# Building UAV-based Testbeds for Autonomous Mobility and Beamforming Experimentation

Yan Shi, John Wensowitch, Alexander Ward, Mahmoud Badi, and Joseph Camp
Department of Electrical Engineering, Southern Methodist University

*Abstract*— **Unmanned aerial vehicles have been deployed in many applications such as search and rescue, reconnaissance, and disaster recovery. However, UAV mobility can threaten the ability to maintain robust transmissions in practical deployments. On one hand, advanced software methodologies and extensive experiments are required to ensure safe and autonomous flights. On the other hand, to unlock additional capacity in drone communications, additional techniques must be leveraged such as directionality via MIMO-based beamforming, requiring accurate channel information to be fed back in-flight. Software defined radio (SDR) platforms play a major role in filling these gaps in multiple frequency bands, customizable design, and performance characterization. In this work, we present the hardware setup as well as software architecture of our proposed testbed leveraged for two different applications: autonomous mobility and beamforming. In the autonomous mobility case, we build a robust UAV-control framework on a customizable drone platform using MAVLink. Our experiments have demonstrated the feasibility of intelligent automated flight patterns. In the beamforming case, we implement a beamforming scheme on a drone-based SDR platform and evaluate its performance in various contexts. Our evaluations reveal that the drone-based beamforming can improve throughput significantly over conventional schemes.**

## I. Introduction

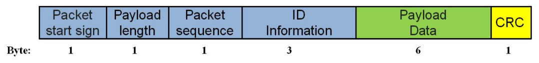
Unmanned aerial vehicles (UAVs) or drones have recently been the subject of many research topics and applications, from conventional missions like surveillance and reconnaissance to special forces like electronic interference, node swarms, and long-haul communication relays, each of which has traditionally relied upon manual solutions or terrestrial-based vehicles. The reason for such an increasing interest in drones lies in the flexibility and vision that this platform provides in novel solutions in remote control, data delivery, security, and machine intelligence [1].

NASA Langley Research Center has been testing with flights for a number of years [2]. However, there still exists a need for autonomous flight design with advanced software methodologies and experimental validation to ensure a technological feasibility for safe autonomous systems [3], motivating the need for a fully customizable testbed for extensive autonomous flight testing. Although many works have explored drone-based applications, they largely use simulated environment [2], [4]–[7]. For example, Ribeiro *et al.* simulated a predictive formation model to allow autonomous flights based on non-linear models [4]. Al-Hourani *et al.* and Kalantari *et al.* studied drone communications to optimize the position of aircrafts in urban environments, but both works lacked in-field experimental support [6], [7].

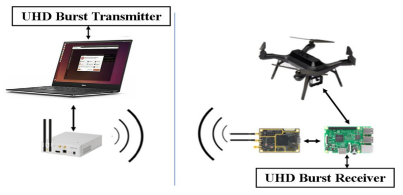
Beamforming techniques are expected to unlock additional capacity in drone communications by precoding the amplitudes and phases of an antenna array, but there are fundamental issues with directionality due to mobility in these networks and the channel feedback overhead that is required [8]. Many works have studied the benefits of beamforming [9], but the vast majority of these works focus on terrestrial networks which are not completely applicable to practical drone-based testbeds. For example, the cellular tower will always be fixed in location and lack the vibrations of a hovering drone. Moreover, many works assume ideal channel state information (CSI) feedback [10], motivating the need for the design of efficient feedback and in-field analysis of UAV-based beamforming characterization.

In this work, we present the hardware setup as well as software architecture of our proposed testbed leveraged for two different applications: autonomous mobility and beamforming-based connectivity. We use GNU Radio, a free and open-sourced software development kit, for software system on SDRs to realize UAV-control message exchange and evaluate the performance of drone-based beamforming transmissions [11]. To explore the feasibility of autonomous mobility, we first build an UAV-control testbed based on a customizable drone platform that supports the MAVLink protocol. We use 3D printing to print a mount housing an SDR platform (a lightweight USB-powered Ettus B200mini), a Pixhawk2 Green Cube (an on-board flight controller), and a Raspberry Pi3 (a embedded Linux system to communicate with B200mini) on a 3DR Solo Quadcopter to realize user-defined packet exchange for UAV control. We then evaluate the specified systems ability to perform a typical autonomous flight pattern inside a hanger, providing the basis for further development of a research platform that enables the study of more advanced autonomous flights. In order to realize drone-based beamforming, we build platforms for allowing mobility testing of a frequency-flexible beamforming system on a SDR platform that supports reconfigurable processing blocks, multiple antennas, channel estimation, and feedback at the receiver, and beamforming precoding at the transmitter. We perform repeatable experiments to understand the impact that the carrier frequency plays in physical beamforming networks including two key transmission channels. We mount a SDR platform (a battery-powered $2 \times 2$ MIMO Ettus E312) on a DJI Matrice 100 drone via 3D printing to realize air-to-ground beamforming as a function of distance, frequency, and drone altitude. The main features of our testbed are as follows:

