amplification following Rapp's model [41] with $p_\mathrm{Rapp} = 1$ and all imperfections combined are compared to the ideal performance without any imperfections. As a result we can conclude that simulations considering ideal RF chains might significantly overestimate the performance of communication systems, especially in high SNR regions where relative impact of hardware effects increases.

Figure 17 displays a range estimate comparison between an FMCW radar sweeping over a bandwidth of 1 GHz, estimating the target range via stretch-processing, and an ISAC radar estimating the target range by correlating a backscattered OFDM Schmidl-Cox pilot symbol over 1000 subcarriers spaced 1 MHz apart. Both signals are transmitted at 70 GHz carrier frequency, have a bandwidth of 1 GHz and a duration of 2 us. A noiseless radar channel models a single target located at 25 m distance with a radar cross section

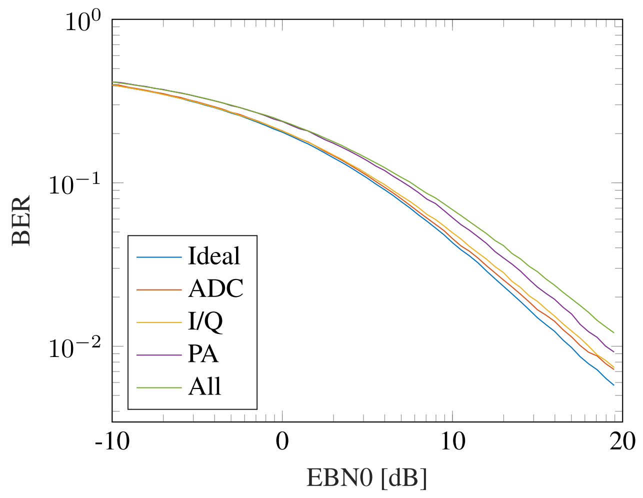


FIGURE 16. Hardware effects modeling.

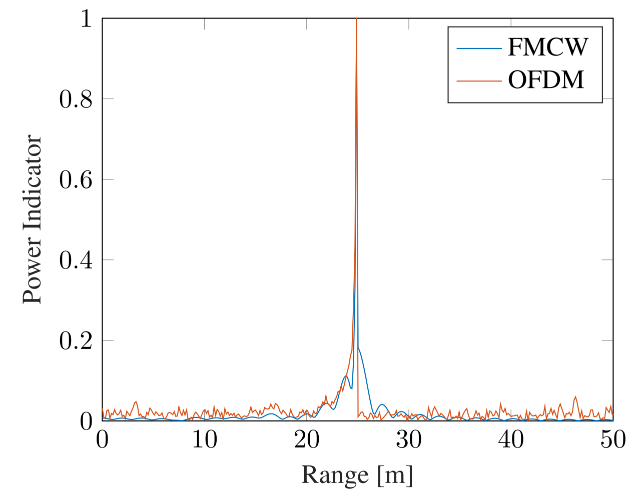


FIGURE 17. FMCW and ISAC OFDM range estimation comparison.

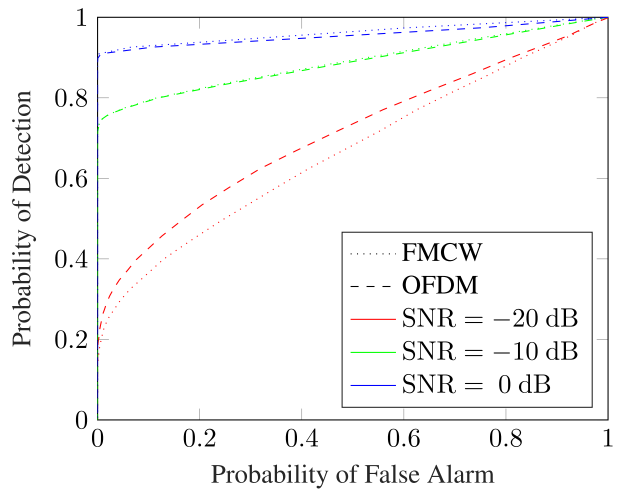


FIGURE 18. FMCW and ISAC OFDM ROC.

