UNIVERSITY OF CALIFORNIA

Los Angeles

UAV Swarm Enabled Communications: System Design for Spectrum and Energy Efficiency with Security Considerations

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Samer Sarwat Nageeb Hanna

2021

ABSTRACT OF THE DISSERTATION

UAV Swarm Enabled Communications: System Design for Spectrum and Energy Efficiency with Security Considerations

by

Samer Sarwat Nageeb Hanna
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2021
Professor Danijela Cabric, Chair

Multi-UAV deployments create new opportunities for wireless communications. By coordinating the UAVs, they can act as a virtual-antenna-array and use multiantenna communication schemes like distributed MIMO and distributed beamforming (BF).

Distributed MIMO enables a swarm of UAVs to transmit multiple data streams simultaneously to a multiantenna ground station (GS), thus improving the spectral efficiency. Due to the line-of-sight propagation between the swarm and the GS, the MIMO channel is highly correlated, leading to limited multiplexing gains. By optimizing the UAV positions, the swarm can attain the maximum capacity given by the single-user-bound. To achieve this capacity, we propose a centralized approach using block coordinate descent and distributed iterative approach using linear controllers.

Distributed BF can extend the communication range of a remotely deployed swarm, avoiding energy waste in travel towards the destination radio. In order to beamform, the UAVs typically rely on the destination feedback, however, noisy feedback degrades the BF gains. To limit the degradation, we developed an analytical framework to predict the BF gains at a given SNR and used it to optimize the signaling with the destination. The proposed framework was verified experimentally in the lab and using UAV-mounted software-defined-radios (SDR). We also developed a feedback-free BF approach that eliminates the need for destination feedback entirely in a LOS channel. In this approach, one BF radio acts as a guide and moves to point the beam of the remaining radios towards the destination. This approach tolerates localization error and was demonstrated using SDRs.

As for the security considerations, they apply beyond UAVs to any wireless device. Security considerations include radio authentication and interpreting unauthorized signals. For device authentication, we leveraged the radios' RF fingerprint extracted using deep learning and formulated an open set classification problem to reject signals from unauthorized transmitters. We compared several approaches and studied the training dataset impact on performance. To blindly decode unauthorized signals, we proposed the dual path network (DPN) combining digital signal processing and deep learning for modulation classification and blind symbol decoding. DPN design yields interpretable outputs and by jointly estimating the unknown parameters, it improves the modulation classification accuracy.

The dissertation of Samer Sarwat Nageeb Hanna is approved.

Ankur M. Mehta

Gregory J. Pottie

Lieven Vandenberghe

Danijela Cabric, Committee Chair

University of California, Los Angeles

2021

To my family

TABLE OF CONTENTS

1 Introduction **1**

1.1 Motivation 1

1.2 Challenges and Objectives 3

1.2.1 UAV Swarms for Communications 3

1.2.2 RF Identification 4

1.3 Contributions 5

1.3.1 UAV Swarms 5

1.3.2 RF Identification 6

1.4 Thesis Organization 7

2 Spectrum Efficient Backhaul for UAVs **9**

2.1 Introduction 9

2.2 Related Work 12

2.3 System Model 15

2.4 Set of Capacity Maximizing Positions 17

2.5 Placement Optimization Problem Formulation 22

2.6 Centralized Offline Solution 24

2.6.1 Problem Relaxation 24

2.6.2 Problem Solution 26

2.6.3 Upper Bound and Time Complexity 28

2.7 Distributed Online Solution 29

2.7.1 Fundamentals of Force Field 30

2.7.2 Force Field Algorithm 32
2.7.3 Force Field URA Extension 33
2.7.4 Initializing Formations 34
2.7.5 Time Complexity of Force Field 36
2.7.6 Comparison with Existing Distributed Algorithms 37
2.8 Optimizing a Linear Uplink Scenario 38
2.9 Numerical Evaluation 39
2.9.1 Simulation Setup 40
2.9.2 Performance Gains of Position Optimization 42
2.9.3 Distributed Algorithms 44
2.9.4 Distance Traveled Per UAV 46
2.9.5 Massive MIMO Evaluation 46
2.10 Summary 48

3 Analysis and Design of BF Systems 49
3.1 Introduction 49
3.2 System Model and Distributed BF Protocol 52
3.2.1 System Model 52
3.2.2 Beamforming Protocol 54
3.3 Frequency Synchronization 57
3.3.1 Frequency Offset Estimation 57
3.3.2 Interpacket Frequency Tracking using Kalman Filter 58
3.4 Phase Estimation and Feedback 60
3.5 Beamforming Gain Analysis 61

3.6 Numerical and Experimental Validation 64
3.6.1 Numerical Validation 64
3.6.2 Experimental Validation 67
3.6.3 Emulation 70
3.7 Beamforming System Design 71
3.7.1 Specifying N and σ_e^2 72
3.7.2 Beamforming Signals Design 72
3.8 Beamforming Design Scenarios 74
3.8.1 Swarm of Small UAVs 74
3.8.2 Weather Balloons 76
3.9 Summary 78

4 Guided Beamforming 80
4.1 Introduction 80
4.2 Related Work 83
4.3 Problem Statement 85
4.4 Guided Distributed Beamforming 87
4.4.1 Approach 88
4.4.2 Guide Separation 89
4.5 Numerical Evaluation 92
4.5.1 Impact of BF Nodes Placement Geometry 93
4.5.2 Localization and Channel Induced Errors 95
4.6 Beamforming Implementation 99
4.7 Experimental Evaluation 102

4.7.1 Baseline Evaluation using Destination Feedback . . . 104
4.7.2 Guided BF Environment . . . 105
4.7.3 Simulating the Expected Beampattern . . . 105
4.7.4 BF Gains . . . 106
4.7.5 SNR and Packet Error Rate . . . 109
4.8 Summary . . . 110

5 Transmitter Authorization . . . 112
5.1 Introduction . . . 112
5.2 Related Work . . . 115
5.3 Classification, Openset Recognition, and Anomaly Detection . . . 119
5.4 System Model and Problem Formulation . . . 120
5.5 Machine Learning Approaches . . . 122
5.6 Dataset . . . 127
5.7 Network Architectures . . . 128
5.7.1 Architecture Comparison . . . 129
5.7.2 Architecture Description . . . 132
5.8 Evaluation Procedure . . . 133
5.9 Transmitter Set Sizes Evaluation . . . 137
5.9.1 Authorized set . . . 138
5.9.2 Known set . . . 141
5.10 Dataset Training Days Evaluation . . . 142
5.11 Summary of Results . . . 143
5.12 Summary . . . 144

6 Blind Signal Decoding . . . 146

6.1 Introduction . . . 146

6.2 Related Work . . . 150

6.3 System Model and Problem Formulation . . . 153

6.4 Dual Path Network (DPN) . . . 154

6.4.1 Dual Path Design . . . 155

6.4.2 Design Motivation and Considerations . . . 156

6.4.3 Design Merits and Drawbacks . . . 157

6.4.4 Training Data and Method . . . 159

6.4.5 Scalable Neural Network Architecture . . . 161

6.4.6 DPN Implementation and Training Parameters . . . 162

6.4.7 Alternative Architectures . . . 163

6.5 Signal Recovery and Symbol Decoding Algorithm . . . 164

6.5.1 Benchmark Signal Recovery Algorithms . . . 165

6.5.2 DPN Post Processing for Signal Recovery . . . 166

6.5.3 Symbol Decoding . . . 166

6.6 Data Generation and USRP Capture . . . 167

6.7 Results . . . 169

6.7.1 Signal Recovery . . . 169

6.7.2 Blind Symbol Decoding . . . 173

6.7.3 Modulation Classification . . . 175

6.7.4 Over-the-Air USRP Evaluation . . . 179

6.7.5 Variable Length Evaluation . . . 180

6.8 Summary 181

7 Conclusion 183

7.1 Summary of Contributions 183

7.2 Future Work 185

A Appendix for Chapter 1 187

A.1 Proof of Lemma 1 187

A.2 Proof of Proposition 1 188

A.3 Proof of Proposition 2 189

A.4 Proof of Lemma 4 189

B Appendix for Chapter 2 190

B.1 Proof of Proposition 4 190

B.2 Proof of Proposition 5 190

B.3 Proof of Proposition 6 191

B.4 Proof of Proposition 7 192

C Appendix for Chapter 4 194

C.1 Discriminator (Disc) 194

C.2 One Vs All (OvA) 195

C.3 OpenMax (OpMx) 195

C.4 AutoEncoder (AutoEnc) 195

D Appendix for Chapter 5 196

D.1 Band Segmentation 196

D.2 Fine Carrier Frequency Estimation 196

D.3 Fine Symbol Rate Estimation 197

D.4 Symbol Timing Offset Estimation 197

D.5 Blind Equalization 198

D.6 Genie Equalization 198

D.7 Symbol Decoding 198

References 199

LIST OF FIGURES

2.1 In the proposed system model, the UAV swarm is in the far-field of a uniform rectangular antenna array GS. 15

2.2 An illustration of the set $\mathcal{P}$ for a swarm placed on the same plane at $y = R$ for a 2×2 URA. The numbered colored shapes identify different positions. Each UAV needs to occupy a different position to maximize capacity. 18

2.3 The closed loop feedback system in UAV n where $n \in \{1, \cdots, N-1\}$. The calculations made on the UAV are highlighted in gray. The value $\Delta\phi_{n-1}$ is obtained from the previous neighbor. 29

2.4 In the simulation setup, the UAVs are initialized in a rectangular area. The GS is tilted towards the swarm. 40

2.5 As we increase the number of UAVs and GS antennas, Cent gives an equal sum rate to URA, which is higher than relying on the initial positions. 42

2.6 Blue dots represent the initial positions, orange dots the final positions of the UAVs. The dashes on the ground represent the point of interest. While URA and Cent optimize capacity, URA moves the UAVs significantly far from their points of interest. 43

2.7 Optimizing positions improves capacity for LOS dominant channel having a large K-factor. Simulation is done for the placement in Fig.2.6a and includes localization errors and channel estimation errors. 44

2.8 The sum rate and distance traveled for the ideal scenario. 44

2.9 The sum rate of distributed approaches under practical disturbances. 45

2.10 The mean distance traveled by the swarm. The error bars represent the range. . 47

2.11 The sum rate of different approaches as we move to the massive MIMO regime. 47

3.1 Timing diagram of BF protocol. The destination is the master and BF radios are the slaves. 55

3.2 The relation between BF gain, N, and σ_e^2. 61

3.3 The distribution of G for different N and σ_e^2 62

3.4 Simulated BF Gains and phase errors at different SNRs for $N = 5$ using the waveform from Table 3.2. 66

3.5 Experimental results collected using $N = 2$ BF software-defined-radios in a lab along with the theoretical results predicted by the BF framework and simulations. 68

3.6 The UAV experiment consists of 2 BF UAVs with SDRs mounted on-board. The UAVs were hovering freely and were not attached to the ground by wires. 69

3.7 Results for minimizing N in a swarm of small UAVs assuming a fixed BF overhead. Using N obtained from our approach, the SNR requirement is satisfied as verified by simulations. 75

3.8 For $N = 4$ weather balloons, in a long coherence time channel, the minimum overheads obtained using our framework satisfy the SNR requirements. 77

4.1 The objective of distributed beamforming is to coherently combine signal from N radios, which are assumed to be mobile, at a distant destination radio. 85

4.2 In the proposed approach, the guide repositions itself such it lies on the line connecting the center of the cluster of followers to the destination. 87

4.3 The signal at from radio i at point m and the signal transmitted from the guide (radio 0) have the same phase. The mismatch between the length of $\overline{mR}$ and $d_{0,R}$ leads to combining error at the destination. 89

4.4 The followers locations are contained in a rectangle of dimensions L_x and L_y, which is symmetric around the x-axis. The separation between the guide and the followers is given by d_x. 91

4.5 The BF gain of random placement of the followers for $L_x = 10m$ for different values of L_y as a function of the separation of the guide d_x. 93

4.6 The beamforming gain and the distance traveled for different values of mismatch tolerance δ. 94

4.7 The beamforming pattern for $L_x = 10m$, $L_y = 1m$. The solid line is the average of 100 random placements and the dotted lines represent ± their standard deviation. 95

4.8 The effect of localization error on the guide based beamforming and location based beamforming. The wavelength used is $\lambda = 0.33$m 96

4.9 The deployment regions corresponding to wider beams are less sensitive to localization errors. 98

4.10 The effect of non-LOS channel components, simulated using a Ricean channel, on the BF gains of different BF approaches. 98

4.11 The timing diagram of the implemented beamforming setup for $N = 3$. The guide initiates the beamforming by sending a synch. preamble. The followers then, after performing time and frequency corrections, send channel estimation preambles. The payload shown in green was designed such that each BF radio transmits individually, then all of them beamform, to enable the evaluation of the beamforming gain. 100

4.12 The magnitude of the payload of a received packet. Each of the beamforming nodes makes an individual transmission followed by a beamformed signal. The beamformed signal consists 4000 samples having constant magnitude followed by 10,000 BPSK modulated symbols. 102

4.13 The placement of the nodes for the experiment evaluation. 103

4.14 The beamforming pattern for $L_x = 0.55$, $L_y = 0.1$ emulating the experimental setup. The solid line is the average of 100 random placements and the dotted lines represent ± their standard deviation. 106

4.15 The results from the experimental evaluation show that the beamforming gains match those predicted by the beampattern in 4.14 with larger gains attained in the destination direction. 107

4.16 The pre-BF and post-BF SNR in the BF direction and opposite to it. The SNR improvement is higher in the BF direction. 109

5.1 Known classes are depicted as circles and outlier classes as squares. Classifiers would mistakenly label oultiers. Anomaly detector rejects outliers but cannot distinguish among the known classes. Openset classification classifies among known classes and rejects outliers. 119

5.2 Signal $\boldsymbol{y}$ is received by receiver $\boldsymbol{R}$. We want to determine if it was sent by an authorized transmitter in the set $\mathcal{A}$ or a new unseen transmitter. 120

5.3 High level architecture of the proposed methods. Autoencoder consists of an encoder and a decoder. The remaining ones consists of a feature extractor followed by one or more classifiers. 122

5.4 Image of orbit testbed. The participating nodes are plotted. 127

5.5 The residual, feature processing, and decision blocks used as building blocks for the neural networks considered in this work. 129

5.6 The three different architectures compared in this work. Feature extractor 2 was eventually selected. The residual block is described in Fig. 5.5. 130

5.7 We considered two OvA architectures, the first having a feature processing block for each output, and the second using a common one. The feature extractors are described in Fig. 5.6 and the feature processing and decision blocks in Fig. 5.5 . 131

5.8 The average performance of the OvA architectures from Table 5.2 are shown as a blue solid line. The error bars represent the standard deviation due to training 10 repetitions of the same network using the same data. 133

5.9 The architecture of the autoencoder. The residual block is described in Fig. 5.5 134

5.10 Box plot of the outlier detection accuracy of OvA is shown for several realizations of the sets. For each set realization, a network is trained for 10 repetitions. The center line represents the median, the box represents the first and third quartiles, and the whiskers represent the range, except for outlier points which are represented as circles. 136

5.11 Average outlier detection performance of several approaches as we change $|\mathscr{A}|$. Error bars represent the standard deviation for different realizations of the sets. 137

5.12 Average outlier detection performance of several approaches as we change $|\mathscr{A}|$. Error bars are omitted for clarity. Solid lines represent same days test, and dashed line represent different day test. 138

5.13 Average outlier detection performance of several approaches as we change $|\mathcal{K}|$. Error bars represent the standard deviation for different realizations of the sets. 139

5.14 Average outlier detection performance of several approaches as we change $|\mathcal{K}|$. Error bars are omitted for clarity. Solid lines represent same days test, and dashed line represent different day test. 140

5.15 Average outlier detection performance against the number of training days. Error bars represent the standard deviation for different set realizations. Solid lines represent same days test, and dashed line represent different day test. 142

5.16 A tree diagram summarizing the feasible architectures as a function of the dataset. The networks are ordered so that the one yielding the better performance comes first. 143

6.1 The Dual Path network consists of feature path and a signal path connected using neural networks (NN) for parameter estimation and feature extraction. A slice of an example input signal and expected outputs of the network are plotted. . . 154

6.2 The layer by layer description of the NNs in DPN. 161

6.3 Alternative DPN designs. 163

6.4 Flow graph for generating samples showing on top the input parameters and the bottom the outputs used for training. 167

6.5 Over-the-air capture setup using USRPs B205 mini. 168

6.6 The mean absolute error of DPN in parameter estimation. 171

6.7 The frequency representation of an example input signal, the signal recovered by DPN, and the ground truth. 172

6.8 The CDF of the symbol error rate (SER) calculated for each signal at SNR=10dB of different approaches. 173

6.9 The packet error rate of different approaches plotted against SNR. 174

6.10 The modulation classification of DPN, its alternatives SepDPN, and Single Path, and networks from the literature. 175

6.11 The confusion matrix of DPN at SNR of 5dB. 176

6.12 Comparison between simulation and over-the-air results. 179

6.13 Comparison of different approaches to scale DPN; dividing the signal into chunks, reusing the weights trained for $N_r = 1024$, retraining using different input lengths. 180

B.1 Summation order for the matrix $\mathbf{X}$. 191

LIST OF TABLES

2.1 Distributed Algorithms Comparison 38

3.1 Kalman Filter relations 58

3.2 Beamforming Waveform Specifications 65

3.3 BF UAV Results 69

3.4 BF Emulation over UAV Channel Trace 71

4.1 Error Rates 108

5.1 Features of Approaches 126

5.2 OvA Network Sizes 132

5.3 Network Size 135

6.1 Loss Functions 159

6.2 Dataset Description 170

6.3 Impact of each stage on Mod. Class. Accuracy 177

6.4 Training and Inference Times 178

ACKNOWLEDGMENTS

First, I would like to thank my adviser Prof. Danijela Cabric for giving me the opportunity to be part of CORES. I would not have come this far withour her continous support and guidance. I would also like to thank the members of my committee Prof. Ankur Mehta, Prof. Gregory Pottie, and Prof. Lieven Vandenberghe for their helpful comments.

I would like to thank my collaborators that helped me in my PhD: Mr. Samurdhi Karunaratne, Mr. Enes Krijestorac, Dr. Han Yan, and Dr. Chris Dick. I also want to thank Dr. Lily Yang and her team for their suggestions that helped shape my work on transmitter authorization.

I am grateful for the many wonderful people that I met during my PhD. I want to thank my labmates: Shailesh Chaudhari, Ghaith Hattab, Han Yan, Veljko Boljanovic, Enes Krijestorac, YenChin Wang, Samurdhi Karunaratne, Benjamin Domae, Aditya Wadaskar, Donar Li, Shamik Sarkar, Manali Sharma. Our discussions and the fun times we spent together helped me throughout this journey. I want to specifically thank Veljko for being a great friend, lab and apartment mate. I am glad that our journeys overlapped.

I want to also thank Enes Krijestorac, Samurdhi Karunaratne, Mahmoud Elhebeary and Alexie Pogue. I am glad for having the chance to collaborate with you on your work. Also, I want to thank Prof. Ankur Mehta for our discussions and his support.

I want to extend my gratitude to my advisers/mentors from Egypt, who helped me get this far; Prof. Noha Korany, Prof. Mina Abd-el-Malek, and Prof. Mustafa El-Nainay. I want to specifically thank Prof. Mina for his continuous support.

I would like to thank the funding agencies; the National Science Foundation (NSF) and the CONIX Research Center — one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA — for funding my research.

And finally, I want to thank my family. You supported me in every step along this journey. Thank you.

VITA

2008–2013	BSc in Electrical Engineering, Alexandria University, Alexandria, Egypt.
2014–2017	MSc in Engineering Mathematics, Alexandria University, Alexandria, Egypt.
2017-2021	Graduate Student Researcher, UCLA
2019	Summer Intern, XILINX, San Jose, CA.
2021	Summer Intern, Qualcomm, San Diego, CA.

SELECTED PUBLICATIONS

[1] S. Hanna, S, Karunaratne, and D. Cabric, "Open Set Wireless Transmitter Authorization: Deep Learning Approaches and Dataset Considerations ," in *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 1, pp. 59-72, Mar. 2021.

[2] S. Hanna, E. Krijestorac and D. Cabric, "UAV Swarm Position Optimization for High Capacity MIMO Backhaul," in *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 10, pp. 3006-3021, Jun. 2021.

[3] S. Hanna, C. Dick, and D. Cabric, "Signal Processing Based Deep Learning for Blind Symbol Decoding and Modulation Classification," to appear in *IEEE Journal on Selected Areas in Communications.*

[4] S. Hanna, E. Krijestorac, and D. Cabric, "Feedback Free Distributed Transmit Beamforming using Guided Directionality," to be submitted.

[5] S. Hanna and D. Cabric, "Distributed Transmit Beamforming: Design and Demonstration from the Lab to UAVs," submitted to *IEEE Transactions on Wireless Communications.*

[6] S. Hanna, C. Dick, and D. Cabric, "Combining Deep Learning and Linear Processing for Modulation Classification and Symbol Decoding," in GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Dec. 2020, pp. 1–7. doi: 10.1109/GLOBECOM42002.2020.9348060.

[7] S. Hanna, S. Karunaratne, and D. Cabric, Deep Learning Approaches for Open Set Wireless Transmitter Authorization, in 2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), May 2020, pp. 1–5. doi: 10.1109/SPAWC48557.2020.9154254.

[8] S. S. Hanna and D. Cabric, "Deep Learning Based Transmitter Identification using Power Amplifier Nonlinearity," in 2019 International Conference on Computing, Networking and Communications (ICNC), Feb. 2019, pp. 674–680. doi: 10.1109/ICNC.2019.8685569.

[9] S. Hanna, E. Krijestorac, H. Yan, and D. Cabric, "UAV Swarms as Amplify-and-Forward MIMO Relays," in 2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Jul. 2019, pp. 1–5. doi: 10.1109/SPAWC.2019.8815538.

[10] S. Hanna, H. Yan, and D. Cabric, "Distributed UAV Placement Optimization for Cooperative Line-of-sight MIMO Communications," in ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2019, pp. 4619–4623. doi: 10.1109/ICASSP.2019.8683875.

CHAPTER 1

Introduction

1.1 Motivation

The number of unmanned-aerial-vehicles (UAVs) is expected to increase to 2.38 million by 2025 [Fed21]. This widespread adoption of UAVs is driven by their mobility and low cost of operation, which makes them suitable for many applications [SSA19]; UAVs can be used to deliver packages, participate in search and rescue operations, wildlife monitoring, provide communications in emergency situations, etc. Many of these applications benefit from using multiple cooperating UAVs or *UAV swarms* [SBS20].

Communications is an integral part of UAV operations, whether the UAVs are used as communications enablers or as user equipment [MSB18]. As communications enabler UAVs can assist the existing infrastructure in providing coverage to users to meet fluctuating demands or to provide emergency communications in case of disasters [ENC17]. UAVs as user equipment rely on communications to accomplish their deployment task. Communications can be solely for control signals in applications like package delivery or can be for the purpose of offloading data for applications like wildlife monitoring.

UAVs as communications enablers can be used to meet the requirements of 5G and future communications systems in challenging scenarios [WXZ21]. These scenarios include events like open air festivals, protests, etc. For instance in an open air festival, a small rural area having minimal coverage is visited by tens of thousands of visitors during a short period, with capacity requirements of up to 900 Gbps/km^2 [FT13]. Other scenarios include traffic jams, which can occur unexpectedly and during which capacity demands increase. In such cases, UAVs are faster to deploy and are more economic than fixed infrastructure. But, one of the challenges of UAVs that unlike basestations, they are not connected using wires to the core network. The connection between the UAV and the core network (backhaul) needs to be done wirelessly [TO21]. This is more challenging, since the backhaul link is shared among UAVs and thus would require higher capacity than the link between users and the UAVs (access link). This drives the need for spectrally efficient approaches for backhaul communication using UAV swarms.

As for UAVs as user equipment, many applications benefit from having a UAV swarm cover a remote area [SBS20]. Examples of these applications include wildlife monitoring, search and rescue, mapping of area, etc. These applications require throughput of in the range of a few Mbps over distances in the range of kilometers [HYM16]. Hence, communications has to be done over a long distance which could exceed the communication range of a single UAV. Since the UAVs are battery powered, their flight time is limited and moving UAVs closer to the ground receiver to offload their data would waste energy. To avoid the energy waste and improve the energy efficiency, techniques to enable the UAV swarm to cooperatively transmit data instead of moving are needed.

Besides, the capacity and energy challenges, there are security considerations that pertain not only to UAVs, but to all wireless devices. The rise in the number of wireless connected devices in the era of Internet of Things (IoT) raises major security concerns in device authentication [SRG15], and poses new challenges due to unauthorized utilization of shared spectrum [LCT09]. For device authentication, traditional systems rely on cryptography, however, many IoT devices are computationally limited and energy-constrained. Thus, a new paradigm to verify the legitimacy of a signal is needed relying only on the transmitted radio-frequency (RF) signals without imposing any overheads on the transmitter [ZHL21]. As for unauthorized spectrum utilization, the first step of addressing it is understanding the nature of the received signals. This requires an approach to blindly understand a received

RF signal [PSY21]. These RF identification challenges drive the need for novel methods improving security in wireless networks.

1.2 Challenges and Objectives

1.2.1 UAV Swarms for Communications

We start by discussing the spectrum and energy challenges pertinent to communications with UAV Swarms.

Spectrum Efficient Backhaul for UAVs Multiple UAVs communicating with the same ground station will interfere with each other. Assuming that the ground station has multiple antennas, MIMO spatial multiplexing can be used to scale the capacity with the number of UAVs. However, the line-of-sight propagation between the UAVs and the ground station, typical of air-to-ground channels, is highly correlated. This correlation makes the multiplexing gains limited and degrades the spectral efficiency. Our objective is to optimize the placement of UAVs in order to attain the maximum capacity given by the single-user-bound (SUB). The optimized positions need to minimize the distance traveled by the UAVs to avoid disturbing UAVs from their deployment objective.

Energy Efficient Range Extension using Beamforming Instead of UAVs wasting their energy traveling towards the destination radio for communications, UAVs can cooperate to transmit their signals using distributed beamforming. Beamforming (BF) enables cooperating radios to extend their communication range by adjusting their signals to ensure coherent combining at a destination radio. BF relies on the BF radios having frequency synchronized carriers and phases adjusted for coherent combining. Both requirements are typically met by exchanging preambles with the destination. However, since BF aims to increase the communication range, the individually transmitted preambles are typically at low

SNR and their lengths are constrained by the channel coherence time. These noisy preambles lead to errors in frequency and phase estimation, which result in the degradation of the BF gains. Our objective is to design BF systems that take into account this degradation or eliminate it by avoiding preamble exchange with the destination entirely.

1.2.2 RF Identification

Transmitter Authorization Using imperfections in transmitters' hardware, wireless signals can be used to verify their identity in an authorization system, thus addressing one the security considerations for IoT devices without imposing any additional overhead on the transmitters. Deep learning excels in learning features from data and has been proposed for transmitter identification. However, existing works have mainly focused on classification among a closed set of transmitters. Malicious transmitters outside this closed set will be misclassified, jeopardizing the authorization system. Our objective is to formulate the problem as open set classification, which recognizes authorized transmitters and rejects unauthorized transmitters unseen during training. We want to compare the different open set approaches and study how they depend on the training dataset structure.

Blind Signal Decoding Blindly decoding a signal can assist in understanding unauthorized signals in the spectrum. To blindly decode a signal, we need to estimate its unknown transmit parameters, compensate for the wireless channel impairments, and identify the modulation type. The difficulty of jointly estimating the unknown parameters using digital signal processing (DSP) and the ease of generating training data makes deep learning a suitable approach for this problem. While deep learning can solve complex problems, DSP is interpretable and can be more computationally efficient. Our objective is design a neural network that combines the performance of deep learning and the interpretability and computational efficiency of DSP.

1.3 Contributions

1.3.1 UAV Swarms

Spectrum Efficient Backhaul for UAVs We derived a set of UAV positions that attains the maximum capacity for a uniform rectangular array ground station. Given an initial swarm placement, we formulated the problem of minimizing the distance traveled by the UAVs to reach a placement within the capacity maximizing set of positions. An offline centralized solution to the problem using block coordinate descent was developed assuming known initial positions of UAVs and was shown to converge with a bounded displacement. An online distributed algorithm was also proposed to maximize the capacity, requiring only sharing of channel estimates between neighboring UAVs. The conditions needed to guarantee its convergence and an upper bound for the traveled distance were derived. Our proposed approaches were shown to significantly increase the capacity by up to 4x at the expense of a bounded translation from the initial UAV placements. This capacity increase persists when using a massive MIMO ground station. Using numerical simulations, we showed the robustness of our approaches in a Rician channel under UAV motion disturbances.

Energy Efficient Range Extension using Beamforming To overcome the challenges associated with the low SNR preambles exchanged with the destination, we proposed two approaches; the first one consists of designing the BF system by optimizing the preambles and the number of BF radios to counter the degradation. The second approach is a novel BF technique called *Guided BF* that does not require any feedback from the destination assuming a line-of-sight channel.

Analysis and Design of BF Systems: Assuming a destination-led BF protocol and Kalman filter for frequency tracking, we optimized the number of BF radios and the preamble lengths to achieve reliable BF gain. To do that, we characterized the relations between the BF gains distribution, the channel coherence time, and design parameters like the SNR, preamble lengths, and the number of radios. The proposed relations were verified using simulations and via experiments using software-defined radios in a lab and on UAVs.

Guided BF: We proposed and demonstrated a feedback free distributed beamforming approach that leverages the radio's mobility and coarse location information in a dominant line-of-sight channel. In the proposed approach, one radio acts as a guide and moves to point the beam of the remaining radios towards the destination. We specified the radios' position requirements and verified their relation to the combined signal at the destination using simulations. A proof of concept demo was implemented using software defined radios, showing up to 9dB SNR improvement in the beamforming direction just by relying on the coarse placement of four radios.

1.3.2 RF Identification

Transmitter Authorization We formulated the problem of recognizing authorized transmitters and rejecting new transmitters as open set recognition and anomaly detection. We considered approaches based on one and several binary classifiers, multiclass classifiers, and signal reconstruction. We studied how these approaches scale with the required number of authorized transmitters. We proposed using a known set of unauthorized transmitters to assist the training and study its impact. The evaluation procedure takes into consideration that some transmitters might be more similar than others and nuances these effects. The authorization's robustness against temporal changes in fingerprints is also evaluated as a function of the approach and the dataset structure. When using 10 authorized and 50 known unauthorized WiFi transmitters from a publicly accessible testbed, we were able to achieve an outlier detection accuracy of 98% on the same day test set and 80% on the different day test set.

Blind Signal Decoding We proposed the dual path network (DPN) that combines DSP and deep learning. It consists of a signal path of DSP operations that recover the signal, and a feature path of neural networks that estimate the unknown transmit parameters. By interconnecting the paths over several recovery stages, later stages benefit from the recovered signals and reuse all the previously extracted features. The proposed design was demonstrated to provide 5% improvement in modulation classification compared to alternative designs lacking either feature sharing or access to recovered signals. The estimation results of DPN along with its blind decoding performance were shown to outperform a blind signal processing algorithm for BPSK and QPSK on a simulated dataset. An over-the-air software-defined-radio capture was used to verify DPN results at high SNRs. DPN design can process variable length inputs and is shown to outperform relying on fixed length inputs with prediction averaging on longer signals by up to 15% in modulation classification.

1.4 Thesis Organization

The rest of the dissertation is organized as follows

- Chapter 2: We present our approaches to optimize the positions of the UAVs swarm to maximize the capacity with a multiantenna ground station. This chapter revises our previous publication [HKC21b].

- Chapter 3: We present our analysis to characterize the impact of noisy preamble exchange with the destination on BF. Using this analysis, we propose approaches to optimize the number of BF radios and the preamble lengths to realize a required destination SNR with a required probability. This chapter revises our previous publication [HC21].

- Chapter 4: We present guided BF as a feedback-free destination approach along with the analysis of its position requirements. This chapter revises our previous publication [HKC21a].

- Chapter 5: We present our study of open set classification approaches for transmitter authorization. This chapter revises our previous publication [HKC20a].

- Chapter 6: We present the dual path network for blind signal decoding and modulation classification. This chapter revises our previous publication [HDC21].
- Chapter 7: We summarize the research contributions and outline future research directions.

CHAPTER 2

Spectrum Efficient Backhaul for UAVs

A swarm of cooperating UAVs communicating with a distant multiantenna ground station can leverage MIMO spatial multiplexing to scale the capacity. Due to the line-of-sight propagation between the swarm and the ground station, the MIMO channel is highly correlated, leading to limited multiplexing gains. In this paper, we optimize the UAV positions to attain the maximum MIMO capacity given by the single user bound. An infinite set of UAV placements that attains the capacity bound is first derived. Given an initial swarm placement, we formulate the problem of minimizing the distance traveled by the UAVs to reach a placement within the capacity maximizing set of positions. An offline centralized solution to the problem using block coordinate descent is developed assuming known initial positions of UAVs. We also propose an online distributed algorithm, where the UAVs iteratively adjust their positions to maximize the capacity. Our proposed approaches are shown to significantly increase the capacity at the expense of a bounded translation from the initial UAV placements. This capacity increase persists when using a massive MIMO ground station. Using numerical simulations, we show the robustness of our approaches in a Rician channel under UAV motion disturbances. This chapter revises our previous publication [HKC21b].

2.1 Introduction

Driven by their low cost, high mobility, and ease of deployment, UAVs are used in many applications including delivery of goods, surveillance, precision agriculture, and civil infrastructure inspection [SSA19]. For wireless communications, UAVs have been proposed as aerial basestations (BS) [AEY18], data aggregators [HC17], and for many other applications [MSB19]. The key advantage of UAV deployments over ground antennas is their mobility [MSB19]. By changing the position of UAVs, the wireless communications channel can be designed to be a line-of-sight channel for a given deployment [CG20]. However, unlike ground BSs which rely on wired communications for backhaul, UAV backhaul has to be wireless, which makes it one of the challenges of UAV BSs [VSD18]. In scenarios requiring high throughput like open-air festivals [FT13, Sec. 15], multiple UAVs need to be deployed due to the lack of nearby fixed infrastructure. In such multi-UAV deployments, the UAVs are positioned close to the ground users, yielding a high SNR access link. In contrast, the backhaul link with a distant ground station has lower SNR and is shared among multiple UAVs thus creating a bottleneck for communications.

Using a multiantenna ground station (GS) makes the backhaul air-to-ground channel between the swarm of UAVs and the GS a MIMO channel. By leveraging spatial multiplexing in this channel, the backhaul capacity can significantly be improved. However, the air-to-ground channel between a UAV swarm and the GS is typically dominated by line-of-sight (LOS) propagation [KCZ18]. Additionally, the GS antenna array is typically implemented as a uniform linear or rectangular array with limited dimensions. These factors can lead to highly correlated low-rank channels that can limit the MIMO capacity gains [TV05, Sec. 7.2.3]. Optimizing the UAV swarm positions can reduce the channel correlation and improve capacity. But, this optimization for backhaul capacity should have a minimum impact on the UAV deployment application.

The concept of positioning antennas to optimize the MIMO channel was first proposed for LOS communications between fixed GSs. By optimizing the spacing of uniform antenna arrays, the maximum capacity given by the single user bound can be achieved [KSH07]. Once synchronized [YHB19], a cooperative UAV swarm can be viewed as a virtual antenna array. But unlike GSs, which are limited in size and typically consist of uniformly spaced antenna elements, UAV swarms can achieve large apertures and are not bound to any geometry.

This flexibility allows a swarm to maximize the backhaul capacity at the cost of moving the UAVs from their initial positions [HYC19]. While it is possible to optimize for access and backhaul link simultaneously [FIG19], this limits the scope of the problem to UAV BSs. Communications intensive UAV applications tolerating some displacement are numerous and can include some video surveillance and remote sensing deployments [SSA19].

In this chapter, for a communications intensive application, we optimize the placements of UAVs within a swarm to attain the maximum MIMO capacity over a backhaul link with a distant uniform rectangular GS. Using the GS geometry and the LOS channel, we derive a set of UAV positions that attain the maximum capacity given by the single user bound. Among the capacity maximizing position set, UAV positions closer to their initial placements pose the least disturbance to the deployment application and are desirable. Based on that, we formulate the problem of finding the positions within the capacity maximizing set that minimize the traveled distance from the initial positions. Two methods are considered to solve this problem; the first one is a centralized offline solution assuming a prior knowledge of the UAVs' initial positions. The second approach is a distributed online approach, where the UAVs iteratively adjust their positions to maximize the backhaul capacity. Our contributions are

- We show that, for a uniform rectangular array ground station, the set of UAV placements maximizing the MIMO capacity is infinite. We derive a subset of placements within this set in the far-field where the distance from the swarm to the ground station is much larger than the size of the swarm.
- Given the UAVs' initial placements, we formulate the problem of finding UAV positions within the capacity maximizing set that minimize the distance traveled. A centralized suboptimal solution to this problem using block coordinate descent is developed and shown to require a bounded traveled distance per UAV.
- An online distributed algorithm is proposed to maximize the capacity, requiring only

sharing of channel estimates between neighboring UAVs. The conditions needed to guarantee its convergence and an upper bound for the traveled distance are derived.

2.2 Related Work

Many of the existing works have focused on optimizing the UAV positions to improve only the access link [AEY18, LCW19, KHC19], thus implicitly assuming an ideal backhaul link. However, as the number of UAVs increase, wireless backhaul becomes challenging [VSD18]. Some works have proposed different approaches for UAV backhaul which we briefly discuss. We also discuss MIMO in UAV networks; some works has envisioned massive MIMO BS serving UAVs, others have proposed multiantenna UAV BSs. Swarms of single antenna UAVs were also proposed communicating with either ground users or GSs.

Approaches for UAV Backhaul To address the challenges associated with UAV wireless backhaul, several approaches were proposed in the literature. Some works have proposed using mmWave backhaul [BPB19, TMG20], however, the high path loss at these frequencies makes them unsuitable for long links. Mechanical antenna steering was proposed for UAVs backhaul at microwave frequencies [POP18], but mechanical steering limits beamforming to one direction and is inherently slower. Integrated access and backhaul (IAB) links optimization was proposed in [YNF19, FIG18, FIG19], where UAVs relay data using the same frequency bands in both links. In [YNF19], the locations, power allocation, and frequency assignment of a swarm of UAVs were optimized to reduce the transmit power in an IAB network using a single antenna GS. The UAVs' frequency assignment is chosen to minimize the interference between the backhaul and access links, which share the same bands. In [FIG18], using IAB, exhaustive search is used to determine the UAV locations, precoder design, power allocations, which maximize the sum-rate to the ground users using a massive MIMO GS for backhaul. However, due to the prohibitive complexity of exhaustive search, only one UAV was considered. A less complex centralized solution to the same problem using a fixed point method and particle swarm optimization was proposed in [FIG19]. Results have shown that increasing the number of UAVs increases the interference in the access link and reduces the network performance. In IAB networks, the main challenge is to minimize the interference in the access link and between both access and backhaul links since the same frequencies are reused. In this chapter, our focus is on maximizing the swarm MIMO backhaul capacity for any application tolerating displacement from a given initial placement. In the case of UAV BSs, the access link is assumed to use a different frequency band than the backhaul link.

