SABRE: Swarm-based Aerial Beamforming Radios: Experimentation and Emulation

Subhramoy Mohanti, Carlos Bocanegra, Sara Garcia Sanchez, Kubra Alemdar, and Kaushik Chowdhury, *Senior Member, IEEE*

Abstract

We propose a novel distributed beamforming framework for UAVs, called SABRE, wherein airborne transmitters synchronize their operations for data communication with target receivers. SABRE chooses the best-suited subset of transmitters that maximizes user-defined QoS, considering relative distances from receivers, traffic characteristics, cumulative SNR desired at the receiver, and individual SNR estimated for each link. This paper makes three main contributions: (i) It shows how to achieve distributed beamforming in challenging, aerial hovering conditions by accurately synchronizing start-times and eliminating relative clock offsets. (ii) It proposes an algorithm with polynomial complexity that groups transmitters and chooses the receiver, maximizing the number of satisfied receivers in each round. (iii) It experimentally validates the concept of aerial beamforming in a testbed composed of four DJI-M100 UAVs in realistic outdoor environments. We follow this up with at-scale emulation involving beamforming with multiple candidate UAV transmitters in Colosseum, the world's largest RF emulator. SABRE keeps the overall network frame error rate below 10% with a probability of 0.95 and manifests a 40% improvement in meeting user QoS thresholds over classical resource allocation methods. From a community viewpoint, the beamforming code, UAV interfacing designs, and the Colosseum container will be released publicly, allowing further independent investigations.

Index Terms

Unamanned aerial vehicle, distributed aerial beamforming, coordinated beamforming, resource allocation, IEEE 802.11be, WiFi 7, 5G, multi-transmitter coordination

I. Introduction

Next generation wireless network protocols like IEEE 802.11be (WiFi 7) are considering breakthrough technologies like multi-Access Point (AP) coordination, along with coordinated beamforming, to enable low-latency/high-throughput applications in dense user deployments while avoiding interference [1, 2, 3]. For different industry segments including automotive,

S. Mohanti, C. Bocanegra, S. Sanchez, K. Alemdar and K. Chowdhury are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, 02115 USA e-mail: (smohanti@coe.neu.edu, bocanegrac@husky.neu,edu, sgarcia@coe.neu.edu, alemdar.k@husky.neu.edu, krc@ece.neu.edu).

transport, logistics and IoT, their service requirements are detailed in standard specifications, like in the 3GPP technical specification for 5G [4]. The latency thresholds for such applications are varied, e.g., 20 ms for sensor data sharing between users and 10 ms for coordinated control of equipment. Furthermore, the RF environment is complex in urban regions, and with high density of users typically seen in such spaces, estimating the net interference caused to any one receiver must consider the propagation environment. Ensuring that each receiver's latency thresholds are satisfied requires timely and correct delivery of packets. One way to ensure this is by increasing the SNR at the receiver, while carefully selecting *who* should be transmitting among all candidate nodes and *which* receiver should be served in the present round [5, 6]. This work serves as a step towards realizing the same goal, while being flexible enough to be deployed on a variety of next generation wireless protocols like WiFi 7, 5G/6G, stand-alone aerial UAV networks, etc.

• **Need for multi-antenna beamforming for UAVs:** We see in [7] that multi-antenna communication improves in-band channel capacity and reduces the network Inter User Interference (IUI). Specifically, transmit beamforming that relies on coherent reception of streams from multiple antennas at the receiver ensures higher SNR than what can be achieved by a single transmitter [8]. If there are $\mathcal{B}$ UAVs, each with a single transmit antenna element, a well-designed beamforming network with perfect phase and time synchronization manifests a linear increase in channel gain, which in turn results in $\mathcal{B}^2$ gain in received power [9]. In this paper, we envisage an architecture that is fully distributed, i.e., each of $\mathcal{B}$ UAV antenna elements are connected to their individual host compute boards, and also lack access to a common reference clock. As shown in Fig. 1, we consider a novel scenario with UAVs mounted with antennas that swarm together, thus forming a flying array of antennas. This results in so called Distributed and Coordinated Aerial Beamforming (DCABF). As long as the same bits are transmitted by the swarm of UAVs (the entire swarm is shown to be partitioned into subgroups $1 \cdots \mathcal{B}$), then DCABF can result in $\mathcal{B}^2$ gain at the target receiver. Already several works have pointed to the feasibility of UAVs serving as aerial base stations to increase range by avoiding no-fly zones [10], enhance coverage [11], capacity [12] and reliability of wireless networks [13]. Through DCABF, we aim to take this capability further in an exciting new direction, while leveraging the ease of programming in most Commercial-Off-The-Shelf (COTS) UAVs, supported capabilities of rapid mobility and flexibility of positioning in 3D space.

• **Challenges:** DCABF brings with it multiple challenges, which we itemize as follows:

- Distributed beamforming requires accurate channel estimation and perfect co-ordination

and synchronization among antenna elements [14, 15]. This problem gets compounded in an aerial setting with spatially distributed UAVs due to their continuous motion, and understandably, wired synchronization methods cannot be used.

- Multi-user communication scenarios that share spectrum lead to IUI, reducing the Packet Delivery Ratio (PDR). Re-transmissions incur not only the risk of violating latency requirements and lower channel utilization, but also require a new round of channel estimation and beam-training. Thus, the transmitter subgroup and the target receiver cannot be arbitrarily chosen; instead as discussed later in the paper, this matching problem is NP Hard.
- We envision a scenario where the UAV flight path is determined by some primary mission objectives like area surveillance, logistics, etc., and DCABF to ground nodes is taken on by the UAVs as secondary tasks, without changing their original flight plan mandated by the primary task requirement. Moreover, it has been proven that, customized flight-trajectories and constant corrections in positioning result in rapid depletion of the UAV batteries [16], reducing their on-air data transmission time, thus lowering the network performance. Thus, DCABF must not be location dependent, assume perfect pre-programmed location assignment for UAVs, or impose static geometric patterns.

In our previous work on Experimental Demonstration of Distributed Beamforming by a Swarm of UAVs (Airbeam) [17] we demonstrated the feasibility of a practical DCABF system under practical SWaP-C constraints, with preliminary experiments and provided extensive systems level implementation on a real test bed. In this paper, we showcase the capability and advantage of DCABF in a real world scenario, wherein the latency thresholds of multiple receivers could be met without the need of implementing exhaustive UAV path planning and location co-ordination, as explained in the challenges as above. To the best of our knowledge, this is the first work that combines a system-level implementation of DCABF while also ensuring the receiver's latency thresholds are met on a best-effort basis.

- **SABRE as a solution:** To address these new challenges, we present SABRE, abbreviated for *Swarm-based Aerial Beamforming Radios with Experimentation and Emulation*. SABRE is a software framework that enables DCABF in a swarm of UAVs, with functional separation into data and control planes as shown in Fig. 1. The *control plane* orchestrates the communication decisions in the swarm through a heuristic for resource allocation, i.e., it jointly creates UAV sub-groups and selects the target receivers that must be served in the current round, so that

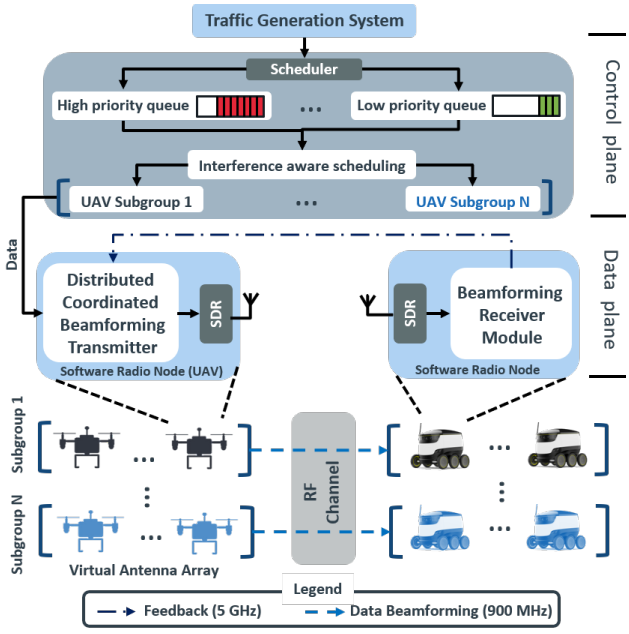


Fig. 1: System architecture for SABRE

the number of receivers whose latency requirements are satisfied is maximized. Thus, this step performs packet/receiver scheduling in polynomial time and takes into account IUI mitigation. The *data plane* is responsible for delivering the scheduled data packets via DCABF. This involves a multi-step process where receiver estimates the channel from each UAV transmitter within the formed sub-group. The channel information is used to compute beamforming weights that are then relayed back to the UAVs. We use Gold sequences for channel estimation that exhibit good auto- and cross-correlation properties. We also ensure synchronization of RF front-ends through a custom-designed software-based wireless time synchronization algorithm. We note that SABRE is designed with the objective of not modifying the UAV flight path while serving multiple users, which preserves UAV flight-time.

- **Paper Contributions** Our main contributions are as follows:
  - Considering the limited availability of transmit antenna elements and scarcity of RF spectrum, we formulate a polynomial time algorithm comprising of (*i*) user scheduling in a time-slotted fashion based on traffic characteristics, and (*ii*) IUI-minimizing UAV transmitter grouping and resource allocation mechanism for multi-receiver scenarios. We showcase through experiments in Sec. VI, that given a default UAV flight pattern, without repeated UAV repositioning to serve multiple users, SABRE performs better than classical methods to meet the latency requirement of different traffic types, and the overall network performance is improved with varying number of users.

- We design the DCABF in SABRE to be robust enough to work with intermittent channel feedback from the receiver. It is capable of precise channel estimation and synchronization among transmitters, without requiring any information exchange among the participating UAVs to set their individual beamforming weights. We implement, demonstrate and validate the feasibility of DCABF on a swarm of COTS DJI-M100 UAVs mounted with Software Defined Radios (SDRs), which required significant systems level software and hardware engineering effort.
- We evaluate the scalability of the resource-allocation algorithm by implementing SABRE in Colosseum [18, 19, 20], the world's largest RF-emulator. Colosseum is used to create a realistic network deployment involving a large number of RF-elements (ground users and UAVs) moving in urban regions. As before, we propose to open-source the Colosseum container for public use, if the paper is accepted.

We discuss the current state-of-the-art in Sec. II. Then, we explain the beamforming theory and the data plane in Sec. III. The control plane UAV grouping and resource allocation is given in Sec. IV. In Sec. V, we describe the steps for implementing SABRE on UAVs, validate our approach for DCABF, and study the impact of frame misalignment and inaccurate beamweights at the receiver. We describe the Colosseum emulation in Sec. VI with evaluation of SABRE in large-scale settings. Finally, we conclude in Sec. VII.

II. Related Works

Recently, a number of works have attempted collaborative beamforming in distributed systems and antenna selection strategies, along with a comprehensive survey on UAV based cellular communications in next generation networks [21]. An extensive survey about emerging technologies aiding in the development of the next generation of wireless networks is presented in [22]. Here the authors give detailed overview of how UAVs can be integrated into the existing and future terrestrial and non-terrestrial networks, including their advantages and existing shortcomings. The authors in [15] present a solution to enhance directivity gain under ideal channel assumptions by performing optimal beampattern formation. However, the effect of interference while scheduling closely grouped receivers is not considered. In addition, there is no calibration and co-ordination between transmitter nodes when beamforming to the receivers. An extended Kalman filter is used in [23] for frequency offset estimation as well as one-bit receiver feedback for distributed phase synchronization between transmitter and receiver, albeit without considering network traffic characteristics or UAV transmitters. The authors in [24] propose an invertible channel matrix as

training symbols for transmitter array nodes and phase offset feedback, but interference mitigation to individual receivers in a multi-user scenario and packet priority-based transmitter antenna assignment is not considered. Finally, none of the aforementioned works focus on mobile or aerial transmitters that have motion-related artifacts. [25] proposes a Multi-Label Convolutional Neural Network (MLCNN)-aided transmit antenna selection scheme for end to end MIMO systems in a single-user scenario. In [26] the authors tackle transmit and receive antenna selection in MIMO systems by using genetic algorithms. Although they do not consider the multi-user case, they include priorities for transmitter and receiver to chose specific antennas in a MIMO system. Regarding multi-user approaches, [27] perform sub-carrier and antenna allocation in an OFDMA system with two users, with different priorities to access the channel, while not considering the effect of interference. A joint antenna selection and user scheduling algorithm is proposed in [28], where scheduling is based on the channel perceived for each user in order to maximize the capacity, without considering different types of traffic and priorities. The authors in [29, 30, 31] study the link performance of UAV communications in dense cellular networks, with PHY layer modifications, but their approach is focused on static network configurations without any performance evaluation considering real-time heterogeneous traffic demands. Finally, in [32] the Hungarian algorithm is used to assign antennas to different users, again not considering different priorities among users, while the validation is limited to a static, indoor testbed environment.