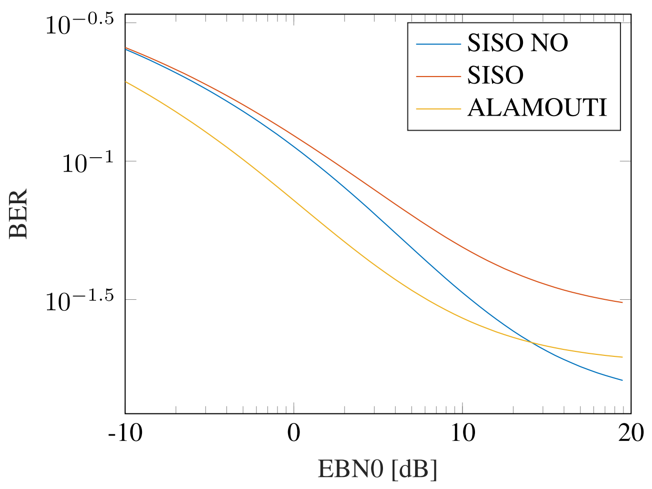


FIGURE 19. Interference system analysis.

(RCS) of 1 m^2. Note that the OFDM radar is similar to the approach presented in [37]. Comparing both range estimates visually, both approaches seem to perform rather similarly. A more formal comparison of radar performance is the ROC, comparing the probability of false alarm when no target is present within the the channel to the probability of detection when a is target present. The evaluation and comparison of ROCs between FMCW and OFDM for transmit signal power to noise power ratios of −20 dB, −10 dB and 0 dB is displayed in Figure 18. Again, both approaches perform rather similarly, with OFDM slightly outperforming FMCW in lower SNR regions.

Figure 19 displays the result of an interference investigation. The simulation considers an industrial communication scenario featuring two devices with with perfect channel state information communicating via OFDM modems over a line-of-sight link in a factory building, being located 50 m apart from one another, modeled by the indoor factory parameterization of the 3GPP CDL channel model [42] assuming a volume of 48000 m^3 and a surface of 80000 m^2. The devices' communication is disturbed by an FMCW radar transmitting within the same frequency band. The BER between a SISO link equalized by zero forcing (SISO ZF) and a 2×1 MIMO link with Alamouti precoding is compared. As a reference, a SISO scenario lacking the interfering FMCW signal (SISO NO) is considered as well. The results show that the MIMO system continuously outperforms the SISO system. Additionally, when comparing the results with and without interference, the expected effect of the interference limiting the system performance, especially in low-noise regions, becomes evident.

VIII. HARDWARE VERIFICATION

The implemented software architecture, more precisely the division between operators generating complex base-band signal samples and devices simulating hardware effects and exchanging signal models over channel links, enables the

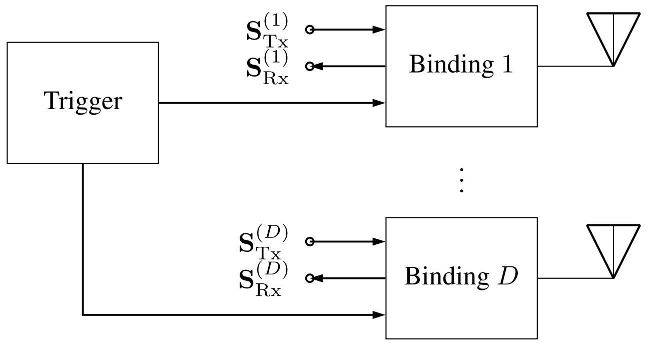


FIGURE 20. Hardware loop architecture.

instantaneous switch from using Hermes as a simulator to verifying algorithms implemented within the Hermes API using real hardware. Instead of simulated devices, operators can be assigned to devices representing bindings to interfaces such as SDRs. The operator configuration remains identical, with the exception of access to perfect channel state information being, naturally, impossible. Therefore, in case of communication operation, configuring synchronization, channel estimation and equalization routines is usually mandatory for hardware verification setups. During transmission, the operator samples are generated and uploaded to the respective device drivers via their binding implementations. Channel propagation modeling is replaced by a trigger operation, triggering synchronous transmission and reception over all devices. The received samples are subsequently downloaded over the device bindings and submitted to the configured receive operators for further processing. The architecture is visualized in Figure 20.