UAVs Served by Massive MIMO BS In [GGG18, GGL19], massive MIMO cellular BSs were proposed for cellular-connected UAVs and are shown to improve the data rates. Unlike our work, the UAVs are treated as user equipment with no control over their positions. Deep reinforcement learning was also proposed for navigation of a single UAV communicating with a massive MIMO BS in [HYW20]. The impact of having a swarm of UAVs on the capacity was not considered.

MIMO using multiantenna UAVs Using UAVs carrying antenna arrays was proposed in many works in the literature. Some works have considered optimizing the positions and the beamforming vectors to improve the ground users' SNR [RIG16] or minimize the transmit power [XSN18]. The trajectory of multiantenna UAVs serving ground users under an uncertain environment was optimized in [XSN19]. However, for UAVs carrying an antenna array, the UAV size and maximum payload for safe flight significantly constrains the antenna array aperture compared to a swarm of single antenna UAVs, thus limiting the multiplexing gains with a distant GS.

MIMO using UAV Swarms Several existing works have proposed UAV swarms leveraging MIMO. These works have either considered the access link with ground users or the backhaul link with a GS. For the access link, in [MSB17], the motion and beamforming weights of linearly arranged UAV swarm were optimized to serve users one at a time. To serve multiple ground users simultaneously, UAVs were proposed as remote radio heads in a coordinated multipoint (CoMP) system and were optimized to improve the capacity in [LZZ18] and physical layer security in [WFC19]. Access link air to ground channel is different from the backhaul channel; in the former, the ground users are typically spread out, closer to the UAVs, and are more likely to get obstructed unlike a distant GS with a dominant LOS channel in the latter.

To improve the capacity in LOS channels, before the interest in UAV networks, the designs of traditional uniform antenna arrays like linear and rectangular were optimized in [BOO07a, BOO07b]. Based on these designs, uniform geometries were proposed for UAV swarms communicating with GS in [SMG13, SEH19, PHN20]. However, these rigid geometrical placements might conflict with positions required by application-driven deployments. In [CDL18], for a given UAV deployment, the massive MIMO GS was optimized to maximize the ergodic LOS channel capacity, thus not benefiting from the UAVs mobility and requiring GS redesign per deployment. In [IQM13], the authors proposed randomly placing the UAVs within a specified area for optimal MIMO capacity. Due to the randomness of this approach, the capacity improvements are probabilistic and a large capacity increase requires having more UAVs than GS antennas. In [HYC19], two iterative distributed algorithms were proposed to optimize the LOS MIMO channel capacity of a UAV swarm; namely gradient descent and brute force, which are described later in this work. However, no convergence proofs nor travel upper bounds were developed for the proposed algorithms. In this work, we leverage the UAV mobility to optimize the backhaul link capacity. Our proposed approaches minimize the UAVs' displacements from given initial positions. Upper bounds on the distance traveled and convergence proofs are derived for our proposed approaches.

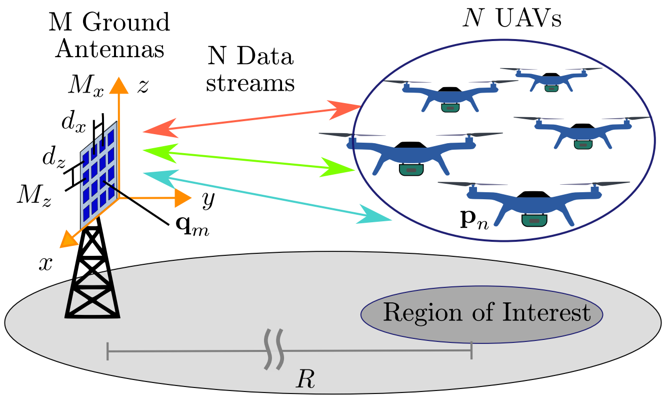


Figure 2.1: In the proposed system model, the UAV swarm is in the far-field of a uniform rectangular antenna array GS.

2.3 System Model

A swarm of N single antenna UAVs is communicating with a multiantenna GS having M antennas in either uplink or downlink. Each UAV has its own data to transmit or to receive. To avoid high inter-swarm communications overhead, the MIMO processing (precoding for downlink and combining for uplink) is done at the GS and each UAV sends or receives only its own stream. Since the maximum number of simultaneous streams possible is $\min(M, N)$, at most $N = M$ UAVs can benefit from the MIMO gains. Hence, throughout this work, we assume that $N \leq M$. If $N > M$, time multiplexing or another technique has to be used, which is not considered in this work.

The channel between the swarm and the ground antennas can be modeled as a Rician channel [KCZ18] and is denoted by $\mathbf{H} \in \mathbb{C}^{M\times N}$ defined as follows [BOO07a]

$$\mathbf{H} = \sqrt{\frac{K}{K+1}}\mathbf{H}_{\text{LOS}} + \sqrt{\frac{1}{K+1}}\mathbf{H}_{\text{NLOS}} \tag{2.1}$$

where K is the K-factor, $\mathbf{H}_{\text{LOS}}$ is the line-of-sight (LOS) component and $\mathbf{H}_{\text{NLOS}}$ is the non-line-of-sight (NLOS) component. The elements of $\mathbf{H}_{\text{NLOS}}$ are independently drawn from a zero-mean circularly symmetric complex Gaussian distribution with variance equal to $\frac{\|\mathbf{H}_{\text{LOS}}\|_F}{MN}$,where $\|.\|_F$ is the Frobenius norm. The normalized LOS channel $\overline{\mathbf{H}}_{\text{LOS}}$ and LOS channel accounting for path loss $\mathbf{H}_{\text{LOS}}$ are given by

$$\left[\overline{\mathbf{H}}_{\text{LOS}}\right]_{m,n} = \exp\left(\frac{-j2\pi\|\mathbf{p}_n - \mathbf{q}_m\|}{\lambda}\right) \tag{2.2}$$

$$[\mathbf{H}_{\text{LOS}}]_{m,n} = \frac{\lambda}{4\pi\|\mathbf{p}_n - \mathbf{q}_m\|}\left[\overline{\mathbf{H}}_{\text{LOS}}\right]_{m,n} \tag{2.3}$$

where λ is the wavelength and $\|\mathbf{p}_n - \mathbf{q}_m\|$ is the distance between the n-th UAV and the m-th GS antenna. The element of the m-th row, n-th column of a matrix $\mathbf{X}$ is denoted by $[\mathbf{X}]_{m,n}$. The n-th UAV is located at position $\mathbf{p}_n \in \mathbb{R}^3$ defined as $\mathbf{p}_n = [x_n, y_n, z_n]$. Similarly, the m-th ground antenna is located at position $\mathbf{q}_m \in \mathbb{R}^3$. The matrix $\mathbf{P} \in \mathbb{R}^{3\times N}$ contains all the UAV positions such that $\mathbf{P} = [\mathbf{p}_0^T, \cdots, \mathbf{p}_{N-1}^T]^T$ with $()^T$ denoting the transpose.

The GS is assumed to be arranged as a $M_x \times M_z$ uniform rectangular array where $M = M_x \times M_z$. Without loss of generality, the GS is assumed to be placed in the x-z plane with $\mathbf{q}_0 = [0, 0, 0]^T$ used as a coordinate reference. The spacing between the antennas in the x and z directions is given by d_x and d_z respectively. Hence, $\mathbf{q}_m = [i_m d_x, 0, j_m d_z]^T$, where i_m and j_m are the antennas indices in x and z directions respectively and satisfy $m = i_m M_z + j_m$. The average separation along the y-axis between the UAVs and the GS is given by $R = \frac{\sum_{n=0}^{N-1} y_n}{N}$. The system model is illustrated in Fig. 2.1.

We assume that the UAV swarm operates within a bounded region in the far-field and that the GS is pointed toward the swarm such that $\mathbf{P} \in \mathcal{F}$ where $\mathcal{F}$ is a position set defined as

$$\begin{aligned}\mathcal{F} = \Big\{\mathbf{P} \mid\ & y_n \gg |y_n - y_m|,\ y_n \gg |x_n|,\ y_n \gg |z_n|, \\ & y_n \gg M_x d_x,\ y_n \gg M_z d_z,\ n, m \in \{0, \cdots, N-1\}\Big\}\end{aligned} \tag{2.4}$$

In this work, the swarm is always assumed to be within the set $\mathcal{F}$. Using these assumptions, the magnitude of all the elements of the LOS channel matrix can be approximated to be constant and equal to $\frac{\lambda}{4\pi R}$ such that

$$\mathbf{H}_{\text{LOS}} \approx \frac{\lambda}{4\pi R}\overline{\mathbf{H}}_{\text{LOS}} \tag{2.5}$$

The single user bound defines the maximum achievable capacity and is given by [NLM14]

$$\begin{aligned} C &= \log\det\left(\mathbf{I} + \rho\mathbf{H}^H\mathbf{H}\right) \\ &\leq \sum_{n=0}^{N-1} \log\left(1 + \rho\|\mathbf{h}_n^{[c]}\|^2\right) = C_{\max} \end{aligned} \tag{2.6}$$

where ρ is the signal to noise ratio and $\mathbf{h}_n^{[c]}$ is the n-th column of $\mathbf{H}$ and $()^H$ denote the Hermitian transpose. The maximum capacity given by the single user bound $C_{\max}$ can be attained when the columns of the channel matrix are mutually orthogonal [NLM14]. When the bound is reached, the N different data streams do not interfere with each other. Using the magnitude approximation, the channel maximizing the capacity has to realize

$$\mathbb{E}\{\mathbf{H}^H\mathbf{H}\} = M\left(\frac{\lambda}{4\pi R}\right)^2\mathbf{I} \tag{2.7}$$

where $\mathbf{I}$ is the $N \times N$ identity matrix and $\mathbb{E}\{\}$ denotes the expectation with respect to the channel NLOS component. Equation (2.7) defines the condition on the channel matrix to attain the single-user bound capacity $C_{\max}$. However, to formulate an optimization problem over the UAV positions, we need to relate the UAV positions with (2.7). Our objective is to define a set of UAV positions that realize equation (2.7) in order to maximize the capacity. Later, this set is used in the problem formulation.

2.4 Set of Capacity Maximizing Positions

For an M antenna GS, the maximum number of UAVs capable of using spatial multiplexing is M. In this section, we aim to define the M positions for these UAVs that maximize the capacity. If the number of UAVs, N, is less than M, the UAVs can be placed to occupy only N of these M positions. So without loss of generality, we consider the matrix $\mathbf{H}$ to be a square matrix of size $M \times M$ and (2.7) can be rewritten in terms of the rows (instead of the

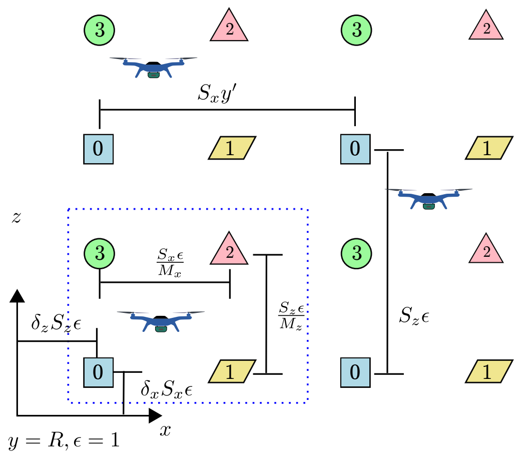


Figure 2.2: An illustration of the set $\mathcal{P}$ for a swarm placed on the same plane at $y = R$ for a 2×2 URA. The numbered colored shapes identify different positions. Each UAV needs to occupy a different position to maximize capacity.

columns) of $\mathbf{H}$ as

$$\mathbb{E}\{\mathbf{h}_l^H \mathbf{h}_k\} = \begin{cases} M \left(\frac{\lambda}{4\pi R}\right)^2 & l = k \\ 0 & l \neq k \end{cases} \tag{2.8}$$

where $\mathbf{h}_k$ is the k-th column of the transposed channel $\mathbf{H}^T$.

Since $\mathbf{H}$ is a Rician channel, due to the NLOS component $\mathbf{H}_{\text{NLOS}}$, its elements are random variables. In our derivations, we consider the expected value of the columns inner product given by $\mathbb{E}\{\mathbf{h}_l^H \mathbf{h}_k\}$. Given our assumption that the elements $\mathbf{H}_{\text{NLOS}}$ are independent and zero mean, $\mathbb{E}\{\mathbf{h}_l^H \mathbf{h}_k\} = \mathbf{h}_l^{H[\text{LOS}]} \mathbf{h}_k^{[\text{LOS}]}$ for $l \neq k$ where $\mathbf{h}_k^{[\text{LOS}]}$ is the k-th column of $\mathbf{H}_{\text{LOS}}^T$. Hence, using the LOS component in our derivations is equivalent to using the average over the Rician channel. To simplify the notations, we drop the expectation operator and we consider the channel matrix to be normalized such that $\mathbf{H} = \overline{\mathbf{H}}_{\text{LOS}}$.

We start by relating the right-hand side of (2.8) to the UAV positions as follows

$$\mathbf{h}_l^H \mathbf{h}_k = \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi}{\lambda}(\|\mathbf{p}_n - \mathbf{q}_l\| - \|\mathbf{p}_n - \mathbf{q}_k\|)\right) \tag{2.9}$$

To simplify the exponent of (2.9), we use our far-field assumption to approximate the distance

$$\begin{aligned}\|\mathbf{p}_n - \mathbf{q}_l\| &= \sqrt{(x_n - i_l d_x)^2 + (y_n)^2 + (z_n - j_l d_z)^2} \\ &\approx y_n \left(1 + \frac{1}{2}\left(\frac{x_n - i_l d_x}{y_n}\right) + \frac{1}{2}\left(\frac{z_n - j_l d_z}{y_n}\right)\right)\end{aligned} \tag{2.10}$$

where i_l and j_l are the antenna indices along x and z respectively and both satisfy $l = i_l M_z + j_l$. The approximation uses the first-order Taylor approximation of the square root assuming that the UAVs are within $\mathcal{F}$. Hence, we can rewrite

$$\mathbf{h}_l^H \mathbf{h}_k \approx \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi}{y_n \lambda}\left((-i_l + i_k)d_x x_n + (-j_l + j_k)d_z z_n\right)\right) \tag{2.11}$$

Note that from this result, we can see that the rate of change of the phase with respect to x_n and z_n is proportional to $1/y_n$, while the rate of change with respect to y_n is proportional to $1/y_n^2$. Hence, to incur a given phase difference by moving the UAV in the y-direction requires a much larger motion than by moving in the x or z directions. This observation will be used later when optimizing the UAV positions.

We start by describing one value of swarm positions $\mathbf{P}$ that make the channel orthogonal in Lemma 1. After determining these positions, we investigate the changes in positions that retain orthogonality in Lemmas 2 and 3. The set of positions maximizing capacity is obtained by combining these Lemmas in Theorem 1.

Lemma 1. *A uniform rectangular arrangement of UAVs within $\mathcal{F}$ having positions given by $x_n = i_n \frac{\lambda y_n}{M_x d_x}$ in the x-direction and $z_n = j_n \frac{\lambda y_n}{M_z d_z}$ in the z-direction realize channel orthogonality, where $i_n \in \{0, \cdots, M_x - 1\}$ and $j_n \in \{0, \cdots, M_z - 1\}$ such $n = i_n M_z + j_n$.*

Proof. See Appendix A. □

Lemma 1 determines one value of swarm positions $\mathbf{P}$, which makes the channel orthogonal. Now, we consider a few changes to positions that do not affect orthogonality.

Lemma 2. *Shifting the UAV swarm in the x-z plane such that each UAV is shifted proportionally to its y separation from the GS does not affect channel orthogonality as long as the swarm remains within $\mathcal{F}$.*

Proof. Let the set of UAV positions $\mathbf{P}$ realize $\mathbf{h}_l^H \mathbf{h}_k = 0$ for all $l \neq k$. Let $\mathbf{h}'_l$ and $\mathbf{h}'_k$ be columns of the channel after shifting UAV n by $\delta_x y_n$ for all n in the x-direction and $\delta_z y_n$ in the z-direction.

$$\begin{aligned} \mathbf{h}_l'^H \mathbf{h}'_k &= \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi}{y_n \lambda}\left((-i_l + i_k)d_x\left(x_n + \delta_x y_n\right)\right.\right. \\ &\qquad \left.\left. +(-j_l + j_k)d_z\left(z_n + \delta_z y_n\right)\right)\right) \\ &= \exp\left(\frac{-j2\pi}{\lambda}((-i_l + i_k)\delta_x + (-j_l + j_k)\delta_z)\right)\mathbf{h}_l^H \mathbf{h}_k \\ &= 0 \end{aligned} \tag{2.12}$$

Hence, UAV shifts scaled with respect to their y coordinate do not affect the orthogonality of the MIMO channel. □

Lemma 3. *Translation of individual UAVs such that UAV n is translated by an integer multiple of $\frac{\lambda y_n}{d_x}$ in x-direction and/or $\frac{\lambda y_n}{d_z}$ in z-direction does not affect the channel orthogonality as long as the swarm remains within $\mathcal{F}$.*

Proof. Let the channel have columns $\mathbf{h}_l$ and $\mathbf{h}_k$ for some GS antennas l and k. Let all UAVs be translated independent of each other, such that UAV n is shifted by $f_n \frac{\lambda y_n}{d_x}$ and $g_n \frac{\lambda y_n}{d_z}$ in the x and z-direction respectively for some UAV specific integers f_n and g_n. Let $\mathbf{h}'_l$ and $\mathbf{h}'_k$ be columns of the channel after the shifting. Calculating the inner product of these columns, we get

$$\begin{aligned} \mathbf{h}_l'^H \mathbf{h}'_k &= \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi}{y_n \lambda}\left((-i_l + i_k)d_x\left(x_n + f_n \frac{\lambda y_n}{d_x}\right)\right.\right. \\ &\qquad \left.\left. +(-j_l + j_k)d_z\left(z_n + g_n \frac{\lambda y_n}{d_z}\right)\right)\right) \\ &= \exp\left(-j2\pi((-i_l + i_k)f_n + (-j_l + j_k)g_n)\right)\mathbf{h}_l^H \mathbf{h}_k \\ &= \mathbf{h}_l^H \mathbf{h}_k \end{aligned} \tag{2.13}$$

□

In addition to scaled swarm translations from Lemmas 2 and individual UAV jumps from Lemma 3, it easy to see that permuting the positions of the swarm does not affect orthogonality. We define a set of positions orthogonalizing the channel by combining the three previous lemmas as follows.

Theorem 1. *Given a URA GS with M antennas, the set $\mathcal{P} \subset \mathcal{F}$ is a set containing placements, $\mathbf{P} \in \mathbb{R}^{3\times M}$, of M UAVs, which realize the channel orthogonality condition given by (2.8). The set $\mathcal{P}$ can be described given the environment constants $S_x = \frac{\lambda R}{d_x}$ and $S_z = \frac{\lambda R}{d_z}$ as follows*

$$\begin{aligned}\mathcal{P} = \{\mathbf{P} : \mathbf{P} = \mathbf{T}_\Pi \widetilde{\mathbf{P}},\ \widetilde{\mathbf{P}} \in \mathcal{F}, x_n = \left[\widetilde{\mathbf{P}}\right]_{0,n}, \epsilon_n = \frac{\left[\widetilde{\mathbf{P}}\right]_{1,n}}{R}, z_n = \left[\widetilde{\mathbf{P}}\right]_{2,n} \\ x_n = i_n \frac{S_x \epsilon_n}{M_x} + f_n S_x \epsilon_n + \delta_x S_x \epsilon_n, z_n = j_n \frac{S_z \epsilon_n}{M_z} + g_n S_z \epsilon_n + \delta_z S_z \epsilon_n\ , n = i_n M_z + j_n, \\ i_n \in \{0, \cdots, M_x - 1\}, j_n \in \{0, \cdots, M_z - 1\}, \mathbf{T}_\Pi \in \Pi^M, \delta_x, \delta_z \in \mathbb{R}, f_n, g_n \in \mathbb{Z} \forall n\} \quad (2.14)\end{aligned}$$

where Π^M is the set of all $M \times M$ permutation matrices.

Proof. This can be proved by the application of Lemmas 2, 3, and applying permutations on the results obtained by Lemma 1. We renamed $y_n = \epsilon_n R$ and the shifts from Lemma 2 to $\delta_x S_x \epsilon_n$ and $\delta_z S_z \epsilon_n$ for convenience of notation. □

If the swarm positions are within $\mathcal{P}$ as defined in Theorem 1, the orthogonality condition (2.7) is satisfied and $C_{\max}$ is attained. By using this set, an optimization problem over UAV positions can be formulated. We show an example of the set $\mathcal{P}$ in Fig. 2.2 for a simple scenario having $M = 4$ with $M_x = M_z = 2$ and $N = 3$ UAVs on the same x-z plane. The positions defined according to Lemma 1 are labeled 0 to 3 in different colored shapes inside the blue dotted square. According to Lemma 3, The $S_x\epsilon$ and the $S_z\epsilon$ jumps along x and z respectively are also within $\mathcal{P}$. This entire grid can be shifted by δ_x and δ_z according to Lemma 2. Orthogonality is attained by assigning the 3 UAVs to any of the 4 positions. After defining the set of UAV positions, $\mathcal{P}$, orthogonalizing the channel and attaining $C_{\max}$, we discuss the problem formulation.

2.5 Placement Optimization Problem Formulation

There are several ways to mathematically formulate the problem of optimizing swarm positions to attain the maximum capacity. The most intuitive way is to optimize over the positions with the capacity as the objective. However, since any swarm positions $\mathbf{P} \in \mathcal{P}$ can achieve $C_{\max}$ and no unique solution exists, using this formulation, the obtained positions can be far from the UAVs' initial positions and hence would cause unnecessary disturbance to the deployment application. Given that the considered deployment application prioritizes communications with no hard constraints on UAV displacement, defining a constraint on the distance traveled by the UAVs is not straightforward; if the constraint is too tight, a suboptimal capacity below $C_{\max}$ will be achieved, if the constraint is too loose, the solution might lead to unnecessary travel by the UAVs. The minimal distance to attain $C_{\max}$ differs from one deployment environment to the other and hence can not be used as a constraint. Instead, we make minimizing the distance traveled our objective. To guarantee that the maximum capacity is attained, we constrain the optimized UAV positions to be within the set $\mathcal{P}$, which attains $C_{\max}$. This formulation attains the maximum capacity with the least traveled distance.

A mathematical formulation of the problem is as follows; Given N UAVs with initial positions $\{\overline{\mathbf{p}}_0, \overline{\mathbf{p}}_1, \cdots, \overline{\mathbf{p}}_{N-1}\}$, where $\overline{\mathbf{p}}_n = [\overline{x}_n, \overline{y}_n, \overline{z}_n]^T$. These initial positions are assumed to be determined by the deployment application. Our objective is to find the nearest UAV positions which belong to $\mathcal{P}$. This problem can be formulated as

$$\begin{aligned} \underset{\{\mathbf{p}_m\},\{b_{m,n}\}}{\text{minimize}} \quad & \sum_{m=1}^{M}\sum_{n=1}^{N} b_{m,n}\|\overline{\mathbf{p}}_n - \mathbf{p}_m\| \\ \text{subject to} \quad & [\mathbf{p}_0, \cdots, \mathbf{p}_{M-1}] \in \mathcal{P} \\ & \sum_{n=0}^{N} b_{m,n} \leq 1 \quad \forall m, \qquad \sum_{m=0}^{M} b_{m,n} = 1 \quad \forall n \\ & b_{m,n} \in \{0,1\} \quad \forall m, n \end{aligned} \tag{2.15}$$

The binary variable $b_{m,n}$ is used to assign each of the N UAVs (indexed using n) to one of the M positions within $\mathcal{P}$ (indexed using m). This problem formulation does not make any assumptions about whether the UAVs are transmitters or receivers and does not make any assumptions about the transmitter and receiver processing. Later in Section 2.8, we consider an uplink scenario and derive the optimal linear precoders and combiners.

Using the definition of $\mathcal{P}$ from Theorem 1 in (2.15), the problem can be rewritten as

$$\begin{aligned}
\underset{\substack{\{x_m\},\{y_m\},\{z_m\},\\ \delta_x,\delta_z,\{\epsilon_n\},\\ \{f_m\},\{g_m\},\{b_{m,n}\}}}{\text{minimize}} \quad & \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} b_{m,n}\sqrt{(x_m-\overline{x}_n)^2+(y_m-\overline{y}_n)^2+(z_m-\overline{z}_n)^2} \qquad (2.16)\\
\text{subject to} \quad & x_m = i_m\frac{S_x\epsilon_m}{M_x} + f_m S_x\epsilon_m + \delta_x S_x \epsilon_m, \quad \forall m\\
& z_m = j_m\frac{S_z\epsilon_m}{M_z} + g_m S_z\epsilon_m + \delta_z S_z \epsilon_m, \quad \forall m\\
& f_n, g_n \in \mathbb{Z} \quad \forall n\\
& \sum_{n=0}^{N} b_{m,n} \le 1 \quad \forall m, \quad \sum_{m=0}^{M} b_{m,n} = 1 \quad \forall n\\
& b_{m,n} \in \{0,1\} \quad \forall m,n\\
& -\frac{1}{2} \le \delta_x \le \frac{1}{2}, -\frac{1}{2} \le \delta_z \le \frac{1}{2}\\
& y_m = R\epsilon_m
\end{aligned}$$

where $i_m \in \{0,\cdots,M_x\}$ and $j_m \in \{0,\cdots,M_z\}$ and both satisfy $m = i_m M_z + j_m$. In the current form, this problem is a non-convex mixed-integer problem that is not tractable.

To solve this problem, we consider both an offline centralized solution in Section 2.6 and an online distributed algorithm in Section 2.7. The centralized solution assumes the initial positions are known apriori and aims to relax and solve (2.16). In the case where the UAVs are already deployed without prior knowledge of their placements, a distributed online algorithm where the UAVs iteratively improve their positions is also proposed.

2.6 Centralized Offline Solution

For the centralized solution, we start by relaxing problem (2.16) to make it more tractable prior to deriving its solution. An upper bound for the distance traveled and the time complexity are also discussed.

2.6.1 Problem Relaxation

We start by eliminating the y-translation variable.

Eliminating y-translation As discussed previously y_m has a small effect on the phase unlike a change in x_m and z_m. A UAV has to travel a much larger distance along the y direction compared to the x or z direction to incur a phase change. So to simplify, we relax the problem by not optimizing over the y-translation, i.e, setting $y_m = \overline{y}_m$ for all UAVs. Hence, we only optimize over x_m and z_m. Given this simplification, the y term in the objective is equal to zero and ϵ_m becomes a constant for all m. The problem can be reformulated as

$$\underset{\substack{\{x'_{m,n}\},\{z'_{m,n}\},\\ \{f_n\},\{g_n\}\\ \{b_{m,n}\}\\ \delta_x,\delta_z}}{\text{minimize}} \quad \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} b_{m,n}\sqrt{(x'_{m,n})^2 + (z'_{m,n})^2} \tag{2.17}$$

subject to

$$x'_{m,n} = i_m\frac{S_x\epsilon_n}{M_x} + f_m S_x\epsilon_n + \delta_x S_x\epsilon_n - \overline{x}_n, \forall n, m \tag{2.18}$$

$$z'_{m,n} = j_m\frac{S_z\epsilon_n}{M_z} + g_m S_z\epsilon_n + \delta_z S_x\epsilon_n - \overline{z}_n, \forall n, m \tag{2.19}$$

$$f_n, g_n \in \mathbb{Z} \quad \forall n \tag{2.20}$$

$$b_{m,n} \in \{0, 1\} \quad \forall n, m \tag{2.21}$$

$$\sum_{n=0}^{N} b_{m,n} \leq 1 \quad \forall m, \quad \sum_{m=0}^{M} b_{m,n} = 1 \quad \forall n \tag{2.22}$$

$$-\frac{1}{2} \leq \delta_x \leq \frac{1}{2}, \quad -\frac{1}{2} \leq \delta_z \leq \frac{1}{2} \tag{2.23}$$

The value of $x'_{m,n}$ is the x-translation difference between the initial position of UAV n and the optimal position m and $z'_{m,n}$ is similarly defined for the z-translation. The integers f_m and g_m define multiple possible solutions, however, we know that the optimal one is closer to the starting positions. We use this intuition to narrow the solution space.

Narrowing Solution Space According to Lemma 3, any integer value of f_m and g_m can achieve orthogonality. However, the values of these variables that minimize the translation are expected to be the ones closest to the starting positions of the UAVs. To simplify the problem using this intuition, we start by rewriting the initial positions of the UAVs $\overline{x}_n$ and $\overline{z}_n$ as a function of our environment constants as follows $\overline{x}_n = c'_n S_x \epsilon_n$ where c'_n is the constant satisfying this relation. By substituting in (2.18), we get

$$\begin{aligned} \frac{i_m}{M_x} S_x \epsilon_n + f_m S_x \epsilon_n - \overline{x}_n &= S_x \epsilon_n \left(f_m + \frac{i_m}{M_x} - c'_n \right) \\ &= S_x \epsilon_n \left(f_m - (f'_n + r'_n) \right) \\ &= \tilde{x}_{m,n} + f_{m,n} S_x \epsilon_n \end{aligned} \tag{2.24}$$

where $f'_n = \left\lfloor \frac{i_m}{M_x} - c'_n \right\rfloor$ is an integer obtained by the floor operation and $r'_n = \left(\frac{i_m}{M_x} - c'_n \right) - f'_n$ has a magnitude smaller than one. The distance $\tilde{x}_{m,n}$ is defined as $\tilde{x}_{m,n} = S_x \epsilon_n r'_n$ and satisfies $0 \le \tilde{x}_n < S_x \epsilon_n$. We define $f_{m,n} = f_m - f'_n$, which redefines the integer translations to use the initial positions of the UAV n as a starting point. Hence, we can rewrite (2.18) as

$$x'_{m,n} = \tilde{x}_{m,n} + f_{m,n} S_x \epsilon_n + \delta_x S_x \epsilon_n \tag{2.25}$$

similarly for the z-direction, we get

$$z'_{m,n} = \tilde{z}_{m,n} + g_{m,n} S_z \epsilon_n + \delta_z S_z \epsilon_n \tag{2.26}$$

Proposition 1. *The value of $f_{m,n}$ and $g_{m,n}$ that minimizes (2.17) is within the set $\{-1, 0\}$ and is given by*

$$\hat{f}_{m,n} = \begin{cases} 0 & -\frac{1}{2} S_x \epsilon_n \le \tilde{x}_{m,n} + \delta_x S_x \epsilon_n < \frac{1}{2} S_x \epsilon_n \\ -1 & \frac{1}{2} S_x \epsilon_n \le \tilde{x}_{m,n} + \delta_x S_x \epsilon_n \le \frac{3}{2} S_x \epsilon_n \end{cases} \tag{2.27}$$

Proof. See Appendix A.2. □

A similar result can be proved for $g_{m,n}$. Hence, among all the values of integer translations from the initial UAV positions, $f_{m,n}$ and $g_{m,n}$, we only need to consider the values of the nearest translations from the UAV's initial locations.

The relaxed problem The problem is thus simplified to

$$\begin{aligned}
\underset{\substack{\{x'_n\},\{z'_n\},\\ \{f_{m,n}\},\{g_{m,n}\}\\ \{b_{m,n}\}\\ \delta_x,\delta_z}}{\text{minimize}} \quad & \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} b_{m,n}\sqrt{(x'_{m,n})^2+(z'_{m,n})^2} \\
\text{subject to} \quad & x'_{m,n} = \tilde{x}_{m,n} + f_{m,n}S_x\epsilon_n + \delta_x S_x \epsilon_n, \quad \forall n, m \\
& z'_{m,n} = \tilde{z}_{m,n} + g_{m,n}S_z\epsilon_n + \delta_z S_z \epsilon_n, \quad \forall n, m \\
& f_{m,n}, g_{m,n} \in \{-1, 0\} \quad \forall n \\
& b_{m,n} \in \{0,1\} \quad \forall n, m \\
& \sum_{n=0}^{N} b_{m,n} \le 1 \quad \forall m, \quad \sum_{m=0}^{M} b_{m,n} = 1 \quad \forall n \\
& -\frac{1}{2} \le \delta_x \le \frac{1}{2}, \quad -\frac{1}{2} \le \delta_z \le \frac{1}{2}
\end{aligned} \tag{2.28}$$

2.6.2 Problem Solution

The problem (2.28) still remains a non-convex mixed-integer problem. The difficulty in solving the problem is because the variables δ_x and δ_z are common to the entire swarm. We show that by for a given value of some variables the problem becomes tractable and we use that fact to solve the problem.

Solution given δ_x and δ_z For a given value of δ_x and δ_z, the problem becomes tractable and it can be solved as follows: first, we minimize over $f_{m,n}$ and $g_{m,n}$ using (2.27) since δ_x and δ_z are given. Once these values have been calculated, the square root term in the objective becomes a constant. What remains is to solve for $b_{m,n}$, which becomes the following integer linear program

$$\begin{aligned} \underset{\{b_{m,n}\}}{\text{minimize}} \quad & \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} b_{m,n}\sqrt{(x'_{m,n})^2 + (z'_{m,n})^2} \\ \text{subject to} \quad & b_{m,n} \in \{0,1\} \quad \forall n,m \\ & \sum_{n=0}^{N} b_{m,n} = 1 \quad \forall m, \quad \sum_{m=0}^{M} b_{m,n} \le 1 \quad \forall n \end{aligned} \tag{2.29}$$

This integer program can be shown to be equivalent to its real relaxation. This problem is indeed an assignment problem that can be solved in polynomial time using the Hungarian algorithm [Kuh55].

Solution given an assignment Again considering (2.28), the challenge in solving for δ_x and δ_z is that they are multiplied by integer variables $b_{m,n}$. Given an assignment defining the values of $b_{m,n}$, the problem (2.28) becomes the following convex problem

$$\begin{aligned} \underset{\substack{\{x'_n\},\{z'_n\},\\ \delta_x,\delta_z}}{\text{minimize}} \quad & \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} b_{m,n}\sqrt{(x'_{m,n})^2 + (z'_{m,n})^2} \\ \text{subject to} \quad & x'_{m,n} = \tilde{x}_{m,n} + \hat{f}_{m,n} S_x \epsilon_n + \delta_x S_x \epsilon_n, \quad \forall n,m \\ & z'_{m,n} = \tilde{z}_{m,n} + \hat{g}_{m,n} S_z \epsilon_n + \delta_z S_z \epsilon_n, \quad \forall n,m \\ & -\frac{1}{2} \le \delta_x \le \frac{1}{2}, \quad -\frac{1}{2} \le \delta_z \le \frac{1}{2} \end{aligned} \tag{2.30}$$

which can be solved using a convex solver like CVXPY [DB16].

Complete Solution We have shown that for a given δ_x and δ_z, (2.28) gets simplified to (2.29) which can be optimally solved. We also have shown that for a given $b_{m,n}$, we get (2.30) which can also be optimally solved. Hence, to solve (2.28), we use block coordinate descent. We optimize over each set of variables in an alternating manner, until the solution stops changing. Since both (2.29) and (2.30) are solved to optimality, Problem (2.28) is guaranteed to converge to a stationary point [Ber99, Prop. 2.7.1]. After solving, we obtain the optimal $\hat{\delta}_x$, $\hat{\delta}_z$, and $\hat{b}_{m,n}$ for all m and n, along with $\hat{f}_{m,n}$ and $\hat{g}_{m,n}$. We need to substitute back to get the UAV positions. From $\hat{b}_{m,n}$, the index of the placement assigned to the n-th UAV $\hat{m}_n$ is given by $\hat{m}_n = \underset{m}{\text{argmax}} \hat{b}_{m,n}$. The assigned position is then calculated using

$$x_n = \tilde{x}_{\hat{m}_n,n} + \hat{f}_{\hat{m}_n,n} S_x \epsilon_n + \hat{\delta}_x S_x \epsilon_n + \overline{x}_n \tag{2.31}$$

similarly for the z position

$$z_n = \tilde{z}_{\hat{m}_n,n} + \hat{g}_{\hat{m}_n,n} S_z \epsilon_n + \hat{\delta}_z S_z \epsilon_n + \overline{z}_n \tag{2.32}$$

The centralized solution algorithm is summarized in Algorithm 1. Hence, using block coordinate descent, we obtained a suboptimal solution of (2.28), which is a relaxation of (2.15).

2.6.3 Upper Bound and Time Complexity

The upper bound for the translation of UAVs is derived in Proposition 2

Proposition 2. *The maximum absolute translation of UAV n is upper bounded by* $\frac{\sqrt{S_x^2+S_z^2}}{2}\epsilon_n$.

Proof. See Appendix A.3. □

As for the algorithm computational complexity, it is the sum of the solution complexities of solving (2.29) and (2.30) times the number of iterations. For the number of iterations, convergence typically occurred within fewer than five iterations, which can be enforced as a maximum number of iterations. The Hungarian algorithm used to solve (2.29) has complexity $O(M^3)$ [EK72]. For problem (2.30), CVXPY [DB16] uses ECOS second-order cone programming solver [DCB13], which relies on an interior-point algorithm based on Mehrorta predictor-corrector method. In general, the interior points algorithms' complexity depends on the number of variables [PW00]. Since problem (2.30) has only two variables (δ_x and δ_z) regardless of the problem size, the solution time is dominated by the Hungarian algorithm. By limiting the iterations to five, the complexity of Algorithm 1 is approximately $O(M^3)$.

```
Algorithm 1: Centralized Solution
input  : The initial positions of the UAV swarm $\{\overline{\mathbf{p}}_0, \overline{\mathbf{p}}_1, \cdots, \overline{\mathbf{p}}_{N-1}\}$, The parameters
         of the GS $M_x, M_z, d_x, d_z$. The wavelength $\lambda$.
output: The optimized UAV positions.
current_obj = $\infty$;
previous_obj = 0;
Initialize $\delta_x = 0$ and $\delta_z = 0$ ;
while current_obj-previous_obj¿1e-5 do
    previous_obj = current_obj ;
    Solve (2.29) for $\{b_{m,n}\}$ using $\delta_x$ and $\delta_z$;
    Solve (2.30) for $\delta_x$ and $\delta_z$ using $\{b_{m,n}\}$ and assign the objective value to
      current_obj ;
end
Calculate the position of UAVs, using (2.31) and (2.32)
```

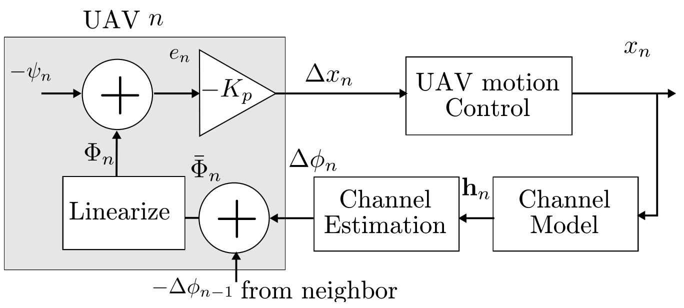


Figure 2.3: The closed loop feedback system in UAV n where $n \in \{1, \cdots, N-1\}$. The calculations made on the UAV are highlighted in gray. The value $\Delta\phi_{n-1}$ is obtained from the previous neighbor.