III. Beamforming theory: Background & Analysis

SABRE is designed as a Multiple-Input Single-Output (MISO) framework with a channel delayed feedback to realize transmit beamforming. It enforces a Maximum Transmit Combining (MTC) scheme to obtain the beamforming weights, which is known to deliver the maximum beamforming gain for MISO systems. Thus, SABRE can deliver a power increase by a factor up to $\mathcal{B}$, where $\mathcal{B}$ is the number of available UAVs, each with a single transmit antenna [33, 34]. In this section, we lay out the theory behind transmit beamforming in MISO, obtain the upper-bound beamforming capacity, and analyze the main factors that lower the deliverable power.

A. Notations

We use some example notations in this subsection to describe how we represent the mathematical formulations in the rest of this manuscript. For example, we denote the cardinal of a set, say, α as $|\alpha|$, representing the number of elements in α. The Hadamard product is denoted by $\odot$, and represents the element-wise product between two vectors or matrices of same dimensions.

The Euclidean norm of a set is denoted using $||\cdot||$, and returns a scalar. The operator $\setminus$ is used to subtract a subset from a given set. We use the bold notation when defining structures that contain more than one element (vectors, matrices), and plain notation for scalars. When defining structures, we reveal the type of their elements (Φ, ζ, Ω, etc.) followed by their dimension(s). When indexing row ζ from a matrix, say $\boldsymbol{\Upsilon} \in \Omega^{(\omega \times \Phi)}$, we denote it as $\boldsymbol{\Upsilon}_{\zeta} \in \Omega^{(\zeta \times \Phi)}$. When indexing rows $\mathcal{V} = \{\zeta, k\}$, it results in $\boldsymbol{\Upsilon}_{\mathbf{V}} \in \Omega^{(|\mathbf{V}| \times \Phi)}$. When indexing a matrix that results in a scalar, we drop the bold notation, i.e., $\Upsilon_{i,j} \in \Omega^{(1\times 1)}, i \in \Phi, j \in \Phi$. We use the curly inequality for element-wise comparison between structures, e.g., $\boldsymbol{\alpha} \succeq \boldsymbol{\xi}$ shows all elements in vector $\boldsymbol{\alpha}$ are greater than or equal to the elements in vector $\boldsymbol{\xi}$. When multiplying two scalar values, and also when multiplying a scalar with a vector, we use the notation '$*$' between the variables. For multiplying vectors, we juxtapose those variables together, e.g., $\boldsymbol{\alpha}\, \boldsymbol{\xi}$. The actual notations which are used in the rest of this paper are given in Table I along with their descriptions.

B. Background on MISO

Consider a MISO system composed of $\mathcal{B}$ UAVs, each of which is equipped with 1 antenna. The relationship between the received signal $\mathbf{y}$ and transmitted signal $\mathbf{x}$ at time instant k is: $\mathbf{y}[k] = \mathbf{h}^* x[k] + \eta[k]$, where $\mathbf{h} = [h_1, ..., h_{\mathcal{B}}]$ are the fixed channel gains from the transmit antenna h_l to the receive antenna, $\mathbf{x}$ represents a matrix of dimensions $[\mathcal{B} * 1]$, $x[k]$ is the transmitted signal

TABLE I: Table of Notations

Symbol	Description	Symbol	Description
k	Time instant	l	Packet size in bits
x	Transmit signal	δ	Packet delivery deadline
$\mathbf{y}$	Received signal	Φ^*	A particular sequence of users
$\mathbf{h}$	Channel gain	i	Traffic type
η	Additive White Gaussian Noise	$\mathcal{Z}$	Packet delivery time
s	Transmitted symbols	Λ	SINR
σ	Standard deviation	t	Current slot index
$\mathbf{w}$	Beamforming weight vector	ψ	Remaining time to deadline
E_s	Average energy of signal	$\mathbf{p}$	Vector of queued packets for a user
$\mathbf{H}$	Channel matrix	$\mathcal{F}$	Packet priority list
$\mathbf{E}$	Expectation	ζ	Data rate
γ_{Beam}	Average mutual information	$\boldsymbol{\vartheta}$	Received power offered to user
C_{Beam}	Total capacity in bits/Hz/sec	$\boldsymbol{I}$	User perceived interference plus noise
ε	Total available beamforming power	$\boldsymbol{\Gamma}$	Antenna to user assignment matrix
h	Channel State Information	$\mathbf{G}$	SINR gap
D	Beamforming feedback delay in samples	$\mathcal{N}$	List of packets for neighbor users
ι_e	Channel estimation errors	GS	Gold sequence
ι_d	Errors due to channel variation	t_{corr}	Cross-correlation time
ρ_d	Channel temporal correlation coefficient	t_{LS}	Least square estimation time
J_0	Bessel functions	t_P, λ	Time between received packets
f_m	Maximum doppler shift	t_s	Baseband sampling time
f_s	Sampling rate	$\mathcal{U}$	Set of available users/receivers
$\mathcal{A}$	Set of available UAVs	e	Epoch
Ξ	Forecasted Interference	$\boldsymbol{\Upsilon}$	Rollback matrix
$\mathcal{B}$	Number of UAVs	β	Length of Gold Sequence

at time instant k and $\eta[k]$ represents the additive white Gaussian noise (AWGN) at the receiver with a normal distribution $\varpi(0, \sigma_\eta^2)$, where σ_η is the standard deviation. The receiver estimates the channel continuously and updates the transmitters with the beamforming weight vector $\mathbf{w}$. This allows the data symbols $s[k]$ to be multiplied by the beamforming weights to construct the transmitted signal $x[k] = \sqrt{E_s}\mathbf{w}^H s[k]$, where E_s is the average energy of the transmitted signal $x[k]$, with normalized constellation symbols at any time instant k, i.e., $E(|s[k]|^2) = 1$. In this closed loop system, the weights are selected, so as to maximize the average beamforming mutual information function described in (1), offering a total beamforming capacity in bits/Hz/s given in (2), where ε is the total available beamforming power and H is the Hermitian transpose [17].

$$\gamma_{Beam}(\mathbf{w}, \varepsilon) = \mathbf{E}[log(1 + \varepsilon|\mathbf{h}^H\mathbf{w}|^2)] \tag{1}$$

$$C_{Beam}(\mathbf{w}, \varepsilon) = \mathbf{E}[log(1 + \varepsilon\frac{|\mathbf{h}^H\mathbf{w}|^2}{\sigma_\eta^2})] \tag{2}$$

SABRE is designed for the ISM band with narrowband channels. Under these conditions, the signal is expected to suffer flat-fading, i.e. $W_{comm} \ll W_{cohr}$, where W_{comm} and W_{cohr} imply the communication and coherence bandwidth, respectively. Thus, we model the multipath effect on every symbol by a single complex number $h[k]$ as in (1) and (2).

C. Losses arising from channel imperfections

The channel estimation may contain errors due to shadowing caused by buildings and other environmental objects, resulting in slight errors, such that $\hat{h}[k] = h[k-D]+\Delta h[k]$. We follow the models in [35] for transmit beamforming in MISO systems under delay feedback and estimation errors, and formalize the channel state evolution as in (3).

$$h[k + D] = \rho_d h[k] + \sqrt{1 + |\rho_d|^2}\iota_d[k + D] = \rho_d[\hat{h}[k] + \iota_e[k]] + \sqrt{1 + |\rho_d|^2}\iota_d[k + D] \tag{3}$$

Here, D represents the number of samples that characterize the delay feedback, i.e., from CSI estimation until its application during transmit beamforming. The model also incorporates the estimation errors (ι_e) and the errors due to temporal changes in the channel (ι_d). Lastly, the coefficient $\rho_d = E[\hat{h}[k]\hat{h}[k+D]]$ captures the temporal correlation of the channel, largely defined by the Doppler shift and the Power Angular Spectrum (PAS) of the environment. For instance, the widely used Clarke's model for outdoor communication over a uniform PAS characterizes ρ_d using the Bessel functions [36], i.e., $\rho_d[D] = J_0(2\pi f_m \Delta_t) = J_0(2\pi f_m D/f_s)$, where f_m and f_s are the maximum Doppler shift and sampling rate, respectively [34].

In a static scenario with no estimation or temporal errors, i.e., $\rho_d = 1$ and $\iota_e = \iota_d = 0$, the beamforming capacity is maximized using MTC, by projecting vector $\mathbf{h}$ onto $\mathbf{w}$, i.e., $\mathbf{w} = (\mathbf{h})^{-1}$.

In this way, the ergodic capacity is $E[log(1+\varepsilon)]$, and is solely restricted by the total transmit power. The aggregation of errors incur in a misalignment on the beamforming direction, captured by θ ($0 \leq \theta \leq \pi/2$), between the beamforming vector $\mathbf{w}$ and the channel $\mathbf{h}$. The misalignment can be computed as $\cos(\theta) = Re\left\{\mathbf{h}^H\mathbf{w}\right\}/(||\mathbf{h}||.||\mathbf{w}||)$, resulting in a lower beamforming capacity captured in (4).

$$C_{Beam}(\mathbf{w}, \varepsilon) = \mathbf{E}[log(1 + \varepsilon\frac{|\mathbf{h}^H\mathbf{w}|^2}{\sigma_\eta^2}\cos(\theta))] \tag{4}$$

While small θ does not increase the beamforming capacity gap as we increase $[\mathcal{B}*1]$, the impact increases for large θ in large $[\mathcal{B}*1]$ regime, i.e. for massive deployments. We show this effect during the practical evaluation of this beamforming theory, through UAV experiments in Sec. V.

IV. DCABF IN SABRE

In this section, we formulate and propose a heuristic solution to the problem of UAV-grouping and receiver selection that is solved at the control plane.

A. Scenario description

SABRE operates under the following presumptions:

Multi-user MISO: SABRE forms multi-user MISO links by distributing the set of $\mathcal{B}$ available transmit antennas, i.e., the number of available UAVs, denoted by $\mathcal{A} = \{a_b|1 \leq b \leq \mathcal{B}\}$, among the set of users $\mathcal{U} = \{u_m|1 \leq m \leq M\}$. We envision a centralized controller undertaking the SABRE scheduling task. This controller is co-located with the UAV command and control ground base station through a wireless backhaul link on a separate control channel, which is primarily used for UAV flight control [37]. The scheduling algorithm is not static but dynamic and is iterated continuously over time. At the beginning of each and every iteration, the algorithm looks for packets in the buffer waiting to be delivered to the users on the ground. The Remaining Time To Deadline (RTTD) counter for each packet instantiates the moment it arrives at the buffer. From here on, SABRE follows through the steps mentioned in Sec. IV-C (Algorithm 1), and allocates the best subset of UAVs as beamformers to the users, so as to deliver these waiting packets within their latency threshold and thereby maximizing the overall network Packet Delivery Ratio (PDR). A closed-loop feedback between transmit antennas and receivers enables beamforming, where the latter estimate the wireless channel between each transmitter (b) and user (m) pair as $h_{b,m}$, thus conforming the full-channel matrix $\mathbf{H} \in \Omega^{\mathcal{B}\times M}$. Based on the channel condition $h_{b,m}$, the system adapts to the modulation and coding scheme of $MCS_{b,m}$. SABRE schedules multi-user parallel transmissions within the same frequency band.

Traffic description: Following the 3GPP technical specification [4], we focus on three scenarios with unique traffic characteristics: (*i*) smart-grid operations in smart cities, (*ii*) coordinated control mechanisms, and (*iii*) sensor data sharing. In this way, each user in set $\mathcal{U}$ is assigned one of the three traffic types. These traffic types are characterized by the following unique triplets: the inter-packet arrival time (λ), the packet size in bits (l) and the packet delivery deadline (δ). The deadline δ is the maximum tolerable packet delay, from the time of arrival at the transmitter queue to the instant it is delivered to the receiver.