Hermes currently officially provides bindings to USRPs via the USRP hardware driver [57] as well as bindings to sound cards, enabling communication experiments using audio signals with affordable consumer-grade hardware. It should be noted that the hardware verification mode of Hermes is not yet optimized for conducting real-time communication experiments. Depending on the configured waveforms and processing steps, generating base-band waveforms may require significant processing power, introducing delays up to the magnitude of several seconds. Additionally, the up- and subsequent downloading of base-band samples to and from devices when evaluating waveforms of large bandwidths might further decrease the overall processing speed.

In order to demonstrate the proposed workflow, we consider a scenario of four single-antenna devices communicating over two simplex links, with a link being established between each device pair, respectively. The configured waveforms are identical 16-QAM modulated single-carrier root-raised cosine pulses, transmitted at a bandwidth of $B = 61.44$ MHz, forming communication frames featuring 16 preamble symbols and 100 data symbols, respectively. The processing chain during signal reception consists of a correlation-based clock synchronization followed by a LS

```python
# Initialize a new pipeline
pipe = Simulation()
pipe = HardwareLoop(UsrpSystem())

# Configure a single carrier waveform modulated with 16-QAM root-raised cosine
# shaped pulses at a rate of 61.44 MHz
wave = RRCWaveform(symbol_rate=6144e4, num_preamble_symbols=16,
                   num_data_symbols=100, modulation_order=16)

# Configure the waveform to perform a correlation-based synchronization followed
# by a least-squares channel estimation and zero-forcing equalization
wave.synchronization = SCCorrelationSynchronization()
wave.channel_estimation = SCLeastSquaresChannelEstimation()
wave.channel_equalization = SCZeroForcingChannelEqualization()

# Configure device nodes, the base carrier frequency should be 3.7 GHz
cf = 3.7e9
devs: List[Device] = []

# Initialize virtual devices within the simulation
for _ in range(4):

    device = pipe.new_device(carrier_frequency=cf)
    devs.append(device)

# Initialize physical devices at remote IPs to be controlled by Hermes
for d in range(4):

    ip = str(IPv4Address("192.168.0.1") + d)
    device = pipe.new_device(ip=ip, carrier_frequency=cf)
    devs.append(device)

# Set up two interfering simplex links between a pair of devices, respectively
for d in range(2):

    # Initialize a new transmitting modem
    tx_operator = TransmittingModem()
    tx_operator.waveform_generator = deepcopy(wave)

    # Initialize a new receiving modem
    rx_operator = ReceivingModem()
    rx_operator.waveform_generator = deepcopy(wave)

    # Configure the devices by assigning them modem operators
    devs[d].transmitters.add(tx_operator)
    devs[d+2].receivers.add(rx_operator)

    # Configure a bit error rate evaluation for each link
    ber = BitErrorEvaluator(tx_operator, rx_operator)
    pipe.add_evaluator(ber)

# Configure  a parameter sweep over the two links' carrier frequency distance
dist = linspace(wave.bandwidth, 0., 101, endpoint=True) + cf
pipe.new_dimension('carrier_frequency', dist, devs[1], devs[3])

# Execute the pipeline
pipe.num_drops = 1000
pipe.run()
```

FIGURE 21. Multi-node evaluation workflow code example.

channel estimation over the the preamble and a final ZF data symbol equalization. The first link operates at a static carrier frequency of $f_{c,a} = 3.7$ GHz, while the evaluation sweeps over the second link's carrier frequency

$$f_{c,b} = f_{c,a} + \frac{(N-n)B}{N} \quad \text{with} \quad n = 0 \ldots N$$

in order to investigate the interference between the two links with respect to their frequency distance.