2.7 Distributed Online Solution

In the case where the UAV positions are not known before deployment, we develop an iterative distributed approach to be applied within the swarm in realtime. This approach uses channel estimates instead of positions for optimization and it attempts to minimize the inter-swarm communication overhead. In this approach, the UAVs agree on a formation, in which each UAV designates another UAV as its neighbor along each axis according to a criterion discussed later. For a ULA GS, this formation is linear, and for a URA it is a rectangular grid. Each UAV estimates its channel and shares the estimates with its neighbors using an ideal control side channel. Using the neighbor channel measurements, the UAVs calculate an error signal. This error signal drives a closed loop feedback system, which decides the magnitude and direction of its motion. In this approach, each UAV moves based on information from its neighbor, as if each UAV exerts a force on its neighbor. Hence, we refer to this approach as Force Field (FF). We start by deriving the fundamentals of this approach and show its convergence on a ULA GS aligned to the x-axis. Then, we discuss how it is applied to a URA GS. In the end, we discuss how the agreement on the formation is performed.

2.7.1 Fundamentals of Force Field

The key idea behind FF is that for any optimal positions $\mathbf{P} \in \mathcal{P}$, the equivalent $M \times M$ channel matrix $\mathbf{H}$ can be shown to be a scaled and permuted DFT matrix [HK03]. The phase difference between successive elements of the l-th column of the DFT matrix is $\frac{2\pi}{M}$. Hence, the phase difference at UAV n due to two consecutive ground antennas l and $l+1$ is given by

$$\Delta\phi_n = \phi_{l,n} - \phi_{l+1,n} = 2\pi f'_n + n\left(\frac{2\pi}{M_x}\right). \tag{2.33}$$

where $\phi_{l,n} = \angle[\mathbf{H}]_{l,n}$ is the phase of the channel between GS antenna l and UAV n for some integer f'_n. If (2.33) is satisfied for all UAVs $n \in \{0, \cdots N-1\}$, and all GS antennas $l \in \{0, \cdots M-2\}$, the channel $\mathbf{H}$ is an orthogonal scaled DFT matrix and the capacity is maximized. However, (2.33) determines the position of each UAV solely on its index n regardless of the remaining UAVs. This might lead to larger distance traveled since each UAV does not consider its neighbors' positions or channels. Additionally, if a UAV suffered from an external disturbance like wind, the remaining UAVs will not adapt. To make each UAV adapt to its neighbors, we reformulate (2.33) to

$$\Delta\phi_n - \Delta\phi_{n-1} = 2\pi f_n + \left(\frac{2\pi}{M_x}\right) \tag{2.34}$$

for any integer f_n. Here we assume that the UAV formation has been established and UAV $n-1$ shares its channel estimates with its neighbor UAV n. If all UAVs realize (2.34), $\mathbf{H}$ becomes orthogonal. We define the difference between the right and the left sides of (2.34) as an error signal as follows

$$e_n = \Delta\phi_n - \Delta\phi_{n-1} - \psi_n \tag{2.35}$$

where $\psi_n = 2\pi f_n + \left(\frac{2\pi}{M_x}\right)$ is the target phase difference for some integer f_n. The objective of each UAV is to move such that this error signal is equal to zero. We define $\overline{\Phi}_n = \Delta\phi_n - \Delta\phi_{n-1}$ as the measured state of our system. When each UAV moves, this state changes and our goal is to make it equal to ψ_n. Using the distance approximation (2.10) based on the far-field assumption, we get

$$\overline{\Phi}_n = (\Delta\phi_n - \Delta\phi_{n-1})\%(2\pi) \tag{2.36}$$

$$\approx \left(2\pi\left(\frac{x_n}{\epsilon_n} - \frac{x_{n-1}}{\epsilon_{n-1}}\right)\frac{1}{S_x}\right)\%(2\pi) \tag{2.37}$$

where the environment constant $S_x = \frac{\lambda R}{d_x}$ (as previously defined) and % denotes the modulus operator. The modulus operation accounts for phase wraps. This relation would have been linear if it was not for the phase ambiguity. Using phase measurements only, we can not tell whether the UAV did not change position or moved to create a 2π phase difference. However, this ambiguity can be mitigated and the phase unwrapped by limiting the translation that each UAV performs at each step as discussed later. After unwrapping the phase, the system becomes a linear system. Based on the error signal, each UAV can change its position to orthogonalize the channel using a closed loop feedback system.

2.7.2 Force Field Algorithm

The proposed feedback system is run iteratively in all UAVs. In iteration k, all the UAVs move simultaneously, except the first UAV which is used as an anchor and does not move. This approach is described as follows: Each UAV estimates its channel and calculates $\Delta\phi_n[k]$ at iteration k. It shares this value with its direct neighbor in the formation, so that UAV n knows $\Delta\phi_{n-1}[k]$ from its neighbor. Each UAV calculates the measured state $\overline{\Phi}_n[k]$ and estimates the unwrapped phase state Φ_n using

$$\Phi_n[k] = \begin{cases} \overline{\Phi}_n[k] + 2\pi \left\lfloor \frac{\Phi_n[k-1]}{2\pi} \right\rfloor & c = 0 \\ \overline{\Phi}_n[k] + 2\pi + 2\pi \left\lfloor \frac{\Phi_n[k-1]}{2\pi} \right\rfloor & c = 1 \\ \overline{\Phi}_n[k] - 2\pi + 2\pi \left\lfloor \frac{\Phi_n[k-1]}{2\pi} \right\rfloor & c = 2 \end{cases} \tag{2.38}$$

where

$$c = \operatorname{argmin}\{|\overline{\Phi}_n[k] - \overline{\Phi}[k-1]|, |\overline{\Phi}_n[k] + 2\pi - \overline{\Phi}[k-1]|, |\overline{\Phi}_n[k] - 2\pi - \overline{\Phi}[k-1]|\} \tag{2.39}$$

It is easy to verify that phase wrap errors will not occur as long the phase transition between iterations is less than π. After linearizing the state, each UAV calculates an error $e_n[k]$ using (2.35). Based on this error signal, it changes its position such that

$$x_n[k+1] = x_n[k] - K_p e_n^{[x]}[k] \tag{2.40}$$

where K_p is a constant creating a proportional controller. The first UAV in the formation having $n = 0$ is used as an anchor, i.e, it does not change positions. The closed loop feedback system at UAV n is shown in Fig. 2.3, with the calculations that run in the UAV highlighted in gray. The input to our approach for UAV n is its phase estimates along with those of UAV $n-1$. The output is the motion given by $\Delta x_n[k] = x_n[k+1] - x_n[k]$. After a predefined time sufficient for calculations in all UAVs, the output $\Delta x_n[k]$ is fed to the UAV motion control system which navigates the UAV. After the swarm settles, these steps can be repeated for a fixed number of iterations until $|\Delta x_n[k]|$ becomes small for all UAVs. Note that since the system was linearized, instead of the proportional controller in (2.40), more sophisticated controllers like PID can speed up the convergence [GK17].

We now discuss the convergence of this approach for a single UAV and then generalize to the entire swarm.

Lemma 4. *The error in UAV n is guaranteed to converge to zero given that its previous neighbor, UAV $n-1$, is fixed, if $0 < K_p < \frac{\epsilon_n S_x}{4\pi}$.*

Proof. See Appendix A.3. □

Theorem 2. *The error of all UAVs is guaranteed to converge to zero if $0 < K_p < \frac{\min\limits_{n}(\epsilon_n) S_x}{4\pi}$.*

Proof. UAV 0 acts as an anchor and does not move. Hence, the error of UAV 1, according to Lemma 4 is guaranteed to converge to zero if $0 < K_p < \frac{\epsilon_0 S_x}{4\pi}$. Once, it converges according to Lemma 4 the error of UAV 2 is also guaranteed to converge to zero if $0 < K_p < \frac{\epsilon_1 S_x}{4\pi}$. Similarly, we can show that all N UAVs will converge if $0 < K_p < \frac{\min\limits_{n}(\epsilon_n) S_x}{4\pi}$. □

Note that although the UAVs closer to the fixed UAV converge first, all the non-converged UAVs move simultaneously. As a consequence of simultaneous motion, oscillations might occur; a UAV might move in some direction in an iteration and in the other direction in the following iteration because its previous neighbor has moved. A smaller value of K_p will reduce the magnitude of oscillations.

2.7.3 Force Field URA Extension

Next, we discuss the extension from the ULA GS in the x-direction to a URA in the x-z plane. For a URA, each UAV needs to meet the orthogonality criterion (2.34) in both the x and z directions. The condition along the x-axis is

$$\Delta\phi_n^{[x]} - \Delta\phi_{n-1}^{[x]} = \psi_n^{[x]} \tag{2.41}$$

where the superscript $[x]$ is to denote x-direction, $\psi_n^{[x]}$ is the x phase objective, and $\Delta\phi_n^{[x]} = \phi_{n,i_m M_z + j_m} - \phi_{n,(i_m+1)M_z+j_m}$ where $i_m M_z + j_m$ and $(i_m + 1)M_z + j_m$ are the indices of two consecutive GS antennas along the x-direction. Similar definitions exist for the z-direction using phase calculated for two consecutive GS antennas along the z-direction: $\Delta\phi_n^{[z]} = \phi_{n,i_m M_z + j_m} - \phi_{n,i_m M_z + j_m + 1}$. To realize (2.41) and its z equivalent, FF is extended to apply the same procedures for a ULA along both directions. Hence, each UAV needs to designate two neighbors, one for each direction. This makes the final FF formation a grid. This grid consists of M_x lines applying linear FF along z direction and M_z lines applying it along the x-direction. The anchor node that does not move in that case is a corner node having both grid indices $i_n = j_n = 0$. We note that for a ULA in the x-direction the set $\mathcal{P}$ is unconstrained in the z-direction. Unlike the ULA, for the URA case, to retain orthogonality over the entire swarm, the UAVs that form a line in the x-direction, need to have the same phase with respect to the z-direction and vice versa. To accomplish that a small modification is made; the first line of the grid in the x-direction (having indices satisfying $n\% M_z = 0$ where % denotes the modulus operator) applies FF along the z-direction to have a phase difference along z equal to zero. The phase objective along the z direction $\psi_n^{[z]}$ for UAV n in (2.35) realizing this condition is

$$\psi_n^{[z]} = \begin{cases} 0 & n\% M_z = 0 \\ 2\pi/M_z & \text{otherwise} \end{cases} \tag{2.42}$$

A similar relation can be derived for the phase objective along x. Since, for a URA, the same FF feedback system is applied along multiple lines with a minor modification, the same convergence proofs apply.

2.7.4 Initializing Formations

Last, we describe how the formations are established. As shown in Theorem 2, the convergence only depends on each UAV picking a node as a neighbor along each axis, such that all the UAVs create a grid formation. The method of choosing the neighbor does not affect whether or not convergence will occur, however, it affects the distance that each UAV will travel to orthogonalize the channel. Our proposed approach relies on UAVs creating the formations based on an initial channel estimate that is shared globally among the swarm.

After sharing the channel, all UAVs pick their closest neighbor based on phase relative to the x-direction and then relative to the z direction. Given that the measured phase states along x and z directions are defined as $\overline{\Phi}_N^{[x]} = \Delta\phi_n^{[x]} - \Delta\phi_{n-1}^{[x]}$ and $\overline{\Phi}_N^{[z]} = \Delta\phi_n^{[z]} - \Delta\phi_{n-1}^{[z]}$ respectively. The assignment is accomplished in two stages, first, we sort the state along x such that $\overline{\Phi}_0^{[x]} \leq \overline{\Phi}_1^{[x]} \leq \cdots \leq \overline{\Phi}_{N-1}^{[x]}$. Then, each M_z UAVs are divided into a group and sorted such that the m-th group satisfies $\overline{\Phi}_{mM_z}^{[z]} \leq \overline{\Phi}_{mM_z+1}^{[z]} \leq \cdots \leq \overline{\Phi}_{(m+1)M_z-1}^{[z]}$. This assignment guarantees that the phase along any line of UAVs in the grid is increasing.

The entire force field algorithm is summarized in Algorithm 2. The **forall** construct is used to indicate that all UAVs act in parallel. For simplicity, we consider using a fixed number of iterations K_c. More adaptive stopping criteria can easily be developed based on the value of the error or the SINR. Since at convergence the interference among data streams is eliminated, the MIMO SINR is equal to the SNR when a single UAV is communicating with the GS. By setting the target SINR below the SNR, we can sacrifice the achievable capacity in favor of less distance traveled by the UAVs. Next, we find an upper bound for the distance traveled.

Proposition 3. *The distance traveled by UAV n when using Force Field is upper bounded by* $\left(\sqrt{S_x^2 + S_z^2}\right)(\max\{\epsilon_0, \epsilon_n\})$

Proof. For the m-th line in the grid formed by the UAVs along the x direction, UAVs are ordered such that

$$-\pi \leq \overline{\Phi}_m^{[x]} \leq \overline{\Phi}_{M_z+m}^{[x]} \leq \overline{\Phi}_{2M_z+m}^{[x]} \leq \cdots \leq \overline{\Phi}_{(M_x-1)M_z+m}^{[x]} \leq \pi \tag{2.43}$$

The first UAV is used as an anchor and it does not move. Each UAV is pushing its neighbor to realize a phase difference of $\frac{2\pi}{M_x}$. Since, the formation guarantees that the UAVs are increasing in phase, the worst-case scenario is when all UAVs start at exactly at the same phase. In that case, the UAV having index n will have to travel to create a phase difference of $n\frac{2\pi}{M_x}$ from the start UAV. Using (2.37), this is equivalent to having $\left(\frac{x_n}{\epsilon_n} - \frac{x_0}{\epsilon_0}\right) = S_x$. From which, $|x_n - x_0| \leq S_x(\max\{\epsilon_0, \epsilon_n\})$. A similar argument can be made for the z-direction. Combining both constraints, we get that the distance traveled by UAV n is upper bounded by $\left(\sqrt{S_x^2 + S_z^2}\right)(\max\{\epsilon_0, \epsilon_n\})$. □

We notice that the traveled distance upper-bound for Force Field is higher than the centralized solution upper-bound from Proposition 2. We also expect that the centralized approach would require less distance traveled than FF for several reasons; First, the centralized approach assumes the knowledge of the UAV initial positions, which define the problem. On the other hand, FF only uses only channel information, from which the positions can not be recovered. Second, compared to the centralized solution, FF does not optimize the displacement of the entire swarm (from Lemma 2) and just uses the first UAV as an anchor. Having a fixed UAV is crucial to guarantee the convergence as shown in Theorem 2. Third, FF assigns the UAVs to the positions in a simple way based on sorting the phases to avoid running a complicated assignment procedure in all the UAVs.

2.7.5 Time Complexity of Force Field

Since FF is a distributed algorithm, we discuss the complexity from the perspective of one UAV. In the initialization stage, each UAV has $2M$ phase measurements from the entire swarm along x and z directions. Each UAV sorts the phases along x across all UAV and along z as groups of size M_z. Assuming the merge sort algorithm is used, the initialization complexity is given by $\mathcal{O}(M \log M + M_x M_z \log M_z)$. After initialization, each UAV interacts only with one neighbor in the x-direction and one neighbor in the z-direction, regardless of the swarm size making the complexity be a function of only the number of iterations $\mathcal{O}(K_c)$. The fact that beyond initialization FF complexity is independent of the swarm size makes

Algorithm 2: Force Field Algorithm

```
input  : $M_x, M_z$
output: Swarm positioned to maximize capacity.
All UAVs estimates channels and share it ;
forall UAV $n = 0$ to $N - 1$ do
    Sort phase estimates to identify neighbors;
    Calculate phase objective using (2.42) along x and z;
end
for iterations k = 1 to $K_c$ do
    forall UAV $n = 0$ to $N - 1$ do
        Esitmate channel and share with neighbors;
        Calculate state (2.36) and linearize (2.38) in x and z;
        Calculate error using (2.35) along x and z ;
        Wait sufficiently for other UAVs calculations;
        Move in x and z according to (2.40);
        Wait sufficiently for other UAVs to move;
    end
end
```

it scalable.

2.7.6 Comparison with Existing Distributed Algorithms

We briefly compare FF to Gradient Descent (GD) and Brute Force (BF) which were both proposed in [HYC19]. GD and BF are both iterative algorithms inspired by numerical optimization algorithm; gradient descent, and steepest descent respectively. GD relies on knowledge of the UAV positions and global channel knowledge within the swarm to calculate the gradient of the capacity with respect to positions. In each iteration, in a sequential man-

Table 2.1: Distributed Algorithms Comparison

Aspect	Force Field	Gradient Descent	Brute Force
Channel Estimations	K_c	NK_c	$6NK_c$
Inter-swarm Comm.	Neighbors	Swarm	Swarm
Convergence Proof	Yes	No	No
Distance Upper Bound	Yes	No	No

ner, all UAVs estimate the channel and one UAV moves in the gradient descent direction. BF also relies on global channel knowledge. In a BF iteration, a UAV takes 6 steps in each of the 6 orthogonal directions. For each direction, the channel is estimated and the orthogonality of the channel is evaluated. The UAV retains the position that improved the channel orthogonality. No upper bounds (UB) on distance traveled nor convergence guarantees were derived for BG and GD in [HYC19]. Since GD and BF are based on numerical methods applied to a non-convex objective, it is not easy to analyze their convergence. Unlike BF and GD, in a FF iteration, each UAV only requires knowledge of the channel from its direct neighbors reducing inter-swarm communications overhead. Also, in an FF iteration, all UAVs move simultaneously, thus requiring fewer channel estimates. The comparison between the algorithms is summarized in Table 2.1.

2.8 Optimizing a Linear Uplink Scenario

So far we discussed algorithms that optimize UAV positions to maximize the channel capacity, which are applicable in the uplink and downlink scenarios regardless of the transmitter and receiver processing. Now, for an uplink scenario with UAVs as transmitters, we consider the joint optimization of the channel $\mathbf{H}$ (through the UAV positions) and the linear precoders $\mathbf{V} \in \mathbb{C}^{N \times N}$ and combiners $\mathbf{W} \in \mathbb{C}^{M \times M}$. We can define the following joint optimization problem

$$\underset{\mathbf{H},\mathbf{W},\mathbf{V}}{\text{maximize}} \left\{ \log\det\left(\mathbf{I} + \frac{1}{N_0 N_f} \left(\mathbf{W}^H \mathbf{H} \mathbf{V} \right)^H \mathbf{W}^H \mathbf{H} \mathbf{V} \right) \right\} \tag{2.44}$$

where N_0 is the noise power spectral density and N_f is the receiver noise figure, given that $\mathbf{H}$ is a channel matrix defined according to our system model. Each UAV is assumed to be carrying a radio with maximum transmitted power P_T. Since each UAV transmits an independent data stream, $\mathbf{V}$ is constrained to be a diagonal matrix. The columns of the combining matrix $\mathbf{W}$ are assumed to be normalized.

If we define the equivalent channel $\mathbf{H}_{eq} = \mathbf{W}^H \mathbf{H} \mathbf{V}$, according to the upper bound (2.6), the maximum occurs when $\mathbf{H}_{eq}$ is orthogonal for a given SNR. Maximizing the SNR for an orthogonal $\mathbf{H}_{eq}$ solves (2.44). Using our proposed position optimization algorithms, the channel $\mathbf{H}$ can be made orthogonal. For orthogonal $\mathbf{H}$, the matched filter combiner given by $\mathbf{W} = \frac{\mathbf{H}}{\|\mathbf{H}\|_F / M}$ makes $\mathbf{W}^H \mathbf{H}$ a scaled identity matrix and hence orthogonal. The optimal diagonal precoding, in this case, is to use the maximum power $\mathbf{V} = \sqrt{P_T}\mathbf{I}$ to maximize the SNR[1]. Hence, $\mathbf{H}_{eq}$ is an orthogonal matrix maximizing the SNR and thus solves the joint optimization. Thus, we have derived the optimal linear precoders and combiners for an uplink scenario.

2.9 Numerical Evaluation

In this section, we evaluate the performance of the proposed algorithms using numerical simulations. The capacity improvements of position optimization are first evaluated along with their robustness to randomness due to the channel and UAV motion. Then, the convergence of Force Fields is evaluated under ideal and practical conditions along with other UAV

[1]Note that for iterative algorithms, before convergence, $\mathbf{H}$ is not orthogonal and the proposed precoders and combiners are not necessarily optimal

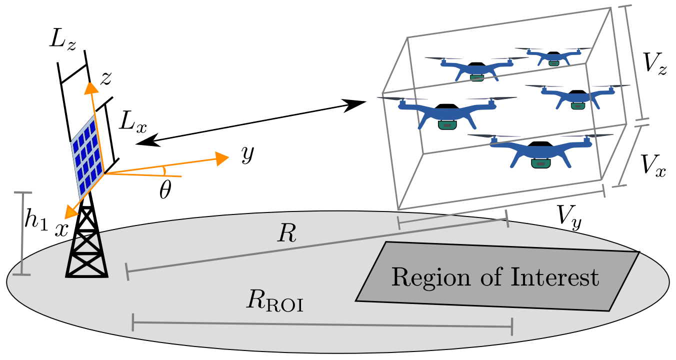


Figure 2.4: In the simulation setup, the UAVs are initialized in a rectangular area. The GS is tilted towards the swarm.

positioning algorithms. The distance traveled per UAV for different swarm positions is then considered and compared to the derived upper bounds. Lastly, we evaluate the impact of position optimization on the capacity as we move to the massive MIMO regime with $M \gg N$.

2.9.1 Simulation Setup

We consider the simulation setup shown in Fig. 2.4. The GS consists of a URA having aperture $L_x = d_x M_x = 6$m and $L_z = d_z M_z = 6$m operating at a frequency of 5GHz. The large GS aperture reduces the distance traveled as shown in the derived upper bounds. However, an extremely large aperture is not practical. Unless otherwise stated, we use $M_x = 6$ and $M_z = 2$ making $d_x = 1$m and $d_z = 3$m. The GS is placed at a height $h_1 = 10m$, which is assumed to be higher than any surrounding buildings making a LOS path exist between the GS and swarm [CG20]. For UAVs deployed in a remote area, the GS can be adjusted to guarantee this condition. To account for non line-of-sight (NLOS) propagation paths, the channel is modeled as a Rician channel. The center of the region of interest is at a distance of $R_{\text{ROI}} = 2$km from the UAV swarm. For simplicity, the initial positions are randomly distributed in a rectangular parallelepiped having sides $V_x = 10$, $V_y = 300$, and $V_z = 300$. The elevation angle of the ground antenna used is $\theta = 0.043$rad making the average height of the swarm approximately 100m.

We consider the uplink scenario, where the UAVs are the transmitters. Channel estimation errors, when considered, are modeled using $\mathbf{H}_{\text{est}} = \mathbf{H} + \tilde{\mathbf{H}}$, where $\mathbf{H}_{\text{est}}$ is the estimated channel, and $\tilde{\mathbf{H}}$ is the channel estimation error. The estimation error is modeled as a matrix with independent complex Gaussian elements with zero mean and variance $\frac{1}{1+\text{SNR } T_\tau}$ where $T_\tau = 10$ is the number of training symbols [HH03]. The UAV motion errors, when considered, are modeled as an independent random Gaussian vector of size 3 having zero mean and a diagonal covariance matrix with a magnitude of 1m. This motion error vector is added to the positions of the UAVs before channel estimation [HYC19].

Each UAV has a transmit power $P_T = 10$dBm and the bandwidth used is assumed to be 1MHz [ZZ17]. The noise power spectral density used is $N_0 = -174$dBm/Hz and the receiver has a noise figure N_F of 3dB, making the noise power equal to -111 dBm. We use the sum rate obtained when using linear minimum mean square error (LMMSE) combiner at the GS as a metric [TV05, 8.3.3]. The LMMSE combining vector w_n for UAV n is calculated using $\mathbf{w}_n = \left(N_0 N_F \mathbf{I} + \sum_{i=0, i\neq n}^{N} \mathbf{h}_i^{[est]} \mathbf{h}_i^{[est]H}\right)^{-1} \mathbf{h}_n^{[est]}$, where $\mathbf{h}_n^{[est]}$ is the n-th column of the estimated channel. The signal-to-interference-and-noise ratio (SINR) of the n-th stream is given by $\text{SINR}_n = \frac{P_T |\mathbf{w}_n^H \mathbf{h}_n^{[c]}|^2}{N_0 N_F + P_T \sum_{i=0, i\neq n}^{n} |\mathbf{w}_n^H \mathbf{h}_i^{[c]}|^2}$. Using the SINR of each stream, the sum rate is calculated using $\text{SR} = \sum_{n=0}^{N-1} \log\left(1 + \text{SINR}_n\right)$

For comparison, we consider relying on the randomness of the initial UAV positions referred to as “Init”. This is similar to what was proposed in [IQM13], although we do not optimize the deployment region. We also consider positioning the UAVs using the technique proposed for traditional planar uniform-rectangular arrays (URAs) [BOO07b]. We also consider BF and GD from [HYC19]. Although URAs were first proposed for fixed antenna arrays, they still can be used to maximize the capacity and along with uniform linear arrangements they have been proposed for UAVs [SMG13, MSB17, CDL18, SEH19].

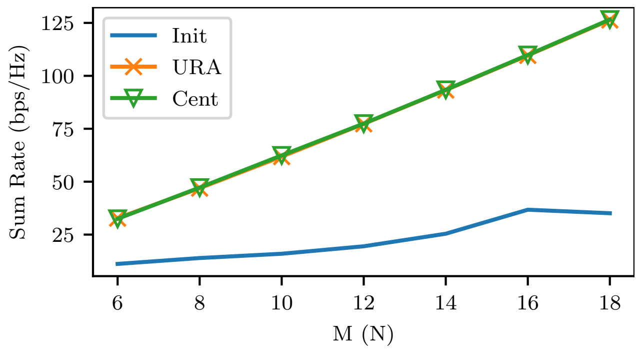
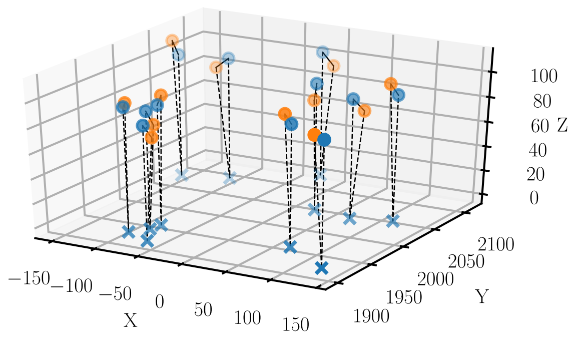


Figure 2.5: As we increase the number of UAVs and GS antennas, Cent gives an equal sum rate to URA, which is higher than relying on the initial positions.

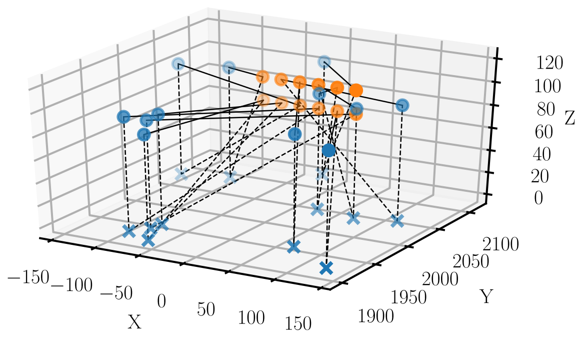
2.9.2 Performance Gains of Position Optimization

We start by demonstrating the performance gains that can be attained by optimizing the UAV swarm. We first consider the case of the swarm and GS having an equal number of antennas $M = N$ and we vary M_x. This is shown in Fig. 2.5. As the M_x increases, the sum rate of the optimized approaches (Cent and URA) increase linearly. This is expected from an optimized MIMO channel. However, this improvement comes at the cost of moving the UAVs from their initial positions. Unlike placing the UAV in a URA, our proposed approach minimizes the distance traveled. An example of a realization of random placement with $M_x = 6$ ($N = 12$) is shown in Fig. 2.6. The initial placements of the UAVs are shown in blue and is assumed to be above the points of interest shown as crosses at $z = 0$. For URA shown in Fig. 2.6b, UAVs need to travel 224m on average, which is far from the point of interest and might conflict with the objective of their deployment. On the other hand, for the centralized approach shown in Fig. 2.6a, each UAV needs to travel only an average distance of 20m from its initial position. This shows the limitation of relying on uniform placements.

After attaining the centrally optimized positions, we evaluate the robustness of our obtained solution against external disturbances. We consider the effects of NLOS propagation, channel estimation errors, and UAV motion errors. We vary the value of the Rician K Factor,



(a) Centralized.



(b) URA.

Figure 2.6: Blue dots represent the initial positions, orange dots the final positions of the UAVs. The dashes on the ground represent the point of interest. While URA and Cent optimize capacity, URA moves the UAVs significantly far from their points of interest.

and for each value, we simulate 100 random realizations of the Rician channel, localization errors, and channel estimation errors. We plot the mean of the sum rates in Fig. 2.7 with the standard deviation shown as error bars along with the single-user upper bound (UB) from (2.6). We see that for small values of the K-factor, the NLOS becomes dominant and both the optimized and non-optimized positions yield the same average capacity. As the LOS becomes more dominant and the K-factor increases, the optimized positions start approaching the capacity upper bound. The random initial positions, on the other hand, converge to a lower sum rate. This is what we expect since an unoptimized LOS MIMO channel is correlated. In practice, the LOS air-to-ground channel typically has a high K-factor. In channel measurement campaigns performed at a frequency of 5-GHz (C-band) for a LOS air-to-ground channel in near-urban and suburban environments, it was shown that the average K-factor was above 25dB [MS17, Table V].

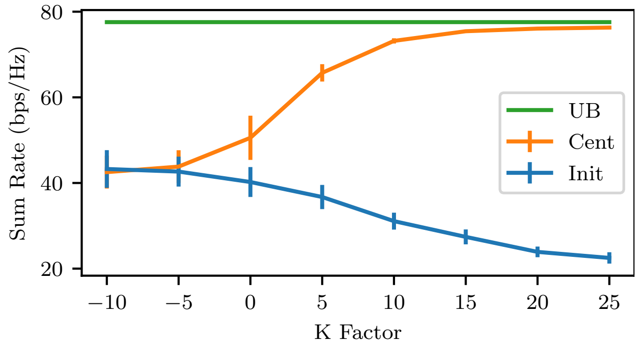
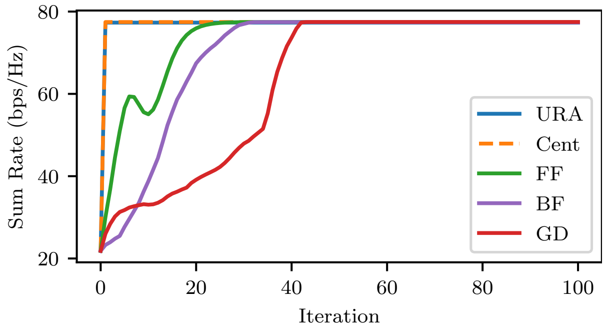
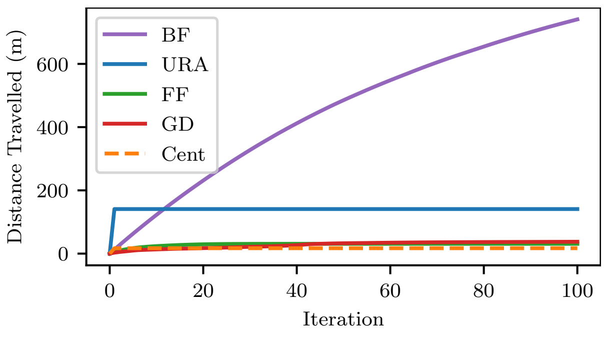


Figure 2.7: Optimizing positions improves capacity for LOS dominant channel having a large K-factor. Simulation is done for the placement in Fig.2.6a and includes localization errors and channel estimation errors.



(a) Average sum rate



(b) Average distance traveled per UAV up to a given iteration. BF and URA were omitted for exceeding 100m.

Figure 2.8: The sum rate and distance traveled for the ideal scenario.

2.9.3 Distributed Algorithms

Next, we evaluate the distributed algorithms performance in optimizing positions. URA approach and the centralized algorithm (Cent) are used as benchmarks. We compare Force Field (FF) against gradient descent (GD) and brute force (BF). For FF, we used $K_p = 0.3\frac{\min_n(\epsilon_n) S_x}{4\pi}$. The convergence results for 100 iterations in an ideal scenario with $K = \infty$ are shown in Fig. 2.8. From Fig. 2.8a, we see that all the methods converge to the optimal sum

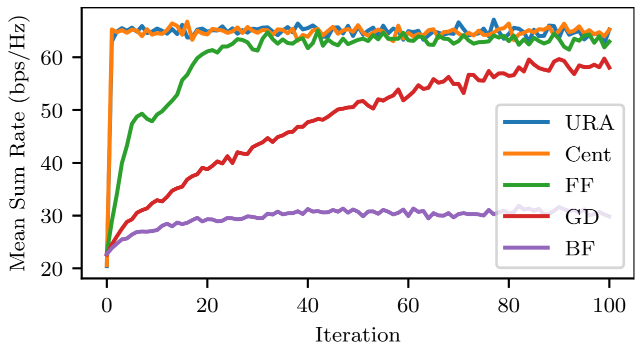


Figure 2.9: The sum rate of distributed approaches under practical disturbances.

rate. The average distance traveled per UAV up to a given iteration is shown in Fig. 2.8b with the curves for URA and BF omitted for exceeding 100m. This Figure along with Fig. 2.8a help characterize the tradeoff between attained capacity and the distance traveled. We can see that after the first two iterations with only 12m average traveled distance, the attained capacity is doubled. This shows that, using FF, significant gains can be attained with a few iterations and a limited traveled distance. Also while moving to optimize the capacity, the UAVs can work on their deployment tasks, hence FF does not impede on the deployment application.

Then, we evaluate the convergence of these methods under practical disturbances. Namely, we consider a K-factor of 20dB along with channel estimation errors and localization errors added after each iteration. Also, in addition to the free space path loss, we consider log-normal fading with 3.2dB standard deviation applied independently to each UAV [MS17]. Using the same initial positions, 100 realizations of these random distortions were simulated. The average sum rate results are shown in Fig. 2.9. Compared to the ideal scenario, the sum rates even for URA and Cent are about 16% lower because fading affects the channel magnitude and hence the SNR per stream. However, we see that FF still converges to the sum rate bound attained by URA and Cent in about 30 iterations similar to the ideal scenario. GD, on the hand, takes more than twice the iterations to converge compared the ideal scenario due to the random changes in the channel magnitude affecting the gradients. Brute

Force is severely impacted by the motion errors and does not converge [HYC19]. Hence, our proposed FF is robust to practical disturbances expected in a swarm of UAVs and can attain the sum rate bound. Compared to GD and BF, FF requires only a fraction of the inter-swarm communications and is guaranteed to converge within a bounded distance in an ideal scenario.

2.9.4 Distance Traveled Per UAV

UAV applications have different tolerance for UAV translations from the initial positions. Hence, it important to evaluate the distance that each UAV needs to travel. To that end, we numerically evaluate the distance traveled by each UAV as a function of $\boldsymbol{R}$. We consider 100 realizations, in which the UAVs are initialized randomly in a cube such that $V_x = V_y = V_z = 10$. The small cube guarantees that the UAVs are within $\mathcal{F}$ as we change the distance $\boldsymbol{R}$. In Fig. 2.10, the solid lines show the mean distance traveled and the whiskers represent the range calculated over all realization and over the entire swarm. The upper bounds for Cent and FF derived in Propositions 2 and 3 respectively are plotted as dashed lines. We can see that the distance that the UAVs need to travel increases as $\boldsymbol{R}$ gets larger. This scaling is captured by our upper bounds, which can be used to estimate the worst case traveled distance. As expected, the distributed algorithm using only channel estimates requires a larger displacement than the centralized algorithm with perfect knowledge of the swarm initial positions. While we only show results for one center frequency and URA design, by using the upper bounds, verified in this section, we can predict the effect of changing the URA design or the center frequency on the distance traveled.

2.9.5 Massive MIMO Evaluation

Now, we evaluate position optimization in the massive MIMO regime, where the number of GS antennas exceeds the number of UAVs $M \gg N$. Massive MIMO was shown to improve

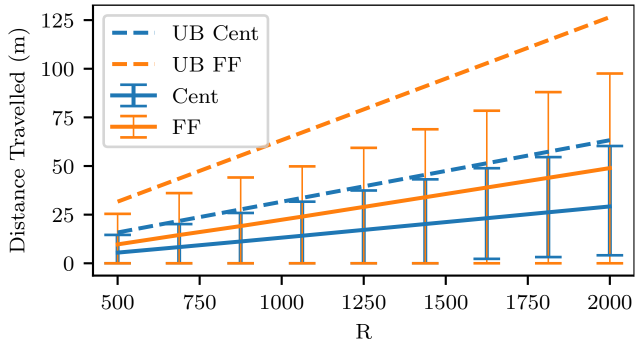


Figure 2.10: The mean distance traveled by the swarm. The error bars represent the range.

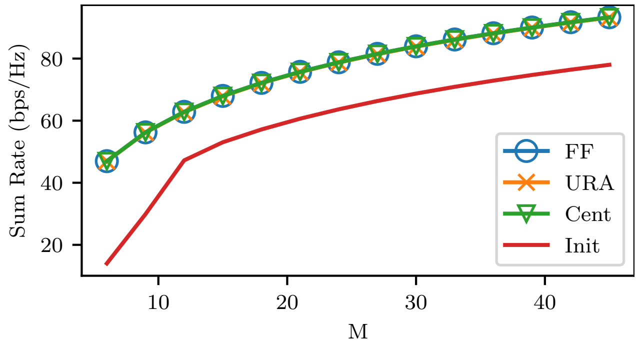


Figure 2.11: The sum rate of different approaches as we move to the massive MIMO regime.

the capacity by increasing the number of GS antennas [LET14]. One might presume that increasing the number of GS antennas eliminates the need for position optimization. We show that this is not the case. We consider 8 UAVs and a GS with a fixed aperture such that $L_x = M_x d_x = 4$ and $L_z = M_z d_z = 6$. The ratio between antennas in the x and z direction was set to be $M_x/M_z = 2$ and the total number of GS antennas is increased [MNC18]. From Fig. 2.11, we see that the optimized approaches provide a higher sum rate than the non-optimized as expected. But as the number of antennas increases the sum rate gap between both optimized and non-optimized approaches does not converge to zero. The suboptimal massive MIMO performance in a LOS channel was also observed and analyzed in [NLM14, Sec. 4.3]. This indicates that even as the number of antennas increases, swarm optimization can provide significant improvements. One way to interpret this result is to consider the grid of optimal positions similar to the one shown in Fig. 2.2. For a fixed aperture massive MIMO setup (assuming the same x-z plane without loss of generality), the smallest distance between two optimal positions $\frac{S_x}{M_x}$ is constant and is equal to $\frac{\lambda R}{L_x}$. This means that by increasing the number of antennas, the optimal point density is the same. Hence, if the initial positions are far from any optimal ones, they will remain far as we increase $\boldsymbol{M}$.