Slotted transmissions: The limitation on the number of antenna resources (i.e., UAVs) may restrict the number of users that can be served together. For instance, scheduling a large number of users may induce an unresolvable IUI among them, increasing packet re-transmissions. SABRE schedules user transmissions into time slots of pre-defined duration, and enforces a scheduling procedure to select the users being served in any given slot.

B. Problem Definition

We pose the following questions: (Q.1) which users should be allocated in any given slot? and (Q.2) how should antennas and weights be assigned to transmitters to guarantee the traffic latency that is unique to each traffic type in a multi-user in-band communication?

The problem described in (Q.1) relates to a scheduling problem, where the number of jobs that overshoot the permissible deadline (δ) needs to be minimized. Thus, the aim is to find the sequence of users Φ^* that follows (5). Here, sub-index i relates to a traffic type requiring a deadline δ_i. The set $\mathcal{Z}$ contains the packet delivery time for different traffic types, hence $\mathcal{Z} = \{z_i | i \text{ represents different traffic types}\}$. In addition, the weights q_i represent the penalty for not meeting such deadline, and z_i is the time to successfully deliver the packet to the user. The problem (Q.1) is NP-Hard when having unequal weights (q_i) and packet arrivals (δ_i). The constraints can be further relaxed by setting the weights uniformly, leading to NP-Completeness.

$$\Phi^* = arg\, min_{\mathcal{U}} \left\{ \sum \{q_i : z_i > \delta_i\} \right\} \tag{5}$$

The requirement captured by Problem (Q.2) is formalized in (6), and relates the SINR Λ_m that user m requires in a given slot with the attainable SINR from a particular antenna assignments. The vector $\mathbf{h}_m$ captures the MISO channel from the set of antennas that were assigned to user u_m, whereas $\mathbf{w}_m$ represents the beamforming weights. Finding such a sub-group of antennas to users in problem (Q.2) is also NP-complete. Thus, there is no optimal solution in polynomial-time.

$$\frac{|\mathbf{h}_m \mathbf{w}_m|^2}{\sum_{z \in \mathbf{U} \setminus m} |\mathbf{h}_k \mathbf{w}_k|^2 + \sigma_m^2} \geq \Lambda_m \tag{6}$$

The high complexity of the presented scheduling and multi-user beamforming problems require a polynomial time approximate solution, which we describe next.

C. Proposed heuristic solution

In the first step, our high-level scheduler prioritizes users in time, depending on their queued packets and their traffic requirements. In the second step, the system sets up multi-user beamforming by allocating antennas to users and computing their beamforming weights.

1) Step 1: Packet priorities: The central scheduler is connected to the network core and is the ingress point for all IP datagrams for applications being executed at each user $u \in \mathcal{U}$. We denote the current slot index as t. Then, the RTTD is the remaining time before violating that packet's application deadline. We denote the RTTD for packet p and user u_m as $\psi_p^{(m)}$, and express it as $\psi_p^{(m)} = \delta_p^{(m)} - t$. Then, each user has its own RTTD vector $\boldsymbol{\psi}^{(m)} = \{\psi_1^{(m)}, \psi_2^{(m)}, ...\}$. We use an iterative method with priorities based on the RTTD metric. Here $\mathbf{p}^{(m)}$ denotes the vector of queued packets for user u_m. The priority list $\mathcal{F}$ as a list of packets belonging to users whose scheduling urgency is defined by the position they take in the list. Then, we define $\mathcal{F}$ as follows:

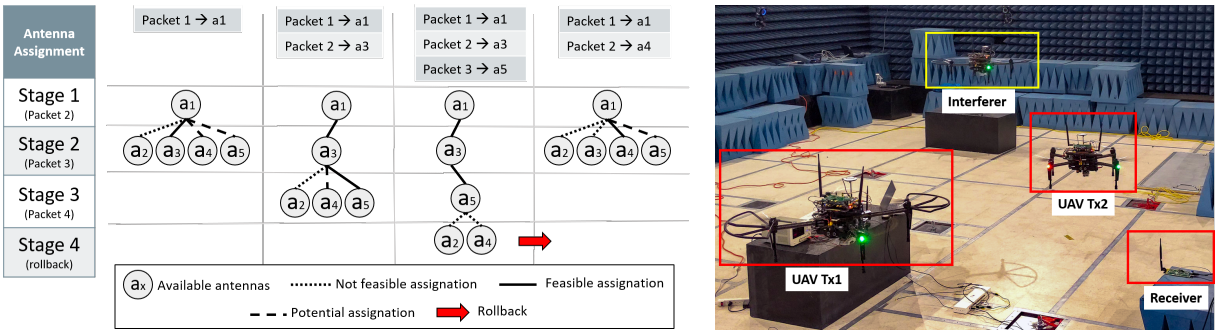
- Combine the individual RTTD's from each user into vector $\boldsymbol{\Psi}$, i.e., $\boldsymbol{\Psi} = [\boldsymbol{\psi}^{(1)}, \boldsymbol{\psi}^{(2)}, ..., \boldsymbol{\psi}^{(M)}]$.
- Sort $\boldsymbol{\Psi}$ in ascending order, starting with the packets having smaller RTTD's, and store the result in $\boldsymbol{\Psi}'$.
- For those packets perceiving same $\boldsymbol{\Psi}'$, i.e., $\boldsymbol{\Psi}'_i = \boldsymbol{\Psi}'_j$, sort them by their packet length in descending order. In this example, F_i and F_j see $i < j$ if $l'_i < l'_j$.

Since the scheduler lacks information about the IUI that can arise due to a given UAV subgroup allocation (problem Q2), it builds the priority list $\mathcal{F}$ and forwards it to the multi-user beamforming algorithm described in Sec. IV-C2. The list $\mathcal{F}$ is updated at the beginning of each slot accounting for the remaining packets in the queue and the new packet arrivals.

2) Step 2: Multi-user beamforming: In this stage, our system allocates antennas from the available set $\mathcal{A}$ to users $\mathcal{U}$ (defined in Sec. IV.A), making use of the sorted packet list ($\mathcal{F}$) and the channel matrix ($\mathbf{H}$). While the former relates to the required SINR per packet, the later allows the assessment of the attainable SINR per packet (see Eq. (6)), in a given slot (t). Let us define the SINR target vector $\boldsymbol{\Lambda}$, where SINR of the packets are indexed over time t as $\boldsymbol{\Lambda}^{(t)}$. We define $\boldsymbol{\Lambda}^{(t)} = \{\Lambda_p^{(t)} | 1 \leq p \leq P\}$, where P is the total number of packets, and $\boldsymbol{\Lambda}^{(t)}$ is represented through $\Omega^{(1\times P)}$ as a row matrix. We denote the required (target) SINR of packet p at slot t as

$\Lambda_p^{(t)}$. To determine $\Lambda_p^{(t)}$, the system follows a three step approach that is widely employed in a variety of wireless standards. First, it assesses the required rate per packet as $\zeta_P^{(t)}$, which is generally obtained by dividing the packet length l_p (in bits) by the duration of total time T. Let us define the required rate vector as $\boldsymbol{\zeta}^{(t)} = \{\zeta_1^{(t)}, ..., \zeta_P^{(t)}\}$. Second, it selects the lowest MCS that achieves the requested rate. Finally, the required SINR $\Lambda_p^{(t)}$ is mandated directly by the selected MCS for packet p.

Having translated the traffic demands $\boldsymbol{\zeta}^{(t)}$ into desired SINR $\boldsymbol{\Lambda}^{(t)}$ in the given slot, the antenna assignment begins. We propose an iterative approach that fits within the Branch-and-Bound (BaB) framework, where candidate antenna assignments to packets (users) are explored systematically in an iterative manner (see Algorithm 1). The algorithm starts allocating antennas to the packet with highest priority order and works its way through the priority list $\mathcal{F}$. The BaB branches out the candidate antenna assignments for each packet, and explores the consequences of its allocations using metrics of received power and perceived interference. We define vectors of offered received power and perceived interference plus noise as $\boldsymbol{\vartheta}$ and $\boldsymbol{I}$, respectively. We index these elements as $\boldsymbol{\vartheta}^{(e)} \in \Omega^{(P\times 1)}$ and $\boldsymbol{I}^{(e)} \in \Omega^{(P\times 1)}$ over epoch e, for all packets in $\mathcal{F}$. The algorithm stops when either all antennas have been assigned, i.e., $|\mathcal{A}| = \emptyset$, or no new assignment can be found for the remaining packets. Note that any new assignment must not conflict with previously assigned packets that have higher priority. Then, it returns a tuple of assignment matrix ($\boldsymbol{\Gamma}$) and packet list ($\mathcal{F}$) for which the antennas have been allocated in the current slot. Let us define $\boldsymbol{\Gamma} \in \Omega^{(P\times\mathcal{B})}$ as the assignment matrix, where $\Gamma_{i,j} \in \{0,1\}, \forall i \in \{1,...,P\}, \forall j \in \{1,...,\mathcal{B}\}$. Hence, it follows that $\mathbf{1}^T\boldsymbol{\Gamma}\mathbf{1} = |\mathcal{A}| = \mathcal{B}$. Naturally, each antenna can only be assigned to one user, and thus $\mathbf{1}^T\boldsymbol{\Gamma} = \mathbf{1}^T$. With each new antenna assignment, the system updates the matrix $\boldsymbol{\Gamma}^{(e)}$, and power vectors $\boldsymbol{\vartheta}^{(e)}$ and $\boldsymbol{I}^{(e)}$. The quality of the current assignment can thus be assessed by the SINR gap $\mathbf{G} \in \Omega^{(P\times 1)}$, defined as the difference between achieved and target SINR, i.e., $\mathbf{G}^{(e)} = \boldsymbol{\Lambda} - \boldsymbol{\vartheta}^{(e)} \ / \boldsymbol{I}^{(e)}$. While it is impossible to assess the final outcome until all antennas are assigned (because of the NP-completeness), the BaB assesses the maximum tolerable interference levels by computing the upper-bounds of the maximum received power. Overall, the algorithm selects the branch that results in the highest received power for the intended packet while the interference to others can be overcome in future epochs. Since the allocation follows the sorted list $\mathcal{F}$, the algorithm only evaluates generated interference for so-called neighboring packets of packet p. We define the neighboring packet list $\mathcal{N}$ of packet p as the packets with higher priority than p, i.e., $\mathcal{N} = \{n | n \in N, \ \forall n < p\}$. Thus, a packet is selected to be allocated antennas iif *(i)*



(a) (b)

Fig. 2: (a) Example of proposed branch-and-bound (BaB)-type antenna assignment; (b) Snapshot of UAV test-bed setup in an RF anechoic chamber [38] when beamforming transmitters are hovering UAVs, forming links with a ground based receiver and affected by a UAV interferer.

Operation	Complexity
Max/min argument of a vector	$\mathcal{O}(n)$
Squared of Euclidean norm	$\mathcal{O}(n)$

TABLE II: Computational complexity per antenna assignment

antennas are available and *(ii)* all packets in its neighboring list $\mathcal{N}$ have fulfilled their SINR requirements, i.e., $\mathbf{G}_{\mathcal{N}} \preceq \mathbf{0}$.

A packet (p) selected to be allocated antennas will accept an antenna assignment as long as its SINR gap is larger than 0, i.e., $g^* > 0$ (Condition C1), and the interference generated to neighboring packets is salvageable (Condition C2). To numerically assess the later, the system computes the tolerable interference for packets in $\mathcal{N}$, i.e., $\mathbf{I}_{max}$, which represents the interference upper-bound that a given user can tolerate if all the remaining antennas in $\mathcal{A}$ were assigned to it. Thus, the requirement is formalized as $\mathbf{I}_{max} \succeq \mathbf{I}$. Failure to meet these two conditions results in the removal of the packet from the priority list $\mathcal{F}$, for its assignment conflicts with others with higher priority in the current slot. Moreover the algorithm also incorporates a procedure to revert assignments that were ascertained as valid but resulted in incompatibility with other packets with higher priorities at later stages. We refer to this procedure as Rollback. Fig. 2a showcases the main stages in the proposed algorithm. In Stage 1, the packet 1 has been assigned to a1 and the packet 2 is selected to be allocated antennas. Here, the candidate antennas are a_2, a_3, a_4, a_5. The algorithm deliberates that a_3 is the one that delivers the highest gain to user owning packet 2 while resulting in salvageable interference to others (Conditions C1 and C2), and hence, it proceeds to assign it. The process is repeated with packet 3, with a_5 being assigned to it. However, a previous assignment that was deemed feasible unveils as infeasible at stage 4, showing no possible assignment to packets with higher priority. The algorithm then proceeds to Rollback to a cleaner state (stage 2), flags a_3 as infeasible, and selects a_4 instead.