- Our testbed supports *autonomous mobility* via the use of MAVLink, a widely-used mechanism for communication of the UAV control over wireless from a ground station.
- Our design supports *beamforming experimentation* via



(a)



(b)

Fig. 1. Autonomous Flight Design: (a) MAVLink frame structure (b) UAV-Control framework

an OFDM-based frame structure, channel estimation, and precoding using USRP-based SDR platforms.

- Our platform has *scalability* to customize autonomous flight configurations of greater complexity.
- Our approach allows experimental observability of the system-level performance of beamforming in terms of Bit Error Rate (BER) or data throughput during in-field experimentation.

The rest of the paper is organized as follows. In Section II, we describe the software design. In Section III, we discuss the hardware setup in detail. We describe the autonomous mobility experiments in Section IV and beamforming experiments in Section V. Finally, we conclude in Section VI.

## II. Background and Software Architecture

In this section, we introduce the design specifics of the autonomous mobility and beamforming testbeds.

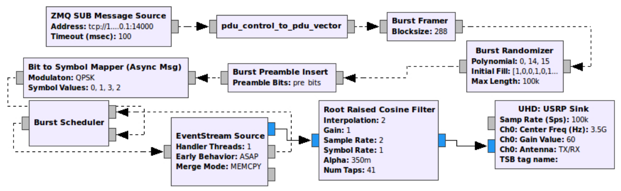
### A. *UAV-Control Architecture*

**UAV-Control Protocol:** In this work, we use MAVLink as the basic control protocol to communicate with UAVs. MAVLink is an open-source, header-only message marshalling library protocol in the flight control world for communicating between a Ground Control Station (GCS) and unmanned vehicles to transmit control commands [12]. It supports sending way-points, remotely adjusting tuning parameters, switching between flight modes, and sending remote UAV-control information and telemetry over MAVLink. The MAVLink frame structure used in this work is shown in Figure 1(a).

PyMAVLink, a Python implementation of the MAVLink Protocol, enables a MAVLink message buffering interface to translate universal UAV-control commands from software generated commands to GNU Radio and finally to MAVLink commands.

**Transmission Protocol:** In order to robustly transmit and receive MAVLink commands between a GCS and a UAV, we utilize a GNU Radio message-based burst framework that modulates command information using QPSK and carries MAVLink messages over the air using SDR platforms. The corresponding out-of-tree (OOT) modules for GNU Radio are gr-evenstream, gr-mapper, and gr-burst [13].

The message exchange operation of our proposed framework is shown in Fig. 2. Starting from the GCS transmitter side, the universal control information (e.g., taking off, altitude hold, and landing) is represented by a ZMQ message (one-byte



ZMQ SUB Message Source
Address: tcp://1....0.1:14000
Timeout (msec): 100

pdu_control_to_pdu_vector

Burst Framer
Blocksize: 288

Burst Randomizer
Polynomial: 0, 14, 15
Initial Fill: [1,0,0,1,0,1...
Max Length: 100k

Bit to Symbol Mapper (Async Msg)
Modulaton: QPSK
Symbol Values: 0, 1, 3, 2

Burst Preamble Insert
Preamble Bits: pre bits

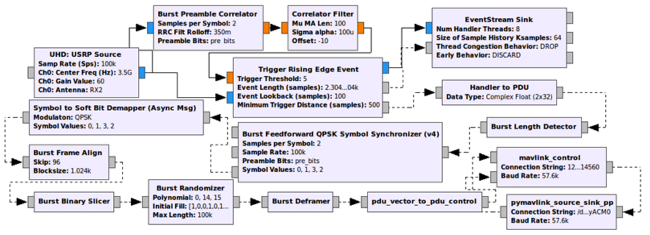
Burst Scheduler

EventStream Source
Handler Threads: 1
Early Behavior: ASAP
Merge Mode: MEMCPY

Root Raised Cosine Filter
Interpolation: 2
Gain: 1
Sample Rate: 2
Symbol Rate: 1
Alpha: 350m
Num Taps: 41