Figure 21 shows the described scenario implemented as a Python script calling Hermes' API. For the sake of readability, initial import statements have been omitted. Cyan boxes indicate the only code modifications required to switch the evaluation process from simulation to hardware loop and

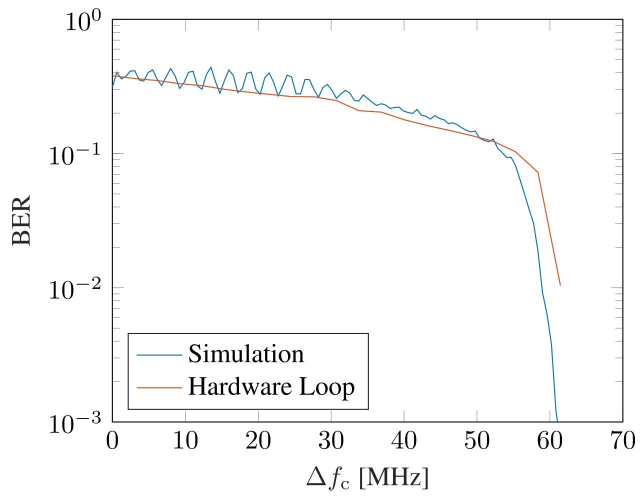


FIGURE 22. Multi-node interference evaluation.

vice-versa: While the simulation pipeline considers virtual devices, the hardware loop pipeline initializes bindings to physical USRP devices within the local network instead, running the server proposed in [57], which require the network address as additional initialization information. Figure 22 displays the average BER over both links with respect to the carrier frequency distance for simulation and hardware loop, respectively. As expected, the error rates significantly decrease with greater frequency distance, approaching zero for a distance greater than the waveform bandwidth. When approaching signal bandwidth carrier frequency distance, the hardware loop performs worse than the initial simulation would suggest, since the simulation considers no noise, assumes ideal antenna frequency characteristics and an ideal channel without time of flight and trigger delays.

IX. CONCLUSION

Physical layer simulations are a core tool for the investigation of novel algorithms in both wireless communications and radar sensing. With the likely adoption of joint communication and sensing in 6G, simulators are required to provide features supporting the evaluation of both services within a unifying software framework. The interference between different services, as well as the role of hardware effects limiting the achievable performance in real-world applications, is currently not being sufficiently addressed by established simulation tools. HermesPy is introduced as a novel open-source software suite, combining both simulation and hardware loop verification abilities and enabling the generation of reproducible results for algorithms implemented within the framework. It provides an easily extensible architecture for the investigation of communication and radar waveforms as well as algorithms within the waveforms' respective signal processing pipelines. Custom algorithms implemented within the API can immediately be verified within hardware testbeds. Future software releases are scheduled to introduce phase noise modeling to the simulation device model, expand the radar operator feature set and optimize the hardware verification performance.

REFERENCES

[1] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6G wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proc. IEEE*, vol. 109, no. 7, pp. 1166–1199, Jul. 2021.

[2] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. E. U. Haq, "Machine learning techniques for 5G and beyond," *IEEE Access*, vol. 9, pp. 23472–23488, 2021.

[3] Y. Liu, X. Liu, X. Mu, T. Hou, J. Xu, M. Di Renzo, and N. Al-Dhahir, "Reconfigurable intelligent surfaces: Principles and opportunities," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1546–1577, 3rd Quart., 2021.

[4] A. Bourdoux, A. N. Barreto, B. van Liempd, C. de Lima, D. Dardari, D. Belot, E.-S. Lohan, G. Seco-Granados, H. Sarieddeen, H. Wymeersch, J. Suutala, J. Saloranta, M. Guillaud, M. Isomursu, M. Valkama, M. R. K. Aziz, R. Berkvens, T. Sanguanpuak, T. Svensson, and Y. Miao, "6G white paper on localization and sensing," 2020, *arXiv:2006.01779*.

[5] T. Wild, V. Braun, and H. Viswanathan, "Joint design of communication and sensing for beyond 5G and 6G systems," *IEEE Access*, vol. 9, pp. 30845–30857, 2021.