3) Complexity Analysis: In this section, we study the worst-case complexity of the proposed analysis. In our case, this maps to an instance where the first packet fulfills its SINR requirement with a single antenna but every remaining packet $(P-1)$ generates neither salvageable interference to the first packet nor fulfills its SINR requirements with the remaining antennas $(\mathcal{A}-1)$. In this way, the algorithm is required to iterate over all the packets and carry out a number of operations. Table II summarizes the steps with highest cost in our proposed solution along with the complexity. Thus, in the worst case at the initial stage, the algorithm selects the highest priority packet $(O(P))$, computes the candidate vector $(O(|\mathcal{A}|))$ and selects the best antenna $(O(|\mathcal{A}|))$. The best antenna will then be assigned to that packet. The algorithm further iterates over all remaining packet, incurring in these same operations plus an additional one to ensure the generated interference to the first packet is salvageable $(O(|\mathcal{A}|-1))$. Therefore, the overall complexity can be summarized as $2O(|\mathcal{A}|)+(P-1)[O(P-1)+3O(|\mathcal{A}|-1)]$, which remains polynomial. Our approach is a relaxed version of the NP-Complete problem that attempts to minimize the number of delayed packets in a co-channel multi-user setup.

V. Implementation of SABRE

We next describe the implementation of SABRE in an aerial testbed. Details on the modules comprising SABRE are provided next, namely the beamforming algorithm, the closed-loop feedback and the accurate synchronization for frame alignment. We validate these in a single-user setting under static, hovering and interfered environment through Bit Error Rate (BER).

A. Preliminary studies towards DCABF

We first characterize the performance in MISO systems operating under channel delay feedback. These are majorly driven by the temporal correlation factor ρ_D and other errors induced by temporal changes in the channel (see Sec. III). For that, we devise an experimental setup with UAVs in both static on-ground and in-air hovering scenarios operating in the 900 MHz ISM band. The snapshot of our experiment setup is given in Fig. 2b, where an anechoic chamber shields our system from external incumbent transmissions within the public ISM band. Each transmitter entity in DCABF comprises of a DJI-M100 UAV [39], to which we attach an NVIDIA Jetson Tx2 [40] host running linux to carry out the DSP, and an Ettus B210 [41] Software Defined Radio (SDR) connected to this Linux host, acts as the RF-front end. The receiver entity comprises 1 B210 SDR, connected to a Linux host. For the preliminary experiment described here, we define 1 receiver entity, equipped with 1 antenna, and 1 transmitter entity, equipped with 2 antennas.

Algorithm 1: BaB based antenna allocation

Inputs: Channel Matrix $\mathbf{H}$, Priority list $\mathcal{F}$, Input SINR objective $\mathbf{\Lambda}$, Available antenna set $\mathcal{A}$
while $\mathcal{A} > 0$, **do**: *(Assign all transmit antennas)*
  Compute packet(s) per user to serve: $\{p^*, u^*\} \leftarrow \{\mathcal{F}\}$, $s.t.$ $\mathbf{G}_p > 0$ *(Pkts sorted by priorities in $\mathcal{F}$)*
  Init. antenna assignment set for p^*: $\mathbf{Y} \leftarrow \emptyset$
  Init. candidate remaining antenna set $\mathcal{A}'' \leftarrow \mathcal{A}'$ and neighbor packet list: $\mathcal{N} = \{n | n \in N \; s.t. \; n < p^*\}$
  Init. SINR gap for p^*: $g^* \leftarrow \mathbf{G}_{p^*} \in \Omega^{(1\times 1)}$ and SINR gap for neighbors of p^*: $\mathbf{G}^* \leftarrow \mathbf{G}_{\mathcal{N}} \in \Omega^{(\hat{\mathcal{N}}\times 1)}$
  Init. Interference gap: $\mathbf{S} \leftarrow \mathbf{0} \in \Omega^{(\hat{\mathcal{N}}\times 1)}$ *(Contains Interference only from the Neighboring set)*
  while $g^* > 0$ **and** $|\mathcal{A}|'' > 0$, **do**: *(Condition C1)*
    Compute candidate power vector: $\mathbf{Q}_{u^*} = \mathbf{H}_{(u^*,\mathcal{A}'')} \odot conj(\mathbf{H}_{(u^*,\mathcal{A}'')}) \in \Omega^{(1\times \hat{\mathcal{A}}'')}$
    Select best antenna for user u^*: $a^* \leftarrow \{\mathbf{Q}_{u^*}\}$ and remove antenna from candidate set: $\mathcal{A}'' \leftarrow \mathcal{A}'' \setminus a^*$
    Update g^*: $g^* \leftarrow \Lambda_{p^*} - (\vartheta^* + ||\mathbf{H}_{(u^*,a^*)}||^2)/I^* \in \Omega^{(1\times 1)}$ ($\vartheta^* \equiv \vartheta_{p^*}$, $I^* \equiv I_{p^*}$)
    for n in $\mathcal{N}$, **do**: *(Update upper-bounds in BaB)*
      Compute Forecasted Interference (Ξ) assigning $a^* \rightarrow u^*$: $\Xi_n \leftarrow I_n + ||\mathbf{H}_{(n,a^*)}||^2 \in \Omega^{(1\times 1)}$
      Compute max. power offered to n: $\varepsilon_{max,n} \leftarrow \varepsilon_n + ||\mathbf{H}_{(n,\mathcal{A}''\setminus a^*)}||^2 \in \Omega^{(1\times 1)}$
      Compute max. tolerable interference (+noise): $I_{max,n} \leftarrow (\; \varepsilon_{max,n} \; / \Lambda_n)$
      Update interference gap: $\mathbf{S}_n \leftarrow I_{max,n} - I_n$ and candidate SINR gap: $\mathbf{G}^*_n \leftarrow \Lambda_n - \; \vartheta_n \; / \; \Xi_n$
    if $\mathbf{S} \succeq 0$, **do**: *(Condition C2)*
      Accept candidate antenna: $\mathbf{Y} \leftarrow a^*$ and candidate SINR gaps: $\mathbf{G} \leftarrow \{g^*, \mathbf{G}^*\}$
      if $\mathbf{G}^* \succ 0$, **do**: *(Fill Rollback matrix)*
        Flag antenna in Rollback matrix: $\mathbf{\Upsilon}_{p^*,a^*} = 1$
    else if $|\mathcal{A}|'' = 0$, **do**: *(p' and $\mathcal{N}$ not compliant & antennas to $<$ priority)*
      Remove packet p' with antennas and lowest priority from current slot: $\mathcal{F} \leftarrow \mathcal{F} \setminus p'$
      Re-activate antennas from packet p': $\mathcal{A}' \leftarrow \mathcal{A}' \cup ind(\mathbf{\Gamma}_{p'})$, $\mathbf{\Gamma}_{p'} \leftarrow \mathbf{0} \in \Omega^{(1\times N)}$
      Rollback to last clean state: $\mathbf{\Gamma}_{\mathcal{N}} \leftarrow \mathbf{\Gamma}_{\mathcal{N}} - \; \mathbf{\Upsilon}_{\mathcal{N}} \in \Omega^{(\hat{\mathcal{N}}\times N)}$
      Reset Rollback matrix: $\mathbf{\Upsilon}_{\mathcal{N}} \leftarrow \mathbf{0} \in \Omega^{(|\mathcal{N}|\times N)}$ and Erase assignments: $\mathbf{Y} \leftarrow \emptyset$
  Update remaining antenna set: $\mathcal{A}' \leftarrow \mathcal{A}' \setminus \mathbf{Y}$ and allocation for user u^*: $\mathbf{\Gamma}_{(p^*,\mathbf{Y})} \leftarrow \mathbf{1} \in \Omega^{(1\times \hat{\mathbf{Y}})}$
  Update power and SINR Gap vectors: $\{\boldsymbol{\vartheta}, \mathbf{I}\} \leftarrow \{\mathbf{\Gamma}, \mathbf{H}\}$, $\mathbf{G} \leftarrow \mathbf{\Lambda} - (\boldsymbol{\vartheta}/\mathbf{I})$
Return Assignment matrix and final packet list: $\{\mathbf{\Gamma}, \mathcal{F}\}$

Although no beamforming is involved at this stage, having 2 antennas helps us generalize our findings with higher assurance. The transmitter sends frames periodically containing solely a training sequence that is known at the receiver. It has good cross-correlation properties to obtain the channel estimates from the transmit antennas *in parallel*. At reception, a simple Least Squares (LS) fit is employed to estimate the channel response, i.e. $\hat{h} = \hat{GS}/GS$, where the estimated response $\hat{h}$ is the division of the received symbols $\hat{GS}$ over the known training sequence GS.

The closed-loop channel feedback naturally incurs in a non-negligible lapse of D samples between channel estimation and beamforming transmission. Here, $D = f_s \Delta_t$, where f_s is the sampling rate and Δ_t is the time lapse between feedbacks. For this experiment, we use a MATLAB-based implementation. To measure the channel variation over time, we compute the

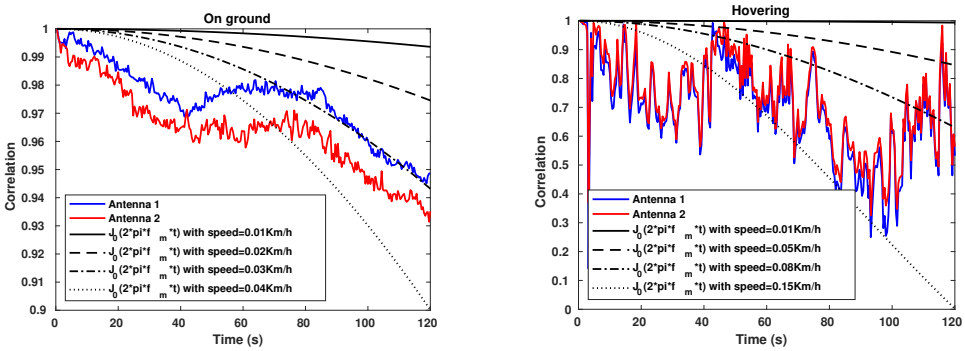


Fig. 3: Channel correlation in a 2x1 MISO w.r.t the origin measurement for static (on the ground, left) and hovering UAV deployment.

normalized correlation coefficient as in shown in (7).

$$\rho_D = \frac{E[h[k]h[k+D]]}{|h[k]|.|h[k+D]|} \tag{7}$$

Fig. 3 reveals the empirical correlation for either antennas in a static and hovering setting. As a reference, we show the Bessel functions that are used to characterize correlations in urban deployments with different mobility patterns (see Sec. III). As expected, a static scenario perceives a highly correlated channel over time and does not require a fast and continuous channel feedback to maintain good channel capacity. On the other hand, the displacement of the UAVs caused by the inevitable hovering motion leads to rapid channel uncorrelation within closely-spaced sampled instances. Consecutive channel measurements can also help us characterize the error that a delay would incur. For instance, estimated channel $h[k]$ will be used after D samples to project over channel $h[k+D]$. Thus, the misalignment is naively formalized as $h[k]-h[k+D]$. These analysis are captured in Fig. 4. Interestingly, the error in both static and hovering scenarios lead to a normal distribution, thus defining (3) in Sec. III. As it follows, the deviation of those errors is substantially more pronounced in the hovering case, with a factor of x10 increase with respect to the static scenario.

Our initial experiments confirm the two main factors that may cause misalignment in DCABF. Namely, (*i*) the variation of the channel over time, i.e., the algorithm processing time needs to match the coherence time of the channel, and (*ii*) an imprecise transmission start timing, i.e., the system must ensure coherent reception for correct packet demodulation.