UHD: USRP Sink
Samp Rate (Sps): 100k
Ch0: Center Freq (Hz): 3.5G
Ch0: Gain Value: 60
Ch0: Antenna: TX/RX
TSB tag name:

(a)



Burst Preamble Correlator
Samples per Symbol: 2
RRC Filt Rolloff: 350m
Preamble Bits: pre bits

Correlator Filter
Mu MA Len: 100
Sigma alpha: 100u
Offset: -10

EventStream Sink
Num Handler Threads: 8
Size of Sample History Ksamples: 64
Thread Congestion Behavior: DROP
Early Behavior: DISCARD

UHD: USRP Source
Samp Rate (Sps): 100k
Ch0: Center Freq (Hz): 3.5G
Ch0: Gain Value: 60
Ch0: Antenna: RX2

Trigger Rising Edge Event
Trigger Threshold: 5
Event Length (samples): 2.304...04k
Event Lookback (samples): 100
Minimum Trigger Distance (samples): 500

Handler to PDU
Data Type: Complex Float (2x32)

Symbol to Soft Bit Demapper (Async Msg)
Modulaton: QPSK
Symbol Values: 0, 1, 3, 2

Burst Length Detector

Burst Feedforward QPSK Symbol Synchronizer (v4)
Samples per Symbol: 2
Sample Rate: 100k
Preamble Bits: pre_bits
Symbol Values: 0, 1, 3, 2

Burst Frame Align
Skip: 96
Blocksize: 1.024k

mavlink_control
Connection String: 12...14560
Baud Rate: 57.6k

Burst Binary Slicer

Burst Randomizer
Polynomial: 0, 14, 15
Initial Fill: [1,0,0,1,0,1...
Max Length: 100k

Burst Deframer

pdu_vector_to_pdu_control

pymavlink_source_sink_pp
Connection String: /d...yACM0
Baud Rate: 57.6k

(b)

Fig. 2. Message-based Transmission Framework based on GNU Radio: (a) UHD Burst Transmitter on GCS (b) UHD Burst Receiver on UAV.

network transport information for distributing stream items between processes) and is produced by a python script. Then, the ZMQ message is translated to a PyMAVlink message via the internal UDP processing block. After that, the burst message is created by passing the UDP message through a QPSK modulator, a burst preamble insert, and a randomizer in order before sending out to the wireless channel. The UAV receiver is designed to receive the burst message. The incoming message is first detected by cross-correlation with the known preamble signal. After QPSK demodulation and randomizer recovery, the obtained PDU message is finally translated back to MAVLink information from the SDR to the Raspberry Pi and serially sent to the on-board flight controller (Green Cube). To facilitate more informed decisions at the ground station, a second channel is established from the drone to the ground station to provide telemetry data, such as aircraft GPS location, altitude, and acceleration. The same burst protocol is utilized on the drone on a separate channel and transmission chain.

**Autonomous Mobility Framework:** The Autonomous Mobility Framework is shown in Figure 1(b). The GNU Radio diagrams for UHD burst transmitter is loaded in the Linux laptop to communicate with GCS USRP. The GNU Radio diagrams for UHD burst receiver running on the Raspberry Pi3 serves two purposes: (*i.*) detecting and decoding burst packets from GCS, and (*ii.*) translating and forwarding UDP control information to the drone's management controller via serial communication. In this work, the drone is piloted by a customized script to follow an autonomous pattern around a starting center point.

### B. *Beamforming Architecture*

Our proposed beamforming testbed consists of a physical layer (PHY) design that implements an OFDM-based frame structure and a media access (MAC) layer design that supports channel estimation and beamforming precoding.