2.10 Summary

In this chapter, we optimized the placements of a UAV swarm to maximize the MIMO backhaul capacity starting from given swarm initial positions. We mathematically defined a set of UAV placements that orthogonalize the channel and maximize the MIMO capacity. The problem of minimizing the distance traveled to reach a placement in this set was formulated. An offline centralized solution was developed by relaxing the problem and decomposing it into two convex problems which were solved iteratively using block coordinate descent. We also proposed FF as a distributed iterative solution to this problem. FF requires sharing channel estimates only between neighbors and we derived the conditions for its convergence. Using numerical simulation, we have shown its robustness under channel and UAV induced disturbances. Upper bounds for the distance that UAVs need to travel using the centralized solution and force field were derived and numerically verified. Our approaches were shown to provide significant sum rate improvements while requiring only bounded displacements. The gains from our approach were shown to remain significant as we transition to the massive MIMO regime with far more ground station antennas than UAVs.

CHAPTER 3

Analysis and Design of BF Systems

Cooperating radios can extend their communication range by adjusting their signals to ensure coherent combining at a destination radio. This technique is called distributed transmit beamforming. Beamforming (BF) relies on the BF radios having frequency synchronized carriers and phases adjusted for coherent combining. Both requirements are typically met by exchanging preambles with the destination. However, since BF aims to increase the communication range, the individually transmitted preambles are typically at low SNR and their lengths are constrained by the channel coherence time. These noisy preambles lead to errors in frequency and phase estimation, which result in randomly changing BF gains. To build reliable distributed BF systems, the impact of estimation errors on the BF gains need to be considered in the design. In this Chapter, assuming a destination-led BF protocol and Kalman filter for frequency tracking, we optimize the number of BF radios and the preamble lengths to achieve reliable BF gain. To do that, we characterize the relations between the BF gains distribution, the channel coherence time, and design parameters like the SNR, preamble lengths, and the number of radios. The proposed relations are verified using simulations and via experiments using software-defined radios in a lab and on UAVs. This chapter revises our previous publication [HC21].

3.1 Introduction

Distributed transmit beamforming (BF) enables a group of radios to act as a virtual antenna array when transmitting a common message to a destination radio. By having N equal power radios beamform, the received power at the destination can increase by up to N^2; N-fold due to transmit power increase and N-fold due to coherent combining [MBM07]. The N^2 increase can theoretically provide up to N fold extension of communication range [MBM09]. Thus, BF can enable long-range communications from cooperating low power devices, unable to communicate individually with a remote destination. This can be useful for power-constrained sensor networks [JRL17] or UAVs deployed in remote regions [MBM19].

For separate radios to act as one virtual array, they need to synchronize their carrier frequencies and adjust their phases for coherent combining at the destination. Both requirements are typically satisfied by exchanging preambles with the destination for channel phase estimation and carrier frequency synchronization [MBM09]. However, given that in typical BF scenarios the radios have low power and/or the destination is remote, the pre-BF SNR of individual radios is low, and there are errors in both channel estimation and destination-led frequency synchronization, which result in phase errors in the combining signals. These combining phase errors will lead to the BF gains being non-deterministic and less than N^2. The BF gain degradation cannot always be mitigated, especially in high mobility radios like UAV-mounted, where the channel coherence time limits the preamble lengths and makes the combining phase errors inevitable. To build a reliable BF system despite of these errors, we need to specify the number of BF radios and the preamble lengths such that a minimum desired post-BF SNR is attained with a given probability.

Existing works have proposed many approaches for BF leveraging different methods for phase adjustment and frequency synchronization [JRL17]. Approaches for phase adjustment include explicit channel feedback from the destination [YP02], 1-bit feedback where the BF radios iteratively adjust their phase based on binary feedback from the destination [MHM10], and roundtrip message exchange among the destination and BF radios [IP08]. Other works proposed using the BF radios' placements to adjust the phase [HKC21a], however, this only works in a line-of-sight channel. For frequency synchronization, some works have relied on external frequency references like GPS [LGS18, KAL19], out of band signaling [RBM12a], and others relied on a destination preamble along with using the extended Kalman filter (EKF) for tracking the carrier drift [QMR12]. While these works have proposed interesting approaches, the relation between the BF gains and the pre-BF SNR, necessary for designing a reliable BF system, was not analyzed.

Using the aforementioned approaches, several BF demonstrations were carried; in a controlled lab experiment, 1-bit feedback was demonstrated using EKF for frequency synchronization in [QMR12, QRM13] and out-of-band signaling in [RBM12a]. Outdoor ground based demonstrations spanning several kilometers using explicit channel feedback were performed in [LGS18, KAL19] relying on GPS for frequency synchronization. Using explicit feedback, in [MBM19], BF was demonstrated from UAVs with the synchronization performed over wires attached to the flying UAVs. These works have shown the potentials for BF in signal combining, yet their results are hard to generalize to different scenarios because they are mostly empirical.

In this Chapter, we consider a destination-led BF protocol using the Kalman filter (KF) for synchronization and explicit channel feedback. For that protocol, assuming equal pre-BF SNRs, we propose an analytical framework relating the statistical distribution of the BF gains, with the system parameters including pre-BF SNR, the number of BF radios, and the duration of the exchanged preambles. Using this framework, for a given channel coherence time, we optimize the number of radios and the length of the preambles for the BF gain to exceed a minimum SNR with a given probability, thus creating a reliable BF system. To derive this framework, we derive the variance of the combining phase errors, which depends on the preamble lengths and the pre-BF SNR. Then, given the variance of the combining phase errors, we approximate the distribution of the BF gains. The proposed framework is verified using simulations and experimentally using two BF software-defined radios (SDRs) in a lab environment. To the best of our knowledge, we are the first to demonstrate fully wireless BF from flying UAVs without any wires attached. Our main contributions are:

- We proposed an analytical framework describing the relations between the BF gains and the pre-BF SNR, the length of the preambles, and the number of BF radios for a destination-led BF protocol under the assumption of equal pre-BF SNRs. These relations were verified using simulations and experimentally using two BF software-defined radios.

- We characterized the distribution of BF gains assuming zero-mean normally distributed phase errors. We derived the variance of the BF gains. For large N, we proved the BF gain distribution approaches Gaussian and for small phase error variance we approximated it using a Gamma distribution.

- Using the BF framework, we proposed an approach to determine the minimum number of BF radios and the shortest BF preambles to meet a minimum post-BF SNR with a given probability and verified that it meets the requirements using simulations.

3.2 System Model and Distributed BF Protocol

3.2.1 System Model

Consider N identical radios collaborating to beamform a message to a destination radio D in a narrowband flat-fading channel. The BF radios can be remotely deployed Internet-of-Things devices communicating with a gateway or UAVs communicating with a ground station. The message is encoded in the complex baseband signal $m(t)$, which is assumed to have unity power. The n-th radio transmits a signal $z_n(t)$ and the combined baseband signal at the destination is given by

$$y(t) = \sum_{n=1}^{N} a_n z_n(t) \exp\{j(2\pi f_n t + \phi_n)\} + w(t) \tag{3.1}$$

where between the destination and the n-th radio, a_n is the channel amplitude, f_n is the carrier frequency offset, and ϕ_n is the phase offset. The white Gaussian noise process is given by $w(t)$ and has power spectral density $N_0/2$. The phase and frequency offsets result from the lack of synchronization between the local oscillators of the radios, the wireless propagation environment, and the Doppler frequency offsets resulting from the relative motion of radios. While these phenomena make the phase and frequency offsets time varying, we assume that the message is shorter than the resulting channel coherence time and we approximate them as constant for one message.

For the signal $m(t)$ to combine coherently at the destination, the BF radios need to compensate for the phase and frequency offsets before transmission. The compensated signal transmitted by radio n, thus, is given by

$$z_n(t) = m(t)\exp\{-j(2\pi \hat{f}_n t + \hat{\phi}_n)\} \tag{3.2}$$

where $\hat{f}_n$ and $\hat{\phi}_n$ are the n-th radio estimates of the frequency and phase offsets obtained through the BF protocol, which is described later. The received signal can be rewritten as

$$y(t) = m(t)\sum_{n=1}^{N} a_n \exp\{j\phi_n^e(t)\} + w(t) \tag{3.3}$$

where the combining phase error from radio n at instant t is given by

$$\phi_n^e(t) = (2\pi(f_n - \hat{f}_n)t + (\phi_n - \hat{\phi}_n)) \tag{3.4}$$

Due to residual frequency errors, the combining phase error increases with time. However, we are only interested in evaluating the BF gain during the payload. Considering the evaluation instance to be t_e seconds after the phase estimation, we get $\phi_n^e = \phi_n^e(t_e)$. The beamforming gain at instant t_e can be defined as the ratio between the energy of the combined signals to that of the individual transmissions

$$G = \frac{\|\sum_{n=1}^{N} a_n \exp\{j\phi_n^e\}\|^2}{\sum_{n=1}^{N} a_n^2} \tag{3.5}$$

Each BF radio is assumed to transmit at its maximum power level P_T, which is common to all radios. We also assume that the BF radios are deployed in proximity from each other far from the destination, and hence they experience similar signal attenuation. Given these assumptions, we get $a_n = a$ for all n, where a is the path loss. In that case, G simplifies to [RBM12b]

$$G = \frac{1}{N} \left| \sum_{n=1}^{N} \exp\{j\phi_n^e\} \right|^2 \tag{3.6}$$

The pre-BF SNR at the destination from one radio is given by

$$\gamma_{\text{preBF}} = \frac{a^2 P_T}{N_0} \tag{3.7}$$

and the post-BF SNR of the combined signal from all N BF radios is equal to

$$\gamma_{\text{postBF}} = N\gamma_{\text{preBF}} G \tag{3.8}$$

The signals transmitted by the destination to the BF radios experience an SNR given by

$$\gamma_{\text{DR}} = \frac{a^2 P_T^D}{N_0} \tag{3.9}$$

where the destination has a transmit power P_T^D. The destination transmit power is assumed to be equal to or larger than that of the BF radios, that is $P_T^D \geq P_T$. Note that the post-BF SNR follows the same distribution of G, which we need to know to realize a minimum post-BF SNR with a given probability. As for G, it depends on ϕ_n^e, which results from the estimation errors during the BF protocol.

3.2.2 Beamforming Protocol

We start by describing the BF protocol, which aims to provide each BF radio with estimates of its phase and frequency offsets $\hat{\phi}_n$ and $\hat{f}_n$. We consider a master-slave beamforming protocol ; the destination radio is used as a master since it has a larger transmit power and the slaves are the beamforming radios. The master initiates the beamforming procedure and sends a preamble for frequency synchronization. After correcting their frequencies, the slaves send a channel estimation preamble to the master. The master calculates a phase estimate $\hat{\phi}'_n$ and transmits it back to the slaves that receive a slightly different value $\hat{\phi}_n$ due to feedback errors. Once each slave knows $\hat{\phi}_n$ and $\hat{f}_n$, they can start transmitting their payload.

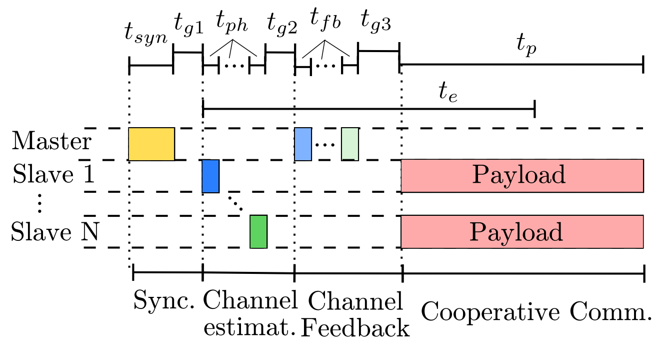


Figure 3.1: Timing diagram of BF protocol. The destination is the master and BF radios are the slaves.

In Fig. 3.1, we illustrate the transmitted signals. All the signaling is performed on the same frequency band, hence, all transmissions are received by all radios. The different beamforming stages can be described as follows

1. Synchronization: The master sends a synchronization preamble of duration t_{syn}. Using this signal each slave estimates its frequency offset $\hat{f}_n$. This preamble is also used as a time reference. A guard time of duration t_{g1} is provided for the slaves to process the signals.

2. Channel Estimation: Each slave during an allocated time slot sends a channel estimation preamble of duration t_{ph}. The master estimates $\hat{\phi}'_n$ from each slave. A guard time t_{g2} is used.

3. Channel Feedback: The master sends $\hat{\phi}'_n$ back to the slaves and due to feedback error each slave receives a slightly different phase estimate $\hat{\phi}_n$. A guard time t_{g3} is used.

4. Cooperative Communication: After estimating $\hat{f}_n$ and receiving $\hat{\phi}_n$, all slaves adjust their signals accordingly and transmit their payload of duration t_{p}.

The duration of the BF overheads incurred by the protocol is given by

$$t_{\mathrm{ov}} = t_{\mathrm{syn}} + N(t_{\mathrm{ph}} + t_{\mathrm{fb}}) + t_{\mathrm{g1}} + t_{\mathrm{g2}} + t_{\mathrm{g3}} \tag{3.10}$$

All the signal processing is assumed to be done in discrete time domain, hence all the time durations are assumed to be an integer multiple of the sampling time T_s. The time overhead can be written in terms of samples as

$$N_{\text{ov}} = N_{\text{syn}} + N(N_{\text{ph}} + N_{\text{fb}}) + N_{\text{g1}} + N_{\text{g2}} + N_{\text{g3}} \tag{3.11}$$

where N_{ov} is defined as $N_{\text{ov}} = t_{\text{ov}}/T_s$ and the remaining number of samples are defined similarly. As we can see from (3.11), the beamforming overheads scale with the number of BF radios N. For short coherence time channels, the overheads N_{ov} are typically constrained, and to increase N while keeping N_{ov} constant, the duration of the preambles needs to be reduced. Note that we assume that the payload is already shared among all the slaves. This can be achieved using a network broadcasting protocol [WC02], which we do not discuss in this work. As for the guard time, it is dependent on the implementation of the BF protocol. A more optimized implementation using an FPGA for instance would require shorter guard times than an implementation using a general purpose processor. Also, note that cooperative communication only requires the BF radios to be synchronized with each other and not necessarily with the destination. However, in order to use channel estimates from the destination, they need to be synchronized with the destination.

Since BF is used to improve the SNR where the individual pre-BF SNR is low, the estimation errors within the protocol can not be neglected and will lead to a combining phase error ϕ_n^e as given by (3.4). At the evaluation time t_e, the variance of the combining phase error σ_e^2 defined as $\text{var}\{\phi_n^e\}$ is given by

$$\sigma_e^2 = (2\pi t_e)^2\sigma_f^2 + \sigma_{ph}^2 + \sigma_{fb}^2 \tag{3.12}$$

where the frequency estimation variance is given by $\sigma_f^2 = \text{var}\{f_n - \hat{f}_n\}$, the phase estimation and feedback variances are given by $\sigma_{ph}^2 = \text{var}\{\phi_n - \hat{\phi}'_n\}$ and $\sigma_{fb}^2 = \text{var}\{\hat{\phi}'_n - \hat{\phi}_n\}$ respectively.

In the following Sections, 3.3 and 3.4, we discuss the waveforms and estimators used for frequency estimation and phase estimation & feedback respectively. We provide expressions for their error variances in terms of the pre-BF SNR and the preamble lengths. We argue that the resulting phase errors follow a zero-mean Gaussian distribution. For zero-mean Gaussian distributed phase errors with variance σ_e^2, we approximate the distribution of the BF gain in Section 3.5 to complete the BF framework. This framework is numerically and experimentally verified in Section 3.6. After verifying the framework, we show how it can be used for designing BF systems in Section 3.7. The BF design procedures are illustrated using example scenarios in Section 3.8.

3.3 Frequency Synchronization

The objective of frequency synchronization is to eliminate the frequency offset between the destination and the BF radios. We start by discussing the signals used for synchronization and the proposed oneshot estimator and its variance. Then we discuss frequency tracking using Kalman filter assuming multiple successive BF cycles.

3.3.1 Frequency Offset Estimation

For frequency synchronization, we use a preamble consisting of N_{ZC} repetition of a Zadoff-Chu (ZC) sequence of length M similar to [YHB19], satisfying $N_{\mathrm{syn}} = N_{\mathrm{ZC}}M$. The frequency estimator calculates the auto-correlation statistic

$$\eta_f = \sum_{k=0}^{(N_{\mathrm{ZC}}-)M-1} y_f^*[k] y_f[k+M] \tag{3.13}$$

where $y_f[k]$ is the noisy received preamble with the frequency offset, and $()^*$ denotes the conjugate operator. The frequency offset estimate at slave n is thus given by $\hat{f}_n = \frac{1}{2\pi T_s M} \angle \eta_f$ where $\angle(\cdot)$ denotes the phase of a complex number calculated using arctan. The term $\angle \eta_f$ calculates the phase difference between two successive sequence repetitions, under the assumption that M is small such that no phase wrapping occurs. The error variance for this

Table 3.1: Kalman Filter relations

Model	Update	Predict
$x_k = x_{k-1} + w_{k-1}$ (3.15) $z_k = x_k + v_k$ (3.16)	$K_k = \frac{p_{k\|k-1}}{p_{k\|k-1} + r}$ (3.17) $x_{k\|k} = x_{k\|k-1} + K_k(z_k - x_{k\|k-1})$ (3.18) $p_{k\|k} = (1 - K_k)p_{k\|k-1}$ (3.19)	$x_{k+1\|k} = x_{k\|k}$ (3.20) $p_{k+1\|k} = p_{k\|k} + q$ (3.21)

estimator is given by [LRP73, eq.70]

$$\sigma_{fe}^2 = \left(\frac{1}{M(N_{\text{ZC}} - 1)^2 \gamma_{\text{DR}}} + \frac{1}{2M(N_{\text{ZC}} - 1)\gamma_{\text{DR}}^2} \right) \frac{1}{(2\pi M T_s)^2} \tag{3.14}$$

This estimator is unbiased thus $\mathbb{E}\{\hat{f}_n - f_n\} = 0$ and was derived using a linear approximation of the arctan assuming η_f has a high SNR. By choosing M to be large, using the central limit theorem, the distribution of η_f can be approximated by Gaussian, thus making $\hat{f}_n - f_n$, which is approximated as linear in η_f, a zero mean Gaussian RV. However, at low SNR of η_f, $\angle\eta_f$ becomes uniform and the expression of σ_{fe}^2 no longer applies. This regime can be avoided by increasing N_{ZC}, otherwise, the BF gains will be too low to be of practical importance. Note that in practice the frequency offset is correlated among successive packets with short separation. This estimator, referred to as a oneshot frequency estimator, does not benefit from this correlation.

3.3.2 Interpacket Frequency Tracking using Kalman Filter

If beamforming is performed periodically at a fixed cycle duration t_{cyc} shorter than the channel coherence time, the frequency estimates between packets at each slave are correlated. Kalman filter (KF), thus, can be used to track the frequency to reduce the estimation variance. The drift system model and the KF equations are given in Table 3.1 for one BF radio following the conventional KF notation [TL98]. The frequency process drift and measurement models are given by (3.15) and (3.16), respectively, where x_k is the true frequency value in Hz (previously denoted by f_n) and z_k is the measured frequency at time kt_{cyc}. The noise terms for the process w_k and the measurement v_k are assumed to be zero mean Gaussian RV and their variances are q and r respectively. For the KF update equations, at step k, K_k is the Kalman gain, $x_{k|k-1}$ is the prediction of x and $p_{k|k-1}$ is the error variance given z_{k-1}. The value of $x_{k|k}$ is the predicted frequency offset and $p_{k|k}$ is its error variance given z_k.

By substituting (3.17) in (3.18) and using (3.20) we get

$$x_{k|k} = \frac{r}{p_{k|k-1} + r} x_{k-1|k-1} + \frac{p_{k|k-1}}{p_{k|k-1} + r} z_k \tag{3.22}$$

from which we can see that the KF creates a weighted average between the previous prediction and the current measurement. The weights of this average are based on the predicted process variance $p_{k|k-1}$ and the measurement variance r. The larger the process variance relative to the measurement variance, the more weight is given to the measured value and vice versa. Since (3.22) is a linear equation, if z_k is a zero mean Gaussian RV, the output of KF will also be zero-mean and Gaussian. For BF, we are interested in calculating the KF error variance.

Proposition 4. *The steady state frequency estimation error variance of KF from Table 3.1 is*

$$\sigma_{fk}^2 = \frac{-q + q\sqrt{1 + 4\frac{r}{q}}}{2} \tag{3.23}$$

The proof is in Appendix B.1. Using (3.23) and assuming the system variances are accurately known, we argue that KF never increases the error variance. By rewriting (3.23), as $\sigma_{fk}^2 = \frac{-q+\sqrt{q^2+4qr}}{2}$, we can see that σ_{fk}^2 is non-decreasing in q and if $q = 0$, at convergence the error variance $\sigma_{fk}^2 = 0$ for any r. For $q >> r$, r/q is small and using the approximation

$\sqrt{1+4\frac{r}{q}} \approx 1+2\frac{r}{q}$, we get $\sigma^2_{fk} = r$. Thus if q and r are perfectly known, the error variance reduction due to KF is higher for large r/q and, in the worst case scenario for small r/q, KF will give the measurement variance $\sigma^2_{fk} = r$, as if we did not use KF. However, if the values of q and r used in KF do not match the system, this result does not hold and KF might deteriorate the frequency estimation. Note that the extended KF (EKF) can track both phase and frequency and might yield a smaller variance than KF which only tracks the frequency. However, EKF can diverge due to phase wrapping [QRM13], which is not desirable in a reliable BF system, and thus was not considered in this work.

3.4 Phase Estimation and Feedback

The objective of the phase estimation and feedback is to have the slaves modify their signals to ensure coherent combining at the destination. In the phase estimation stage, each slave transmits a known signal $x_{\text{ph}}[n]$ consisting of N_{ph} samples. The master receives the noisy signal $y_{\text{ph}}[k]$. The proposed estimator calculates the correlation $\eta_{\text{ph}} = \sum_{k=0}^{N_{\text{ph}}-1} x_{\text{ph}}[k]y_{\text{ph}}[k]$, from which the phase estimate is calculated using $\hat{\phi}'_n = \angle\eta_{\text{ph}}$. The variance of this estimator is given by [Tre85]

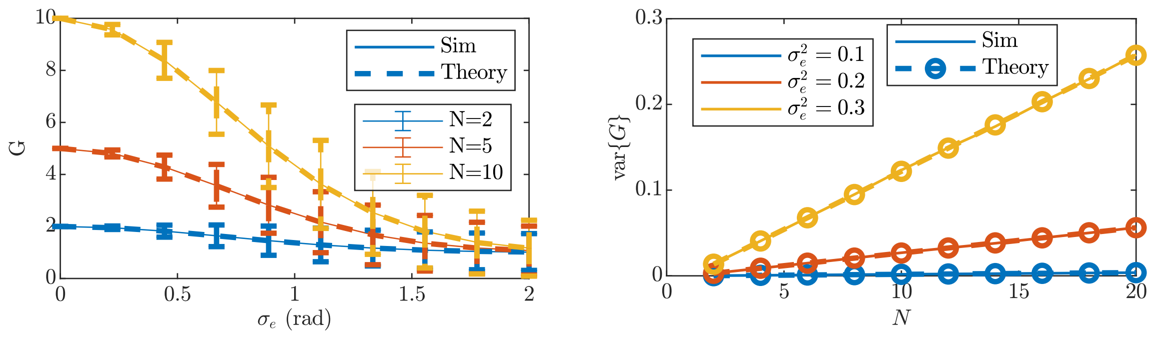
$$\sigma^2_{phe} = \frac{1}{2N_{\text{ph}}\gamma_{\text{preBF}}} \tag{3.24}$$

where $N_{\text{ph}}\gamma_{\text{preBF}}$ is the SNR of η_{ph}. The phase error $\hat{\phi}'_n$ follows a zero mean Gaussian distribution as long as the SNR of $\eta_{\text{ph}} >> 1$ [Tre85], which is the regime of interest.

As for the phase feedback, we use in-band feedback where the value of $\hat{\phi}'_n$ is encoded in the phase difference between two identical preambles to counter hardware phase ambiguity. Let the phase feedback preamble be given as a vector $\mathbf{x}_{\text{fbp}} \in \mathbb{C}^{N_{\text{fb}}}$. The master transmits the sequence

$$\mathbf{x}_{\text{fb}} = [\mathbf{x}^T_{\text{fbp}} \;\; \mathbf{x}^T_{\text{fbp}}e^{j\hat{\phi}'_1} \cdots \mathbf{x}^T_{\text{fbp}}e^{j\hat{\phi}'_n} \cdots \mathbf{x}^T_{\text{fbp}}e^{j\hat{\phi}'_N}]^T \tag{3.25}$$

Once received as $y_{\text{fb}}[k]$ with added noise, slave n estimates the phase difference between the first preamble and the n-th preamble using the statistic $\eta_{\text{fb}} = \sum_{k=0}^{N_{\text{fb}}-1} y_{\text{fb}}[k]y_{\text{fb}}[k+nN_{\text{fb}}]$



(a) The mean BF Gain and its standard deviation as error bars.

(b) The relation between the BF variance and N for fixed σ_e^2.

Figure 3.2: The relation between BF gain, N, and σ_e^2.

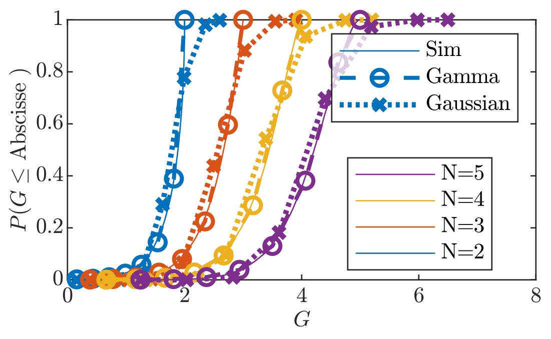
and calculates the angle $\hat{\phi}_n = \angle \eta_{\text{fb}}$. The variance of the feedback is similar to that used for frequency estimation in (3.14) (with $N_{\text{ZC}} = 2$, $M = N_{\text{fb}}$) and is given by

$$\sigma_{fbe}^2 = \left(\frac{1}{N_{\text{fb}} \gamma_{\text{DR}}} + \frac{1}{2 N_{\text{fb}} \gamma_{\text{DR}}^2} \right) \tag{3.26}$$

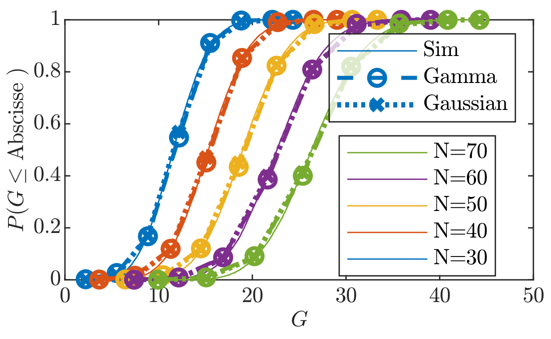
Note that there are other ways to feedback the phase estimates, however, this approach is simple to implement. Another alternative was to encode the values of $\hat{\phi}'_n$ as floating-point numbers and transmit them using digital modulation. However, since we are considering a low SNR and a mistake in one of the most significant bits can be detrimental, we would need to implement channel coding. This would add unnecessary complexity to our protocol.

3.5 Beamforming Gain Analysis

In this section, our objective is to approximate the distribution of G, assuming that the ϕ_n^e are independent Gaussian random variables (RVs) with zero mean and variance σ_e^2. The Gaussian assumption applies to our protocol because the errors of the proposed estimators are independent and can be approximated by a zero-mean Guassian RVs. Hence, their sum according to (3.4) is also zero-mean Gaussian. We start by calculating the mean and variance



(a) The distribution of G for small N and small $\sigma_e^2 = 0.1$.



(b) The distribution of G for large N and large $\sigma_e^2 = 1$.

Figure 3.3: The distribution of G for different N and σ_e^2

of the distribution.

Proposition 5. *For signals combining from N radios having independent zero mean Gaussian phase with variance σ_e^2, the mean and the variance of the BF gains G are given by*

$$\mathbb{E}\{G\} = 1 + (N-1)e^{-\sigma_e^2} \tag{3.27}$$

$$var\{G\} = \frac{(N-1)}{N}(1-e^{-\sigma_e^2})^2\left((1-e^{-\sigma_e^2})^2 + 2Ne^{-\sigma_e^2}\right) \tag{3.28}$$

The proof is in Appendix B.2. Note that the mean was previously derived in [RBM12b]. In Fig. 3.2a, we plot the average BF gain using (3.27) as a function of σ_e with the error bars representing the standard deviation ($\sqrt{\text{var}\{G\}}$). For $\sigma_e = 0$, we get a BF gain of N as we ideally expect. As σ_e increases, the mean BF gains decrease and their variances increase and this happens faster for larger N. Thus when designing a BF system unless N and σ_e are small, we can not assume a perfect N fold power increase due to BF. To verify the derived mean and variance, for each value of N and σ_e, we sampled 100,000 zero mean Gaussian RVs of variance σ_e^2 for each radio and added them to calculate G numerically. The simulations shown in Fig.3.2a as thick dashed lines with dashed error bars overlap the derived expressions verifying Proposition 5.

To better understand the variance behavior with N, for small σ_e^2, we simplify (3.28) to get $\mathrm{var}\{G\} \approx 2Ne^{-\sigma_e^2}(1-e^{-\sigma_e^2})^2$. Thus the variance increases linearly with the number of slaves for small σ_e^2. The linear relation between $\mathrm{var}\{G\}$ and N is illustrated in Fig.3.2b. The higher the value of σ_e^2, the larger the slope. The large discrepancy in the values of the variance with N shows the importance of considering the distribution of G and not just its mean in the design of reliable BF systems. Next, we approximate the distribution of G. First, we consider the case of large N using the central limit theorem. Then, we consider the case for a small N and small σ_e^2 and use the Taylor series to derive the approximation.

Proposition 6. *For large N, the distribution of G tends to a Gaussian distribution with mean and variance given by Proposition 5.*

Proposition 7. *For small combined phase error variance σ_e^2 or for large N, the distribution of G can be approximated by $N - X_\gamma$ where X_γ is a random variable following the Gamma distribution $X_\gamma \sim \Gamma(K,\theta)$ with*

$$K = \frac{N(N-1)}{(1-e^{-\sigma_e^2})^2 + 2Ne^{-\sigma_e^2}} \tag{3.29}$$

$$\theta = \frac{1}{N}(1-e^{-\sigma_e^2})\left((1-e^{-\sigma_e^2})^2 + 2Ne^{-\sigma_e^2}\right) \tag{3.30}$$

The proofs are in Appendices B.3 and B.4 respectively. We start by plotting the empirical cumulative distribution function (CDF) of G for small N and a small $\sigma_e = 0.1$ in Fig. 3.3a. We can see that the distribution is not Gaussian and is accurately approximated by the Gamma distribution. Then, we consider a large $N \geq 30$ and relatively large value of $\sigma_e = 1$ in Fig. 3.3b. From that Figure, we can see that all three CDFs overlap for large N and large σ_e verifying Prop. 6 and 7. Based on these results, since the Gamma distribution applies to a wider range of N and σ_e^2, we use it later to approximate the BF gain distribution. Note that neither approximation is accurate for small values of N and a large value of σ_e^2, however, in this regime the BF gains are small with a large variance, which is not of practical importance. It is important to note that the derived variance and distribution approximation in this section apply to any BF protocol where the phase error ϕ_n^e is independent for all n and can be approximated by zero-mean Gaussian RVs. For our protocol, the value of σ_e^2 can depend on N for scenarios where the BF overhead N_{ov} is constrained by the channel coherence time. In such scenarios, the duration of each preamble decreases as N increases to satisfy the fixed N_{ov}. Thus the estimators error variances and consequently σ_e^2 increase with N. The dependence between N and σ_e^2 is considered when designing the BF preambles in short coherence channels later in Section 3.8.1.

3.6 Numerical and Experimental Validation

In this Section, after deriving the BF framework, we verify it numerically and experimentally and we show that it can be used to predict the BF gains at different SNRs. Using UAV experiments and emulation over a UAV channel trace, we evaluate the impact of the channel coherence time on the BF gains.

3.6.1 Numerical Validation

We simulated the BF protocol between a destination radio and N BF radios. During a BF cycle, signals transmitted from BF radio n to the destination is multiplied by $e^{j(2\pi f_n t+\phi_n)}$ with noise added to realize the SNR γ_{preBF}. Any signal transmitted the other way uses the negative value of f_n with noise added to realize the SNR γ_{DR} . At the start of each BF cycle, for BF slave n, we sample uniform random phase ϕ_n and f_n is generated using a discrete Wiener process as described in (3.15) having variance q. Since we are assuming that the signal is transmitted within the channel coherence time, both frequency and phase are assumed to be constant during the same BF cycle.

The signals transmitted follow the BF protocol. For phase estimation and feedback, we used the estimators discussed in Section 3.4 and for frequency offset we either used the oneshot estimator from Section 3.3.1 alone or combined with KF. To avoid errors in

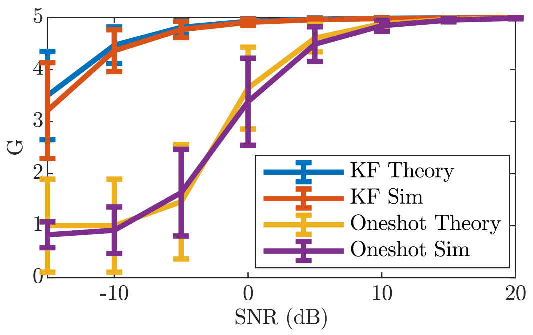
Table 3.2: Beamforming Waveform Specifications

Scenario	Parameters
Simulation	$N_{\text{ZC}} = 10$, $M = 63$, $t_{\text{syn}} = 0.63ms$, $t_{\text{ph}} = 0.1ms$, $t_{\text{fb}} = 0.1ms$, $t_{\text{g1}} = t_{\text{g2}} = t_{\text{g3}} = 1ms$, $t_{\text{p}} = 12ms$, $t_e = 9ms$, $t_{\text{cyc}} = 50ms$
Lab	$N_{\text{ZC}} = 10$, $M = 63$, $t_{\text{syn}} = 0.63ms$, $t_{\text{ph}} = 0.1ms$, $t_{\text{fb}} = 0.1ms$, $t_{\text{g1}} = 6ms$ $t_{\text{g2}} = 4ms$ $t_{\text{g3}} = 16ms$, $t_{\text{p}} = 10ms$, $t_{\text{cyc}} = 180ms$
UAV	$N_{\text{ZC}} = 10$, $M = 63$, $t_{\text{syn}} = 0.63ms$, $t_{\text{ph}} = 0.1ms$, $t_{\text{fb}} = 0.1ms$, $t_{\text{g1}} = 6ms$ $t_{\text{g2}} = 4ms$ $t_{\text{g3}} = 11ms$, $t_{\text{p}} = 1ms$, $t_{\text{cyc}} = 75ms$

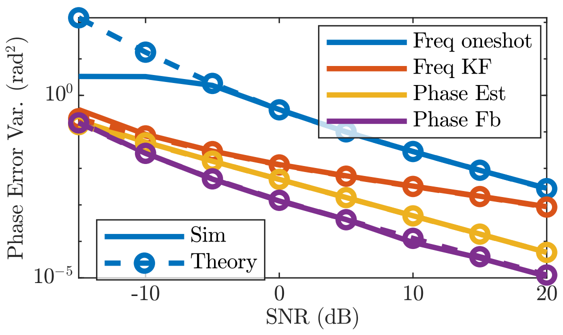
measuring the BF gain, the combined signal magnitude was evaluated at time t_e before adding the noise.

In our simulations, we considered $N = 5$ BF radios using a sampling rate of 1MHz ($T_s = 1\mu s$). The exact duration of each preamble is given in the first row of Table 3.2 and we used $q = 0.18$.The evaluation time $t_e = 9ms$ is in the middle of the payload. One million BF cycles were simulated.

We start by discussing the results obtained when using the oneshot frequency estimation. The average BF gain obtained from simulations is plotted in Fig. 3.4a with the error bars representing its standard deviation. For the oneshot results, the theoretical value is obtained by calculating the variance of each estimator using (3.14), (3.24), and (3.26), calculating σ_e^2 using (3.12), then the BF gain mean and variance using Proposition 5. From that Figure, we can see that the theoretical mean matches the simulations to a large extent. As for the variances, they match except for SNRs below 0dB. By plotting a breakdown of the



(a) BF Gains using oneshot and KF for frequency synch. The standard dev. is shown as error bars.



(b) Phase error variance breakdown with respect to protocol stages.

Figure 3.4: Simulated BF Gains and phase errors at different SNRs for $N = 5$ using the waveform from Table 3.2.

phase error for the slave $n = 3$ using (3.12) in Fig. 3.4b, we see that at SNRs below 0dB the theoretical oneshot frequency variance is overestimated. This happened because the phase error becomes uniform and the Gaussian assumption no longer holds leading to the discrepancy in Fig. 3.4a. At these low SNRs, the BF gains are negligible and this is not a useful BF design. From Fig. 3.4b, since the phase error from the frequency estimation error is dominant, it would be beneficial to allocate more time to frequency estimation or use the KF to reduce its variance.

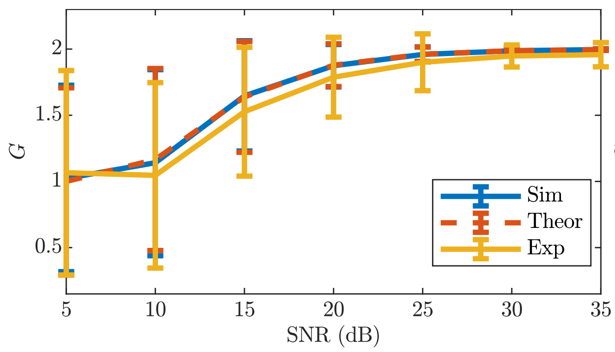
Next, we discuss the BF results when using KF using the same Figures 3.4a and 3.4b. The theoretical KF variance is calculated using (3.23) with the measurement variance r being the oneshot variance and q perfectly known. From Fig. 3.4a, we can see that both theoretical and simulated curves overlap. A small discrepancy exists at low SNR, which we attribute to an insufficient number of BF cycles. Since KF is a recursive filter, its output depends on all previous cycles and convergence is slower for high measurement noise variance [CS03]. Compared to the oneshot BF, at low SNR, KF provides significant BF gain improvements by reducing the frequency estimation variance and the resulting phase errors as shown in Fig. 3.4b. From that Figure, we also see that as the SNR (above 0dB) becomes larger, the gap between oneshot and KF decreases. This happens because as r decreases at high SNR, the ratio r/q becomes small and the benefit from using KF decreases.