B. Details of SABRE

In this section, we present a systems level implementation of SABRE which comprises of, *(i)* a novel frame structure for fast algorithm and processing in DCABF (Sec. V-B1), and *(ii)*

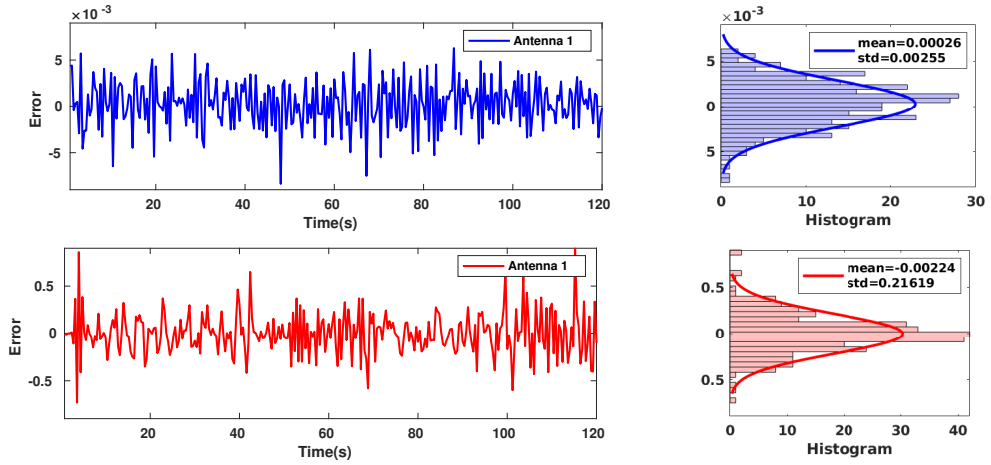
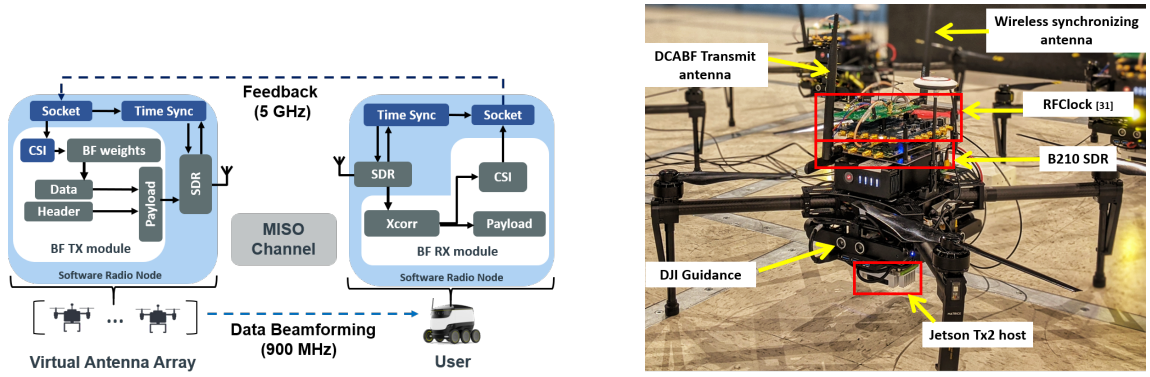


Fig. 4: Estimate errors induced in a delayed channel feedback for static (blue) and hovering (red)

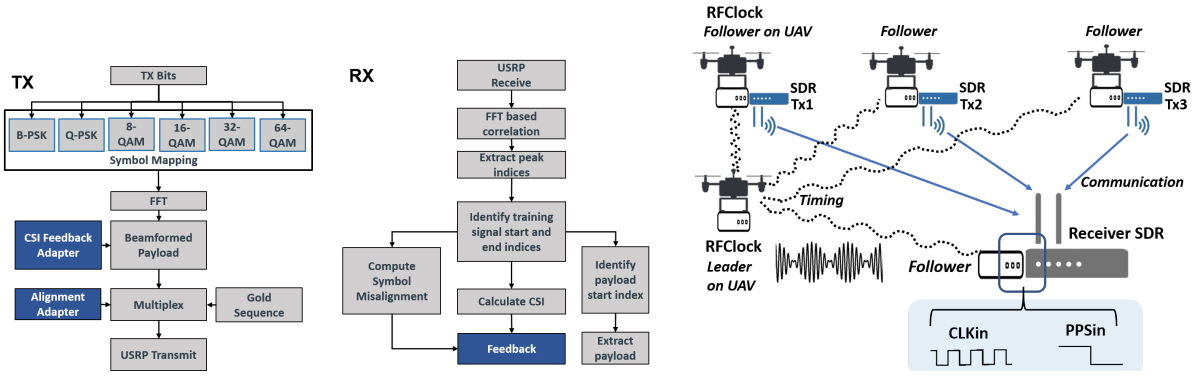


(a) (b)

Fig. 5: Systems implementation of Data Layer of SABRE on UAVs and ground receivers (a) and (b) SABRE implementation showing hardware components on a DJI-M100 UAV

a software-based synchronization method among the distributed UAVs for aligning the transmit streams at the receiver (Sec. V-B2). We will show that the design of SABRE complies with the guidelines presented in Sec. V-A by evaluating the perceived BER at reception in a practical DCABF environment (Sec. V-C). The main blocks that define SABRE's operation are defined in Fig. 5a, where we highlight the components of (*i*) in grey and the components of (*ii*) in blue. These blocks are implemented on the hardware as shown in Fig. 5b, where a Jetson TX2 defines the DSP application in a Linux-based system while a B210 SDR acts as the radio front-end operating in the ISM band. These are attached to a DJI-M100 UAV and are complemented with a wireless clock synchronizer module and antenna to ensure time synchronization in our distributed setting.

1) Frame structure and Beamforming: The core DSP functions are handled by our custom-built GNURadio-based application, thus allowing for a faster processing and overcoming the main limitations that we laid out in Sec. V-A. The frame structure in SABRE comprises a training



(a) (b) (c)

Fig. 6: (a) and (b) Distributed beamforming TX and RX chains, (c) Network architecture showing the distributed synchronization on SDRs installed on UAVs, enabled by RFClock.

sequence, acting as a preamble, followed by a payload of data that needs to be coherently received by the user. We use Gold sequences [42] as our preambles to aid the beamforming process, since they are known for having good auto- and cross-correlation properties. The auto-correlation enables accurate channel estimation under low SNR conditions, while the cross-correlation enables parallel detection from initially unsynchronized transmitted streams. In this way, each transmitter antenna in SABRE is assigned a unique Gold sequence extracted from a pool of orthogonal sequences. Advocating for fast processing to overcome the issues presented in Sec. V-A, SABRE uses the simple LS channel estimation algorithm. The process is divided into two stages. Stage 1 performs the correlation of the received preamble against the expected Gold sequences to detect which specific sequence is received, and thus identifying the original transmitter. This operation consists of 2β multiplications and 2β additions per transmitter for each expected sequence of length β. Let the computation time of this operation be t_{corr}. In stage 2, we take the detected preambles and perform the LS estimation against the associated stored sequences, which requires $6\beta + 2$ multiplications and $6\beta + 2$ additions for each expected sequence. Let the computation time of this operation be t_{LS}. For transmit beamforming relying on a delayed channel feedback we must ensure that the baseband sampling period (t_s) is less than the time between received packets (t_P). We attain this by imposing $t_{corr} < t_s$ and $t_{LS} < t_P$. However, since $t_s \ll t_P$ and the algorithmic complexity of both correlation (Stage 1) and LS estimation (Stage 2) are $O(\beta)$, the limiting factor on real-time operation is the correlation, for which $t_{corr} < t_s$ must hold.

To construct the payload, SABRE takes the incoming data bits and encapsulates them into OFDM blocks, each containing multi-variate modulation schemes, i.e., BPSK, QPSK, 8-QAM,

16-QAM, 32-QAM and 64-QAM. To ensure coherent reception of the payload, our system makes use of the LS channel estimation to conform the beamweights following the MTC procedure described in Sec. III. This is captured in (8).

$$w[m] = \left(\frac{\hat{h}[k-D]}{||\hat{h}[k-D]||} \right)^{-1} \tag{8}$$

The complete flow of this process is depicted in Fig. 6a and 6b, showing the realization of the closed-loop feedback between the transmitter (UAV) and receiver (users) in SABRE. The receiver after performing the preamble detection and channel estimation, assesses the misalignment between the received frames from the distributed antennas by inspecting the correlation indices in the time domain. To ensure coherent reception of the payload in upcoming transmissions, the receiver conforms a feedback packet containing (*i*) a time correction for aligned reception, and (*ii*) the channel estimation, used by the transmitter to compute the beamweights following (8). The corresponding blocks for this operation are highlighted in blue in Fig. 6a and 6b.

2) Timing and Synchronization: We now probe how to mitigate the second cause of misalignment in DCABF, i.e., time synchronization. For scenarios like distributed beamforming, an accurate coherence of the transmission start time between spatially distributed nodes is needed to achieve an alignment at the sample level. For example, a communication bandwidth of 1 MHz would require a time alignment at the microsecond level. This is challenging given that current state-of-the-art wired synchronization methods cannot be applied to untethered UAVs.

We obtain this precision in SABRE with our software-based synchronization algorithm for distributed entities, which is implemented in a two-staged sequential manner. First, we perform a *coarse synchronization* by setting all of the UAVs Linux host clocks up to an accuracy of less than a second. We follow this up with the *fine synchronization* stage, where the internal hardware clock of the SDRs are synced to achieve $< 1\,\mu$s level accuracy between the transmit streams from each UAV.

• **Coarse synchronization:** To set the Linux host's time, we use a local Network Time Protocol (NTP) daemon called Chrony [43, 44]. Chrony encompasses advanced NTP features which make it a good fit for our DCABF environment. For instance, it can synchronize to the local time server much faster than the default NTP service, which is dependent on network connection; compensate for fluctuating clock frequencies in the host; and it never steps the clock after the initial time sync. The later ensures stable and consistent time intervals for system services and applications.

Before starting the Chrony service in the Linux host, we give highest priority to the time sync thread by locking the other concurrent software threads from causing context-switches. This is done in order to mitigate undesired but highly likely software lags or thread delays while setting the time value on the SDR. Once the lock is achieved, we start the Chrony service by designating one of the nodes (which in our case is the receiver) as the Chrony server. The rest of the nodes take that server's time as the reference and synchronizes their own system clocks with it.

• **Fine synchronization:** As for the fine synchronization stage, we implement a fine-grained, hardware-based, over-the-air timing synchronization through our in-house developed wireless clock synchronizer, RFClock [45]. Without modifying existing physical/link layer protocols, as a standalone unit, RFClock provides timing, frequency, and phase synchronization by generating a 10 MHz/1 PPS reference signal suitable for most COTS SDRs today. By utilizing a leader-follower architecture, RFClock-leader allows follower clocks on to synchronize with mean offset under 0.107 Hz, and then corrects the time/phase alignment to be within a 5 ns deviation. As shown in Fig. 6c, RFClock follows the *leader-follower* model, with the leader mounted on a standalone UAV, generating the reference clock that is distributed to all followers on DCABF UAVs. RFClock is designed to provide (1) carrier frequency synchronization that overcomes clock frequency offsets and locks each device to the same reference frequency, (2) timing synchronization so that each device can perform the desired action at coordinated intervals, such as the rising/falling edge of the clock, and (3) carrier phase synchronization, so that clock signal arrives with the same phase for all followers.

(1) Carrier Frequency Synchronization: RFClock *leader* transmits a two tone frequency signal at f_1 and f_2 over the air, separated by the desired input clock frequency (typically, 10 MHz). The RFClock *follower* extracts the envelope of the transmitted signal and passes it through a customized filtering process to obtain the reference clock. Thus, all nodes have the same LO drift, as they are locked to a common reference, and do not lose synchronization even if there is a frequency drift in the leader's clock. At the follower, RFClock's envelope detector measures the *beat* frequencies $f_2 - f_1$ and $f_2 + f_1$. The difference frequency $f_2 - f_1$ drives the virtual LO of 10 MHz, and in turn, the receiver's phased locked loop (PLL). RFClock's front-end extracts the 10 MHz signal with ultra-low power, passive, off-the-shelf components, consuming only 6.6 µW.