**Beamforming PHY Design:** In this work, we use the IEEE 802.11 PHY frame for data transmission, which is composed of a preamble, a header symbol, and OFDM-based data

TABLE I
IEEE 802.11 BASED FRAME PARAMETERS

| **Parameters** | **Preamble** | **Data** |
|---|---|---|
| **Modulation Schemes** | BPSK | QPSK |
| **Total Subcarriers** | 52 | 52 |
| **Occupied Subcarriers** | 52 | 48 |
| **Pilot Subcarriers** | 0 | 4 |
| **FFT size** | 64 | 64 |
| **CP Interval** | 0.25 | 0.25 |

symbols of payload length $L$. Consider a typical beamforming system with $M$ transmit antennas, one single receive antenna, and $K$ subcarriers. At the $k$th subcarrier, the same copies of signal symbol $s(k)$ is coded by the beamformer prior to being sent to the UE from the $m$th transmit antenna. We represent $h_m(k)$ as the CSI obtained in the path from the $m$th transmit antenna to the single receive antenna at the $k$th subcarrier. The length of one OFDM data frame is assumed to contain a fixed number of $L$ OFDM symbols. The preamble has two OFDM symbols with known training data. Therefore, the received symbol at the $k$th subcarrier and $l$th OFDM symbol interval ($l = 1, ..., L$) can be written as:

$$r(k,l) = \sum_{m=1}^{M} h_m(k) w_m(k) s(k,l) + n(k,l) \tag{1}$$

Here, $w_m(k)$ represents the beamforming vector at the $k$th subcarrier, and $n(k,l)$ denotes the additive noise. In our previous work, we have experimentally examined the optimal packet length with frequency-dependencies for drone-based systems [14]. Other parameters of our testbed are set as suggested in IEEE 802.11 and shown in Table I.

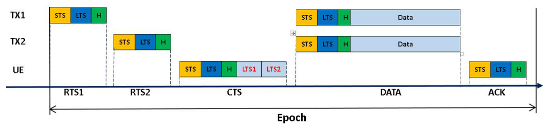
In addition, we use a conjugate beamforming vector to maximize the signal-to-noise ratio (SNR) at the receiver and hence the throughput.

$$\boldsymbol{p_n} = \frac{\bar{\boldsymbol{h}}_{\boldsymbol{n}}^{*}}{\|\bar{\boldsymbol{h}}_{\boldsymbol{n}}\|} \tag{2}$$

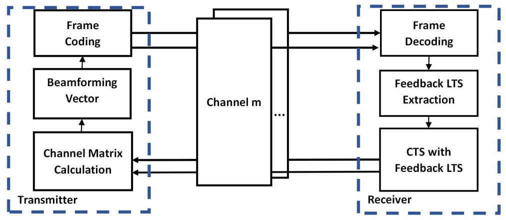
Here, $\bar{\boldsymbol{h}}_{\boldsymbol{n}}$ is the estimated 1xM complex channel gain at the $k$th subcarrier, obtained by transmitting known training symbols prior to frame decoding and channel estimation.

**Beamforming MAC Design:** In order to efficiently achieve beamforming, the receiver broadcasts back its estimate of CSI to the transmitter via the time division duplex (TDD) schedule. Considering aerial communication, one challenge is trying to expedite the feedback procedure at minimal loss, since a large feedback overhead will greatly degrade the throughput rate. A previously proposed criterion focused on the precoder codebook to choose the matrix index, while the receiver still needed to process the CSI and prepare the feedback with extra computational cost [15].

The MAC layer operation of our proposed CSI feedback method is shown in Fig. 3 (a). In each epoch, CSI information is "relayed" to the transmitter by eliminating CSI processing at the receiver. In particular, the transmitting antenna takes turns sending RTS training messages to the receiver. Instead of performing CSI estimation on these received messages, the receiver directly attaches the received LTS to the end of the CTS feedback message to be sent. Finally, the data symbols are precoded by the beamformer at the transmitter prior to being sent to the receiver. We have demonstrated that our approach greatly expedites the feedback as well as limits



(a)



(b)

Fig. 3. Drone-based beamforming system design: (a) Beamforming Timeline (b) Transmission Diagram.

power consumption, compared with conventional feedback schemes [14].

**Beamforming Framework:** We have designed and implemented PHY and MAC layers that carry out the proposed beamforming scheme discussed above using GNU Radio. The GNU Radio diagram for the design of our proposed beamforming testbed is shown in Fig. 3(b). The received signals are amplified and down modulated to baseband. The digital samples are processed by GNU Radio blocks running on a Linux-based laptop. The path loss and throughput are evaluated as a function of various horizontal distances and altitudes.