3.6.2 Experimental Validation

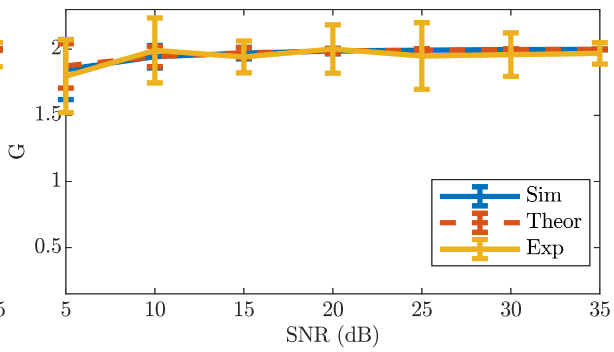
The proposed BF protocol was implemented using three USRP B205-mini software-defined radios (SDR); two were used as BF radios and one as the destination radio. The destination radio initiates a BF cycle by transmitting the frequency synchronization preamble. The BF radios are always running the autocorrelation given by (3.13) and using its output power level to detect the preamble. Once detected, the frequency offset is estimated (using oneshot or KF) and corrected. Each BF radio transmits the phase estimation preamble in a pre-assigned time slot. The destination radio estimates the phase and feeds it back to the BF radios using the same previously discussed waveforms and estimators. Once the feedback is obtained, the radios transmit a known payload, which is received and stored by the destination. The payload consists of three parts; each of the two BF radio transmits individually at first, then both BF radios transmit simultaneously. The magnitude of each part of the payload is estimated by averaging, then the BF gain is calculated by dividing the power of the simultaneous transmission by the sum of the individual transmissions as per (3.5). All the signal processing was implemented using GNURadio [GNU] and timed burst transmissions were used for the different stages of the protocol. The destination processing was performed on a laptop and the BF radios on ODROID XU4 single board computers (SBC). We conducted the experiments in the lab and on UAVs at a frequency of 915MHz with a sampling period of $T_s = 1\mu s$.

3.6.2.1 Lab Experiment

We started by verifying our simulations in a lab environment with a favorable channel. The beamforming slaves were placed in proximity from each other, 2.5 meters away from the



(a) BF gains using oneshot frequency estimation.



(b) BF gains using KF for frequency estimation.

Figure 3.5: Experimental results collected using $N = 2$ BF software-defined-radios in a lab along with the theoretical results predicted by the BF framework and simulations.

destination in an undisturbed line-of-sight environment with a measured coherence time of 0.3s and $q = 0.18$. Both the destination and BF radios were set to use the same transmit gain, which was varied in increments of 5dB to obtain different SNRs. At each SNR, 900 beamforming cycles were performed. The timing of the protocol is shown in Table 3.2. Notice that the guard times are much longer than in the simulations to allow the BF signal processing to operate in real-time, which makes t_e larger in (3.4), and thus increases σ_e^2 and degrades the BF gains.

The experimental results along with its simulated and theoretical equivalents are shown in Fig. 3.5a. We can see that the measured results are close to the simulation and theoretical results, which overlap. The improvement from using KF follows a similar trend to what was observed in Fig. 3.5b. This result experimentally verifies our simulation setup and analysis.

3.6.2.2 UAV Experiment

Next, we move our setup from the lab to UAVs. The BF radios, consisting of the SBC and USRPs along with a battery, were mounted on two DJI Phantom 3 drones as shown in Fig. 3.6. The destination radio was placed on the ground about 5m away from the UAVs

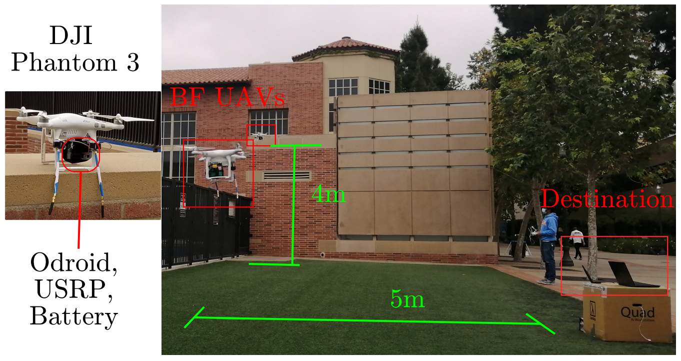


Figure 3.6: The UAV experiment consists of 2 BF UAVs with SDRs mounted on-board. The UAVs were hovering freely and were not attached to the ground by wires.

Table 3.3: BF UAV Results

Setup	Freq	SNR (dB)	G Mean	G Stdev
Ground	KF	26.9	1.825	0.319
Flying	oneshot	23.7	1.636	0.525
Flying	KF	24.9	1.632	0.438

which flew at a height of about 4m. The wind speed at the day of the experiment was 15Km/hr. Due to the wind and the noise of the UAV sensors, the UAVs were not stable and drifted within about a meter. The UAV operators frequently intervened to stabilize them.

Based on channel estimation performed before the experiment, the coherence time was estimated to be about $\tau_c = 85ms$. Thus, the lab experiment BF cycle ($t_{\text{cyc}} = 180ms$) is too long for the UAV channel. For the BF to work from the UAVs, the BF cycle was redesigned to have shorter guard times and a 10 times shorter payload as detailed in Table 3.2, yielding a reduced $t_{\text{cyc}} = 75ms$, which is shorter than τ_c but only with a small margin. The experiment was performed with three settings: 1) UAVs were on the ground and used KF for frequency synchronization, 2) UAVs were flying and used oneshot for frequency synchronization and

3) UAVs were flying and used KF. The BF results are shown in Table 3.3 along with the average SNR. The BF UAVs attained about 80% of the ideal BF gains despite the low coherence time channel. These gains are lower than the ground scenario as expected because of the shorter coherence time. As for the comparison between KF and oneshot, there is no significant difference because r/q is small; r is small because of the high SNR and q is large because of the short coherence time.

3.6.3 Emulation

To overcome the large delays of the BF implementation and have a fair comparison between KF and Oneshot, we emulated BF over a channel trace. The channel trace was obtained by capturing a repeating ZC sequence from a flying UAV over a period of 100s. Using this trace, we emulated the BF protocol as follows; we used a duration t_{syn} to estimate the frequency offset and corrected for it, then we estimated the phase offset after a delay equivalent to the protocol $(t_{\text{g1}} + Nt_{\text{ph}} + t_{\text{g2}})$ and corrected for it. The feedback stage was not emulated and was assumed to be ideal. At the evaluation time t_e, we estimated the phase error ϕ^e which for a static channel and perfect estimation should equal zero. The variance of ϕ^e calculated by emulation over the entire trace provides an estimate of σ_e^2 if BF was applied in this channel. Note that the channel trace was collected over one capture with a USRP operating in half-duplex. Hence, the emulation over that trace does not capture distortions due to burst transmissions and having both transmit and receive chains powered on simultaneously in the protocol implementation.

The measured phase errors are reported in Table 3.4. The first row emulates the timing used in the UAV experiment and using the value of $\tau_c = 85ms$, which is the true one, to calculate the KF q. The calculated phase error variance σ_e^2 is shown for KF and oneshot, and the theoretically predicted mean BF gain G using (3.27) and $N = 2$. Due to the more favorable half-duplex capture and the ideal feedback, the predicted emulation BF gains (≈ 1.7) are better than the measured ones (≈ 1.6). For the relatively long BF packets at a

Table 3.4: BF Emulation over UAV Channel Trace

#	SNR	t_{cyc}	τ_c	KF σ_e^2 [G]	Onesh. σ_e^2 [G]
1	24dB	75ms	85ms	0.569 [1.725]	0.567 [1.723]
2	24dB	18ms	85ms	0.11 [1.99]	0.141 [1.98]
3	0dB	18ms	85ms	0.573 [1.85]	0.4 [1.72]

the high SNR of the capture, the predicted BF gains using both oneshot and KF are very close (1.725 abd 1.723) similar to our experimental results. Yet the BF gains are still below 2 due to the long BF cycle, so in row 2, we emulate the protocol using a shorter cycle of $18ms$ by scaling down t_e and the phase delay. Using this shorter cycle, the BF gains increase significantly for both KF and oneshot and approach the ideal gain of 2. This is the result we would expect using an optimized implementation of the BF protocol having shorter guard times. Due to the high SNR, both KF and oneshot still give a similar performance. Then in row 3, we added Gaussian noise to the channel trace to make its SNR drop to 0dB. The expected BF gains for KF become significantly better than those from oneshot. This result shows that if $t_{\text{cyc}} << \tau_c$, KF can attain significantly higher BF gains than oneshot for distantly deployed UAVs.

3.7 Beamforming System Design

After verifying the framework, we discuss how it can be used in designing BF systems. To design a reliable BF system, we need to specify the number of BF radios N and the duration of the preambles to exceed a minimum post-BF SNR with a given probability. The design procedure is over two steps; first we determine N and σ_e^2 that meet the requirements and then we design the preambles' lengths to realize σ_e^2 at a given pre-BF SNR. Later in Section 3.8, we apply the proposed design procedures for specific scenarios.

3.7.1 Specifying N and σ_e^2

Although the pre-BF SNR (γ_{preBF}) is assumed constant, due to the phase error variance, the post BF SNR (γ_{postBF}) varies randomly. A reliable BF system has to exceed a specified outage probability p_{out} such that $P\left(\gamma_{\text{postBF}} < \gamma_{\min}\right) \leq p_{\text{out}}$, where $\gamma_{\min}$ is the minimum SNR. Using the gamma approximation of the BF gain distribution and the post-BF SNR definition (3.8), we can rewrite $P\left(\gamma_{\text{postBF}} < \gamma_{\min}\right) = 1 - F_{X_\gamma}\left(N - \frac{\gamma_{\min}}{\gamma_{\text{preBF}} N}\right)$ where $F_{X_\gamma}(x)$ is the CDF of the Gamma distribution from Proposition 7 whose mean and variance depend on N and σ_e^2. Hence, our objective is to determine N and σ_e^2 which satisfy

$$F_{X_\gamma}\left(N - \frac{\gamma_{\min}}{\gamma_{\text{preBF}} N}\right) \leq 1 - p_{\text{out}} \tag{3.31}$$

We know the distribution of X_γ and how N and σ_e^2 affect it, however, inverting (3.31) to obtain an explicit relation between N and σ_e^2 is intractable. The fact that σ_e^2 can depend on N under fixed BF overheads further complicates analytical solutions. It is easy, however, to check whether a given choice of N and the corresponding σ_e^2 satisfies the requirements given by (3.31). Thus, we resort to numerical trial-and-error methods to find N and σ_e^2 satisfying the requirements. The exact method depends on the scenario and whether N is fixed or not, and thus its discussion is deferred to Section 3.8 where example scenarios are presented.

3.7.2 Beamforming Signals Design

For given values of N, γ_{preBF}, and γ_{DR}, we want to optimize the time allocated to each preamble for σ_e^2 to meet the system requirements. We identify two problems of interest; the first one is to minimize σ_e^2 for limited BF overheads and the second problem is to minimize the BF overheads N_{ov} to meet a maximum allowable phase error variance. The first problem is suitable for short coherence time channels, where the BF overheads are constrained to allow time for communication within the coherence time. The second problem, on the other hand, is suited for relatively large coherence time channels, where large BF overheads are possible. An example of each problem is provided later in Section 3.8.

Next, we formulate both problems. The total overheads in samples defined in (3.11) can be written as a function of the duration of each stage $N_{\text{ov}}(N_{\text{syn}}, N_{\text{ph}}, N_{\text{fb}})$. For fixed N, γ_{preBF} and γ_{DR}, the phase variance σ_e^2 becomes a function of the number of samples allocated to each stage $\sigma_e^2(N_{\text{syn}}, N_{\text{ph}}, N_{\text{fb}})$ defined as

$$\sigma_e^2(N_{\text{syn}}, N_{\text{ph}}, N_{\text{fb}}) = (2\pi t_e)^2\sigma_f^2 + \sigma_{ph}^2 + \sigma_{fb}^2 \tag{3.32}$$

The values of σ_f^2, σ_{ph}^2, and σ_{fb}^2 are dependent on the choice of estimators and are a function of N_{syn}, N_{ph}, and N_{fb} respectively. For our choice of estimators $\sigma_{ph}^2 = \sigma_{phe}^2$ defined by (3.24), and $\sigma_{fb}^2 = \sigma_{fbe}^2$ defined by (3.26). As for the frequency error variance, if we use oneshot estimation $\sigma_f^2 = \sigma_{fe}^2$ as defined by (3.14) and if we use the KF $\sigma_f^2 = \sigma_{fk}^2$ as defined by (3.23) with $r = \sigma_{fe}^2$.

Note that for a chosen Zadoff-Chu sequence of a length M, we can only optimize the number of repetitions N_{ZC} to change N_{syn}. Hence, for fixed N, the problem P1 can be written as

$$P1 : \underset{N_{\text{ZC}}, N_{\text{ph}}, N_{\text{fb}}}{\text{minimize}} \quad \sigma_e^2(N_{\text{ZC}}M, N_{\text{ph}}, N_{\text{fb}}) \tag{3.33}$$

$$\text{subject to} \quad N_{\text{ov}}(N_{\text{ZC}}M, N_{\text{ph}}, N_{\text{fb}}) \le \delta_{N_{\text{ov}}}, \quad N_{\text{ZC}}, N_{\text{ph}}, N_{\text{fb}} \in \mathbb{Z}^+, \quad N_{\text{ZC}} \ge 2 \tag{3.34}$$

where $\delta_{N_{\text{ov}}}$ is maximum overhead length which depends on the channel coherence time, and $\mathbb{Z}^+$ is the set of positive integers. For a maximum allowable phase error $\delta_{\sigma_e^2}$, the second problem P2 can be written as

$$P2 : \underset{N_{\text{ZC}}, N_{\text{ph}}, N_{\text{fb}}}{\text{minimize}} \quad N_{\text{ov}}(N_{\text{ZC}}M, N_{\text{ph}}, N_{\text{fb}}) \tag{3.35}$$

$$\text{subject to} \quad \sigma_e^2(N_{\text{ZC}}M, N_{\text{ph}}, N_{\text{fb}}) \le \delta_{\sigma_e^2}, \quad N_{\text{ZC}}, N_{\text{ph}}, N_{\text{fb}} \in \mathbb{Z}^+, \quad N_{\text{ZC}} \ge 2 \tag{3.36}$$

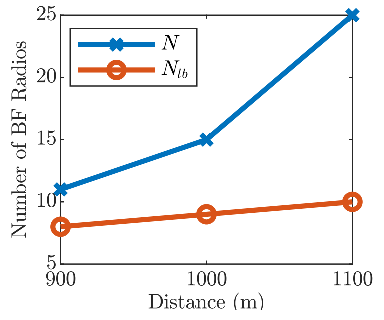
Then, we argue that for our choice of estimators, both problems are convex with respect to their variables and thus are easy to solve. Except for the KF, all these estimators take the form $f(x) = \frac{c_1}{x} + \frac{c_2}{x^2}$ with respect to their variables for some positive c_1 and c_2 where x is strictly positive, hence they are all convex over their domain. As for the KF, when substituting for r, it takes the form $f(x) = c_1 + \sqrt{c_2 + \frac{c_3}{x} + \frac{c_4}{x^2}}$ with respect to its positive variable x for some positive c_1, c_2, c_3 and c_4. This can be rewritten as $f(x) = c_1 + \|\mathbf{y}\|$ where $\mathbf{y} = [\sqrt{c_2}, \frac{\sqrt{c_3}}{\sqrt{x}}, \frac{\sqrt{c_4}}{x}]^T$. The norm is convex and non decreasing and $\frac{\sqrt{c_3}}{\sqrt{x}}$ and $\frac{\sqrt{c_4}}{x}$ are convex for positive x. By applying the composition rule [BV04], the KF variance is convex. Hence, $\sigma_e^2(N_{\text{ZC}}M, N_{\text{ph}}, N_{\text{fb}})$ is convex with respect to its arguments for all of our estimators. As for $N_{\text{ov}}(N_{\text{ZC}}M, N_{\text{ph}}, N_{\text{fb}})$, it is an affine combination of its arguments. This makes both problems P1 and P2 integer convex problems, which can be optimally solved using CVX with a mixed integer solver [GB14].

3.8 Beamforming Design Scenarios

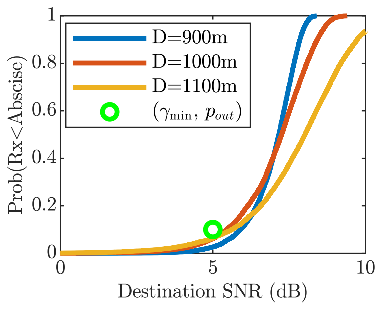
The proposed BF framework and the derived relations can be applied to many BF scenarios. In this ection, we discuss the design procedures for two example scenarios. In the first example, we consider a large swarm of small UAVs; we want to determine the minimum N to satisfy the SNR requirements. Due to the UAVs' high mobility, the channel coherence time is small and the BF overheads are constrained. This example maps to the problem P1. In the second example, we consider $N = 4$ weather balloons sending short payloads. Due to the long coherence time resulting from the balloons slow motion, large BF overheads are possible. However, to avoid energy wasted on unneeded transmission, our objective is to minimize the BF overheads while satisfying the SNR requirement. This example maps to the problem P2.

3.8.1 Swarm of Small UAVs

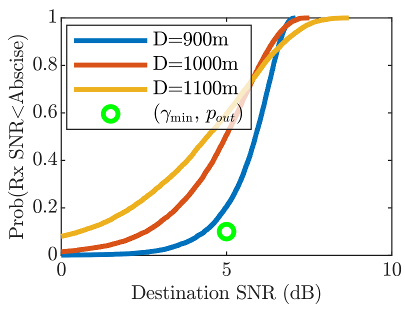
A swarm of N_{ub} small UAVs is deployed in an urban environment for an application like crowd monitoring [TMD20]. A large number of small UAVs is deployed and they continuously



(a) Number of BF Radios obtained assuming ideal BF (N_{lb}) and obtained using our framework (N).



(b) CDF of simulated BF gains using N (from our framework). The requirement given by $\gamma_{\min}$ and p_{out} is satisfied.



(c) CDF of simulated BF gains using N_{lb} (assuming ideal BF). The requirement given by $\gamma_{\min}$ and p_{out} is violated.

Figure 3.7: Results for minimizing N in a swarm of small UAVs assuming a fixed BF overhead. Using N obtained from our approach, the SNR requirement is satisfied as verified by simulations.

transmit data. To avoid a large overhead in data sharing among UAVs for BF, we want to determine the minimum number of UAVs to beamform such that the destination SNR exceeds a minimum of $\gamma_{\text{min}} = 5dB$ for 90% of the time ($p_{\text{out}} = 0.1$).

For the urban channel, we consider a channel having a path loss coefficient of 3.7 [Gol05] and a coherence time of 10ms [CFP18]. The maximum transmit power of each UAV is $P_T = 0dBm$ and of the destination $P_T^D = 20dBm$. Communication takes place over a frequency of 915MHz using a sampling time of $T_s = 1\mu s$ and a BW of 1MHz and all radios have a noise figure of 3dB. By performing the link budget calculation, the SNR from an individual UAV at 1Km is close to -13dB, so the minimum required BF gain is $G_{\text{req}} = 18$dB. Assuming ideal BF gain of N^2, the required gain can be achieved using only 8 BF radios. However, due to the short coherence time, the entire BF packet is assumed to be limited to 5ms and based on the payload required by the application only 1ms of BF overhead is allowed. At this low SNR and with this short BF overhead, the ideal beamforming gains are not achievable and large BF variance is expected. We need to use more than 8 BF radios so that the SNR exceeds 5dB for 90% of the time as required. Our objective is to determine the minimum N and the duration of each preamble.

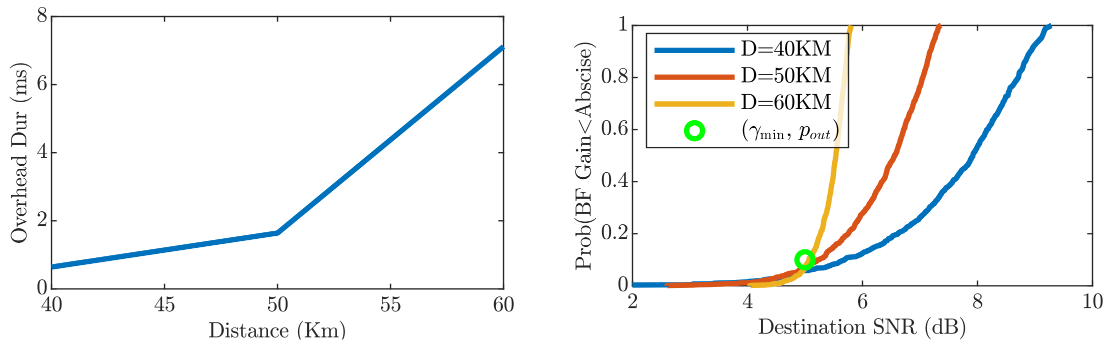
We use our analytical framework to find the minimum N. Since for fixed overheads N_{ov}, σ_e^2 depends on N, we need to solve P1 to calculate σ_e^2 for each N. The proposed approach is summarized in Algorithm 3 and it works as follows; we start from the lower bound on N, which occurs when assuming ideal BF $N_{\text{lb}} = \lceil \sqrt{G_{\text{req}}} \rceil$ and increment N until the requirement is satisfied. For each N, we solve the minimum phase error problem $P1$ to obtain σ_e^2. Using the resulting σ_e^2, we substitute in (3.31) to determine if the requirement is satisfied or not. The first N satisfying the requirement is the minimum N meeting the SNR requirements. If the maximum number of available BF radios N_{ub} was reached without satisfying (3.31), another approach needs to be considered to meet the requirements like increasing the BF overhead or the transmit power of the radios. Since the BF is performed periodically and $t_{\text{cyc}} < \tau_c$, we assume that KF is used for frequency tracking.

The calculated N for different distances is shown in Fig. 3.7a along with N_{lb} calculated assuming ideal BF gain. To verify that the obtained solution meets our design criteria, we simulated 10K BF cycles of the BF protocol using using the calculated N and the optimized waveforms obtained from $P1$ at each distance. The destination SNR was measured and its empirical CDF for the proposed N and the ideal N_{lb} are plotted in Fig. 3.7b and 3.7c respectively. From these Figures, we see that the required outage probability is met using the proposed N. Thus, our problem solution and the underlying analysis can be used to design reliable BF systems satisfying the design requirements as verified by simulations. On the other hand, relying on the ideal N_{lb} is expected to yield lower BF gains than the desired ones in realistic deployment scenarios.

3.8.2 Weather Balloons

Weather balloons are deployed at high altitudes to perform atmospheric measurements and report them back to the ground. We consider $N = 4$ weather balloons deployed at a distance

```
Algorithm 3:
input  : $N_{\text{lb}}, N_{\text{ub}}, N_{\text{ov}}, p_{\text{out}}, \gamma_{\min}$
output: Solved, $N$, BF waveform
Solved := False ;
for $n_i = N_{\text{lb}}$ to $N_{\text{ub}}$ do
    Solve P1 to determine $\sigma_e^2$;
    if $n_i$ and $\sigma_e^2$ satisfy (3.31) then
        Set Solved to True, $N$ to $n_i$, and BF waveform to solution of P1, and exit ;
    end
end
```



(a) The minimum BF overhead to satisfy the SNR requirement.

(b) CDF of simulated BF gains using the minimum overhead. The SNR requirements given by $\gamma_{\min}$ and p_{out} are met.

Figure 3.8: For $N = 4$ weather balloons, in a long coherence time channel, the minimum overheads obtained using our framework satisfy the SNR requirements.

of 50KM from the destination radio. Due to their high altitude, the channel is dominated by line-of-sight propagation and we consider a path loss coefficient of 2 and a large channel coherence time exceeding 100ms. The large channel coherence time allows for much longer BF overheads. However, to economize the balloon payload battery power, we want to minimize the transmission time. Our objective is to determine the smallest BF overheads to attain a received SNR exceeding a minimum of $\gamma_{\min} = 5dB$ for 90% of the time ($p_{\text{out}} = 0.1$). We use the same power, frequency, bandwidth, and noise parameters as the previous scenario except $P_T = 10$dBm is larger. The SNR from a single radio is -4.6dB and thus the required BF gain at 50KM distance is 9.6dB. Assuming that the measurements are infrequent and not periodic, we use oneshot frequency estimation.

To design this system, we find the minimum phase error needed to satisfy the requirement ($\delta_{\sigma_e^2}$), then we find the shortest overhead to meet this phase error. Since, for fixed N, increasing σ_e^2 decreases the average BF gain and vice versa, we determine $\delta_{\sigma_e^2}$ by applying the bisection method on (3.31). Then, using $\delta_{\sigma_e^2}$, we solve the problem $P2$ to determine the minimum overhead. If the minimum overhead makes the BF packet exceed the channel coherence time, the solution is not valid and we need to consider another alternative like increasing the transmit power. The minimum overheads obtained are shown in Fig. 3.8a for different distances. Then, we simulated the BF protocol at these SNRs using the waveforms obtained from P2 and plotted the empirical CDF of the destination SNR in Fig. 3.8b. We can see that the proposed solution approach meets the required outage probability, which verifies the solution and all the underlying analysis.

3.9 Summary

In this Chapter, we developed and verified a mathematical framework to model the BF performance for a destination-led BF protocol. The BF gains distribution was approximated by a gamma distribution assuming a zero mean normally distributed combining phase errors and the proposed distribution was verified using simulations. The effect of the pre-BF SNR and the preamble lengths on the combining phase error was derived for our choice of estimators. Using software-defined radios, in a lab, we experimentally verified the predictions of our BF framework. The BF radios were mounted on UAVs and were shown to exceed 80% of the ideal BF gains despite the low coherence time channel. The proposed framework can be used to design BF systems for a given deployment as illustrated by two example scenarios.

Even though we only considered a specific BF protocol and only two example scenarios, the proposed framework can support many protocol variations and use cases. For the protocol, the framework is applicable for any other choice of estimators as long as their phase variance can be expressed mathematically. As for the scenarios, heuristics can easily be developed to optimize over a combination of the SNR, preamble lengths, and the number of BF slaves, enabling the framework to adapt to many different deployment scenarios.

CHAPTER 4

Guided Beamforming

Distributed transmit beamforming enables cooperative radios to act as one virtual antenna array, extending their communications' range beyond the capabilities of a single radio. Most existing distributed beamforming approaches rely on the destination radio sending feedback to adjust the transmitters' signals for coherent combining. However, relying on the destination radio's feedback limits the communications range to that of a single radio. Existing feedback free approaches rely on phase synchronization and knowing the node locations with sub-wavelength accuracy, which becomes impractical for radios mounted on high-mobility platforms like UAVs. In this chapter, we propose and demonstrate a feedback free distributed beamforming approach that leverages the radio's mobility and coarse location information in a dominant line-of-sight channel. In the proposed approach, one radio acts as a guide and moves to point the beam of the remaining radios towards the destination. We specify the radios' position requirements and verify their relation to the combined signal at the destination using simulations. A proof of concept demo was implemented using software defined radios, showing up to 9dB SNR improvement in the beamforming direction just by relying on the coarse placement of four radios. This chapter revises our previous publication [HKC21a].

4.1 Introduction

Distributed transmit beamforming (BF) is a cooperative communications technology that enables a group of radios to act as a virtual antenna array. In distributed transmit beamforming, a group of synchronized radios with the same message adjusts their signals to ensure coherent combining at the destination. For a group of N radios, distributed beamforming provides N^2 increase in the received power [MBM09]. This power increase can be used to extend the communications range or reduce the power transmitted from the radios. Both these factors are of great importance for unmanned aerial vehicles (UAVs), which have a limited power budget and are advantageous to deploy in large numbers at remote areas [SSA19].

To achieve coherent combining using BF, the radios need to synchronize their carrier frequencies and their symbol timing as well as adjust their phases to ensure coherent combining at the destination [MBM09]. Synchronization needs to happen among the beamforming radios and it does not depend on the destination radio. The phase correction, however, depends on the destination. There are two methods to adjust the phases for distributed beamforming [JRL17, OMP05]: the first one relies on the destination radio assisting the nodes in obtaining channel phase estimates, while the second method relies on the nodes knowing their locations and the beamforming direction.

The first approach assumes that the destination can communicate with the beamforming radios. This communication can be in the form of a preamble transmitted from the destination [OMP05], or the destination sending back the channel estimates [YP02], or just providing binary feedback with the BF radios randomly perturbing their phases [MHM10]. Many of these methods were demonstrated using software defined radios [JRL17]. However, while these approaches can correct the phase to attain coherent combining, they rely on the destination radio having sufficient transmit power to reach the BF radios which limits the communication range to that of the destination radio regardless of how many BF radios are used.

The second approach does not need any feedback from the destination and relies only on the nodes knowing their relative locations. Using this information and the direction towards the destination, the radios can calculate the phases needed for beamforming [JRL17]. However, to have the full beamforming gains using this approach, location information accurate to a fraction of a wavelength is necessary, and the gains degrade rapidly due to localization errors [OMP05]. This requirement places stringent localization requirements and limits the applicability of this approach for high mobility platforms like UAVs where typically only coarse location is available using satellite navigation systems. Additionally, this approach assumes that the BF nodes are aligned in phase which is not easy to realize using radios having independent oscillators. To the best of the authors' knowledge, no demonstration of distributed BF relying only on locations was implemented in the literature. Another approach avoids destination feedback by relying on the randomness of the combining gain from unsynchronized radios along with repeating transmissions [SAB13]. However, this approach is not scalable and has a low throughput.

In this chapter, we propose guided distributed beamforming as an approach for cooperating mobile radios to attain coherent combining at a distant destination radio unable to provide feedback. To overcome the lack of feedback, the BF radios, which are assumed to be in proximity of each other, need to know the beamforming direction to the destination and have a LOS channel between them. These radios can be mounted on ground robots, UAVs, or handheld as long as they can be coarsely positioned relying on satellite navigation for instance. Guided distributed BF relies on assigning one of the BF radios as a guide and the rest as followers. The followers adjust their signals to ensure coherent combining at the guide. Since the guide is close to the followers, it can provide them with feedback for BF unlike the destination. Using radios mobility and coarse localization, the followers cluster and the guide moves towards the desired BF direction to point the combined signal at the desired BF direction. We verify this concept using simulations and analyze the position requirements of the guide and followers along with its sensitivity to localization errors and non-LOS channel components. Then, we demonstrate this approach using software defined radios. Using 4 BF radios we were able to attain more than 3 fold increase in the signal magnitude (9x increase in power received) towards the direction of interest. To the best of the authors' knowledge, this is the first demonstration of distributed beamforming that achieves coherent combining at the destination without any destination feedback and without sacrificing throughput with repeated transmissions. This approach can be used to extend the range of communications towards a distant destination radio unable to provide any feedback to the BF radios. Our main contributions can be summarized as follows:

- We developed guided distributed beamforming as an approach to enable beamforming towards a destination unable to provide feedback, assuming a LOS channel between the BF radios. Our proposed approach leverages the radio's mobility and coarse localization to achieve coherent combining at the destination.
- We derived a criterion for the guide and the followers positions to limit the phase mismatch of the combining signals at the destination.
- Using simulations, we showed that the proposed approach can tolerate BF radios location errors within multiple wavelengths in contrast to location based beamforming, which requires location accuracy within a fraction of a wavelength.
- We experimentally verified the proposed approach using software defined radios. An average signal combining gain of over 3x was achieved in the intended direction when using 4 beamforming radios leading to 9dB SNR improvement on the average. The combining gains measured in different directions were shown to follow the expected BF pattern predicted by simulations.

4.2 Related Work

As discussed earlier, existing BF approaches either rely on destination feedback or highly accurate knowledge of radio locations.

Destination Feedback Many existing works have relied on destination feedback for coherent combining [MBM09, JRL17]. A system using explicit channel feedback was proposed in [YP02] and demonstrated in [LGS18, MBM19]. To reduce the feedback overhead, a 1-bit feedback algorithm was developed [MHM10]. Using this approach, the nodes randomly perturbate their phase and the receiver provides binary feedback indicating whether the channel has improved or no. This approach was used in several experimental evaluations of distributed beamforming for instance [QMR12, RBM12a]. Joint location and beamforming optimization was considered using destination feedback in [GPY20, GYP20]. Motion and communications energy was optimized for mobile robots in [MM18] using destination feedback along with channel predictions. In [IP08], a synchronization algorithm based on roundtrip message exchanges was developed. Other works have proposed using channel reciprocity for channel estimation [OMP05] and this approach was demonstrated in [PMG16]. All these approaches are not feasible if the destination does not have sufficient transmit power to provide feedback.

Location Based Beamforming Other works have relied on the knowledge of the locations for the beamforming radios to adjust the phases. These works mostly focused on either studying the beampattern of random placements of radios or optimizing the beampattern [JRL17]. In [OMP05], the beampatterns obtained using uniform random deployments of transmitters within a disk area was considered. The effects of phase jitter and location estimation errors on the beampattern were studied. Other works have studied the beampattern of beamforming nodes following a Gaussian distribution [AV09] or arbitrary distributions[HWW12]. Among the works that considered beampattern optimization, some have proposed using node selection or coefficient perturbation to create a null in a certain direction [KDS20], minimize the beamwidth [ZAG09], or control the sidelobes [LFS19, SLC19]. These approaches typically rely on accurate localization and to the best of the authors knowledge there are no implementations of location based distributed beamforming.

Other works have proposed and demonstrated zero feedback beamforming [SB11, SAB13], which works by sending multiple repetitions of the signal and using the fact that unsynchro-

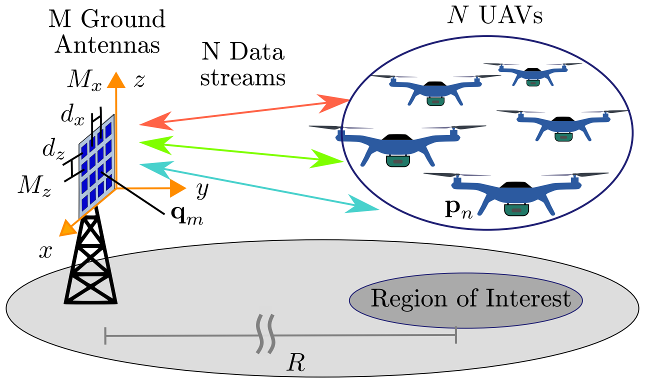


Figure 4.1: The objective of distributed beamforming is to coherently combine signal from N radios, which are assumed to be mobile, at a distant destination radio.

nized carriers occasionally combine constructively. While this approach avoids relying on destination feedback, it negatively affects the throughput and is not scalable.

4.3 Problem Statement

Consider N mobile radios that use BF to send a critical message to a distant destination radio $\boldsymbol{R}$. The BF vehicles are cooperating on the same task and hence are assumed to be close to each other and far from the destination radio $\boldsymbol{R}$ beyond its communication range. Hence, the destination cannot provide any feedback for BF. The BF radios are assumed to know the beamforming direction toward the destination and their locations coarsely using a satellite navigation systems like GPS. For example, the radios can be mounted on UAVs performing a search and rescue operation in a remote area. They are communicating with the destination radio placed at the operation center established at their takeoff location. Due to their elevation above the ground and assuming a deployment in a remote non-urban area, an air-to-ground channel is dominated by LOS propagation [KCZ18]. Hence, the knowledge of locations is sufficient for the BF radios to determine the BF direction without feedback from the destination.

BF radio i is located at $\mathbf{p}_i = [p_i^x, p_i^y, p_i^z]^T$, where p_i^x, p_i^y, p_i^z are the x, y, and z coordinates of the node i, with respect to node 0 which is used as a reference, i.e, $\mathbf{p}_0 = [0, 0, 0]^T$. The destination is located at $\mathbf{p}_R = [p_R^x, p_R^y, p_R^z]^T$ in the far field of the BF radios such that $d_{i,R} \gg d_{i,j}$ for all i and j from 0 to $N-1$, where the distances are defined as $d_{i,R} = \|\mathbf{p}_i - \mathbf{p}_R\|$ and $d_{i,j} = \|\mathbf{p}_i - \mathbf{p}_j\|$. Without a loss of generality, we assume that the known BF direction is the positive x. If a LOS channel exists towards the destination, the destination receiver would be located far on the x-axis such that $|p_R^x \gg \sqrt{(p_R^y)^2 + (p_R^z)^2}$. This setup is shown in Fig. 4.1.

Assuming that the BF radio are synchronized in time and frequency using an over-the-air synchronization protocol as discussed later in Section 4.6, the signal transmitted by radio i is given by

$$x_i(t) = \Re\{s(t) w_i e^{j2\pi f_c t}\} \tag{4.1}$$

where $s(t)$ is the complex baseband message, f_c is the carrier frequency, and $\Re\{\cdot\}$ denotes the real part. To ensure coherent combining at the destination, each radio precodes its signal with a complex weight $w_i = \kappa_i e^{j\theta_i}$, where κ_i is the magnitude and θ_i is the phase.. The received signal at R is given by

$$y(t) = \sum_{i=0}^{N-1} \Re\{w_i h_i e^{j2\pi f_c t} s(t)\} + \gamma(t) \tag{4.2}$$

$$= \sum_{i=0}^{N-1} \Re\{\kappa_i |h_i| e^{j(2\pi f_c t + \theta_i + \angle h_i)} s(t)\} + \gamma(t) \tag{4.3}$$

where the narrowband channel is given by $h_i = |h_i| \exp(j\angle h_i)$, $|h_i|$ is its magnitude, $\angle h_i$ is its phase, and $\gamma(t)$ is the additive Gaussian noise.

The normalized magnitude of the beamforming gain is the ratio between the attained combining gain and the ideal combining gain and is given by

$$\Gamma = \frac{|\sum_{i=0}^{N-1} \kappa_i |h_i| \cdot e^{j(\theta_i + \angle h_i)}|}{\sum_{i=0}^{N-1} \kappa_i |h_i|} \tag{4.4}$$

and it takes a value between 0 and 1. Perfect coherent combining at the destination occurs, if combining phases $(\theta_i + \angle h_i)$ are equal for all i, which this corresponds to $\Gamma = 1$. A phase mismatch between the combining signals will lead to degraded BF gains.

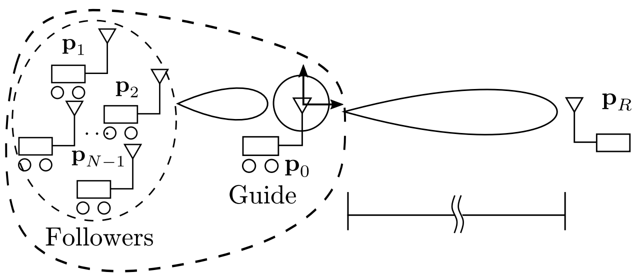


Figure 4.2: In the proposed approach, the guide repositions itself such it lies on the line connecting the center of the cluster of followers to the destination.

Since the radios are separate, each radio has a maximum transmit power given by P_T independent of the other radios. To maximize the power at the end receiver, it is optimal for each radio to set its gain to the maximum which is given by $\kappa_i = \sqrt{P_T}$ regardless of the channel magnitude $|h_i|$ [MBM09]. Hence, our objective is to find the phases θ_i for coherent combining at the destination receiver. Since $|h_i|$ has no impact on the phase, for simplicity, we consider that $|h_i| = 1$ making $h_i = e^{j\angle h_i}$ for all i.

Our objective is to determine the beamforming phases θ_i and the BF radio positions $\mathbf{p}_i$ for $i \in \{0, \cdots, N-1\}$, to ensure coherent combining at the destination receiver (large Γ). To enable BF beyond the communications range of the destination, we do not rely on its assistance in calculating θ_i. Instead, we rely on the cooperation of the mobile radios under coarse localization using guided distributed BF.

4.4 Guided Distributed Beamforming

We start by explaining the concept behind guided distributed beamforming, then we analyze its requirements in terms of node positions, and its impact on the phases of the combining signals at the destination.