[6] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA, USA: CreateSpace, 2009.

[7] *GNU Affero General Public License*. Accessed: Oct. 27, 2022. [Online]. Available: http://www.gnu.org/licenses/agpl-3.0.en.html

[8] *Hermespy Package*. Accessed: Oct. 27, 2022. [Online]. Available: https://pypi.org/project/hermespy/

[9] *Python Package Index—PYPI*. Accessed: Oct. 27, 2022. [Online]. Available: https://pypi.org/

[10] J. Adler, A. N. Barreto, and T. Kronauer. (2022). *HermesPy GitHub Repository*. Accessed: Oct. 27, 2022. [Online]. Available: https://github.com/Barkhausen-Institut/hermespy

[11] P. K. Gkonis, P. T. Trakadas, and D. I. Kaklamani, "A comprehensive study on simulation techniques for 5G networks: State of the art results, analysis, and future challenges," *Electronics*, vol. 9, no. 3, p. 468, Mar. 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/3/468

[12] *MATLAB Communications Toolbox*, The MathWorks Inc., Natick, MA, USA, 2022. Accessed: Oct. 27, 2022. [Online]. Available: https://www.mathworks.com/help/comm/

[13] *MATLAB Communications Toolbox*, The MathWorks Inc., Natick, MA, USA, 2022. Accessed: Oct. 27, 2022. [Online]. Available: https://www.mathworks.com/help/5g/

[14] T. Dominguez-Bolano, J. Rodriguez-Pineiro, J. A. Garcia-Naya, and L. Castedo, "The GTEC 5G link-level simulator," in *Proc. 1st Int. Workshop Link-Syst. Level Simulations (IWSLS)*, Jul. 2016, pp. 1–6.

[15] S. Pratschner, B. Tahir, L. Marijanovic, M. Mussbah, K. Kirev, R. Nissel, S. Schwarz, and M. Rupp, "Versatile mobile communications simulation: The Vienna 5G link level simulator," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, no. 1, p. 226, Dec. 2018.

[16] *MATLAB Parallel Computing Toolbox*, The MathWorks Inc., Natick, MA, USA, 2022. Accessed: Oct. 27, 2022. [Online]. Available: https://de.mathworks.com/help/parallel-computing/

[17] G. F. Riley and T. R. Henderson, "The *ns*-3 network simulator," in *Modeling and Tools for Network Simulation*, K. Wehrle, M. Güneş, and J. Gross, Eds. Berlin, Germany: Springer, 2010, pp. 15–34.

[18] M. Mezzavilla, M. Zhang, M. Polese, R. Ford, S. Dutta, S. Rangan, and M. Zorzi, "End-to-end simulation of 5G mmWave networks," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2237–2263, 3rd Quart., 2018.

[19] N. Patriciello, S. Lagen, B. Bojovic, and L. Giupponi, "An E2E simulator for 5G NR networks," *Simul. Model. Pract. Theory*, vol. 96, Nov. 2019, Art. no. 101933. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X19300589

[20] R. L. Graham, T. S. Woodall, and J. M. Squyres, "Open MPI: A flexible high performance MPI," in *Parallel Processing and Applied Mathematics*, R. Wyrzykowski, J. Dongarra, N. Meyer, and J. Waśniewski, Eds. Berlin, Germany: Springer, 2006, pp. 228–239.

[21] J. M. Eckhardt, C. Herold, B. K. Jung, N. Dreyer, and T. Kürner, "Modular link level simulator for the physical layer of beyond 5G wireless communication systems," *Radio Sci.*, vol. 57, no. 2, Feb. 2022, Art. no. e2021RS007395.