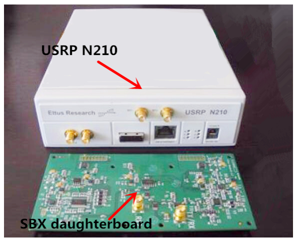
## III. HARDWARE SETUP

### A. UAV-Control Hardware Setup

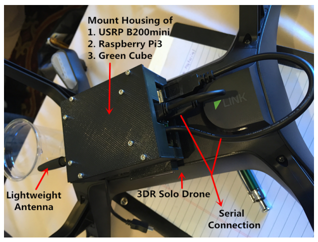
For UAV-Control experiments, we use the USRP N210 as the ground control transmitter controlled by a Linux laptop (HP ZBook Mobile Workstation) and USRP B200mini (light weight and USB powered) as the UAV receiver controlled by a Raspberry Pi3 with an OpenEmbedded Linux system, as shown in Figure 4. The housing mount is modeled using 3D CAD software and printed using a ROBO 3D printer. The ground control transmitter USRP is equipped with an SBX daughterboard that covers a frequency ranging from 400 to 4400 MHz with a bandwidth of 40 MHz. The ground control USRP is equipped with an omni-directional, multi-band antenna, and is signaled wirelessly over 3.5 GHz to not interfere with other operation networks (WiFI, cellular, etc.) and USRP data transmission channels. The UHD burst transmitter software and GUI are operating on the ground USRP for the purpose of sending out control messages to the UAV receiver. The UHD burst receiver software is set to run on booting the UAV USRP, becoming ready to receive messages in the air. A lightweight omnidirectional antenna is attached on the bottom of drone body for better signal reception.

### B. Beamforming Hardware Setup

To build the drone-based beamforming testbed, we use a USRP E312 since it is battery-powered and has 2x2 MIMO capabilities. We also use the 3D printer to design and print secure mounts to fix the USRP E312 and two omni-directional vertical antennas on a DJI Matrice 100. We choose DJI



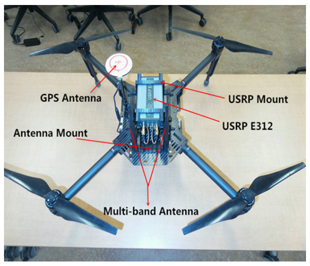
(a)



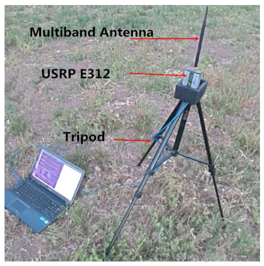
(b)

Fig. 4. Equipment Settings for Autonomous Mobility Experiments: (a) Transmitter USRP used as a Ground Control station (b) Receiver USRP mounted on a 3DR Solo Drone

Matrice 100 for in-field experiments due to its 1–kg load capability and better stabilization, as shown in Fig. 5. The antenna mounts guarantee a 10-cm separation between two antennas, allowing little correlation between two streaming channels and permitting experimental repeatability. We mount two dual-band VERT900 omni-directional antennas for carrier frequencies of 900 MHz and 1800 MHz, and two dual-band VERT2450 antennas for a carrier frequency of 5 GHz. Both antenna types provide an isotropic gain of 3 dBi. During the experiments, the receiver is mounted on a tripod at a 1-m height. The received signals are processed by software blocks, and the outcomes are recorded by a laptop. We also develop shell scripts to allow the USRP to operate in an automated fashion after beginning experimentation.



(a)



(b)

Fig. 5. Equipment Settings for Beamforming Experiments: (a) Beamformer USRP mounted on a drone (b) Beamformee Receiver mounted on a tripod

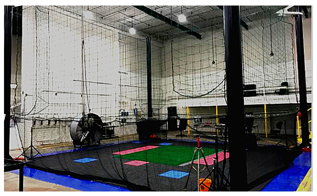
### C. In-lab Calibration

In-lab calibration of USRP RF transmission power on different frequency bands is performed by directly connecting the Rohde & Schwarz FSH8 Spectrum Analyzer to the transmitting USRP nodes [16]. We fix the total transmission power to be 2 dBm regardless of the number of transmitting antennas (single antenna and multiple antenna schemes) by equally distributing the transmission power along each RF chain.

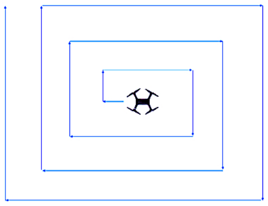
## IV. Autonomous Flight Experiments and Results

In this section, we describe our autonomous flight control experiments using a user-defined script on a custom flight control system. We evaluate our flight operation in a hangar at the Flight Operation Center located in the NASA Ames Conference Center (NACC), as shown in Figure 6(a) (from the DARPA SDR Hackfest [17]).