4.4.1 Approach

The proposed approach consists of having one of the BF radios act as a guide to the remaining radios, which are referred to as the followers. The followers, using feedback from the guide, adjust their phases for coherent combining at the guide. By leveraging the radio's mobility, the guide moves to be on the line originating from the centroid of the followers towards the desired beamforming direction, making the beamformed signals have a large combining gain at the destination receiver as shown in Fig. 4.2. It is easy to see that if the guide was placed in the close vicinity of the destination, coherent combining at the guide would imply a large combining gain at the destination. However, having one of the beamforming nodes move near the destination defeats the purpose of beamforming. We want to attain the beamforming gain at the destination receiver without having any of the nodes travel a large distance. To that end, we study the relation between the positions of the nodes and the combining gains.

Without loss of generality, we assume that the node 0 acts as the guide and that the followers are nodes 1 to $N-1$. To use guided distributed BF, the followers need to cluster around the x-axis ($\sum_{i=1}^{N-1} p_i^y \approx 0, \sum_{i=1}^{N-1} p_i^z \approx 0$) with $p_i^x \leq 0$ for $i \in \{1 \cdots N-1\}$, making the line between the cluster of followers and the guide (which is the coordinate reference) point towards the BF direction. If a LOS channel exists with the destination, the guide would lie on the line in between the followers and destination $p_R^y = 0, p_R^z = 0$ with $p_R^x \gg 0$. The phase of the channel between the guide and follower i is given by $\angle g_i$. Since the followers beamform towards the guide, they set their phases to $\theta_i = -\angle g_i$ where $\angle g_i$ is obtained using the guide's feedback. Hence, the guide is considered as a reference for phase for all the followers, i.e, $\theta_i + \angle g_i = 0$ for all i. In that case, the guide sends its signal without phase compensation, i.e, $\theta_0 = 0$, assuming it is hardware calibrated for phase reciprocity [GSK05]. Phase reciprocity implies that both its transmit and receive chains are phase calibrated to leverage channel reciprocity.

Note that since guided distributed beamforming relies on the radios motion to steer the

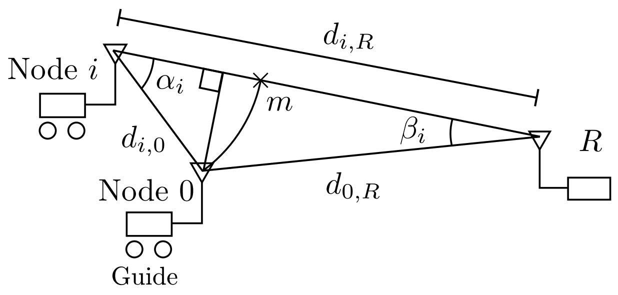


Figure 4.3: The signal at from radio i at point m and the signal transmitted from the guide (radio 0) have the same phase. The mismatch between the length of $\overline{mR}$ and $d_{0,R}$ leads to combining error at the destination.

beam, it is more suited to applications with one fixed destination (like the ground station of search and rescue UAVs) than those with multiple destinations. As stated earlier guided distributed beamforming only requires knowing the direction to point the beam and does not require a LOS channel with the destination nor knowing its exact location. However, using these assumptions it easier to determine the BF direction. Thus for the remaining of this work, we assume a LOS channel between the BF radios and the destination. This is the case if the destination radio is mounted on a high tower and the BF radios are ground based vehicles in a rural area or UAVs. Since the followers are adjusting their signals based on the guide and not the destination, the combining signals will have a phase mismatch at the destination. This phase mismatch will lead to degraded BF gains. We want to determine the separation between the followers and the guide to bound the phase mismatch and prevent the degradation of the BF gains.

4.4.2 Guide Separation

To analyze the phase mismatch caused by using the guide for feedback instead of the destination, we start by describing the geometry between follower node i, the guide, and destination as shown in Fig 4.3. The phase of node i signal at the guide is equal to zero (as stated earlier), which is the same phase at point m which lies on the line between i and R at an equal distance of $d_{i,0}$. Our objective is to have the phases of the signals from node i and from the guide to be equal at the destination. This is equivalent to having the segment $\overline{mR}$ to be equal to $d_{0,R}$. Since $\overline{im} = d_{i,0}$, this is equivalent to having $d_{i,R}$ to be equal to $d_{i,0} + d_{0,R}$. Using the triangle inequality, we know that $d_{i,R} \leq d_{i,0} + d_{0,R}$ with equality holding if and only if points i, 0, and R are on the same line. So, by placing the nodes on a perfect line pointing toward the receiver, beamforming towards the guide would guarantee coherent combining at the destination. However, in practice due to the imperfect positioning due to inaccurate location information or other mobility constraints due to the radio's mobility, this might not be practical to achieve. If the followers are not on the same line, there will be a mismatch in the propagation paths, which will eventually lead to phase mismatch. This path mismatch is given by

$$\begin{aligned} e_i &= d_{i,0} + d_{0,R} - d_{i,R} && (4.5)\\ &= d_{i,0}(1 - \cos(\alpha_i)) + d_{0,R}(1 - \cos(\beta_i)) && (4.6)\\ &\approx d_{i,0}(1 - \cos(\alpha_i)) && (4.7)\\ &\approx d_{i,0} - p_i^x && (4.8)\end{aligned}$$

where the approximations are due to the assumption that the receiver is in the far field making $\beta_i \approx 0$ and $\overline{iR}$ and $\overline{0R}$ are almost parallel, thus $\cos(\beta_i) \approx 1$ and $|p_i^x| \approx d_{i,0}\cos(\alpha_i)$.

This mismatch of the different propagation paths will be translated to a phase error between the signals from the guide and radio i given by

$$\phi_i = \frac{2\pi e_i}{\lambda} \tag{4.9}$$

After deriving the phase error for a single node, we generalize it for all the followers. For simplicity, we limit our analysis to the 2D case ($p_R^z = 0, p_i^z = 0$ for all i) although the geometry can easily be extended to the 3D case. Hence, we define a rectangle of dimensions $L_x = \max_{i,j} |p_i^x - p_i^x|$ and $L_y = 2\max_i |p_i^y|$ having the vertical line $y = 0$ at its center, which

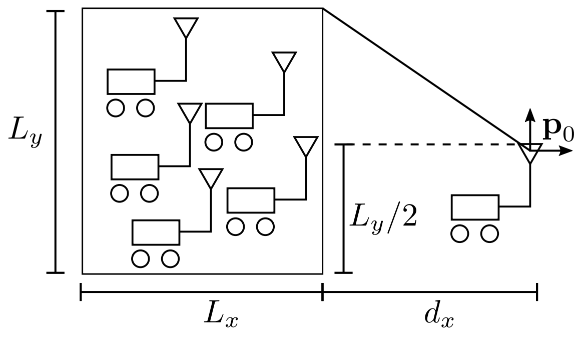


Figure 4.4: The followers locations are contained in a rectangle of dimensions L_x and L_y, which is symmetric around the x-axis. The separation between the guide and the followers is given by d_x.

contains all the followers. The distance between the guide and the closest receiver in the x-dimension is given by $d_x = \min_i |p_i^x|$. This is shown in Fig. 4.4. The upper bound of the propagation path mismatch due to using the guide, given by $e_{\max}$ is equal to

$$\begin{aligned} e_{\max} &= \max_i e_i && (4.10)\\ &= \max_i \sqrt{(p_i^x)^2 + (p_i^y)^2} - p_i^x && (4.11)\\ &\le \max_i \max_j \sqrt{(p_i^x)^2 + (p_j^y)^2} - p_i^x && (4.12)\\ &= \max_i \sqrt{(p_i^x)^2 + (L_y/2)^2} - p_i^x && (4.13)\\ &\le \sqrt{d_x^2 + (L_y/2)^2} - d_x && (4.14) \end{aligned}$$

where (4.12) adds another variable and can not decrease the maximization objective, and (4.14) uses the fact that the function $\sqrt{a + x^2} - x$ is a strictly decreasing function in x for any positive a and x, and that $d_x \le |p_i^x|$ for all i by definition. This makes the largest phase deviation from the guide equal to $\phi_{\max} = \frac{2\pi e_{\max}}{\lambda}$. The smaller $\phi_{\max}$, the larger the BF gains at the destination. The exact value of the BF gain (Γ) depends on the placements of the followers and is further considered in simulations.

Based on the tolerable amount of phase errors, we want to upper bound the mismatch

$e_{\max}$ by a chosen value of δ such that

$$e_{\max} \leq \delta \tag{4.15}$$

The smaller the value of δ, the smaller $\phi_{\max}$, which means less phase mismatch and larger BF gains. Note that the chosen δ has to be less than λ for the maximum phase error $\phi_{\max}$ to be less than 2π. By manipulating (4.14), the relation between the guide separation d_x and vertical spread of the follower L_y to realize (4.15) for a given δ is

$$d_x \geq \frac{(L_y/2)^2 - \delta^2}{2\delta} \tag{4.16}$$

Hence, the separation between the guide and the followers (d_x) to achieve a given mismatch δ scales quadratically with the vertical spread of the followers (L_y). For a given follower placement (fixed L_y), using a smaller δ to reduce the phase errors requires the guide to travel further to increase its separation. Hence, the choice of δ trades off between the distance traveled by the guide and the BF gains as we will illustrate using simulations.

4.5 Numerical Evaluation

Using numerical simulations, we study the impact of the arrangement of the followers and the separation of the guide on the beamforming gain. Then we compare guided beamforming with the approach relying only on location information under localization errors. The impact of non-LOS channel components on guided beamforming and feedback based beamforming are also evaluated.

In our simulations, we consider $N = 11$ beamforming radios (1 guide and 10 followers). The follower nodes are assumed to be randomly placed in a rectangle of dimensions $L_x \times L_y$ at a distance d_x from the guide as shown in Fig. 4.4. The receiver R is placed at a distance of 10KM from the guide. The frequency used in the simulation is 900MHz making the wavelength equal to 33.3cm. The channels are modeled as LOS channels with $\angle h_i = \frac{2\pi}{\lambda} d_{i,R}$ and normalized to have $|h_i| = 1$. The channel estimates between the guide and the followers

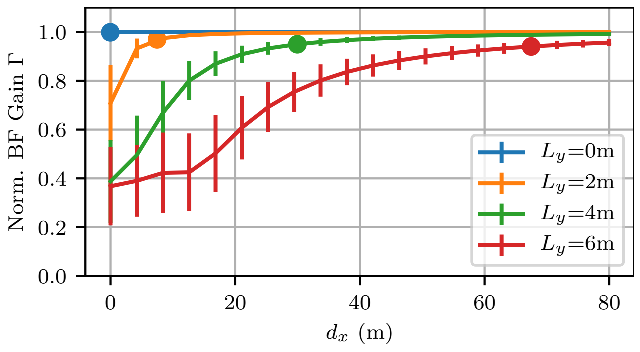


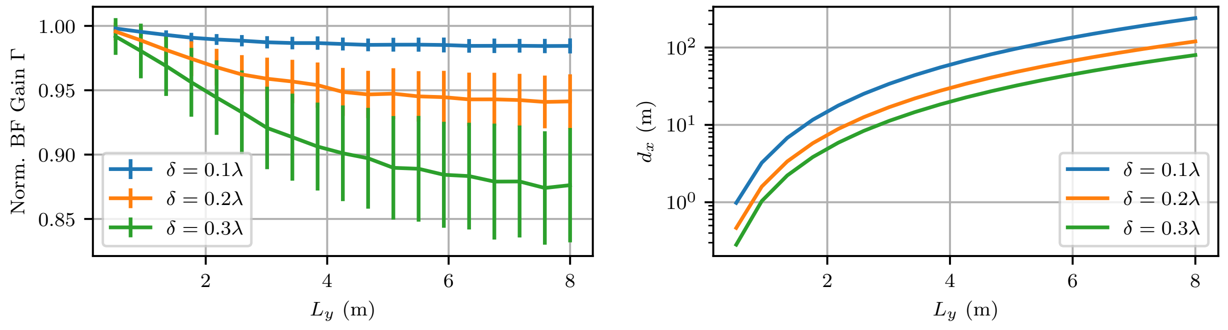
Figure 4.5: The BF gain of random placement of the followers for $L_x = 10m$ for different values of L_y as a function of the separation of the guide d_x.

are assumed to be perfect, making the combining gain at the guide always equal to 1. Our evaluation focuses on the beamforming gain of all the BF radios at the destination receiver.

4.5.1 Impact of BF Nodes Placement Geometry

First, we consider the effect of the separation between the guide and the followers d_x on the combining gain at the destination receiver Γ. This evaluation is performed for $L_x = 10m$ and for different L_y as shown in Fig. 4.5. For each point, we consider 100 uniform random placements of the followers within the deployment rectangle and plot the mean BF gain with error bars representing the standard deviation. We can see that as L_y increases the guide needs to be further from the followers to ensure coherent combining at the receiver. In the case of linear array, $L_y = 0$, the optimal combining gain can be attained with no separation ($d_x = 0$). In Fig. 4.5, the solid circles show the combining gain when using the optimal separation calculated using (4.16) for a tolerable mismatch given by $\delta = 0.2\lambda$. These circles show that using (4.16) and for this choice of δ, most of the BF gains are attained. The relation between δ and the BF gains is further discussed later.

To get an understanding of the separation between the guide and the followers as a function of the vertical spread of the followers (L_y), we plot the lower bound from (4.16) in Fig. 4.6b on a logarithmic scale. For our simulation setup, a vertical spread below 1m would



(a) The beamforming gain for mismatch tolerance δ as a function of L_y.

(b) The separation of the guide obtained using (4.16) for different value of the mismatch tolerance δ as a function of L_y.

Figure 4.6: The beamforming gain and the distance traveled for different values of mismatch tolerance δ.

require separation below 2m. For larger spreads up to 8m, the separation can be over 100m. This is expected since (4.16) is quadratic in L_y. Hence, it is beneficial to align the followers to avoid large displacement of the guide.

We also consider the effect of the chosen tolerance on the distance traveled and the combining gain. As predicted by (4.16), a tighter tolerance requires larger separation. In terms of combining gain at the end receiver, in accordance with the result in [MBM07], the combining gain is tolerant to phase errors which are due to mismatch between the guide and the destination. Fig. 4.6a shows that a tolerance of 0.2λ is able to attain over 90% of the combining gain while requiring at about half the separation of 0.1λ as shown in Fig. 4.6b. Note that the BF gains change for the same δ because it is an only an upper bound on the phase error. For the same δ, since the radios are randomly placed within a rectangle, the distribution of the phases vary with L_y and d_x leading to varying BF gains.

We also consider the beampattern obtained in the far field when the followers beamform toward the guide. In Fig. 4.7, we plot the far field beampattern when the followers are deployed randomly in a region of $L_x = 10$m and $L_y = 1$m with the guide placed at d_x to

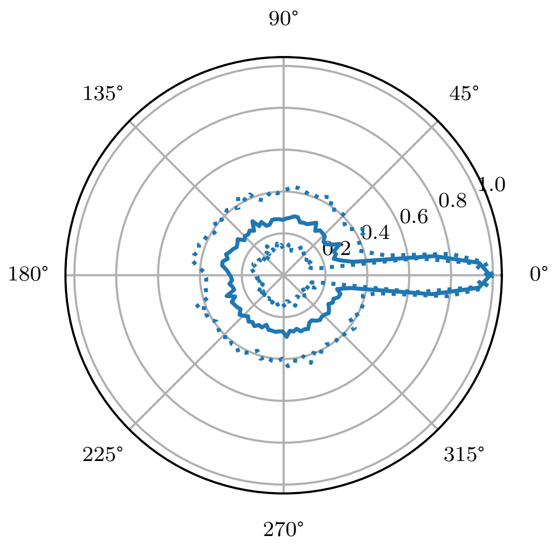
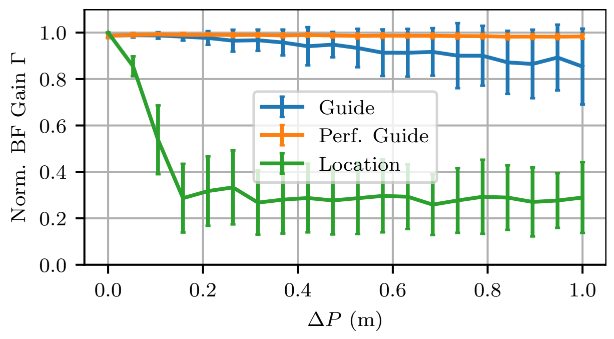


Figure 4.7: The beamforming pattern for $L_x = 10m$, $L_y = 1m$. The solid line is the average of 100 random placements and the dotted lines represent ± their standard deviation.

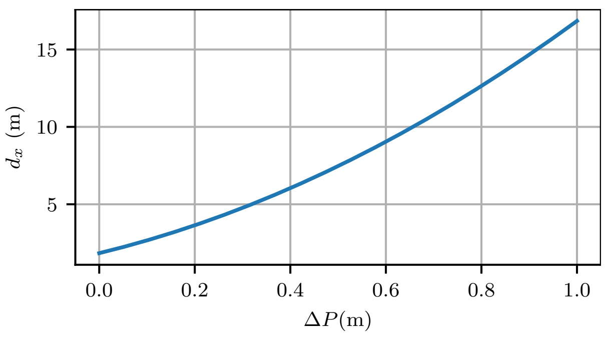
achieve $\delta = 0.2\lambda$ using (4.16). The solid curve shows the average beampattern of 100 random placements of the followers with the dotted curve representing the mean plus and minus one standard deviation. We can observe that the realized beampattern guarantees a narrow beam toward the destination, while in other directions a smaller combining gain is expected on the average.

4.5.2 Localization and Channel Induced Errors

We evaluate the sensitivity of our proposed approach to localization errors of the BF radios and compare it against using only locations to calculate the beamforming weights. The localization errors are modeled as uniform random variables Δp_i^x and Δp_i^y which are unknown to the nodes. These errors take values between $-\Delta P/2$ and $\Delta P/2$ added to the positions of the nodes where ΔP is the error range, such that node i would be located in $[p_i^x + \Delta p_i^x, p_i^y + \Delta p_i^y]$ for $i \in \{0, 1, \cdots, N-1\}$ (the origin is assumed to be fixed regardless of localization errors). For location based beamforming, we set the beamforming of node i to $\theta_i = e^{-j\frac{2\pi}{\lambda}d_i}$, where $d_i = p_i^x + L_x$. This choice of d_i only uses the known position p_i^x and ensures that the resultant



(a) The effect of localization errors for guided beamforming (Guide), guided beamforming assuming an the guide perfectly placed (Perf. Guide), and using location-based beamforming.



(b) The sepration of the guide as a function of the range of localization error ΔP.

Figure 4.8: The effect of localization error on the guide based beamforming and location based beamforming. The wavelength used is $\lambda = 0.33$m

wave from the radios is aligned pointing towards the receiver in the case of no localization errors. We consider the same initial setup with $L_x = 10$ and $L_y = 1$. For the guided beamforming, we accounted for the worst case localization error by increasing the separation d_x using $L_y = 1 + \Delta P$ in (4.16) for $\delta = 0.2\lambda$. We consider both the case where the guide suffers from localization error similar to the rest of the nodes and a perfect guide which does not suffer from localization errors.

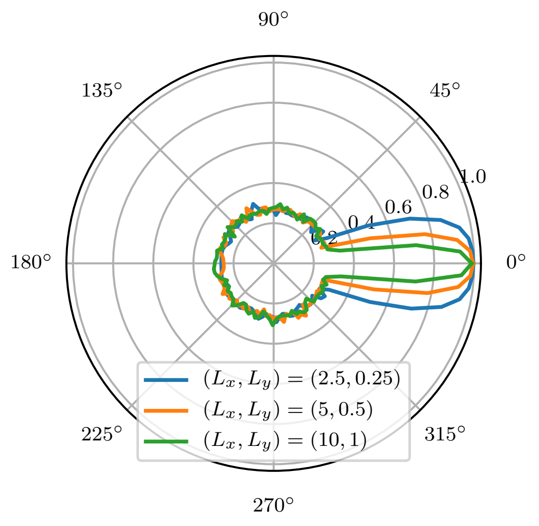
The combining gain obtained only using location information is shown in Fig. 4.8a against the localization error range ΔP for $L_x = 10$ and $L_y = 1$. We can see that for perfect location information, a gain of 1 is attained using the location based approach but as localization error range increase, the beamforming gain falls rapidly reaching the bottom when the magnitude of localization error approaches $\lambda/2$ (16.6cm). This shows that location based beamforming requires location information accurate within a fraction of a wavelength to work. While localization systems that can attain this accuracy exist, they require a large bandwidth [KBB19], which is not always available. The guide based approach is maintaining average BF gains above 0.8 even as the localization error reach 1m, which is equivalent to 3λ. However, it still decays to some extent. This decay is explained by the localization errors in the guide making the beam formed by the followers point slightly towards the wrong direction. Since the beam is narrow as we have shown in Fig. 4.7, this leads to suboptimal combining gains. However, this can be resolved by choosing the placement to attain a wider beam (using smaller L_x and L_y for instance). The used separation of the guide, which increases with the range of the error is shown in Fig. 4.8b and it does not exceed 18m for our setup. This shows that our proposed approach can compensate for localization errors by reasonably increasing the separation of the guide to account for the worst-case vertical spread.

Then, we verify that the BF gain degradation of guided BF caused by location errors is due to the width of the beam. To that end, we considered different deployment region dimensions, that is different values of L_x and L_y in Fig. 4.9. In Fig. 4.9a, we plotted the beampatterns, which show that smaller deployment regions correspond to wider beam. Then, we repeated the evaluation of the impact of the localization error. The results are shown Fig. 4.9b, from which we can see that the smaller regions corresponding to wider beams are less impacted by localization errors. This result demonstrates that by using wider beams, we can further reduce the sensitivity of guided BF to localization errors.

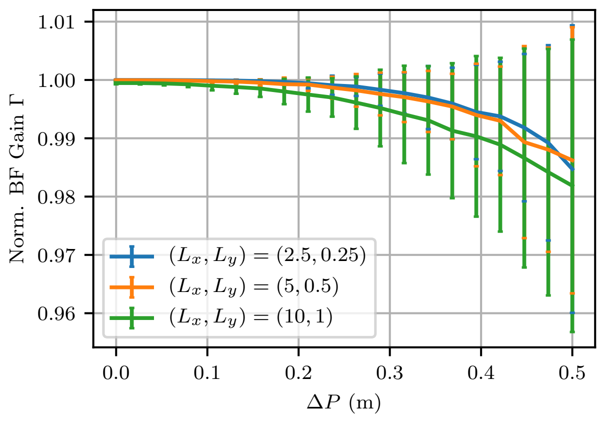
In our previous simulations, we assumed a LOS only channel h_i between each BF radio and the destination. In practice, even if the LOS path is dominant, there can be other non-LOS paths due to reflections from the surrounding objects. We also consider the case of a guide with non reciprocal phase. Our objective is to characterize the amount of BF degradation due the non-LOS channel components and non reciprocity on guided distributed BF and how it compares to other BF approaches. To do that, we consider a Ricean channel h_i^R modeled as follows

$$h_i^R = \sqrt{\frac{K}{K+1}} h_i + \sqrt{\frac{1}{K+1}} h_i^N \tag{4.17}$$

where h_i^N is a standard Gaussian random variable modeling the non-LOS channel components. The magnitude of these components is determined by the factor K, such that the



(a) The average beamwidth is determined by the dimensions of the deployment region.



(b) The normalized BF gain for different deployment regions.

Figure 4.9: The deployment regions corresponding to wider beams are less sensitive to localization errors.

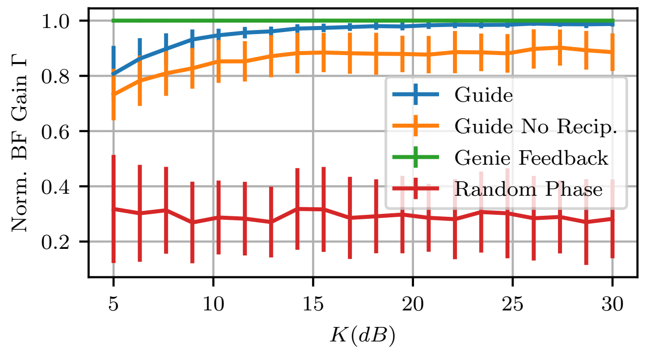


Figure 4.10: The effect of non-LOS channel components, simulated using a Ricean channel, on the BF gains of different BF approaches.

smaller $\boldsymbol{K}$, the stronger the non-LOS components. These non-LOS components are unknown to the BF radios unless the destination provided channel estimation feedback.

We compare the BF gains when using guided distributed beamforming and the random phase approach, which was proposed in [SB11, SAB13] and relies on the assumption that unsynchronized carriers occasionally lead to constructive combining. This approach was simulated by considering beamforming weights with uniformly distributed random phase. The results using the same simulation setup are shown in Fig. 4.10. From this Figure, as expected BF using feedback is not affected by the non-LOS components since it knows h_i^R assuming a genie carried the destination feedback. For reciprocal guided BF, the BF gains degrade with larger the non-LOS components simulated using a smaller K. However, the BF gains remain above 90% for K-factors as low as 10dB. In air-to-ground channels, the average K-factors were found to exceed 10dB as shown in measurement campaigns [MS17].

Depending on the RF front end implementation, the phase offset between the guide's transmit and receive chains can be varying between transmissions making the guide phase non reciprocal. When we consider a non reciprocal guide, simulated by making its phase uniformly random, the BF gains drop by a approximately $1/N$ because the signal transmitted by the guide is not necessarily coherently combining. This incoherent combining is because the followers are adjusting their signals based on the guide's receive chain, which has a different phase from its transmit chain. As for relying on random phase, the BF gains are not affected by the K-factor since it does not adjust the signal based on the channel. However, its average BF gains are are much lower than the other approaches, and it relies on re-transmitting the same data multiple times, in the hope of constructive combining occurring. Note that in practice, even using destination feedback, the BF gains do not attain 1, since there are residual timing and frequency errors leading to phase errors at combining.

4.6 Beamforming Implementation

In this section, we describe the setup that achieves coherent combining at the guide radio. This setup is used later for demonstrating guided beamforming at the destination without any feedback. In addition to adjusting their phases, since each BF radio has its own local oscillator and timing clock, the radios need to first synchronized in frequency to avoid phase

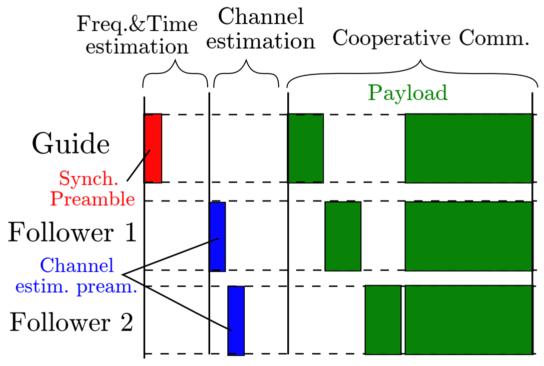


Figure 4.11: The timing diagram of the implemented beamforming setup for $N = 3$. The guide initiates the beamforming by sending a synch. preamble. The followers then, after performing time and frequency corrections, send channel estimation preambles. The payload shown in green was designed such that each BF radio transmits individually, then all of them beamform, to enable the evaluation of the beamforming gain.

drift and in time to avoid intersymbol interference. To achieve these requirements each node i estimates its frequency offset Δf_i and timing offset Δt_i, relative to the guide along with the channel phase estimate $\mathbf{p}_i$. After estimating and digitally correcting for these errors, coherent combining can be attained at the guide.

A protocol to achieve coherent combining was developed using software defined radios having a sampling time T_s. A timing diagram of the protocol is shown in Fig. 4.11. It consists of a frequency and timing estimation stage which aims to estimate Δf_i and Δt_i, followed by a channel estimation stage to obtain $\angle h_i$. Afterward, the radios transmit their payload.

The beamforming is initiated when the guide transmits a synchronization preamble. The timing and frequency estimation is performed simultaneously using this preamble using the approach from [YHB19] as follows: The BF radios obtain a one-shot frequency estimate from this preamble and apply the extended Kalman filter (EKF) as an averaging filter [QRM13]. For the timing estimation, the time of arrival (TOA) of the preamble is used as a reference for timing [EGE03]. The TOA is estimated by using correlation for samples level timing accuracy and maximum likelihood is used for sub-sample-time accuracy [YHB19].

After frequency and timing estimation, the channel is estimated. We use explicit channel estimation, where each of the followers is assigned a time slot to transmit a preamble. The guide estimates the phase of the received preamble and feeds it back to each follower through a side channel, which was implemented over WiFi. At the last stage, radio i transmits its payload after correcting for timing and frequency offsets and using $\theta_i = -\phi_i$. In practice, there are error in channel estimation, frequency and timing synchronization, which lead to imperfect combining gains at the guide despite of having feedback.

The protocol was implemented using GNU Radio [GNU] and the USRP B205-mini software defined radio [Ett]. The center frequency used is 915MHz and the sampling rate used is 1Msps making $T_s = 1\mu s$. The synchronization preamble consisted of 630 samples and the channel estimation preamble of 200 samples. The first stage was allocated 60ms, the second stage 20ms, and the third stage 30ms. The time assigned for each stage contains guard times to allow for the processing to take place. The timing of the different transmissions within the packets was ensured by using the USRP hardware driver (UHD) timing tags. While the entire payload can be used for beamforming, the payload was designed such that each of the followers transmits individually in a portion of the time, and a portion assigned for beamforming as illustrated in Fig. 4.11. To evaluate the beamforming gain of a single packet, the magnitude of the beamformed signal and that of the sum of the individual transmissions of the nodes are substituted in (4.4).

Single board computers (SBC), namely the ODROID XU4 [HAR] having a Samsung Exynos5 Octa ARM processor and 2GB of RAM, were used to power the guide and the followers. The end receiver was operated using a Dell Precision 3520 laptop having a 3GHz XEON E3-1505 V6 processor with 8GB of RAM. Note that since the BF protocol needs to run in real time, the waveforms were designed to work efficiently using SDRs powered by computationally constrained SBC and do not follow any existing protocol.

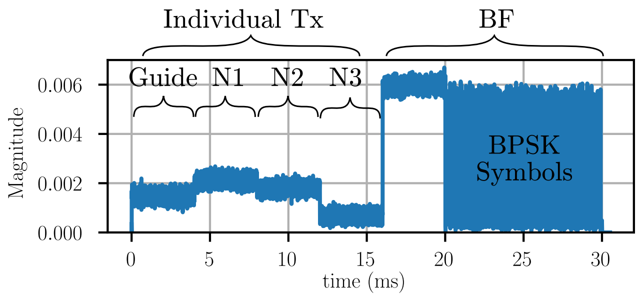
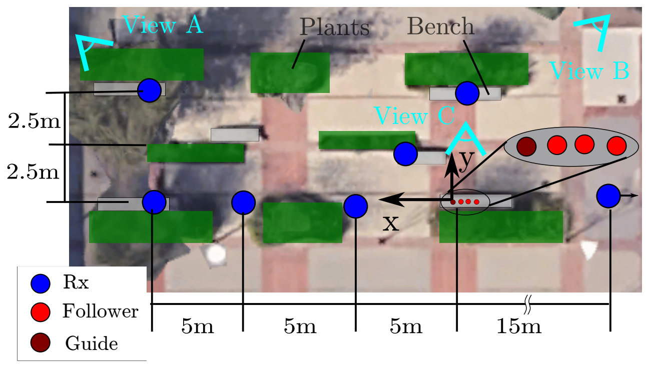


Figure 4.12: The magnitude of the payload of a received packet. Each of the beamforming nodes makes an individual transmission followed by a beamformed signal. The beamformed signal consists 4000 samples having constant magnitude followed by 10,000 BPSK modulated symbols.

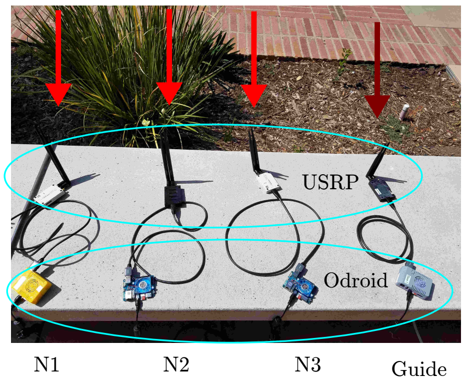
4.7 Experimental Evaluation

For the experimental evaluation, we start by describing the procedure for making measurements. Beamforming performance when using destination feedback is evaluated and used as a baseline for guided beamforming. Then, the environment to evaluate guided beamforming is described and the expected BF patterns is simulated. The results shown include the BF gains attained along with the signal-to-noise ratio (SNR) before and after BF and the packet error rates (PER).

The results consist of several measurements. A single measurement consists of 900 packets captured over a period of 5 minutes. In a packet, each node transmits individually, then they beamform. The individual transmission of one node consists of 4K samples chosen to be all ones, hence yielding an unmodulated carrier. The beamforming portion consists of 14K samples, out of which 4K samples are ones and the remaining 10K samples are BPSK modulated using root raised cosine pulse shaping with 2 samples per symbols. This makes the signal bandwidth at 1Msps equal to 500KHz. An example of the magnitude of a received packet at the destination is shown in Fig. 4.12. The beamforming gain, SNR, and PER are presented as the statistics of the 900 packets.



(a) Atop level view of the area where the experiment was conducted. The beamforming direction correspond the positive x-axis.



(b) View C showing the beamforming radios.

(c) View A of the environment and some of the destination positions. The destination radio is placed at (15,5).

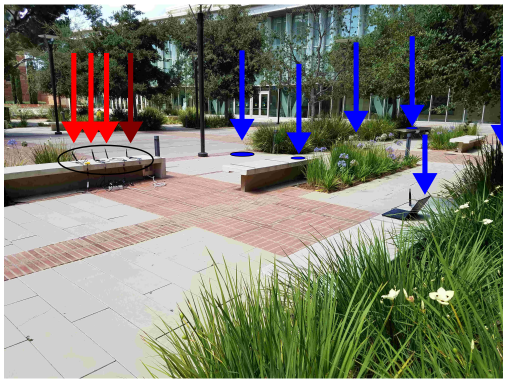

(d) View B of the environment and the remaining destination positions. The destination radio is placed at (0,5).

Figure 4.13: The placement of the nodes for the experiment evaluation.

Note that while the driving applications of guided beamforming involves communications over distances in the range of kilometers in a remote place using radios mounted on vehicles, for practical reasons, our experimental evaluation is performed at a much smaller scale as a proof of concept. By showing that radios meeting our placement criteria provide a large combining gain towards the desired direction compared to the other directions, we can infer that the range of communications can be dramatically increased at large distances using guided beamforming.

4.7.1 Baseline Evaluation using Destination Feedback

We start by evaluating our beamforming setup using 4 beamforming nodes and a destination providing feedback according to the previously mentioned protocol. In that case, the cooperative radio is the destination and the results obtained will serve as an upper bound for the guided beamforming. This experiment was conducted indoor. The beamforming nodes were placed in a line of length 0.7m broadside to the receiver at a distance of 2.5m. For feedback based BF, since the destination provides channel feedback, BF is not dependent on the exact placement of the BF radios and not affected by any non-LOS channel components. Using this setup, the mean BF gain obtained was 0.97 with a standard deviation of 0.025. The reason for not obtaining the ideal gain of 1 is because of frequency and channel estimation errors.

The destination feedback combining gain represent an upper bound for what our proposed approach can achieve for several reasons. First, when using feedback, the channel estimation measures the true channel between the BF nodes and the destination, and thus yields the best combining possible given the system phase errors. Our method leverages only the LOS channel and the radios' placements and thus is unable to account for the signal reflections that occur in the environment. Second, the guide contribution to the transmission relies on the fact that the guide radio has phase reciprocity. However, the USRP hardware is not reciprocal and there is a discrepancy between the transmitted and received phases [PMG16]. These discrepancies are expected to decrease the beamforming gain and increase its variance. However, these phenomena exist regardless of the receiver placement and hence by comparing measurements for different receiver positions, the relative gains should change as expected.

4.7.2 Guided BF Environment

The guided beamforming experiment was performed in an outdoor environment as shown in Fig. 4.13. Five nodes were used in the experiment, three followers, one guide, and one node as the destination. The four beamforming nodes were all placed in an arrangement close to linear as shown in Fig. 4.13b, which can be described according to our system model as having $L_x = 0.55$m, $L_y = 0.1$m. The guide separation was set to $d_x = 0.32$m, which exceeds the value calculated using (4.16), $\delta = 0.1\lambda$. The receiver was moved to several locations to evaluate the beamforming gains. A top view of the environment is shown in Fig. 4.13a, with the beamforming node highlighted in red and the positions of the measurements highlighted in blue. The beamforming direction is the direction of the positive x-axis as denoted on the Figure 4.13a, i.e, pointing towards the left. Fig. 4.13c and 4.13d show images of the environment from the point of view B and C respectively as denoted in Fig. 4.13a. The colored downward arrows highlight the placement of the nodes in the environment. The magnitude of a packet captured at the destination when the radio was placed at (15,0) is shown in Fig. 4.12. From this Figure, the magnitude of the signal from node 3 (N3), which is the closest to the destination is not the largest. As discussed earlier, this is due to the reflections in the environment, which are expected to degrade the beamforming gain at the destination.

4.7.3 Simulating the Expected Beampattern

We start by using simulations based on our experimental setup to obtain an expectation of the beampattern formed by our nodes. Since the experiment is not conducted in an anechoic chamber with perfectly synchronized nodes, the simulation does not aim to accurately replicate the environment but to give an expectation of the beampattern. To do that we use that same procedure of randomly placing $N = 3$ nodes within L_x and L_y. The results are shown in Fig. 4.14 with the solid line representing the mean and the dotted lines for $\pm$ the

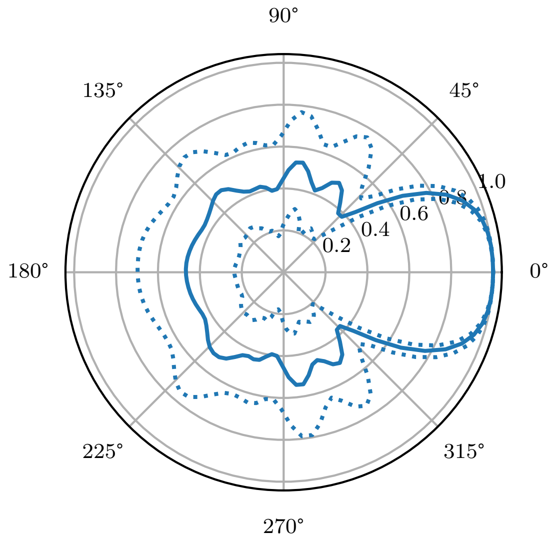


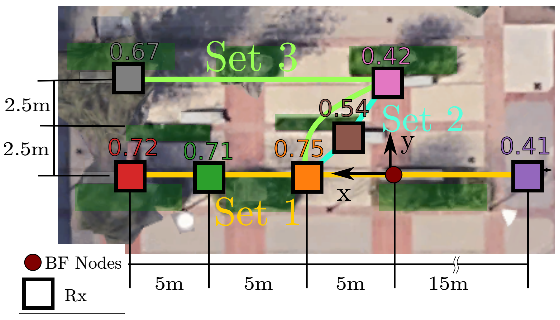
Figure 4.14: The beamforming pattern for $L_x = 0.55$, $L_y = 0.1$ emulating the experimental setup. The solid line is the average of 100 random placements and the dotted lines represent ± their standard deviation.

standard deviation. From this Figure, we expect a wide beam with width around 45° in the pointing region, with about less than half the beamforming magnitude gain elsewhere.