(2,3) Accurate Time/Phase Estimation: In addition to the 10 MHz reference, each SDR requires a PPS signal to perform processing tasks at the same time. However, even if all the devices have their LO driven by the reference clock frequency, there can still be phase difference

between clock edges. Thus, any time offset between PPS edges for individual RFClock followers needs to be compensated. RFClock includes a clock alignment algorithm and an auxiliary correction mechanism to increase resiliency, which selects inputs from a cheap, off-the shelf GPS module costing around $35 and/or ultra-wide band (UWB) module. Whenever GPS is available, RFClock followers correct their individual time offsets with respect to this global PPS reference. In GPS denied environments, RFClock receivers use UWB ranging to produce high-resolution timestamps (with pico-second precision) and estimate phase offset with respect to the RFClock leader, which eliminates explicit pair-wise messaging. We consider moderate user and UAV mobility models, ranging from 0.5m/s to 2m/s in indoor and outdoor settings. From our experiments with RFClock in [45], we observe in that this type of mobility introduces jitter in the received clock signal, resulting in Carrier Frequency Offset (CFO) error of up to 3.73Hz when the PLL loop bandwidth is 100Hz. We decrease the loop bandwidth to 10Hz, which reduces error down to 1.8Hz at running speed, thus enabling the distributed beamforming on hovering UAVs within these speed limits. The results of this operation from real life experiments are displayed in Fig. 8 and 9, respectively.

We compared the performance of RFClock with existing GPS disciplined oscillators (Ettus GPSDO) [46] that provide 10 MHz clock and PPS signals. We observe that the relative time error between two GPSDO-sourced PPS is $\pm\,500$ns. However, the relative phase drift between two clock outputs of GPSDOs is not stable, varying between 0-100 ns, which results in skewed constellations across all MCS values at the receiver-side, resulting in high BER, shown in Fig. 8a and 8c. We also show through experiments (in [45], at the Section 6.2 and Fig. 18b) that RFClock can cover an area of radius 45.72 m in the 900 MHz ISM band, which is enough for operating with multiple UAVs. RFClock thus ensures that all the SDRs on the UAVs collectively lock their hardware time to the same PPS instant, which guarantees that the spatially distributed SABRE entities start transmitting on their SDRs simultaneously even when the GNURadio applications themselves are not initialized at the same time. At this stage any other process in the Linux host is put to sleep to prevent buffer overflows at the radio UHD driver software [47].

Even when the SDRs start their transmissions at the same PPS instant, we find that it is not enough to satisfy the $\leq 1\,\mu$s precision required for data beamforming. To achieve this desired precision, the receiver computes the symbol misalignment between all the transmit streams by setting the last received stream in that iteration as a reference. It then piggybacks this information along with the CSI as a feedback to the individual transmitters. The transmitters

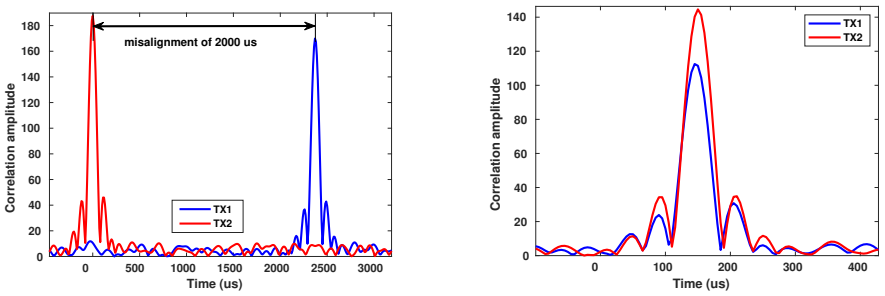


Fig. 7: Receiver cross-correlating transmit signals from two distributed transmitters

TABLE III: Real life experimental configuration

Operating frequency	900 MHz	Antenna Characteristics	Omnidirectional
Bandwidth	400 KHz	Number of Tx UAVs	4 DJI M100s
Radio front ends	Ettus B210 SDR	Wireless clock synchronizer	RFClock
Number of users	1	Location	Outdoor

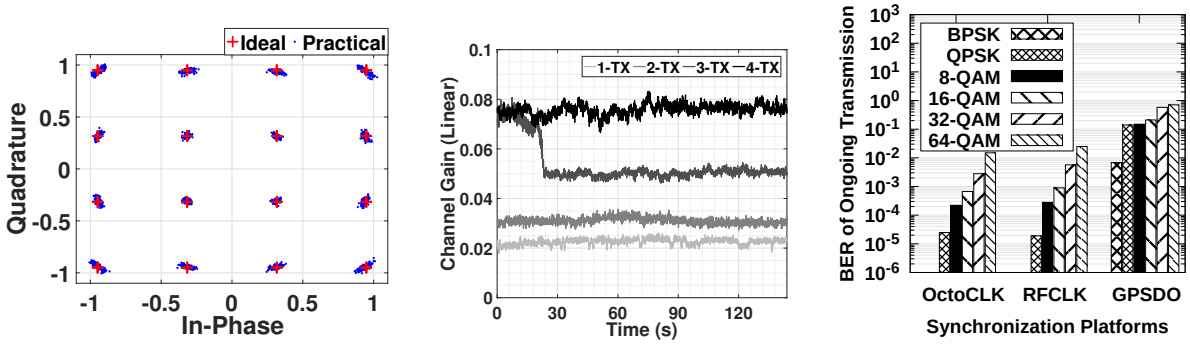
delay their signals accordingly with additional zero-padding to sync with the transmitter radio peer having the latest transmission start time. These modules are integrated with the SABRE beamforming algorithm in Fig. 6a and are highlighted in blue. At this stage we are able to get the desired correlation accuracy of within 1 μs between all the beamforming transmitters. These synchronization steps start after the UAVs reach their intended GPS-mandated location to start the collaborative beamforming process.

We validate this process through a testbed experiment where we use two hovering UAV transmitters to beamform two unique Gold sequences to a static ground based receiver using the time synchronization algorithm. The results for time alignment at reception are captured in Fig. 7, where we observe that the preambles lack coherence in absence of time synchronization. After implementing the sync steps, we achieve perfect correlation between the transmitted preambles with $\leq 1\,\mu$s accuracy, thereby ensuring coherent reception.

C. Beamforming Validation

After ensuring the desired synchronization of the transmitted signals, our next step is to validate the data beamforming result at the receiver. For this, we use a testbed setup consisting of four UAVs, in an outdoor environment, with the components as explained in Sec. V-B, and also given in Table III, to beamform the pre-constructed payload with the unique Gold sequences (explained in Sec. V-B1, and zero-padding (explained in Sec. V-B2), and calculate the BER of the received beamformed payload.

The synchronization accuracy of RFClock in DCABF is showcased in Fig. 8a, resulting in near-zero phase and frequency offsets on the received I/Q symbols over time throughout the

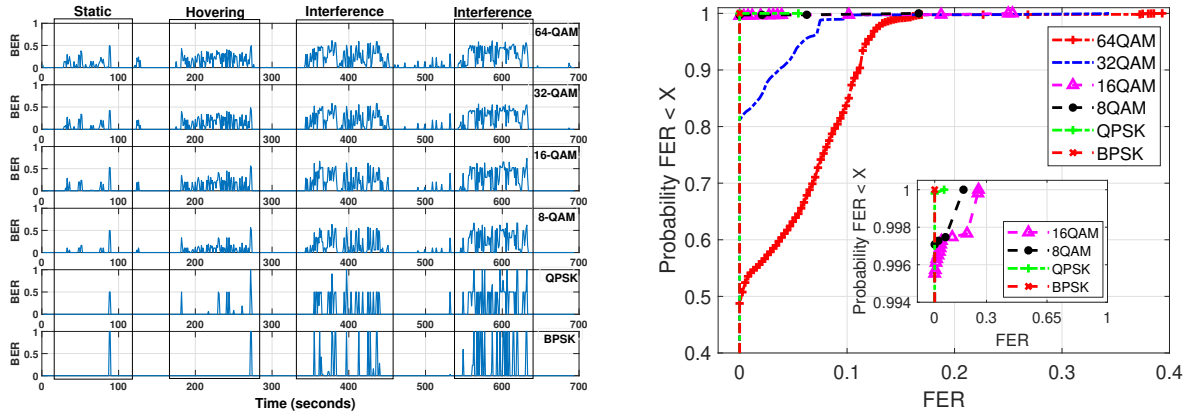


(a) (b) (c)

Fig. 8: (a) Constellation diagram of 16-QAM with RFClock synchronization in DCABF. The constellation points (in blue) were superimposed over 100 iterations; (b) Channel gain with increasing number of transmitters; (c) BER performance in different modulation schemes and synchronization methods.

duration of the experiment. Fig. 8b showcases the expected effect of linear increase in channel gain due to DCABF, as we increase the number of transmitters. This improvement in channel gain is a result of in-phase arriving signals from the transmitters, which in turn improves BER at the receiver. The impact of RFClock on BER for different modulation schemes, when compared with Octoclock and GPSDO, is shown in Fig. 8c. We see that the BER performance of RFClock is similar to the wired setup of Octoclock for modulation schemes up to 8-QAM (10^{-6} for BPSK and QPSK), but degrades slightly for 16-QAM, 32-QAM and 64-QAM. The BER performance with GPSDO fares worse in comparison, with the BER staying near 10^{-2} for BPSK and rising to 10^{-1} for higher modulation schemes. Our results demonstrate that RFClock's performance is close to current wired synchronization approaches used in the industry, such as Octoclock, and performs better than the state-of-the art, GPSDO.

Fig. 9a, showcases the progression of the BER over the various modulation schemes within the OFDM-modulated payload (see Sec. V-B1) under various deployment scenarios. Namely, static deployments with the UAVs on the ground, flying deployments with the UAVs hovering at a distance of 2 m from one another, and communications suffering from in-band interference generated by neighboring UAV. This particular experiment was carried in the indoor anechoic chamber in order to generate beamforming performance data under these controlled scenarios, without being corrupted by uncontrollable external influences like unknown interferers, wind, GPS inaccuracies, etc. The drop of the BER to 0 ascertains the frame alignment at reception, and hence a working beamforming system. During the hovering scenarios, the UAVs experience a maximum displacement of ˜3 m from their intended static positions due to a variety of factors, such as GPS inaccuracies. The embedded control system in the UAVs attempts to correct it



(a) (b)

Fig. 9: (a) Effect on instantaneous BER values when beamforming with distributed aerial transmitters, under static, hovering and interference conditions in the anechoic chamber; (b) Probability of FER below certain value with various modulation schemes, 4 UAV-Tx and 1 ground-Rx in outdoor environment

but it inevitably leads to an oscillating drifting motion around the intended location. In this case, the BER degrades slightly for higher modulation schemes as a result of increased channel variance. To improve this condition, we reduce the hovering displacement of the UAVs using extra localization modules as described in Sec. VI.

Next, with the interferer present, we observe that the BER degrades across all modulation schemes despite having power gains brought by beamforming. This reinforces the need for our IUI mitigation scheme in SABRE, presented earlier in this paper in Sec. IV.

VI. Performance Evaluation

After validating the DCABF approach, we next describe the performance evaluation of the entire SABRE system. We first carry out a standalone evaluation of SABRE to test the beamforming efficiency in a real-world deployment, in the absence of interference. Then, we carry out an extensive experimental campaign on Colosseum to test the scalability of SABRE using UAV swarms and multiple users, handling dissimilar traffics flows.

1) Evaluation on real world testbed: We used an outdoor experimental setup of 4 transmitter UAVs that beamform data to a single B210 SDR connected to a ground based static Linux host. We installed DJI Flight Controller on the UAVs, with GNSS and Guidance [48], for more stability when the UAVs are hovering, thereby improving the BER at the receiver. The UAVs receive the GPS coordinates for hovering via the DJI SDK system. Once all the UAVs are at the expected location, the transmitter Linux hosts synchronize the SDRs' clock with the local server, then proceed to fine tune the radio transmissions to within 1 μs delay of each another and start the beamforming transmission. The receiver, after extracting the payload, multicasts the

CSI feedback to the transmitters along with piggybacking the symbol misalignment information, if required.