Our proposed autonomous flight pattern is an outward spiraling path to ensure robust and repeatable control, as shown in Figure 6(b). In order to make our experiments and data gathering repeatable, we need to automate the flight of the drone when analyzing wireless signal data. The general design of the algorithm, as shown in Algorithm 1, pilots the drone in a spiral pattern around a starting center point. The drone is placed in the center of the hanger, issued a takeoff command and a control script from the ground station. The use of GPS enables the drone to act on the commands issued through our framework with little influence from flight conditions, such as turbulence and wind. Feedback and telemetry data is available from a terminal on the ground station laptop while flight tests were in progress. As verification in the accuracy of the drones movements is difficult, we rely on choosing the same starting point and monitoring our GPS information for consistency. We are able to demonstrate a controllable and repeatable flight path based on our burst communication and control protocol design.



(a)



(b)

Fig. 6. Automated Flight Test Setup: (a) Flight Operation Center (b) Automated Outward Spiral Flight Path

```
Input: distanceInterval, maxIteration
directionToMove, distanceToMove, iteration = 0
TimerInitiation(timeout)
initialization (taking off and holding altitude)
for each iteration from 0 do
    if iterations % 2 == 0 then
        distanceToMove += distanceInterval
        timeout = TimeoutSet(distanceToMove)
        while timeout do
            Move(Direction(directionToMove[iterations % 4]))
        Altitude_Hold for 3s and clear timeout
    else
        timeout = TimeoutSet (distanceToMove)
        while timeout do
            Move(Direction(directionToMove[iterations % 4]))
        Altitude_Hold for 3s and clear timeout
```

**Algorithm 1:** Pseudocode of Autonomous Flight Pattern

## V. Beamforming Experiments and Results

In this section, we move to understanding the link performance of beamformed (2x1) transmissions that are housed on a drone using the aforementioned experimentation setup in a LOS environment by performing in-field unmanned aerial vehicle (UAV) measurements across a wide range of communication bands (900 MHz, 1800 MHz, and 5 GHz). During the experiments, we move the receiver within a radius of five times the wavelength to average fast-fading effects. For each test case, we transmit 10000 frames to obtain experimental reliability. We compare the link budget of the beamforming framework proposed in this work and the SISO scheme without beamforming, at three drone altitudes (10 m, 20 m, and 30 m)