[22] J. Hoydis, S. Cammerer, F. A. Aoudia, A. Vem, N. Binder, G. Marcus, and A. Keller, "Sionna: An open-source library for next-generation physical layer research," 2022, *arXiv:2203.11854*.
[23] M. Abadi. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org/
[24] A. Cassagne, O. Hartmann, M. Léonardon, K. He, C. Leroux, R. Tajan, O. Aumage, D. Barthou, T. Tonnellier, V. Pignoly, B. Le Gal, and C. Jégo, "AFF3CT: A fast forward error correction toolbox!" *SoftwareX*, vol. 10, Oct. 2019, Art. no. 100345. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352711019300457
[25] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, and I. Stoica, "Ray: A distributed framework for emerging AI applications," 2017, *arXiv:1712.05889*.
[26] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.
[27] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.
[28] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.
[29] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 2, May 1993, pp. 1064–1070.
[30] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.
[31] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *J. Soc. Ind. Appl. Math.*, vol. 8, no. 2, pp. 300–304, Jun. 1960, doi: 10.1137/0108018.
[32] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Inf. Control*, vol. 3, no. 1, pp. 68–79, Mar. 1960.
[33] L. Kahn, "Ratio squarer," *Proc. IRE*, vol. 42, no. 11, pp. 1698–1704, Jan. 1954.
[34] D. G. Brennan, "Linear diversity combining techniques," *Proc. IRE*, vol. 47, no. 6, pp. 1075–1102, Jun. 1959.
[35] A. Chorti and M. Brookes, "A spectral model for RF oscillators with power-law phase noise," *IEEE Trans. Circuits Syst., I, Reg. Papers*, vol. 53, no. 9, pp. 1989–1999, Sep. 2006.
[36] G. Fettweis, M. Lohning, D. Petrovic, M. Windisch, P. Zillmann, and W. Rave, "Dirty RF: A new paradigm," in *Proc. IEEE 16th Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, vol. 4, Sep. 2005, pp. 2347–2355.
[37] S. H. Dokhanchi, A. N. Barreto, and G. Fettweis, "A half-duplex joint communications and sensing system using ZP-OFDM," in *Proc. 2nd IEEE Int. Symp. Joint Commun. Sens.*, Mar. 2022, pp. 1–6.
[38] F. Bozorgi, P. Sen, A. N. Barreto, and G. Fettweis, "RF front-end challenges for joint communication and radar sensing," in *Proc. 1st IEEE Int. Online Symp. Joint Commun. Sens.*, Feb. 2021, pp. 1–6.
[39] M. Valkama, M. Renfors, and V. Koivunen, "Advanced methods for I/Q imbalance compensation in communication receivers," *IEEE Trans. Signal Process.*, vol. 49, no. 10, pp. 2335–2344, Oct. 2001.
[40] A. A. M. Saleh, "Frequency-independent and frequency-dependent nonlinear models of TWT amplifiers," *IEEE Trans. Commun.*, vol. COM-29, no. 11, pp. 1715–1720, Nov. 1981.
[41] C. Rapp, "Effects of HPA-nonlinearity on a 4-DPSK/OFDM-signal for a digital sound broadcasting system," in *Proc. 2nd Eur. Conf. Sat. Commun.*, Liege, Belgium, 1991, pp. 179–184. [Online]. Available: https://elib.dlr.de/33776/
[42] *Study on Channel Model for Frequencies From 0.5 to 100 GHz*, 3GPP, document TR 38.901, TSG-RAN, 2020, [Online]. Available: https://www.etsi.org/deliver/etsi_tr/138900_138999/138901/16.01.00_60/tr_138901v160100p.pdf
[43] B. Mondal, T. A. Thomas, E. Visotsky, F. W. Vook, A. Ghosh, Y.-H. Nam, Y. Li, J. Zhang, M. Zhang, Q. Luo, Y. Kakishima, and K. Kitao, "3D channel model in 3GPP," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 16–23, Mar. 2015.
[44] X. Chen, S. Zhang, and Q. Li, "A review of mutual coupling in MIMO systems," *IEEE Access*, vol. 6, pp. 24706–24719, 2018.
[45] Y. Fei, Y. Fan, B. K. Lau, and J. S. Thompson, "Optimal single-port matching impedance for capacity maximization in compact MIMO arrays," *IEEE Trans. Antennas Propag.*, vol. 56, no. 11, pp. 3566–3575, Nov. 2008.
[46] R. Janaswamy, "Effect of element mutual coupling on the capacity of fixed length linear arrays," *IEEE Antennas Wireless Propag. Lett.*, vol. 1, pp. 157–160, 2002.
[47] C. E. Shannon, "Communication in the presence of noise," *Proc. IEEE*, vol. 86, no. 2, pp. 447–457, Feb. 1998.
[48] C. H. Knapp and G. C. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoust., Speech Signal Process.*, vol. ASSP-24, no. 4, pp. 320–327, Aug. 1976.
[49] T. M. Schmidl and D. C. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Trans. Commun.*, vol. 45, no. 12, pp. 1613–1621, Dec. 1997.
[50] T.-A. Truong, M. Arzel, H. Lin, and B. Jahan, "DFT precoded OFDM—An alternative candidate for next generation PONs," *J. Lightw. Technol.*, vol. 32, no. 6, pp. 1228–1238, Mar. 15, 2014.
[51] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE J. Sel. Areas Commun.*, vol. SAC-16, no. 8, pp. 1451–1458, Oct. 1988.
[52] M. S. Bartlett, "Periodogram analysis and continuous spectra," *Biometrika*, vol. 37, nos. 1–2, pp. 1–16, Jun. 1950. [Online]. Available: http://www.jstor.org/stable/2332141
[53] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proc. IEEE*, vol. 57, no. 8, pp. 1408–1418, Aug. 1969.
[54] N. Jindal, "MIMO broadcast channels with finite-rate feedback," *IEEE Trans. Inf. Theory*, vol. 52, no. 11, pp. 5045–5060, Nov. 2006.
[55] J. Echard, "Estimation of radar detection and false alarm probability," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 27, no. 2, pp. 255–260, Mar. 1991.
[56] C. Bayer, H. Hoel, E. von Schwerin, and R. Tempone, "On NonAsymptotic optimal stopping criteria in Monte Carlo simulations," *SIAM J. Sci. Comput.*, vol. 36, no. 2, pp. A869–A885, Jan. 2014.
[57] T. Kronauer and M. Matthé. (2022). *UHD Wrapper*. Accessed: Oct. 27, 2022. [Online]. Available: https://github.com/Barkhausen-Institut/usrp_uhd_wrapper