Beamforming performance to single receiver: Fig. 9b provides the statistical information on the FER at the receiver. With SABRE enabled aerial beamforming, the probability of FER to be around 0 is 0.99 for BPSK, QPSK, 8-QAM and 16-QAM, while for 32-QAM and 64-QAM the probability of FER remaining below 10%, is ˜0.98 and ˜0.84 respectively. This drop in performance in the higher modulation schemes is because of the increasing number of bits being sent per frame as the MCS increases, with a single bit flip rendering the entire frame in error. Assuming we are sending frames in each of the modulation schemes, and there is no packet fragmentation, then the number of bits per frame in each modulation scheme we consider are, BPSK = 64 bits, QPSK = 128 bits, 8-QAM = 192 bits, 16-QAM = 256 bits, 32-QAM = 320 bits, and 64-QAM = 384 bits.

After validating the beamforming process through practical implementation on hovering UAVs, our next step is to carry out the performance evaluation of SABRE with increasing number of transceivers, to showcase the reliability, capacity, and scalability of our proposed model. Given the large number of nodes, we port SABRE to Colosseum [18] [19].

A. Large Scale RF Emulation

Colosseum is the world's largest RF emulator designed to support research and development of large-scale, next generation radio network technologies in a repeatable and highly configurable RF environment. It combines 128 Software Radio Nodes (SRNs) with a Massive digital Channel Emulator (MCHEM) backed by an extensive FPGA routing fabric. Each of the SRNs provides a platform for SDR with two key hardware components, namely, a Dell R730 computation module, which comes with an NVIDIA K40M GPU, and an Ettus Research USRP X310 [49] SDR that is equipped with a Xilinx Kintex 7 FPGA [50]. The MCHEM facilitates real-world wireless RF channel emulation between the SRNs and can emulate fading, multipath, etc., for up to 256 x 256 independently customizable channels. This allows for large scale RF testing with up to 256 independent radio nodes each with powerful computational capabilities. Accessible as a cloud-based platform, Colosseum also provides other resources to create a real-time high-fidelity radio scenario such as traffic generation, timing, and GPS synthesis.

1) Colosseum Architecture: As shown in Fig. 10a, a user can access the Colosseum resources via ssh and https. As per the user's request, based on the type of experiment to be conducted,

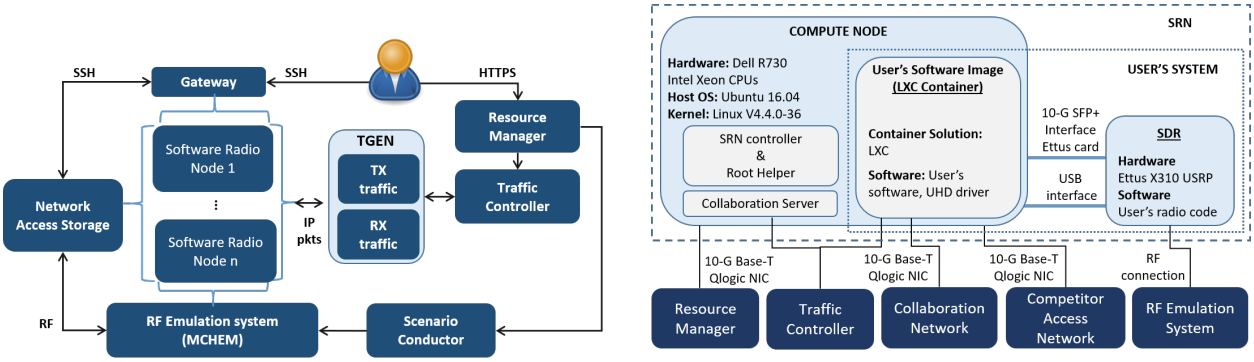


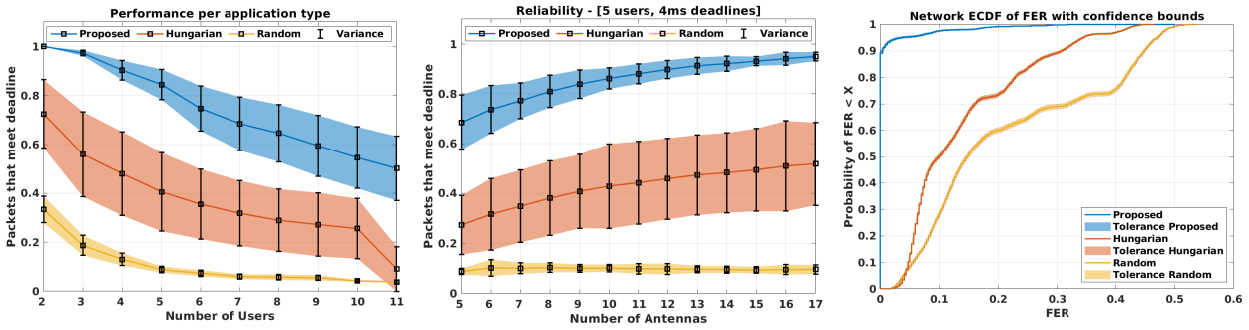
Fig. 10: (a) Colosseum architecture; (b) Software Radio Node in Colosseum.

the 'Resource Manager' module allocates available SRNs for the user. These SRNs exchange data packets between each other through the X310 SDRs over the RF MCHEM, as per the user's customization. The resource manager oversees the traffic controller and scenario conductor modules. The scenario conductor manages the MCHEM, which emulates specific RF channels between the SRNs, as per the experiment scenario.

The Traffic Controller module manages the Traffic Generator (TGEN) application, which generates and receives data traffic in the form of IP packets, based on the experiment scenario. These IP packets are made available to the SRNs, which are then transmitted as I/Q samples from the X310 SDRs over the MCHEM to the destination SRN, undergoing channel induced distortions as per the scenario description. These I/Q samples are then converted to IP packets through custom user scripts and are then fed back to the TGEN receiver.

2) Software Radio Nodes (SRNs) as UAVs: Next, we give some details of the SRNs, which are further illustrated in Fig. 10b for clarity. The SRN is composed of a Linux based computation module for data processing and a radio module for transmit/receive. In our case, each UAV and ground receiver is mapped to a single SRN. These machines have the SRN controller and root helper functions which help to run separate ubuntu installations in software based LXC containers [51] within the machine. The users' custom scripts and scripts to interface with the radio module are installed in these container images as per the user's discretion. The computation module is connected to the radio module via ethernet and USB interface. The radio module is an Ettus X310 SDR, capable of running user's radio-based applications, with tools like GNURadio, etc. For time synchronization of the SRNs, the 10 MHz reference and the 1 PPS is provided by the Ettus Octoclock [52] solution. The SRN USRPs are connected to the MCHEM USRPs via RF cables. The MCHEM emulates the RF channel by processing the baseband I/Q samples according to the selected scenario and sends the resulting RF signal to the receiving SRN. The

Fig. 11: Network topology of large scale experiment with 29 nodes, performing DCABF from UAV transmitters (T) to users (R) on the ground, in Colosseum wireless network emulator



(a) (b) (c)

Fig. 12: (a) SABRE capacity, with 17 UAV transmitters serving increasing number of users seeing traffic with 4 ms latency requirement. (b) SABRE reliability, with 5 users and increasing number of UAV transmitters seeing traffic with 4 ms latency requirement. (c) Outage Probability of FER under various allocation schemes, with 12 users, and 17 UAV transmitters, for three different traffic types, having latency requirements of 4 ms, 9 ms and 19 ms.

MCHEM emulates real-world over-the-air channels, with fading (multipath, environmental, etc.), between 256 independent radio channels. For each destination channel, the signals provided by source channels are filtered individually according to the channel model. The source channel contributions are summed and transmitted out of the MCHEM.

B. Evaluation in multi-user scenario

For the performance evaluation of SABRE in large scale settings, we deploy in Colosseum, an emulation scenario of 29 nodes, having 12 receivers on ground and 17 UAVs. For this purpose, we set up a local LXC container image where we install the SABRE code along with the necessary GNURadio modules and libraries. Then, this container image is installed on all the SRN nodes which are used in the experiment scenario. It must be noted here that the same SABRE code which we used on the UAV testbed setup with the B210 SDRs was ported to the Colosseum SRNs without any re-configurations.

Implementation details: The minimum target performance for SABRE in multi-user scenario is to fulfill the latency requirement of three types of 3GPP standard mandated traffic [4], having latency thresholds of 4 ms, 9 ms and 19 ms respectively, with increasing number of users while keeping the number of UAV transmitters constant. For the emulated experiment environment, we modify the DARPA Spectrum Collaboration Challenge scenario called as *Alleys of Austin* [53], which emulates a platoon from the Texas Army National Guard at Camp Mabry, practicing urban maneuvers and communications in Austin, Texas. This scenario is emulated to be as realistic to the actual environment as possible, including both LoS and NLoS conditions, through the representation of the channel model with 4 taps, created from actual geographical measurements [20]. At any instance of the scenario runtime, the channel impulse response based on the delay values, takes into account the actual environmental parameters where the nodes are located at that time instant. The network topology snapshot is shown in Fig. 11, in their actual emulated location in the Austin neighborhood, with the transmitters labelled as T and receivers as R. The platoon is split into 3 squads, with the edge squads consisting of 4 soldiers on the ground, 5 UAVs hovering over the soldiers and one UAV circling around the squad. The squad in the middle having 4 UAVs hovering over the squad's ground soldiers and one UAV circling around the squad. The scenario is designed to run for 930 s, during which the squads move from the 39th street to 45th street, covering a region of 1 km^2.

The main SABRE algorithm is orchestrated from a centralized controller. In real-life experiments, this controller is co-located with the UAV command and control ground base station through a wireless backhaul link on a separate control channel, which is primarily used for UAV flight control [37]. In the Colosseum emulation, this controller and remote ground base station modules are replicated as functions on remote host machines. SABRE knows the real time location of the UAVs and the users and also it knows the geometrical layout of the urban environment. While allocating UAV beamforming resources to users, SABRE takes all these information into account which translates to the available SNR to the user for a specific UAV-beamforming array configuration, at a specific instance of time. When UAV and users experience NLoS conditions, their corresponding SNR suffers and based on the SABRE logic, those specific UAVs experiencing NLoS conditions to the concerned user will not be considered in the beamforming array configuration for that user. This is not a static resource allocation mechanism, but rather a dynamic one where for every instance of packet delivery SABRE makes sure the optimal UAV beamforming configuration is made for each and every user.

We compare the performance of SABRE with the classical Hungarian method of maximum weighted matching based resource allocation [54], using the link quality based weights between the transmitters and receivers. We also run the same set of experiments with a random allocation method of pairing the transmitters and receivers as a baseline approach.

Beamforming performance with multiple receivers: First, we measure the capacity of the network by keeping the number of UAV transmitters static at 17. We gradually increase the number of users which need to be served and measure the network QoS in terms of packets that meet the application latency requirement. Fig. 12a shows the normalized ratio of the packets that meet the application specific latency requirement of 4 ms, with increasing number of users for the three different resource allocation schemes. SABRE performs considerably better than the others, with the packets that are delivered within the application deadline being more than 50% over the random resource allocation method, and an increase of ˜34.5% over the Hungarian approach. As the user density increases, the Hungarian method reaches a threshold, where after 10 users, its performance in delivering packets within the latency threshold, drops sharply, similar to the random allocation method. This shows the advantage of SABRE in avoiding interference in a high user density scenario. Since the Hungarian method allocates UAVs to users based solely on the link quality between UAV-user pair before antenna assignment and does not consider the resulting interference to neighboring users, its performance degrades in a scenario with large number of users. In this case, SABRE is able to deliver 40% more packets than either Hungarian or the Random allocation method. SABRE is also able to deliver 100% of the traffic to the intended receivers, in 70% of the time. In Fig. 12b, we showcase the reliability of SABRE. Here, the number of users is kept static at 5, while the number of UAV transmitters is sequentially increased to 17, and the packet delivery ratio (PDR) is measured at each step with all the different resource allocation methods. Here also, SABRE proves to be more reliable than the other two methods, with a PDR increase of ˜40% over Hungarian and between 70-90% over Random allocation method. Notably, the advantage of SABRE can be seen with higher number of UAV transmitters as it approaches a PDR of more than 0.9 with 17 transmitters. Fig. 12c shows the performance of the network as a whole, while considering all three types of traffic, being served to the 12 users with the 17 UAV transmitters. Here, we show the Empirical Cumulative Distributive Function (ECDF) of the FER in the entire network, along with considering the upper and lower confidence bounds in the different resource allocation schemes. The probability of FER to be below 10% is above 0.95 for SABRE, which proves its capability in meeting heterogeneous

traffic demands, in a multi-user setting while avoiding interference. The same for the Hungarian method is ˜0.5, whereas for the random resource allocation method is ˜0.3. This proves that in a multi-user setting with diverse traffic demands, an IUI-avoiding resource allocation scheme with aerial distributed beamforming can be beneficial.