TABLE II
ESTIMATED THROUGHPUT (MBPS) IN LOS ENVIRONMENT

| Scenarios | | Frequency | 900 MHz | | | | 1800 MHz | | | | 5 GHz | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $h$ / $d_h$ | Ground | 10 m | 20 m | 30 m | Ground | 10 m | 20 m | 30 m | Ground | 10 m | 20 m | 30 m |
| **LOS** | Beamforming (Mbps) | 10 m | 37.64 | 34.87 | 27.58 | 12.59 | 28.73 | 26.58 | 24.87 | 16.39 | 26.81 | 24.73 | 21.59 | 14.58 |
| | | 20 m | 34.56 | 32.47 | 28.02 | 19.27 | 26.69 | 23.59 | 22.74 | 22.54 | 24.73 | 21.84 | 18.00 | 11.89 |
| | | 30 m | 31.88 | 30.11 | 29.25 | 21.83 | 23.60 | 18.20 | 22.20 | 19.16 | 22.23 | 19.02 | 16.15 | 10.07 |
| | | 40 m | 29.28 | 27.70 | 28.94 | 22.95 | 21.42 | 18.32 | 22.01 | 18.97 | 20.07 | 19.90 | 14.96 | 9.09 |
| | | 50 m | 26.70 | 26.19 | 24.70 | 22.55 | 20.19 | 16.01 | 17.54 | 16.53 | 15.42 | 14.55 | 5.78 | 5.15 |
| | | 60 m | 23.35 | 22.90 | 20.98 | 20.83 | 19.59 | 14.71 | 14.47 | 12.37 | 5.08 | 4.73 | 1.39 | 0 |
| | | 70 m | 21.23 | 20.22 | 17.96 | 15.67 | 17.18 | 14.39 | 12.88 | 10.33 | 1.39 | 1.95 | 0 | 0 |
| | | 80 m | 14.70 | 16.33 | 16.11 | 13.55 | 14.58 | 13.11 | 9.79 | 4.38 | 0 | 0 | 0 | 0 |
| | | 90 m | 11.07 | 10.94 | 9.22 | 7.81 | 8.81 | 8.90 | 5.33 | 4.30 | 0 | 0 | 0 | 0 |
| | | 100 m | 9.11 | 8.85 | 7.62 | 6.57 | 6.72 | 7.39 | 3.44 | 3.16 | 0 | 0 | 0 | 0 |
| | SISO (Mbps) | 10 m | 27.16 | 26.04 | 20.06 | 8.55 | 21.79 | 20.64 | 14.33 | 12.16 | 21.28 | 20.15 | 18.20 | 12.62 |
| | | 20 m | 25.36 | 24.49 | 21.12 | 13.65 | 20.54 | 18.50 | 14.92 | 13.81 | 20.10 | 18.15 | 15.54 | 10.81 |
| | | 30 m | 23.80 | 22.93 | 22.18 | 16.17 | 18.43 | 14.42 | 15.72 | 14.56 | 18.52 | 16.12 | 14.29 | 9.15 |
| | | 40 m | 22.24 | 21.31 | 23.15 | 18.53 | 16.99 | 14.66 | 15.72 | 14.37 | 17.07 | 16.86 | 13.24 | 8.56 |
| | | 50 m | 20.64 | 20.15 | 19.76 | 18.37 | 16.26 | 12.81 | 13.52 | 13.91 | 13.77 | 12.84 | 5.38 | 5.03 |
| | | 60 m | 18.43 | 18.20 | 17.30 | 18.83 | 16.11 | 12.13 | 12.43 | 10.71 | 4.78 | 4.36 | 1.37 | 0 |
| | | 70 m | 17.53 | 16.62 | 15.29 | 14.44 | 14.71 | 12.25 | 11. | 9.21 | 1.38 | 1.88 | 0 | 0 |
| | | 80 m | 12.72 | 13.90 | 14.16 | 12.73 | 13.02 | 11.52 | 9.01 | 4.02 | 0 | 0 | 0 | 0 |
| | | 90 m | 10.07 | 9.65 | 8.38 | 7.48 | 8.22 | 8.09 | 5.01 | 4.07 | 0 | 0 | 0 | 0 |
| | | 100 m | 8.73 | 8.10 | 7.17 | 6.42 | 6.56 | 6.95 | 3.37 | 3.06 | 0 | 0 | 0 | 0 |
| | Gain | 10 m | 0.386 | 0.339 | 0.375 | 0.473 | 0.318 | 0.288 | 0.736 | 0.348 | 0.260 | 0.227 | 0.187 | 0.155 |
| | | 20 m | 0.363 | 0.326 | 0.327 | 0.412 | 0.299 | 0.275 | 0.524 | 0.632 | 0.230 | 0.204 | 0.158 | 0.101 |
| | | 30 m | 0.340 | 0.313 | 0.318 | 0.350 | 0.281 | 0.263 | 0.412 | 0.316 | 0.202 | 0.181 | 0.129 | 0.101 |
| | | 40 m | 0.317 | 0.30 | 0.25 | 0.239 | 0.261 | 0.250 | 0.401 | 0.321 | 0.176 | 0.18 | 0.132 | 0.062 |
| | | 50 m | 0.294 | 0.302 | 0.251 | 0.227 | 0.242 | 0.251 | 0.298 | 0.188 | 0.120 | 0.133 | 0.073 | 0.023 |
| | | 60 m | 0.267 | 0.258 | 0.212 | 0.106 | 0.216 | 0.212 | 0.164 | 0.155 | 0.064 | 0.086 | 0.017 | 0 |
| | | 70 m | 0.211 | 0.217 | 0.175 | 0.086 | 0.168 | 0.175 | 0.171 | 0.122 | 0.083 | 0.038 | 0 | 0 |
| | | 80 m | 0.155 | 0.175 | 0.137 | 0.065 | 0.118 | 0.137 | 0.087 | 0.089 | 0 | 0 | 0 | 0 |
| | | 90 m | 0.099 | 0.134 | 0.102 | 0.044 | 0.072 | 0.101 | 0.064 | 0.056 | 0 | 0 | 0 | 0 |
| | | 100 m | 0.043 | 0.092 | 0.059 | 0.021 | 0.024 | 0.063 | 0.020 | 0.029 | 0 | 0 | 0 | 0 |