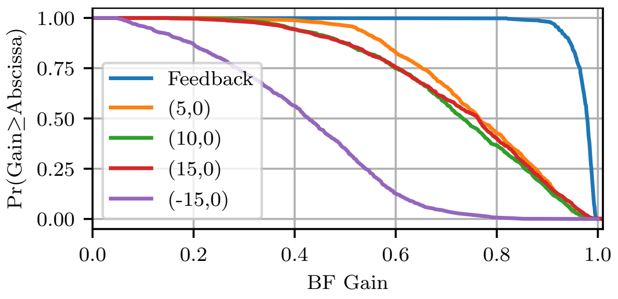
4.7.4 BF Gains

We show the measured BF gains and compare them with the expected BF pattern to demonstrate guided BF. The results are shown in Fig. 4.15. In Fig. 4.15a, we repeat the top level view with each receiver location given the same color as the color used for its curve. Using the same colors, the average BF gains are also written above each location. The BF gains distributions are represented in 3 plots each representing a group of locations forming a Set in Fig. 4.15a. The reference is at the location of the BF radios as illustrated in the same Figure.

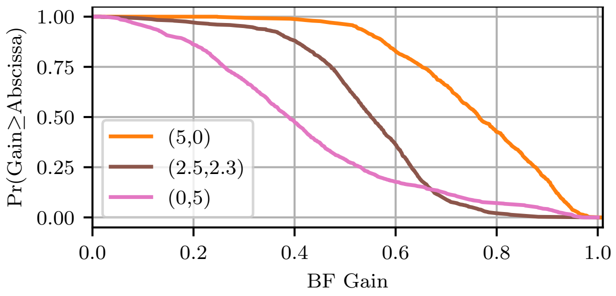
For the positions in a linear arrangement formed by Set 1, the results are shown in Fig. 4.15b as a complementary cumulative distribution function (CCDF) over the 900 packets



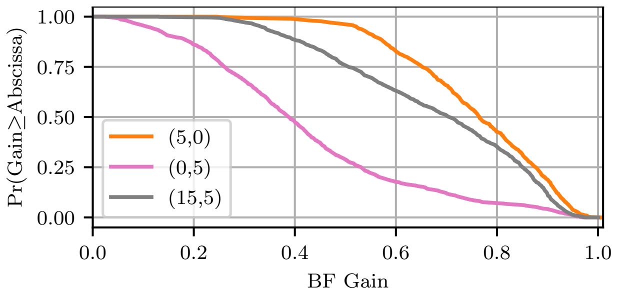
(a) The color codes of the locations correspond to color of its curve in this figure. The average BF gain at each point is written above it. The sets of points considered are connected by lines.



(b) The results of Set 1 and the results for the feedback is added as a reference. The combining gain opposite to the beamforming direction are much lower.



(c) The results along the line given by Set 2. The combining gain decreases as the angle increases.



(d) The results along Set 3. The beamforming gain at (15,5) is close to (5,0) indicating a large beamwidth.

Figure 4.15: The results from the experimental evaluation show that the beamforming gains match those predicted by the beampattern in 4.14 with larger gains attained in the destination direction.

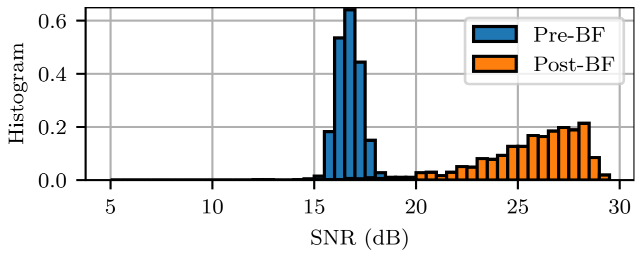
with the results obtained when using feedback included as a reference. We see that all the positions in front of the beamforming nodes obtain similar beamforming gains. This shows that the gains are consistent along the same direction and thus are expected to hold for further distances. As expected due to the unaccounted reflections and phase reciprocity mismatch the combining gains are lower than when using full feedback. Considering the results of the point (-15,0) shown in purple, the average gain significantly decreases to an average of 0.41, which is about 57% from the beamforming gain in the BF direction which

Table 4.1: Error Rates

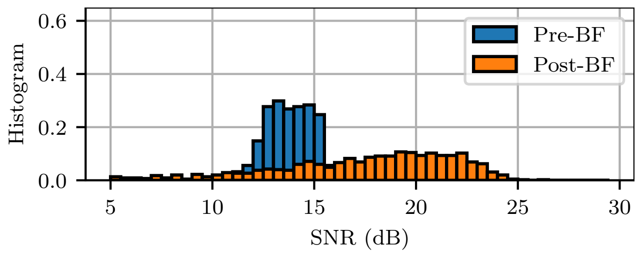
Location	Average PER	Average BER	Stdev BER
(5,0)	0.3 %	0.00015	0.00426
(10,0)	1.18 %	0.00086	0.01841
(15,0))	2.00 %	0.00102	0.01166
(-15,0)	13.41 %	0.02596	0.10168
(2.5,2.3)	1.01 %	0.00192	0.02554
(0,5)	13.85 %	0.02891	0.09211
(15,5)	1.52 %	0.0071	0.05776

matches our expectations from the simulated beampattern. In the BF direction, we observe a 3 fold increase of the received signal magnitude (9x power gain) on the average. This gain is achieved solely using the coarse placement of the radios in a line without any type of cooperation or assistance from the destination receiver.

Then, we consider Set 2 from Fig. 4.15a which allows us to investigate the combining gain at different angles, with (2.5,2.3) corresponding to 42.5° and (0,5) to 90°. From Fig. 4.15c, we see that the results match our expectations that the beamforming gain decreases as we get further from the direction of interest with a 50% decrease in combining gain orthogonal to the virtual array. This further verifies that our approach behaves as predicted by our simulations. In Fig. 4.15a, we show the results of Set 3, which shows that for the (15,5) corresponding to an angle of 18.5°, we still see most of the combining gain similar to (5,0) and about twice the gain at (0,5). This shows that as predicted the beam achieved using this arrangement is wide. Hence, this placement is still practical when only approximate knowledge of the destination location is available to point the beam.



(a) SNR before and after beamforming at (15,0)in the BF direction



(b) SNR before and after beamforming at (-15,0) opposite to the BF direction

Figure 4.16: The pre-BF and post-BF SNR in the BF direction and opposite to it. The SNR improvement is higher in the BF direction.

4.7.5 SNR and Packet Error Rate

In Fig. 4.16, we show the histograms of the prebeamforming SNR (calculated by measuring the average power received from the 4 nodes) and postbeamforming SNR. We do that for the receiver at (15,0) in Fig. 4.16a and the receiver at (-15,0) in Fig. 4.16b. We see that at (15,0), which is in the beamforming direction, the SNR improved by about 9dB on the average. This feedback free SNR improvement is what enables our approach to extend the range of communications. From the same Figure, we see that the standard deviation of the SNR also increased from 0.94dB to 2.8dB due to the randomness caused by estimation errors and the channel. Consider (-15,0), which is opposite to the beamforming direction, we see a much smaller improvement in the SNR, with an average of only 3.32dB and a larger standard deviation of 5.07dB. We also observe that at many points the post BF SNR are lower than the preBF SNR, which is expected due to destructive combining.

Unlike other feedback free approaches [SAB13], which rely on sending multiple repetitions of the sequence, guided beamforming does not penalize throughput and is able to retain a low packet error rates (PER) even without any channel coding. The PER is calculated as the percentage of packets having errors in the payload, which is obtained by demodulating the received symbols at the destination and comparing them to the transmitted symbols. Since the destination does not participate in the over-the-air synchronization, carrier frequency and symbol timing needs to be recovered and this is performed in post processing. The carrier frequency offset is estimated using the all one sequence by using linear regression on the unwrapped phase [Tre85]. The symbol timing is recovered by interpolating the received signal by a factor of 16 and downsampling it based on the minimum Gardner timing error [Gar86]. The received symbols were determined by a minimum distance Euclidean receiver.

The PER results at different destinations are shown in Table 4.1. The average bit-error-rate (BER) of all the packets is shown in the same table along with its standard deviation. We can see that the results are consistent with the combining gain results. For points on the beamforming direction, the PER is less than or equal to 2%. For the positions (-15,0) and (0,5), which had low beamforming gains, we get significantly higher error rates with the PER exceeding 13% and more than 20 fold increase in BER compared to points in the beamforming direction. Part of these errors is attributed to residual timing and frequency offsets between the beamforming nodes, which lead to interference even at high SNR.

4.8 Summary

In this chapter, we proposed guided distributed beamforming as an approach for mobile radios sharing a LOS channel to make their signals coherently combine at a remote destination, without feedback from the destination and without having a strict requirement on location information. We have derived and verified using simulations a geometric criterion for the placement of the radios, to bound the phase errors leading to degraded beamforming gains at the destination. The proposed approach was implemented using software defined radios. The results show a 9 dB SNR improvement in the beamforming direction when using 4 radios, with a weaker gain in other directions as predicted using simulations. Thus, our approach enables an extended range of communications towards a distant destination not providing feedback.

CHAPTER 5

Transmitter Authorization

Due to imperfections in transmitters' hardware, wireless signals can be used to verify their identity in an authorization system. While deep learning was proposed for transmitter identification, existing work has mainly focused on classification among a closed set of transmitters. Malicious transmitters outside this closed set will be misclassified, jeopardizing the authorization system. In this chapter, we formulate the problem of recognizing authorized transmitters and rejecting new transmitters as open set recognition and anomaly detection. We consider approaches based on one and several binary classifiers, multiclass classifiers, and signal reconstruction. We study how these approaches scale with the required number of authorized transmitters. We propose using a known set of unauthorized transmitters to assist the training and study its impact. The evaluation procedure takes into consideration that some transmitters might be more similar than others and nuances these effects. The authorization's robustness against temporal changes in fingerprints is also evaluated as a function of the approach and the dataset structure. When using 10 authorized and 50 known unauthorized WiFi transmitters from a publicly accessible testbed, we were able to achieve an outlier detection accuracy of 98% on the same day test set and 80% on the different day test set. This chapter revises our previous publication [HKC20a].

5.1 Introduction

With the growth in the number of wireless connected devices, securing them has become more challenging. Unlike wired communications, a wireless network is accessible by any device with sufficient transmit power. This makes *authentication*, the process of verifying the identity of devices, challenging. After authentication, devices are granted access, the process known a *authorization*. While cryptographic methods are used for authentication, many devices like Internet-of-Things (IoT) devices don't possess the energy nor computational power to run them, leading to many authentication based attacks [NBC19].

Physical Layer Authentication (PLA) leverages the dynamics of physical layer attributes to address these challenges and to enhance wireless security [WHH16]. While active PLA typically requires changes in transmitters, passive PLA is performed on the receiver side, making it more practical. Passive PLA uses RF fingerprints; combining channel state information (CSI) and transmitter hardware fingerprints to authenticate devices. Transmitter fingerprints result from the imperfections in their RF chain components like ADCs, power amplifiers, etc. The interaction between these non-idealities makes signals from identical transmitters exhibit unique characteristics typically modeled as carrier frequency offset, IQ imbalance, among others [WSP16].

While there has been many approaches for using RF fingerprints based on handcrafted features [BBG08, NZH11, VVN16, RPB18, PHZ19, CDS18, ZHL19, XGM08, GGD13, WKS17, SBG19, ZTJ19, FWH18], it was shown to be highly dependent on the quality of the receiver hardware [RSC12] and requires manual feature engineering which are protocol dependent. For these reasons, recently, there has been wide interest is using deep learning approaches to address this problem [RSI18, YHL19, GCM19, BGG19, AAP19, WFK18, MRS18, HC19, YBH17, GWJ19, HKC20b]. Deep learning has the ability to learn a richer set of features from raw IQ samples leading to improved performance over manually selected features, as has been demonstrated in [RSI18].

While previous deep learning work in this area has addressed many of the challenges of RF fingerprinting, this body of work has posed the problem as a *closed set* classification which assumes a known set of transmitters, except for [GWJ19] and our prior work [HKC20b]. No matter how large the set is, if any new unseen transmitter gets within communication range, its signal will get misclassified leading to security vulnerabilities. This calls for *open set* approaches which are capable of rejecting signals from unseen transmitters. While [GWJ19] proposed their own approach to address this problem, rejecting samples from a new distribution is not a novel problem for the machine learning community. A plethora of approaches have already been proposed for similar problems in computer vision, natural language processing, intrusion detection, etc. Two problems are most relevant; openset classification [GHC19]: classifying among known classes and rejecting unseen classes, and anomaly detection [CC19]: identifying abnormal samples. Instead of reinventing the wheel, we aim to adapt the most prominent approaches for these problems and evaluate their performance. Unlike other domains, transmitter authorization has its own challenges and requirements: (1) RF fingerprints arise from random channel and hardware variations, hence, generalizable conclusions can not be derived from single point evaluations (2) the number of authorized transmitters is a system requirement that can vary significantly (3) the ease of collecting data (compared to image classification, for instance) raises questions about how to construct a training dataset, and (4) the robustness of the approach against time varying fingerprints needs to be evaluated.

In our previous work [HKC20b], which we extend in this work, we started investigating a few approaches from the existing openset recognition literature. In this work, our contributions can be summarized as follows:

- We formulate the problem of rejecting signals from unseen transmitters as both an openset recognition problem and an anomaly detection problem. We consider 5 deep learning approaches to the problem and evaluate their performance; (1) Binary classification using "Disc" (2) Multiclass classification using "DClass" (3) A binary classifier for each transmitter using "OvA" (4) Evaluating signal reconstruction errors using "AutoEnc" (5) Analyzing classifier's activations using "OpenMax"

- We discuss several considerations for network architecture design for transmitter authorization. We show that making minor changes to the neural network architecture and data labeling strategy yields a conceptually different approach with different performance. We show that classification within a closed set is not always an indicator for performance in an open set.

- We compare the performance of the considered approaches with respect to the number of authorized transmitters required by the system. We propose using a set of known unauthorized transmitters and show its benefits.
- We show that the results obtained are dependent on the choice of transmitters and time of evaluation. Then we address this by showing results in terms of statistics of multiple transmitter choices using data captures on same day as training and different day.

5.2 Related Work

Physical Layer Authentication (PLA) can be classified as active or passive. Active PLA typically overlays a tag over the message used for authentication, thus requiring changes to the physical layer of the transmitters [XZ18, GCX20]. Passive PLA on the other hand uses the channel state information and the RF fingerprint due to hardware imperfections to identify transmitters [WSP16], requiring no change to transmitter signals, and hence is easier to apply. Approaches for passive PLA either use a set of handcrafted features or use deep learning to learn the features.

5.2.0.1 Handcrafted Feature PLA

In these works, a set of manually designed features are extracted from the signals, which are then used for distinguishing transmitters according to some procedure. A variety of features were considered as transmitter fingerprints in the literature [XZS16]. These features include transient ones like the patterns at the start of packets [DC09], and steady-state ones like carrier frequency offset [YHP16], IQ imbalance, sampling frequency offset or a combination of these features [BBG08, NZH11, VVN16, RPB18, PHZ19, CDS18, ZHL19]. Other works have used channel state information (CSI) as features. This kind of approach has been considered for SISO [XGM08, GGD13, WKS17, SBG19] and massive MIMO [ZTJ19] communications. Combining CSI with transmitter fingerprints has also been proposed [FWH18].

As for the procedure for distinguishing transmitters using the features, some works used traditional methods like a distance metric among features [DC09, CDS18, PHZ19], and hypothesis testing [WKS17]. Other works have used machine learning applied to the features; K-nearest neighbor (KNN) [BBG08], support vector machines (SVM) [BBG08], and neural networks [CDS18] were proposed for classification. For authentication, Gaussian mixture models were also used [NZH11, XGM08, GGD13].

5.2.0.2 Deep Learning Based PLA

In contrast to handcrafted features, deep learning approaches are able to extract features from the high dimensional signals without requiring manual feature engineering. Deep learning became popular due to its superior performance in computer vision [HZR15]. It was successfully applied in many problems in wireless communications like modulation classification [ORC17, HDC20, HKC20b], detecting anomalies in spectrum utilization [RML19, TKF20], spectrum sensing [LWL19], and backscatter signal detection [LWN20].

Due to the better performance of deep learning compared to handcrafted features in transmitter identification [RSI18], it has gained widespread interest [RSI18, YHL19, GCM19, BGG19, AAP19, WFK18, MRS18, HC19, YBH17]. Some of these works fall under the category of active PLA, requiring changes in the transmitters, while others are passive.

In active approaches, modifications are intentionally added to the signal to improve classification. A protocol inserting IQ imbalance and DC offset impairments to improve the RF fingerprints was proposed in [SBZ18, SBZ19]. FIR filtering was also considered in [RDA19] to optimize RF fingerprints. This work requires modification in the transmitter side, which is not alway feasible. The work considering passive PLA focused on the data representation, the network type, or studying the impact of a specific transmitter characteristic.

Data representation The work in [BGG19] has compared different data representations like wavelet transform and Short Time Fourier transform while in [BGD19], the authors considered recurrence plots. Applying the Hilbert-Huang transform to the signal and deep residual networks were proposed in [PYP19]. Higher order statistics like bispectrum were proposed in [DWW18].

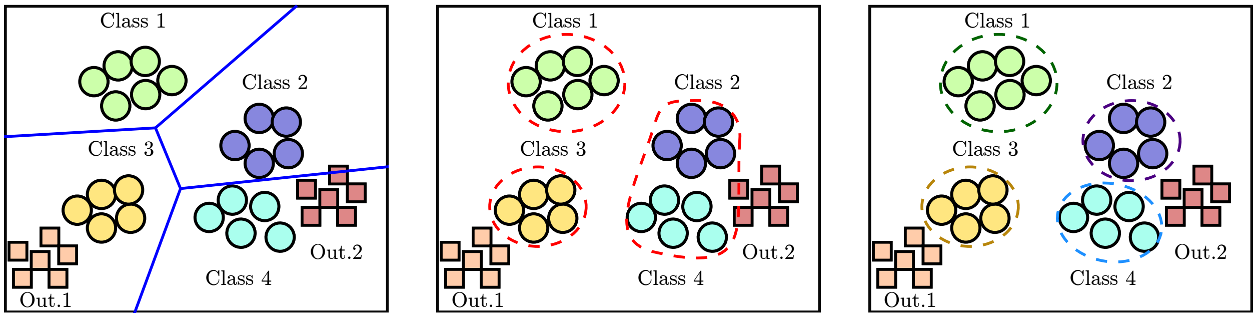
Network Architecture In [YBH17], authors compared different types of neural networks and machine learning techniques. CNNs and RNNs to classify IoT devices over a wide range of SNR in [JOA18]. In [MRS18], multiple CNN architectures were tested on indoor and outdoor data with a focus on cognitive radio applications. Complex neural networks were proposed in [AAP19] using convolutional and recurrent architectures. The effect of a dynamic channel on deep learning RF fingerprinting along with the type of data was the focus of the work in [MCH19]. In [YHL19], the authors have considered a multisampling neural network using LOS and NLOS datasets. Denoising autoencoders were also proposed for the same problem [YHZ19]. Adversarial learning was adapted [RMC19] in to detect rogue transmitters.

Transmitter Characteristic Some works used deep learning while focusing on a specific type of impairment. The effect of power amplifier nonlinearity and signal type on classification performance was studied in [HC19]. In [WHM18], CNNs were used to learn IQ imbalance as a modulation-independent way of transmitter identification.

The main limitation of this body of work is that it focuses on classification among a closed set of known transmitters. To the best of the authors' knowledge, two works have considered the problem of using deep learning for transmitter authorization that generalizes to unseen transmitters. The first work has proposed a novel approach for outlier detection that works on a per-packet basis [GWJ19]. The classifying neural network is applied to slices of the packet and statistics of the slices predictions are compared to a threshold. This approach is discussed further later in this work in Section 5.5.0.4. In their work, several datasets using WiFi and ADS-B were considered using 50, 250, and 500 devices. The data is said to have been captured ”in the wild” with no further details provided. We only mention their results most similar to our work; using 50 authorized WiFi devices, they were able to detect new devices with an accuracy of 73% at the cost of a drop in classification accuracy from 63% to 43%.

The second work that has considered this problem was our previous work [HKC20b], which is extended in this work. In [HKC20b], three approaches based on open set recognition were contrasted using a dataset consisting of WiFi preambles captured in a publicly accessible wireless testbed [RSO05]. We have considered the effects of the number of authorized transmitters and demonstrated the benefit of using known outliers in training. Using 40 authorized transmitters, we were able to identify new devices from authorized devices with an accuracy of 84%. The improved results obtained in our work can be partially attributed to relying on the packet preambles, as discussed in Section 5.6.

Compared to [HKC20b], in this work, we consider two additional approaches which are OpenMax [BB15] and autoencoders [CC19] and the latter is shown to improve transmitter authorization with fewer transmitters. The results are presented as probability of false alarm and probability of detection providing more insights on the obtained performance. We further explore considerations for neural network architecture design with regard to outlier detection and how it differs from classification. The dataset used is expanded to include more transmitters, captured over a period of five days, increasing the confidence of our results and exploring temporal generalization.



(a) Classification (b) Anomaly Detection (c) Openset Recognition

Figure 5.1: Known classes are depicted as circles and outlier classes as squares. Classifiers would mistakenly label oultiers. Anomaly detector rejects outliers but cannot distinguish among the known classes. Openset classification classifies among known classes and rejects outliers.

5.3 Classification, Openset Recognition, and Anomaly Detection

In this section, we highlight the difference between classification, openset recognition, and anomaly detection. A closed set classifier determines boundaries that separate a pre-defined finite set of classes, shown as colored circles in Fig. 5.1a. As such, for samples from new classes (shown as squares in Fig. 5.1a) the classifier will predict the nearest class. This poses a grave security risk for a wireless authentication system. Since, it is impossible to train a classifier on all the transmitters in the world, an approach that generalizes to signals from new unseen transmitters is needed.

We consider two formulations to address this limitation. The first one is posing the problem as an anomaly detection [CC19]. Anomaly detection aims to identify instances which are different from the normal. Hence, it finds boundaries around the seen classes (considered as normal cases), as shown in Fig. 5.1b. The limitation, however, is that it treats all authorized transmitters as a single class. An overestimation of the determined boundaries could lead to errors in outlier detection as illustrated for outlier 2 in Fig. 5.1b. Open set

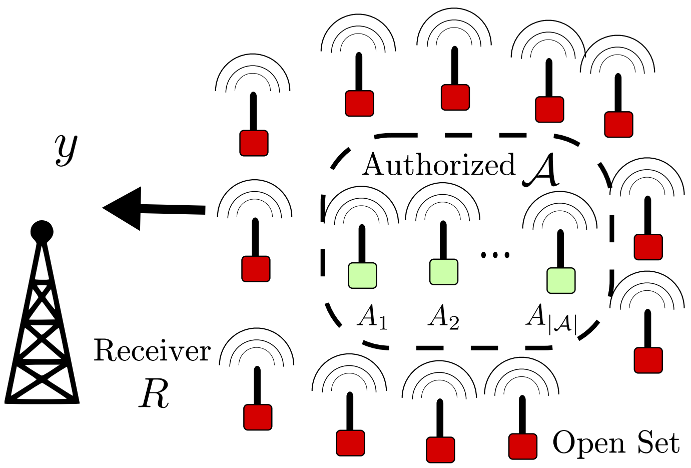


Figure 5.2: Signal y is received by receiver R. We want to determine if it was sent by an authorized transmitter in the set $\mathcal{A}$ or a new unseen transmitter.

classification [GHC19] takes an approach similar to anomaly detection while additionally classifying among the known samples. Hence, it isolates each class on its own as shown in Fig. 5.1c.

Classifying among transmitters using only received signals in a robust manner is a challenging problem on its own. Extending it to open set poses even more challenges. Unlike approaches using handcrafted features [GGD13, XZS16, XGM08, NZH11], which use well separated features like CSI, in order to leverage the power of deep learning, we use raw IQ samples. The challenge for the deep neural networks is to learn features which separate the known classes from the unknown classes, for which no training samples are available.

5.4 System Model and Problem Formulation

We consider a finite set of authorized transmitters given by $\mathcal{A} = \{A_1, A_2, \cdots, A_{|\mathcal{A}|}\}$ that are allowed to send data to a receiver R, where $|\mathcal{A}|$ is the size of the set $\mathcal{A}$. When some transmitter T sends a set of symbols $\mathbf{x}$, the signal received by R is $f_T(\mathbf{x}, t)$. The function f_T represents the time variant RF fingerprint, which captures the transmitter hardware fingerprints and wireless channel effects. Since the channel depends on the environment surrounding the transmitters, it is more prone to temporal variation.

The authorization problem can be formulated as shown in Fig. 5.2: receiver R receives a signal $\mathbf{y}$ from some transmitter T and wants to determine whether the transmitter T belongs to the authorized set or not without decoding $\mathbf{y}$. This can be formulated as the following hypothesis test:

$$\begin{matrix} \mathcal{H}_0 : \mathbf{y} = f_T(\mathbf{x}, t), T \in \mathcal{A} \\ \mathcal{H}_1 : \mathbf{y} = f_T(\mathbf{x}, t), T \notin \mathcal{A} \end{matrix} \quad \forall \quad t \tag{5.1}$$

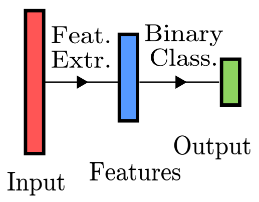
Here, $\mathcal{H}_0$ corresponds to an authorized transmitter and $\mathcal{H}_1$ corresponds to an outlier.

Additionally, in cases where each authorized transmitter has different privileges, we might be interested in classifying it within the authorized set, which can be formulated as finding $\hat{A}$ that is most likely to have generated $\mathbf{y}$, defined as

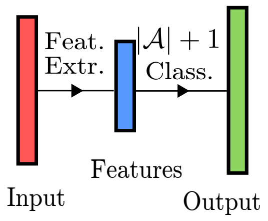
$$\hat{A} = \underset{T}{\operatorname{argmax}} \Pr(f_T(\mathbf{x}, t) = \mathbf{y} | \mathbf{y}), \qquad T \in \mathcal{A}, \quad \forall \ t \tag{5.2}$$

While the anomaly detection problem addresses only problem (5.1), the open set problem addresses both (5.1) and (5.2). Since classification has been studied extensively in the literature, our main focus in this work is on the results of outlier detection when using either formulation.

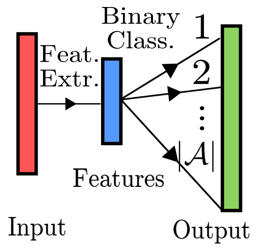
To improve outlier detection, we propose using an additional class of *known* outlier transmitters $\mathcal{K} = \{K_1, K_2, \cdots, K_{|\mathcal{K}|}\}$, where $\mathcal{K} \not\subset \mathcal{A}$. Samples from transmitters in $\mathcal{K}$ will be used during training to assist the outlier detector to differentiate between authorized and non-authorized transmitters. In practice, samples from the set $\mathcal{K}$ can be obtained by capturing data from a finite number of non-authorized transmitters. For evaluating performance on unseen transmitters, we will use a set of *unknown* outlier transmitters $\mathcal{O}$. All signals from transmitters in $\mathcal{O}$ are reserved only for the test set.



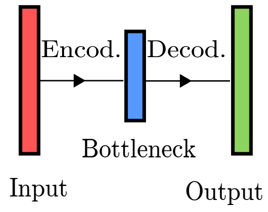
(a) Disc



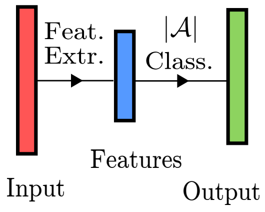
(b) DClass



(c) OvA



(d) AutoEnc



(e) OpenMax

Figure 5.3: High level architecture of the proposed methods. Autoencoder consists of an encoder and a decoder. The remaining ones consists of a feature extractor followed by one or more classifiers.

5.5 Machine Learning Approaches

In this section, we discuss machine learning approaches to address this problem. An approach consists of a neural network, followed by an output processing stage to decide whether a signal is an outlier or not. The neural networks are used as function approximators [GBC16]. They a map an input $\mathbf{y}$ to an output $\mathbf{z}$ using parameters θ, such that $\mathbf{z} = g_\theta(\mathbf{y})$. The parameters θ are learned by applying stochastic gradient descent on labeled training data. For this work, the input is the raw IQ samples and the output and its interpretation differs from one approach to the other. As for the processing stage, for some approaches, it involves modifying a threshold to control the sensitivity to outliers.

5.5.0.1 Discriminator (Disc)

This the most intuitive approach for outlier detection. It consists of a binary classifier that outputs a decision on whether the signal is an outlier or no. While simple, the main limitation of this method is its complete reliance on the known outlier set for training. In terms of architecture, the discriminator has as single scalar output z as shown in Fig. 5.3a. z is generated by a sigmoid function and takes a value between 0 and 1. The labels for authorized transmitters and outliers are $l = 0$ and $l = 1$, respectively. A threshold γ is used to make a decision with $\mathcal{H}_1$ declared if $z > \gamma$; $\mathcal{H}_0$ is declared otherwise.

5.5.0.2 Discriminating Classifier (DClass)

This approach detects outliers and classifies transmitters within the authorized set. It consists of a multiclass classifier having $|\mathcal{A}| + 1$ outputs. The first $|\mathcal{A}|$ outputs correspond to the authorized class and the last class corresponds to outliers. This classifier is expected to perform better than Disc, as individually labelling transmitters should help it extract better features. To train this network, we also need known outliers similar to Disc. A signal is classified as an outlier if the maximum activation corresponds to the last class; it is considered authorized otherwise. For this architecture, it is not straightforward to design an adjustable threshold as in Disc.

5.5.0.3 One Vs All (OvA)

OvA consists of multiple binary classifiers, one for each authorized transmitter. While the Disc network can serve as a binary classifier, reusing Disc increases the network size due to repeating the feature extractor. A better way to implement OvA is shown in Fig. 5.3c. In this approach, all $|\mathcal{A}|$ binary classifiers share the same feature extractor similar to what was proposed in [SXL17]. Unlike Disc, OvA does not require a known set of outliers as long as $|\mathcal{A}| \geq 2$, since for binary classifier i, signals from all transmitters $j \neq i$ are considered outliers. The output of this network will be a vector $\mathbf{z}$ of $|\mathcal{A}|$ real numbers such that $\mathbf{0} \leq \mathbf{z} \leq \mathbf{1}$, where $\mathbf{0}$ and $\mathbf{1}$ are the vectors of all-zeros and all-ones, respectively. Following the notation in [SXL17], the labels for a sample from authorized transmitter A_i will have $l_i = 1$ and $l_j = 0 \ \forall j \neq i$ and a known outlier, if used, will have $\boldsymbol{l} = \mathbf{0}$.

The decision is based on $|\mathcal{A}|$ thresholds given by $\boldsymbol{\gamma}$, where element γ_i is the threshold for z_i. Each binary classifier i declares a sample to belong to its class if $z_i > \gamma_i$. A signal is declared to be an outlier (corresponding to $\mathcal{H}_1$), if all discriminators declare the signal to be not within their class ($\mathbf{z} \leq \boldsymbol{\gamma}$), and to be within the authorized set (corresponding to $\mathcal{H}_0$) otherwise.

5.5.0.4 OpenMax (OpMx)

OpenMax is a popular approach for openset recognition [BB15]. It consists of modifying a trained classifier having $|\mathcal{A}|$ softmax outputs. This modification is based on the statistical analysis of activations of authorized transmitters. It works as follows; the output activation vector $\mathbf{v}$, obtained prior to softmax, is processed to generate a modified activations vector $\mathbf{v}'$ having $|\mathcal{A}| + 1$ outputs, with the additional output corresponding to outliers. The modified activation vector is given by

$$v_i' = \begin{cases} v_i \omega_i, & i \in \{1, \cdots, |\mathcal{A}|\} \\ \sum_{i=1}^{|\mathcal{A}|} v_i (1 - \omega_i) & i = |\mathcal{A}| + 1 \end{cases} \tag{5.3}$$

where ω_i represents our confidence in the membership of the given sample to class i. The concept behind calculating the vector $\boldsymbol{\omega}$ is that the activation vectors of samples belonging to the same class are similar, while those belonging to unseen classes are different from all classes. This is implemented by calculating the mean activation vector $\bar{\mathbf{v}}_i$ for each class i using the training set. The distance $d_i(\mathbf{v}) = \|\bar{\mathbf{v}}_i - \mathbf{v}\|$ represents the similarity of the sample generating vector $\mathbf{v}$ to class i. On the training set, for the correctly classified samples, the distance of each sample belonging to a class i is calculated. Using extreme value theorem [KN00], the tail of the distribution is calculated by fitting the τ samples with the largest d_i to a Weibull distribution having parameters (m_i, η_i). Then, for the α classes having the highest activations, ω_i is calculated by evaluating the probability of belonging to the tail of distribution of i using

$$\omega_i = 1 - R_\alpha(i) \times \text{WeibullCDF}(d_i(\mathbf{v}), (m_i, \eta_i)) \tag{5.4}$$

where $R_\alpha(i) = \frac{\alpha - i}{\alpha}$ is a calibrator with parameter α and $\text{WeibullCDF}(x, (m, \eta)) = \exp\left(-(x/\eta)^m\right)$, as explained in [BB15, YSK19]. After calculating $\mathbf{v}'$, uncertain outputs are rejected if the confidence is below some threshold ϵ. This is done by applying the softmax function to $\mathbf{v}'$ to obtain the vector $\mathbf{z}$. Then we calculate $i = \text{argmax}(\mathbf{z})$; the sample is considered an outlier if $i = |\mathcal{A}| + 1$ or $z_i < \epsilon$. Since OpenMax is based on a classifier, it does not benefit from known outliers in training.

The approach for transmitter authorization in [GWJ19] consists of modifying a classifier similar to OpenMax. However, their approach only uses the maximum value of the softmax output, while OpenMax uses the entire activation vector. Hence, it uses more information from the neural network output.

5.5.0.5 AutoEncoder (AutoEnc)

Autoencoders are commonly used for anomaly detection [CC19]. They detect anomalies by reconstructing their input and evaluating the reconstruction error. They consist of an encoder mapping the data into a smaller dimension, the bottleneck, followed by a decoder trying to reconstruct the original input. During training, autoencoders learn the distribution of the training data. When the autoencoder processes anomalous data, it generates a higher reconstruction error, which can be used to detect anomalies. During training, the objective of the autoencoder is to reduce the mean squared error, $\text{MSE} = \|\mathbf{y} - \hat{\mathbf{y}}\|$ where $\mathbf{y}$ is the input and $\hat{\mathbf{y}}$ is the output of the autoencoder. A signal is considered an outlier if this error is bigger than some threshold γ.

Table 5.1: Features of Approaches

Approach	Samples from $\mathcal{K}$	Adjustable Threshold	Classifies $\mathcal{A}$
Disc	Necessary	Yes	No
DClass	Necessary	No	Yes
OvA	Optional	Yes	Yes
OpMx	Unsupported	No	Yes
AutoEnc	Unsupported	Yes	No

To summarize, the proposed approaches are either based on classifiers (Disc, DClass, OvA, OpMx) or uses signal reconstruction (AutoEnc). The classifier-based approaches share a neural network-based feature extractor and differ only in the output labels and the last layers activation function. Some of the classifier-based approaches, beside outlier detection, classify the authorized signal among the set $\mathcal{A}$. As for the known outlier set $\mathcal{K}$, it is necessary for the training of some approaches, can be used to improve the performance of others, and cannot be used in some other approaches. Table 5.1 provides a high level comparison of approaches.

For the approaches which have an adjustable threshold, a tight threshold would lead to signals from authorized transmitters being mistakenly rejected (high probability of false alarm P_{FA}) and a loose threshold would fail to recognize many outlier signals (low probability of detection P_D). The desired value of P_{FA} can be obtained by empirically calibrating the thresholds on the training set. The method for setting the thresholds along with any other hyperparameters used in this work is discussed in Appendix C.

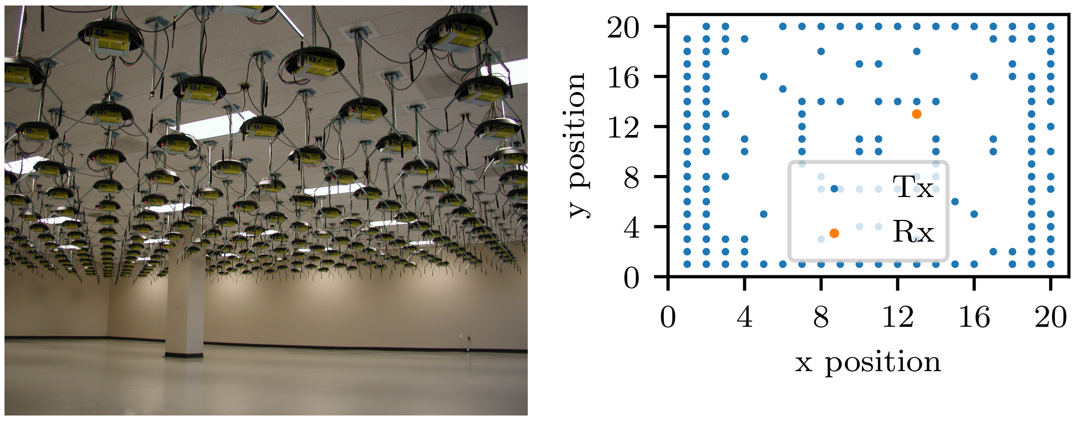


Figure 5.4: Image of orbit testbed. The participating nodes are plotted.

5.6 Dataset

The dataset was captured using off-the-shelf WiFi modules (Atheros 5212, 9220, and 9280) as transmitters and a software defined radio (USRP N210) as a receiver. The capture was performed in the Orbit testbed grid [RSO05]. Orbit testbed grid consists of 400 nodes arranged in a 20×20 grid with a separation of one meter. The receiver was chosen near the center of the grid and 163 nodes surrounding it were used as transmitters. An image of the testbed along with the location of the transmitters and receivers are shown in Fig. 5.4.

The capture was done over IEEE 802.11g Channel 11 which has a center frequency of 2462 MHz and a bandwidth of 20 MHz. Captures were taken using a sampling rate of 25 Msps, over a duration of one second. After the IQ capture was complete, the packets were extracted using energy detection. The number of packets captured during this period for each transmitter varied according to the WiFi rate control and their total number is over 300,000.

While it is possible to use the entire packet payload for training, this would make the data contained in each slice different. In our previous work [HC19], we have shown that using the slices with the same data leads to a better performance than using slices containing random data. This was also verified in [MCH19]. Hence, from each captured WiFi packet, we used the first 256 IQ samples containing the preamble. The IQ samples were normalized to have a unity average magnitude without any further preprocessing. For transmitter authorization in [GWJ19], the entire packet payload was used for training and inference was done on slices of a packet, which are combined to obtain one decision per packet. While this method leads to having more data, it makes the learning task of the neural network harder, as was previously demonstrated in [HC19, MCH19]. Since in this work we only consider preambles and due to the similarity of their approach to OpenMax, we don't consider their approach in our evaluation.

As was pointed out in recent works [CGM20, ARD20], the fingerprints learned by neural networks can be dependent on the channel and not the transmitter. This causes significant degradation in the recognition performance if the testing data was captured on a different day than the training capture. To this end, using the previous setup, we made five data captures over 5 different days. The data from the first capture is the one used in our previous work [HKC20b], and contained data from less transmitters. Also, it was made two months earlier than the remaining captures. The data from the last day was kept exclusively for testing.