JAN ADLER received the M.Sc. degree in electrical engineering and information technology from TU Darmstadt, in 2019. From 2019 to 2021, he was a Research Associate with the Fraunhofer Society. In 2021, he joined the Wireless Connectivity Group, Barkhausen Institut, Dresden, Germany, as a Research Associate and a Software Engineer. His research interests include signal processing techniques and channel modeling for multi-node joint communication and sensing networks.

TOBIAS KRONAUER received the M.Sc. degree in mechanical engineering with a focus on information technology from the Karlsruhe Institute of Technology (KIT), in 2019. From 2019 to 2022, he was worked with the Barkhausen Institut, Dresden, Germany, as a Research Associate and a Software Engineer. In 2022, he joined Optonic GmbH, Freiburg, Germany, as a Software Developer. The company specializes in software development for robotics. His research interests include software engineering, robotics, and state estimation.

ANDRÉ NOLL BARRETO (Senior Member, IEEE) received the M.Sc. degree in electrical engineering from Catholic University (PUC-Rio), Rio de Janeiro, Brazil, in 1996, and the Ph.D. degree in electrical engineering from Technische Universität Dresden, Germany, in 2001. He held several positions with academia and industry in Switzerland (IBM Research) and Brazil (Claro, Nokia Technology Institute/INDT, Universidade de Brasília, and Ektrum). He was the Chair of the Centro-Norte Brasil Section of the IEEE, in 2013 and 2014, and the General Co-Chair of the Brazilian Telecommunications Symposium, in 2012. He worked at the Barkhausen Institut, Dresden, Germany, from 2018 to 2022. Since September 2022, he has been a Researcher at Nokia, Paris. His current research interests include joint communications and sensing and physical-layer security for 6G.

• • •