at a transmitter-receiver separation distance ranging from 10 to 100 meters with 20 meter linear granularity. Table II shows the throughput results of both schemes at various distances. We observe that the highest air-to-ground throughput gains occur at the shortest distance (10 m), where beamforming provides improvements over SISO of up to 47.3%, 73.6%, and 22.7%, at drone altitudes of 30 m, 20 m, and 10 m, respectively, from lowest to highest carrier frequency. Furthermore, as distance increases, there is a significant decrease in gains at 900 MHz and 1800 MHz. When the distance is beyond 60 m, throughput for both beamforming and SISO schemes reduce to nearly zero. This is explained by the large path loss experienced at 5 GHz that causes clipping beyond the sensitivity level of the receiver as compared to the shorter distances. We conclude that beamforming results in significant throughput gains at shorter distances (10 to 50 meters) as opposed to more distant distances (60 to 100 meters). The results for NLOS experiments can be found in our previous work [18].

## VI. CONCLUSION

In this work, we presented hardware and software architecture of our proposed testbeds for two different applications: autonomous mobility and beamforming. We first demonstrated a UAV autonomous scheme that will allow for a customizable control of mobility and the backing protocols. We built a UAV-control framework on a customizable 3DR Solo drone platform using the MAVLink protocol and burst frame transmissions. To evaluate the feasibility of autonomous mobility, we used a pre-defined script to pilot the drone in a autonomous spiral pattern. In addition, we implemented a beamforming scheme on a commercialized DJI Matrice 100 platform and evaluate its performance at various altitudes and horizontal distances. We have demonstrated that significant gains can be obtained compared to the omni-directional scheme, and our design is flexible to allow operating on customized frequency bands for analysis on a full range of spectrum. Future work will explore the ability to allow greater levels of coordination using higher levels of mobility control and pack greater number of radios and antennas on UAV-based testbeds.

## REFERENCES

[1] E. Vattapparamban, I. Guvenc, A. I. Yurekli, K. Akkaya, and S. Uluagac, "Drones for smart cities: Issues in cybersecurity, privacy, and public safety," in *IEEE IWCMC*, 2016.

[2] M. Motter, M. Logan, M. French, and N. Guerreiro, "Simulation to flight test for a uav controls testbed," in *25th AIAA Aerodynamic Measurement Technology and Ground Testing Conference*, 2005.

[3] E. M. Arkins, "Certifiable autonomous flight management for unmanned aircraft systems," 2010.

[4] T. T. Ribeiro, "Nonlinear model predictive formation control for quadcopters," in *International Federation of Automatic Control*, 2015.

[5] A. AlHourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *IEEE GLOBECOM*, 2014.

[6] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal lap altitude for maximum coverage," in *IEEE Wireless Commun. Lett.*, 2014.

[7] E. Kalantari, H. Yanikomeroglu, and A. Yongacoglu, "On the number and 3d placement of drone base stations in wireless cellular networks," in *IEEE VTC*, 2016.

[8] C. F. Shih and R. Sivakumar, "Fastbeam: Practical fast beamforming for indoor environments," in *IEEE ICNC*, 2014.

[9] J. Mietzner, R. Schober, L. Lampe, W. H. Gerstacker, and P. A. Hoeher, "Multiple-antenna techniques for wireless communications - a comprehensive literature survey," in *IEEE Communications Surveys & Tutorials*, 2009.

[10] H. Yang and T. Marzetta, "Performance of conjugate and zero-forcing beamforming in large-scale antenna systems," in *IEEE J. Sel. Areas Commun.*, 2013.

[11] "Gnu radio." [Online]. Available: https://wiki.gnuradio.org/

[12] "Mavlink." [Online]. Available: http://qgroundcontrol.org/mavlink/start

[13] T. Rondeau and T. Flynn, "DARPABay area SDR hackfest workshop," in *GRCon 2017*, 2017.

[14] Y. Shi, R. Enami, J. Wensowitch, and J. Camp, "UABeam: UAV-Based beamforming system analysis with in-field air-to-ground channels," in *IEEE SECON*, 2018.

[15] D. Love and R. W. Heath, "Limited feedback precoding for spatial multiplexing systems," in *IEEE GLOBECOM*, 2013.

[16] Y. Shi, J. Wensowitch, E. Johnson, and J. Camp, "A measurement study of user-induced propagation effects for uhf frequency bands," in *IEEE SECON*, 2017.

[17] "Hackfest." [Online]. Available: https://darpahackfest.com/

[18] Y. Shi, R. Enami, J. Wensowitch, and J. Camp, "Measurement-based characterization of los and nlos drone-to-ground channels," in *IEEE WCNC*, 2018.