VII. Conclusion

We demonstrated the feasibility of a practical DCABF system through theoretical and preliminary experimental analysis. We carried out systems-level implementation of our theoretical models on a real test bed setup and showcased the scalability of our approach through large scale experiments in Colosseum. We verified through our test bed experiments, that SABRE achieves ˜99% probability of FER to be near 0, in the absence of any other interfering nodes. We also formulated an interference-aware resource allocation scheme for the aerial distributed beamformers. Our approach achieves an improvement of ˜40%, over classical methods to meet the latency requirement of different traffic types with varying number of users. Also, SABRE improves the network performance as a whole, resulting in a performance improvement of 50% when compared to the classical resource allocation metrics. As part of our future work, we plan to focus on demonstrating enhanced beamforming algorithms that can operate even with intermittent CSI information resulting from losses in the receiver generated feedback packets and reducing the training signal overhead for beamformed payloads to improve further the network QoS in low latency applications.

VIII. Acknowledgment

This research was accomplished under the Cooperative Agreement W911NF-19-2-0221 between the US Army Research Laboratory and Northeastern University.

References

[1] M. Yang, B. Li, Z. Yan, and Y. Yan, "Ap coordination and full-duplex enabled multi-band operation for the next generation wlan: Ieee 802.11 be (eht)," in *IEEE Conference on Wireless Communications and Signal Processing*, 2019, pp. 1–7.

[2] A. Garcia-Rodriguez, D. Lopez-Perez, L. Galati-Giordano, and G. Geraci, "Ieee 802.11 be: Wi-fi 7 strikes back," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 102–108, 2021.

[3] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta, "Time-sensitive networking in ieee 802.11 be: On the way to low-latency wifi 7," *Sensors*, vol. 21, no. 15, p. 4954, 2021.

[4] 3GPP, "Service requirements for the 5G system," 2017. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/ SpecificationDetails.aspx?specificationId=3107

[5] M. A. Lema, A. Laya, T. Mahmoodi, M. Cuevas, J. Sachs, J. Markendahl, and M. Dohler, "Business Case and Technology Analysis for 5G Low Latency Applications," *IEEE Access*, vol. 5, pp. 5917–5935, 2017.

[6] P. e. a. Schulz, "Latency Critical IoT Applications in 5G: Perspective on the Design of Radio Interface and Network Architecture," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 70–78, 2017.

[7] K. C. Manar Mohaisen, YuPeng Wang, "Multiple Antenna Technologies," 2008.

[8] P. B. et. al., "Implementation and demonstration of receiver-coordinated distributed transmit beamforming across an ad-hoc radio network," *Conference Record - Asilomar Conference on Signals, Systems and Computers*, 2012.

[9] S. Mohanti, E. Bozkaya, M. Y. Naderi, B. Canberk, and K. Chowdhury, "Wifed: Wifi friendly energy delivery with distributed beamforming," in *IEEE INFOCOM*, 2018, pp. 926–934.

[10] P. Sarunic and R. Evans, "Hierarchical model predictive control of UAVs performing multitarget-multisensor tracking," *IEEE Transactions on Aerospace and Electronic Systems (TAES)*, vol. 50, no. 3, pp. 2253–2268, 2014.

[11] Vodafone, "Vodafone Beyond Visual Line of Sight Drone Trial Report," Tech. Rep. November, 2018. [Online]. Available: https://www.vodafone.com/content/dam/vodafone-images/me- dia/Downloads/VodafoneBVLOSdronetrialreport.pdf

[12] Y. Zeng, J. Lyu, and R. Zhang, "Cellular-connected UAV: Potential, challenges, and promising technologies," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 120–127, 2018.

[13] M. Mozaffari, W. Saad, M. Bennis, Y. H. Nam, and M. Debbah, "A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[14] R. Mudumbai, D. R. Brown, U. Madhow, and H. V. Poor, "Distributed Transmit Beamforming : Raghuraman Mudumbai , University of California at Santa Barbara," *IEEE Communications Magazine*, no. February, pp. 102–110, 2009.

[15] H. Ochiai, P. Mitran, H. V. Poor, and V. Tarokh, "Collaborative beamforming for distributed wireless ad hoc sensor networks," *IEEE Transactions on Signal Processing (TSP)*, vol. 53, no. 11, pp. 4110–4124, 2005.

[16] K. Chang, P. Rammos, S. A. Wilkerson, M. Bundy, and S. A. Gadsden, "LiPo battery energy studies for improved flight performance of unmanned aerial systems," *Unmanned Systems Technology XVIII*, vol. 9837, no. 1, p. 98370W, 2016.

[17] S. Mohanti, C. Bocanegra, J. Meyer, G. Secinti, M. Diddi, H. Singh, and K. Chowdhury, "Airbeam: Experimental demonstration of distributed beamforming by a swarm of uavs," in *2019 IEEE MASS*, pp. 162–170.

[18] Northeastern University, "Colosseum at Northeastern University." [Online]. Available: https://www.northeastern.edu/colosseum/

[19] John Hopkins University Applied Physics Laboratory, "The DARPA SC2 Colosseum Test Bed." [Online]. Available: www.jhuapl.edu/TechDigest/Detail?Journal=JVolumeID=35IssueID=1

[20] M. Tehrani-Moayyed, L. Bonati, P. Johari, T. Melodia, and S. Basagni, "Creating rf scenarios for large-scale, real-time wireless channel emulators," in *Proceedings of MedComNet*.

[21] G. Geraci, A. Garcia-Rodriguez, M. M. Azari, A. Lozano, M. Mezzavilla, S. Chatzinotas, Y. Chen, S. Rangan, and M. Di Renzo, "What will the future of uav cellular communications be? a flight from 5g to 6g," *arXiv preprint arXiv:2105.04842*, 2021.

[22] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. M. Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani *et al.*, "Evolution of non-terrestrial networks from 5g to 6g: A survey," *arXiv preprint arXiv:2107.06881*, 2021.

[23] F. Quitin, M. M. U. Rahman, R. Mudumbai, and U. Madhow, "A scalable architecture for distributed transmit beamforming with commodity radios: Design and proof of concept," *IEEE TWC*, vol. 12, no. 3, pp. 1418–1428, 2013.

[24] S. Leak and et. al., "Distributed Transmit Beamforming Expanding the Capacity and Range of Tactical Communications," in *Military Communications and Information Systems Conference (MilCIS)*, 2018.

[25] W. An, P. Zhang, J. Xu, H. Luo, L. Huang, and S. Zhong, "A Novel Machine Learning Aided Antenna Selection Scheme for MIMO Internet of Things," *Sensors*, vol. 20, no. 8, p. 2250, 4 2020.

[26] H. Y. Lu and W. H. Fang, "Joint transmit/receive antenna selection in MIMO systems based on the priority-based genetic

algorithm," *IEEE Antennas and Wireless Propagation Letters*, vol. 6, pp. 588–591, 2007.

[27] A. Kühne and A. Klein, "Transmit antenna selection and OSTBC in adaptive multiuser OFDMA systems with different user priorities and imperfect CQI," in *International ITG Workshop on Smart Antennas (WSA)*. IEEE, 2010, pp. 218–225.

[28] H. Li, H. Zhang, D. Li, Y. Liu, and A. Nallanathan, "Joint Antenna Selection and User Scheduling in Downlink Multi-User MIMO Systems," in *International Conference on Computer and Communications (ICCC)*. IEEE, 2018, pp. 1072–1076.

[29] A. Garcia-Rodriguez, G. Geraci, D. López-Pérez, L. G. Giordano, M. Ding, and E. Bjornson, "The essential guide to realizing 5g-connected uavs with massive mimo," *IEEE Communications Magazine*, vol. 57, no. 12, pp. 84–90, 2019.

[30] H. H. Yang, G. Geraci, T. Q. Quek, and J. G. Andrews, "Cell-edge-aware precoding for downlink massive mimo cellular networks," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3344–3358, 2017.

[31] D. López-Pérez, M. Ding, H. Li, L. G. Giordano, G. Geraci, A. Garcia-Rodriguez, Z. Lin, and M. Hassan, "On the downlink performance of uav communications in dense cellular networks," in *2018 IEEE GLOBECOM*, pp. 1–7.

[32] C. Bocanegra, K. Alemdar, S. Garcia, C. Singhal, and K. R. Chowdhury, "NetBeam: Networked and Distributed 3-D Beamforming for Multi-user Heterogeneous Traffic," in *IEEE DySPAN*, 2019, pp. 1–10.

[33] M. Y. Naderi, K. R. Chowdhury, and S. Basagni, "Wireless sensor networks with RF energy harvesting: Energy models and analysis," in *IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2015, pp. 1494–1499.

[34] D. Tse and V. Pramod, *Fundamentals of wireless communication*, 2005, vol. 9780521845.

[35] A. Firag, P. J. Smith, H. A. Suraweera, and A. Nallanathan, "Performance of beamforming in correlated MISO systems with estimation error and feedback delay," *IEEE TWC*, vol. 10, no. 8, pp. 2592–2602, 2011.

[36] Bird, Trevor, "Bessel Functions." [Online]. Available: www.onlinelibrary.wiley.com/doi/pdf/10.1002/9781119127451.app2

[37] Y. C. Paw and G. J. Balas, "Development and application of an integrated framework for small uav flight control development," *Mechatronics*, vol. 21, no. 5, pp. 789–802, 2011.

[38] A. Adhyapak, M. Diddi, M. Kling, A. Colby, and H. Singh, "Exploration of anechoic chamber characterization with autonomous unmanned aerial systems," in *2021 IEEE EuCAP*, pp. 1–5.

[39] DJI, "Matrice 100 - Product Information." [Online]. Available: https://www.dji.com/matrice100/

[40] NVIDIA, "Jetson TX2." [Online]. Available: www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/

[41] Ettus Research, "USRP B210 SDR," 2019. [Online]. Available: https://www.ettus.com/all-products/x310-kit/

[42] Z. Xinyu, "Analysis of M-sequence and Gold-sequence in CDMA system," *IEEE ICCSN*, no. 1, pp. 466–468, 2011.

[43] D. L. Mills, "ntp.org: Home of the Network Time Protocol." [Online]. Available: http://www.ntp.org/

[44] R. Curnow and M. Lichvar, "chrony – Introduction," 2019. [Online]. Available: https://chrony.tuxfamily.org/

[45] K. Alemdar, D. Varshey, S. Mohanti, U. Muncuk, and K. Chowdhury, "Rfclock: timing, phase and frequency synchronization for distributed wireless networks," in *2021 ACM MobiCom*, pp. 15–27.

[46] Ettus Research, "Board Mounted GPSDO." [Online]. Available: https://www.ettus.com/all-products/gpsdo-tcxo-module/

[47] Ettus, "UHD (USRP Hardware Driver)." [Online]. Available: https://www.ettus.com/sdr-software/uhd-usrp-hardware-driver/

[48] DJI, "High precision vision positioning." [Online]. Available: https://www.dji.com/guidance

[49] Ettus Research, "USRP X310." [Online]. Available: https://www.ettus.com/all-products/x310-kit/

[50] Xilinx, "Kintex-7 FPGA Family." [Online]. Available: http://www.xilinx.com/products/silicon-devices/fpga/kintex-7/

[51] Linux, "Linux Containers - LXC - Introduction," 2019. [Online]. Available: https://linuxcontainers.org/lxc/introduction/

[52] Ettus Research, "OctoClock CDA-2990." [Online]. Available: https://kb.ettus.com/OctoClockCDA-2990

[53] RCR Wireless News, "DARPA SC2: Alleys of Austin." [Online]. Available: https://www.youtube.com/watch?v=8k8ctvxtI4U

[54] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics*, vol. 2, no. 1-2, pp. 83–97, 1955.