5.7 Network Architectures

In this section, we describe the architecture of the proposed networks. The design of neural networks (NN) plays a crucial role in the performance achieved. While there are many variants of networks, in this work we use feed forward neural networks. They are typically built using convolutional and dense neural layers [GBC16, II]. While there is no systematic way to design NN, there are known guidelines for optimizing their hyperparameters [GBC16, 11.4].

Since the feature extractor is an essential component for Disc, DClass, OvA, and OpMx, we consider several alternatives for it and compare to the architecture used in our previous work [HKC20b]. All the architectures in this work are built using the blocks shown in

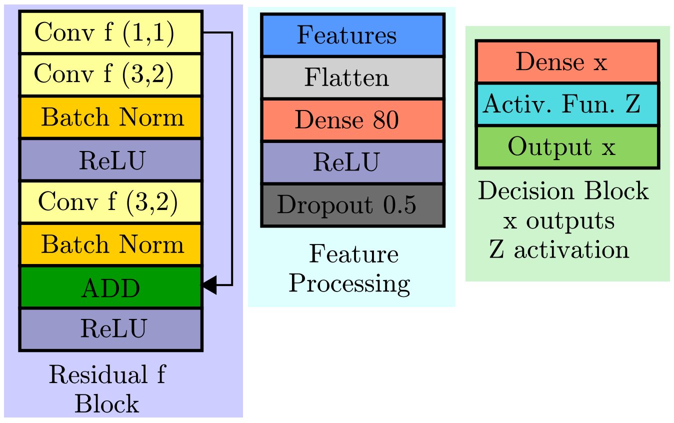


Figure 5.5: The residual, feature processing, and decision blocks used as building blocks for the neural networks considered in this work.

Fig. 5.5; the residual block which has f filters [HZR15], the feature processing block, and the decision block using activation function Z to generate a given number x outputs. The values of f, Z, and x are specified when the block is used. For all the figures used in this section, we use a color code; each type of layer (convolutional, dense, etc) is given a unique color, and the background color of each modular block is used as a layer color when this block is used.

5.7.1 Architecture Comparison

For choosing the best architecture and contrasting to the architectures in our previous work, we use OvA as a benchmark using data from 10 authorized transmitters ($|\mathcal{A}| = 10$) collected on the same 4 days and no known outliers. Details for the network training is deferred for later in the chapter.

We consider 4 architectures of OvA. OvA 1 (from [HKC20b]) which has $|\mathcal{A}|$ Feature Processing block and OvA 2, which uses the same Feature Extractor 1 with a common Feature Processing blocks as shown in Fig. 5.7. Moving from OvA 2 to OvA 3 and OvA 4, we use smaller feature extractors 1,2, and 3 respectively which are described in Fig. 5.6.

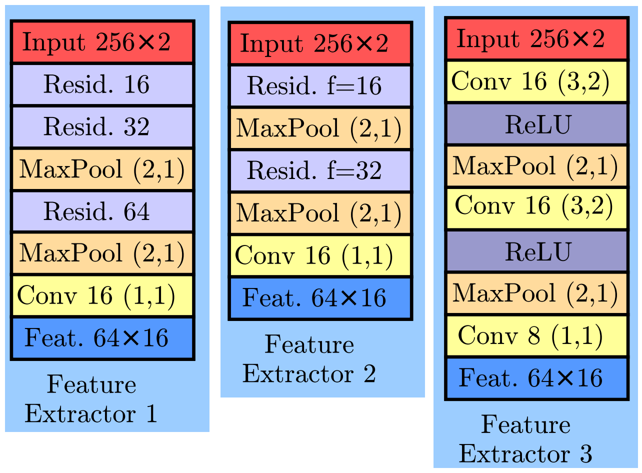
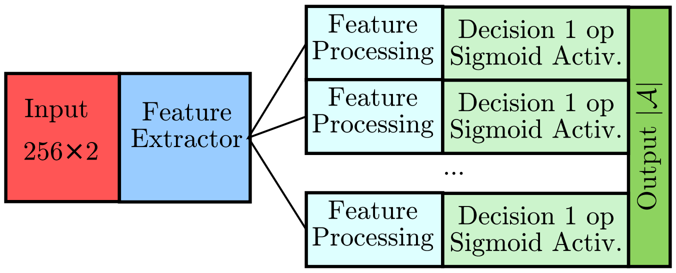


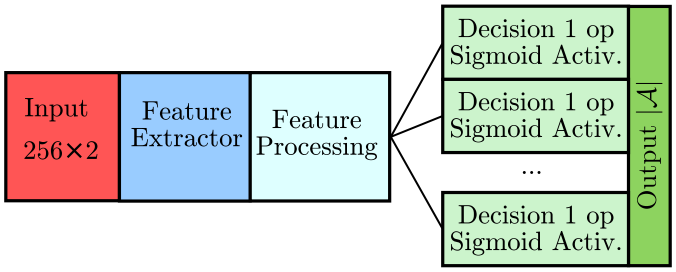
Figure 5.6: The three different architectures compared in this work. Feature extractor 2 was eventually selected. The residual block is described in Fig. 5.5.

The summary of each architecture and the number of trainable parameters are shown in Table 5.2 from which we see that from OvA 1 to OvA 4 the network gets smaller.

Due to the random initialization of the weights, along with randomness in batch division during training, the same network trained using the same data can give different results. To have confidence on the significance of our results, each network is trained from scratch using the same data for ten repetitions and the statistics of the results are shown. Fig 5.8a shows the classification results of the authorized nodes. The larger networks perform better, as expected; by having more learning capacity, the networks are better at distinguishing between transmitters. But looking at the accuracy of outlier detection in Fig. 5.8b for OvA 1 and 2, we see a rather surprising result; the largest network actually performs worse than some of the smaller networks. Since we want the network to generalize to new transmitters, we want each binary classifier of OvA to learn only the characteristics of its designated transmitter, while rejecting any other transmitter. But once the learning capacity of the per transmitter branches increases beyond a certain point, it starts to learn the characteristics of the remaining transmitters. Although this improves classification and minimizes training and validation loss, it does not generalize well to outlier detection.



(a) OvA used in our prior work [HKC20b] with a feature processing block per output.



(b) OvA used in this work having a shared feature processing block. Several feature extractors were compared and eventually Feature Extractor 2 was used.

Figure 5.7: We considered two OvA architectures, the first having a feature processing block for each output, and the second using a common one. The feature extractors are described in Fig. 5.6 and the feature processing and decision blocks in Fig. 5.5

As we decrease the capacity of the common feature extractor in OvA 2 to 4, the performance of both classification and outlier detection degrades. Since this shared block extracts features and does not make a decision, the more learning capacity it has, the better the results. Although feature extractor 1 performs about 1% better, as a design choice, we use feature extractor 2 for the rest of this work because of its smaller network size. For the remaining of this work, we use OvA 3, having the common feature processing architecture.

Table 5.2: OvA Network Sizes

OvA	Description	# params.
1	Feat. Ext. 1 + 10 Feat. Process.	890,170
2	Feat. Ext. 1 + Common Feat. Process.	152,170
3	Feat. Ext. 2 + Common Feat. Process.	99,754
4	Feat. Ext. 3 + Common Feat. Process.	46,226

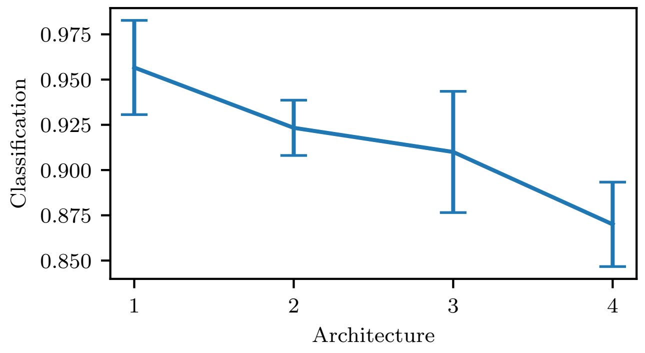
5.7.2 Architecture Description

The architectures for Disc, DClass, and OpMx consist of Feature Extractor 2 followed by a feature processing block and a decision block with 1 sigmoid, $|\mathcal{A}| + 1$ softmax, and $|\mathcal{A}|$ softmax respectively. Disc was provided with a larger feature processing network having an additional Dense network with 100 neurons after the flattening to be comparable in size to the other networks. The autoencoder architecture is shown in Fig. 5.9. It consists of encoder with a bottleneck consisting of 32 samples followed by a decoder which reconstructs the signal.

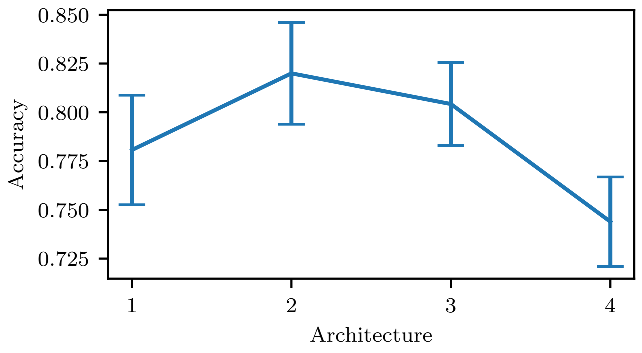
The number of trainable parameters of each network is shown in Table 5.3. The network sizes of OvA, DClass, and OpMx scale with $|\mathcal{A}|$[1]. AutoEncoder and Disc have a fixed number of parameters for any value of $|\mathcal{A}|$. Notice that OvA and OpMx have an identical number of parameters while DClass differs by only an additional 81 parameter.

From an architecture perspective, Disc, OvA, OpMx, and DClass are very similar. They share the same feature extractor and feature processing blocks, which constitute most of their neural network. The difference between these methods come from the type of activation

[1]In our previous work [HKC20b], since we repeated the feature processing block for OvA, the network size increased with $|\mathcal{A}|$ at a much higher rate



(a) Classification Accuracy



(b) Outlier Detection Accuracy

Figure 5.8: The average performance of the OvA architectures from Table 5.2 are shown as a blue solid line. The error bars represent the standard deviation due to training 10 repetitions of the same network using the same data.

function, data labeling, and post processing performed. These differences lead to conceptual differences in the approach as we have discussed, leading to different characteristics as we summarized in Table 5.1, and will lead to significantly different performance.

5.8 Evaluation Procedure

In this section, we describe the evaluation procedures used through out this work. We discuss the evaluation metrics used, steps to avoid the dependence on the specific division of the dataset to different sets ($\mathcal{A}$, $\mathcal{K}$, $\mathcal{O}$), and the way we evaluate how the approach generalizes across time.

As stated earlier, for many of these networks, there is a threshold which defines the trade-off between detection and false alarm. This trade-off is typically represented by the Receiver Operating Characteristic (ROC) curve relating the probabilities of false alarm and detection. To compactly visualize the results, we consider the area under the ROC curve (AUC) calculated using integration, which measures performance in a manner independent of the threshold [SWK09]. The procedure for calculating the ROC curve for each approach is

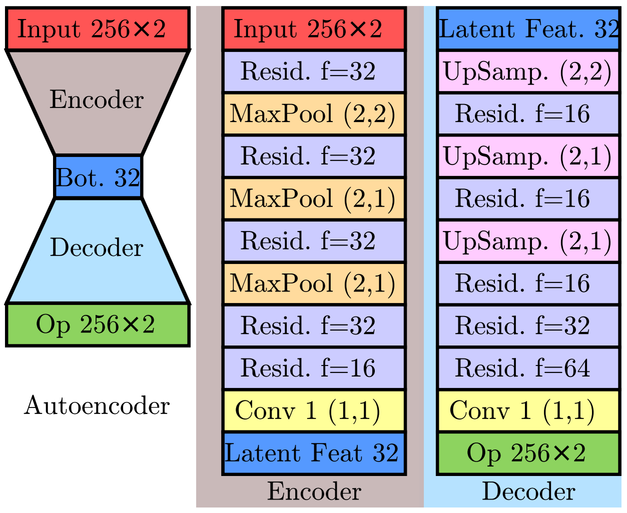


Figure 5.9: The architecture of the autoencoder. The residual block is described in Fig. 5.5

described in Appendix C. Still, when the network is deployed, we have to choose a threshold. Given a specific threshold-set, we calculate the accuracy of correctly identifying outliers over a balanced test set, such that random guessing would yield a 50% accuracy. In that case, the accuracy is the average of the performance on the authorized samples given by $1 - P_{FA}$, and the outliers given by P_D. The thresholds are chosen to provide a high value of accuracy according to the procedures discussed in Appendix C. We use the accuracy instead of the F1 score since it is an easier metric to interpret when the dataset is balanced. Classification accuracy results included for the authorized samples are evaluated for a balanced version of the test set having the same number of the samples from each authorized transmitter, where any trivial or random guess would yield an accuracy of $1/|\mathcal{A}|\%$. Note that as stated earlier, not all methods have an adjustable threshold (as summarized in Table 5.1) and hence won't have a corresponding AUC result.

Unlike with classification, where typically all transmitters available are used, outlier detection involves dividing the transmitters into sets of authorized transmitters, outliers, and possibly known outliers. Since RF fingerprints are random, some transmitters are more similar than others, and a comprehensive evaluation cannot be done using only one realization of

Table 5.3: Network Size

Net	# of trainable prams.
OvA	98944 + 81 $\lvert\mathcal{A}\rvert$
Disc	127,605
DClass	99,025 + 81 $\lvert\mathcal{A}\rvert$
AutoEnc	109,362
OpMx	98944 + 81 $\lvert\mathcal{A}\rvert$

the sets. This adds another source of variability to our results, besides the inherent randomness in training neural networks. To demonstrate this, we train OvA using 10 authorized nodes and evaluate it using 63 outlier nodes picked randomly from the 163 transmitters. Ten random realizations of these sets are compared and for each we train 10 repetitions. The results are shown as a box plot in Fig. 5.10, from which we can see up to 9% difference in the median due to the different realizations of the sets. This is more significant than the difference between the first and third quartiles due to training randomness which did not exceed 3%. Based on these findings, our evaluation considers multiple realization of the sets while only considering one repetition.

As for assessing the ability of our network to generalize through time, we create two test sets: a same-day test set which was captured on the same days as the training set and the different-day test set, which was captured on a different day than the training data.

Based on the previous considerations, we describe our evaluation procedure. For certain values of $|\mathcal{A}|$, $|\mathcal{K}|$, and $|\mathcal{O}|$, we randomly partition the dataset to $\mathcal{A}$, $\mathcal{K}$, and $\mathcal{O}$ to form 10 realizations of $\{\mathcal{A}, \mathcal{K}, \mathcal{O}\}$. All approaches are evaluated using the same 10 realizations and the results are shown in terms of mean and standard deviation. For the training data and

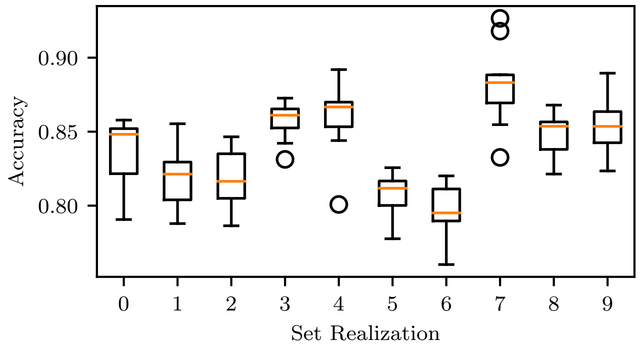
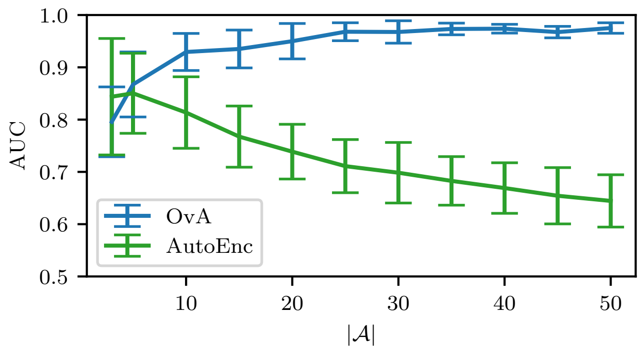


Figure 5.10: Box plot of the outlier detection accuracy of OvA is shown for several realizations of the sets. For each set realization, a network is trained for 10 repetitions. The center line represents the median, the box represents the first and third quartiles, and the whiskers represent the range, except for outlier points which are represented as circles.

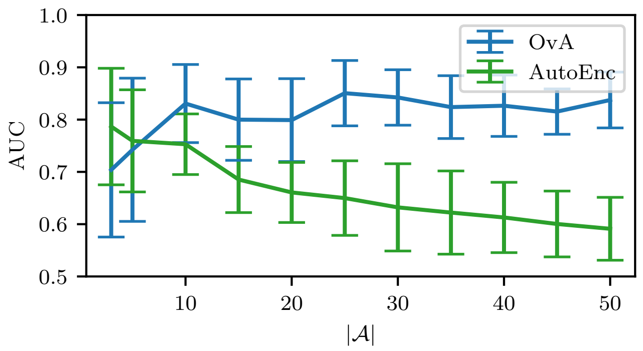
the same-day test data, we use only samples from the captures made on the first four days, while the last day capture is entirely left for different day testing.

Given a realization of $\mathcal{A}$, $\mathcal{K}$, and $\mathcal{O}$, for training and validation, we use 70% of the samples belonging to $\mathcal{A}$, and all the samples belonging $\mathcal{K}$, from the same day data. The combination of this data is split into 80% for training and 20% for validation. The same day test set contains all samples from $\mathcal{O}$ and the remaining 30% of $\mathcal{A}$. For different realizations of the sets, the dataset can get highly imbalanced. To avoid degenerate solutions, where the network always predicts the class with the majority of samples, the training loss is weighted based on class frequency. As for the different day test set, it is obtained by combining all samples from $\mathcal{A}$ and $\mathcal{O}$ from the last day capture.

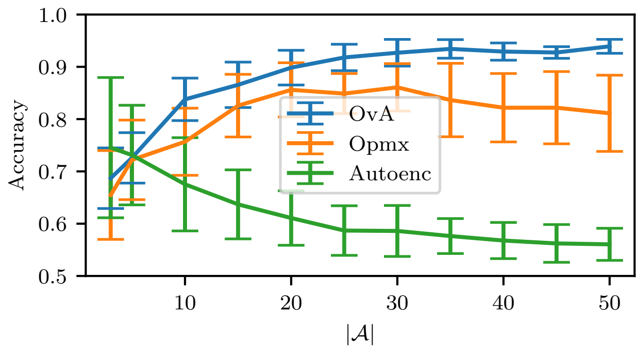
The training used 10 epochs using the ADAM optimizer with a learning rate of 0.001. The weights corresponding to the epoch which produced the lowest validation loss are kept. Data was first normalized, then augmented by adding noise with a variance of 0.01 and applying a uniform random phase shift. Cross-Entropy was used as the loss function for Disc, DClass, OvA, and OpMx with classes weighted depending on the number of samples of each class. AutoEnc used MSE loss.



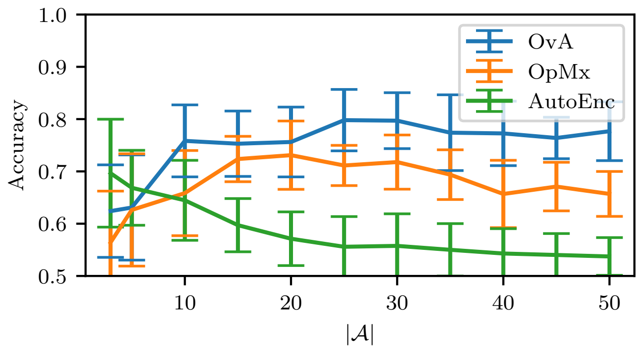
(a) AUC – Same Days



(b) AUC– Different Day



(c) Accuracy – Same Days

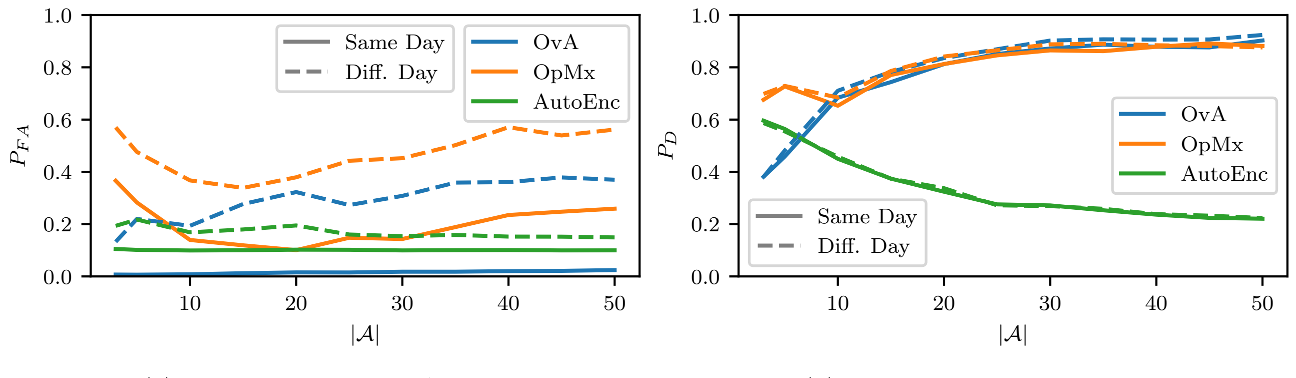


(d) Accuracy – Different Day

Figure 5.11: Average outlier detection performance of several approaches as we change $|\mathcal{A}|$. Error bars represent the standard deviation for different realizations of the sets.

5.9 Transmitter Set Sizes Evaluation

In this Section, we explore the effect of changing the size of the required authorized set $\mathcal{A}$, and evaluate the effect of having a known outlier set $\mathcal{K}$ and its size on the ability of the network to distinguish authorized signals from outliers. Throughout this Section, we used $|\mathcal{O}| = 63$ for the evaluation.



(a) Probability of False Alarm

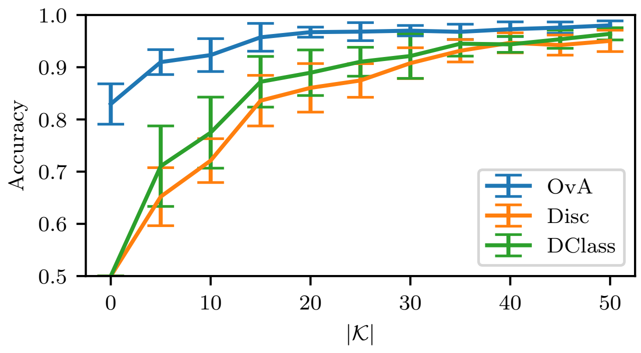
(b) Probability of Detection

Figure 5.12: Average outlier detection performance of several approaches as we change $|\mathcal{A}|$. Error bars are omitted for clarity. Solid lines represent same days test, and dashed line represent different day test.

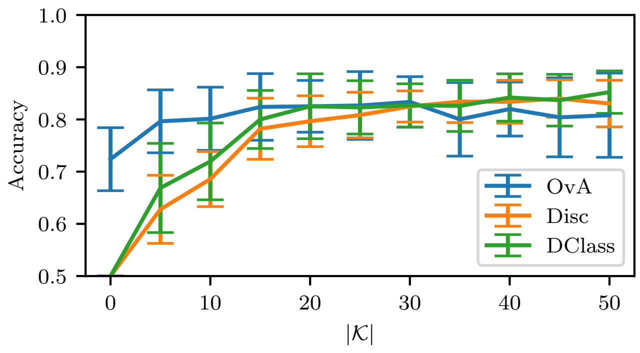
5.9.1 Authorized set

We start the evaluation by considering no known outliers, i.e, $|\mathcal{K}| = 0$. We want to know how large the set $\mathcal{A}$ has to be for good outlier detection and what performance can be achieved thereof. Results are shown for AUC and accuracy in Fig. 5.11a and Fig. 5.11c for the same-day test. For OvA, we see that as we increase the number of authorized nodes, the average AUC increases and its standard deviation decreases, showing less dependence on the set realization. The accuracy, shown in Fig. 5.11c, follows the same trend, and we are able to achieve accuracies above 90% on the average when $|\mathcal{A}|$ is more than 20. The reason behind this pattern is that as $|\mathcal{A}|$ increases, each binary classifier has more signals from other transmitters, helping it learn better its designated transmitter without memorizing others, leading to better generalization. This interpretation is supported by the observation that the improvement in accuracy is due to the decrease in P_D for the same P_{FA} as shown in Fig. 5.12b and Fig. 5.12a, respectively.

As for autoencoders, the trend seems to be reversed in Fig. 5.11a and Fig. 5.11c. As $|\mathcal{A}|$ increases, both the accuracy and AUC decrease. Autoencoders generate a compressed



(a) Accuracy – Same Days

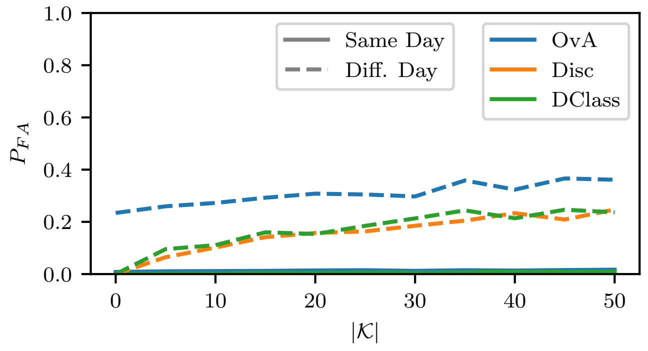


(b) Accuracy – Different Day

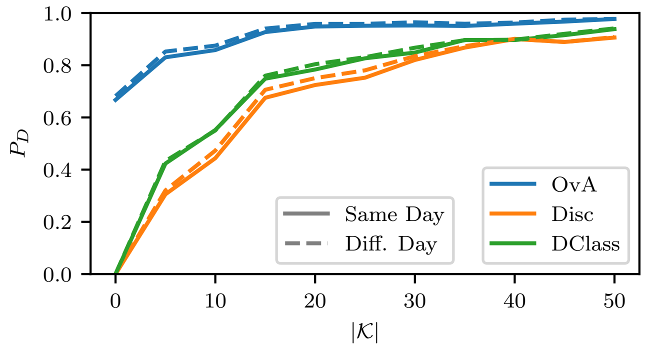
Figure 5.13: Average outlier detection performance of several approaches as we change $|\mathcal{K}|$. Error bars represent the standard deviation for different realizations of the sets.

representation of their input by memorizing their distribution. For small $|\mathcal{A}|$, this is the distribution of the authorized transmitters. As $|\mathcal{A}|$ increases, they learn the distribution of signals in general, independent of the transmitter. Hence, they reconstruct signals for unknown transmitters with low MSE and fail to detect them. This is verified by looking at the decreasing P_D curve for AutoEnc in Fig. 5.12b, while P_{FA} is almost constant in Fig. 5.12a. This trend coincides with our visualization in Fig. 5.1b.

For OpMx, the accuracy increases until $|\mathcal{A}| = 20$ and then slightly decreases. This fluctuation is mostly attributed to P_{FA}, as shown in Fig. 5.12a. The results of OpMx depend on the value of the activation vectors (AV) with respect to tail distributions and the uncertainty threshold ϵ. The key is understanding that the modified activations $\mathbf{v}'$ calculated using (5.3) reduces the AV of the top α classes. After calculating the output $\mathbf{z} = \text{softmax}(\mathbf{v}')$, the maximum $z_i = \max\{\mathbf{z}\}$ is thresholded using ϵ. For small $|\mathcal{A}|$, classification for authorized is highly confident, leading to AV belonging to the tails of other classes, pushing z_i below ϵ, and leading to false alarms. As $|\mathcal{A}|$ becomes larger, this confidence decreases, leading to an improvement in P_{FA}. However, the similarity between AVs of different classes decreases, pushing them to the tail of other distributions as $|\mathcal{A}|$ increases beyond a certain point, leading to an increase in P_{FA}.



(a) Probability of False Alarm



(b) Probability of Detection

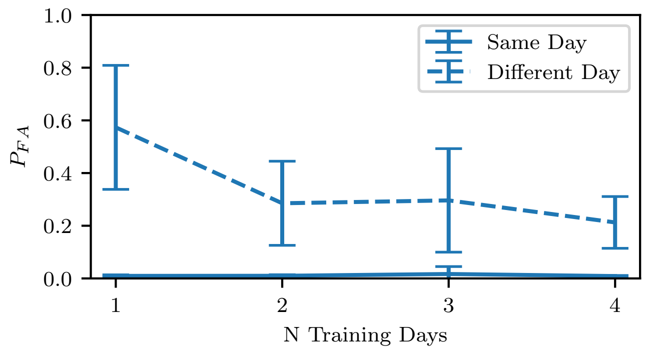
Figure 5.14: Average outlier detection performance of several approaches as we change $|\mathcal{K}|$. Error bars are omitted for clarity. Solid lines represent same days test, and dashed line represent different day test.

In comparison, we see that for small $|\mathcal{A}|$, namely $|\mathcal{A}| = 3$, AutoEnc gives the highest accuracy on average because it is able to capture the distributions of small number of transmitters. At $|\mathcal{A}| = 5$, all methods are equally as good. As $|\mathcal{A}|$ increases, OvA gives the best performance because each branch uses all the data to learn its transmitter without memorizing the other transmitters.

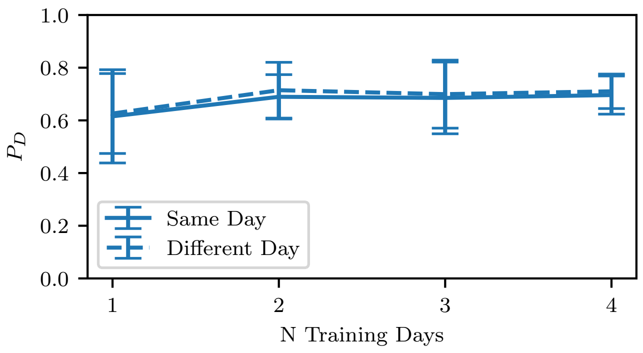
In Fig. 5.11d, we plot the accuracy for a different day test. The plots follow the same trends, but the accuracy of OvA and OpMx drop by about 15% while AutoEnc drops by only 5%. The reason behind this drop is clearer by inspecting the P_{FA} and P_D separately shown as dashed lines in Fig. 5.12. The P_{FA} and P_D curves represent the same information as confusion matrices—applied to binary classification—in a more compact manner. From Fig 5.12b, we see that the performance in detecting the new transmitters is almost unaffected. The drop in accuracy is a result of the failure to identify authorized transmitters as shown in the P_{FA} curves in Fig 5.12a. This is reasonable since any change on unseen transmitters should not have any effect, unlike changes in the learned transmitters. For OvA, we see that as we increase $|\mathcal{A}|$, P_{FA} increases, since identifying transmitters from data captures on different days becomes even more challenging as we increase the number of transmitters. The smaller gap on different day test using AutoEnc is explained by the encoders ability to learn more general features of the signal compared to OvA and OpMx, which leads to an overall smaller drop in accuracy, which also causes the lower P_D.

5.9.2 Known set

We expect that seeing more known outliers would help the network differentiate the authorized transmitters from the outliers. To show this, we evaluate the performance of the approaches that support having $\mathcal{K}$ as input, as a function of $|\mathcal{K}|$ given $|\mathcal{A}| = 10$. The accuracy curves are shown in Fig. 5.13a. As stated earlier, at $|\mathcal{K}| = 0$, DClass and Disc don't have any outlier samples for training and predict everything as authorized. From Fig. 5.13a, we see that the accuracy of all methods improve as we increase the number of known outliers. We also note that OvA is performing noticeably better than the others. This can be understood by realizing that in OvA, each binary classifier sees more samples to reject, the known outliers and the samples from other authorized transmitters. Thus, it is able to isolate its class better. DClass and Disc, on the other hand, only learn to reject samples from $\mathcal{K}$. This is further supported by looking at the curves for P_{FA} and P_D shown in Fig. 5.14a and Fig. 5.14b, where we see that the accuracy improvement is due to the probability of detection. DClass slightly outperforms Disc because the labels of $\mathcal{A}$ help it extract better features compared to Disc. So it can be concluded that even if we are not interested in classifying among the nodes in $\mathcal{A}$, including these labels in training improves the outlier detection performance. Fig. 5.13b shows the accuracy curves when using the data from a different day for testing. Again, we see the same trend from the previous section, both AUC and accuracy drop by about 15% for all methods due to the degradation of P_{FA}.



(a) Probability of False Alarm



(b) Probability of Detection

Figure 5.15: Average outlier detection performance against the number of training days. Error bars represent the standard deviation for different set realizations. Solid lines represent same days test, and dashed line represent different day test.

5.10 Dataset Training Days Evaluation

In this section, we evaluate the effect of the dataset construction on the ability of the proposed approaches to generalize over time. While developing methods to counter temporal variation in RF fingerprints is not the main focus of the chapter and has been discussed in other works [CGM20] , we study its effect on transmitter authorization. For our evaluation, we only consider the OvA architecture with $|\mathcal{A}| = 10$ and $|\mathcal{K}| = 0$. We built four datasets, where dataset i contains the data captures on dates prior to and including day i. The network was trained according to the same procedures discussed earlier and the results are shown in Fig. 5.15. From Fig. 5.15a, we see that as the number of days included in training increases, P_{FA} decreases. On the other hand, P_D is almost unaffected. The larger improvement from 1 day capture to two day captures is explained by the fact that the capture on day 1 was two months earlier than the remaining captures. During this long period, the transmitter fingerprints changed more significantly. The remaining captures were done on consecutive days, during which fingerprints had less severe changes, and hence smaller improvements to P_{FA}. Hence, a simple way to improve the robustness against temporal variation is to collect data from the authorized transmitters over an extended period of time. Still, more sophisticated approaches are needed to close the gap between same day and different day testing.

5.11 Summary of Results

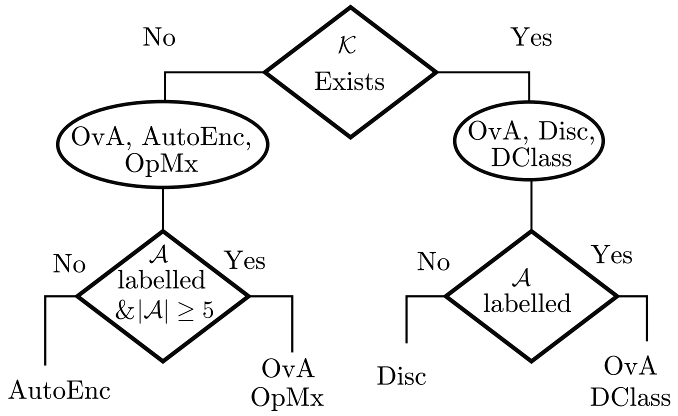


Figure 5.16: A tree diagram summarizing the feasible architectures as a function of the dataset. The networks are ordered so that the one yielding the better performance comes first.

The results we obtained can be summarized as follows. *Regarding the dataset:* It is better to label the authorized transmitters even if we are not interested in classifying among them, as it enables us to use openset methods (OvA, OpMx, DClass) which outperform the anomaly detection methods (Disc, AutoEnc) in many cases. Furthermore, the data for authorized transmitters should ideally be collected over a span of multiple days; as we have demonstrated, the drop in outlier detection accuracy is due to misclassification of authorized signals, i.e P_{FA}, and can be reduced by collecting the data over multiple days.

Regarding the approach: The dataset structure; whether it is labeled or not, and the sizes of $\mathcal{A}$ and $\mathcal{K}$, determines which approaches are feasible or are better. If we have no data from known outliers and no labels, our only option is AutoEnc. Without known outliers and with the availability of labels, for $|\mathcal{A}| \leq 5$, ignoring the labels and using AutoEnc is a better choice driven by its superior performance for small $|\mathcal{A}|$; if $|\mathcal{A}| > 5$, OvA gives the best performance. If we have a pre-trained classifier, then OpMx is the best option. If we have $\mathcal{K}$, we can use OvA, Disc, and DClass. Without labels, we are limited to Disc. If we have labels, OvA is typically better than DClass with tunable thresholds. This is summarized in Fig. 5.16.

Regarding the network architecture: We have shown that if we have labeled data, the performance on classifying the transmitters does not necessarily correlate with the performance on outlier detection. A known outlier set is recommended to optimize the architecture.

Eventually, if we are able to collect the data as we wish (having labels, data for authorized transmitters captured over multiple days, and with data for known outliers) the best approach is OvA. When using 10 authorized transmitters and 50 known outliers, it yielded an outlier detection accuracy of 98% on the same day test set and 80% on the different day test set.

5.12 Summary

In this chapter, we have considered the problem of transmitter authorization using RF fingerprints captured from raw IQ samples. Since this problem has been scarcely investigated in the wireless domain, we performed a comprehensive evaluation of the most prominent machine learning approaches from the openset recognition and the anomaly detection literature, as applied to our problem definition. The dependence of the evaluation results on the choice of transmitters was demonstrated and a simple strategy was proposed to reduce it. We have also shown that the performance of a given neural network model on closed set classification is not an indicator of its performance in outlier detection, indicating the need for architectures specifically designed for this problem. Also, we demonstrated that minor change in network architecture and data labeling can lead to a significantly different approaches. Using a known outlier set was proposed and was shown to improve the outlier detection accuracy. While classification based OvA gives the highest accuracy in most cases, it is outperformed by reconstruction based AutoEnc for small number of authorized. This opens the door for hybrid approaches combining classification and reconstruction. We also pointed out that the temporal variation of fingerprints is an open problem for transmitter authorization.

CHAPTER 6

Blind Signal Decoding

Blindly decoding a signal requires estimating its unknown transmit parameters, compensating for the wireless channel impairments, and identifying the modulation type. While deep learning can solve complex problems, digital signal processing (DSP) is interpretable and can be more computationally efficient. To combine both, we propose the dual path network (DPN). It consists of a signal path of DSP operations that recover the signal, and a feature path of neural networks that estimate the unknown transmit parameters. By interconnecting the paths over several recovery stages, later stages benefit from the recovered signals and reuse all the previously extracted features. The proposed design is demonstrated to provide 5% improvement in modulation classification compared to alternative designs lacking either feature sharing or access to recovered signals. The estimation results of DPN along with its blind decoding performance are shown to outperform a blind signal processing algorithm for BPSK and QPSK on a simulated dataset. An over-the-air software-defined-radio capture was used to verify DPN results at high SNRs. DPN design can process variable length inputs and is shown to outperform relying on fixed length inputs with prediction averaging on longer signals by up to 15% in modulation classification. This chapter revises our previous publication [HDC21].

6.1 Introduction

In a typical wireless communication system, the transmitter and receiver exchange waveforms following an agreed upon protocol. However, a prior agreement on waveforms is not always possible and heterogeneous radios might need to communicate without a protocol predefining waveforms. Without a known protocol, a blind receiver has to reconstruct and decode an unknown waveform in many applications. Civilian applications include decoding unknown signals to enable exchanging messages between heterogeneous radios using dynamic spectrum access. Military applications of blind decoding include intercepting hostile communications.

Decoding a blindly received signal is a challenging problem. In addition to all the channel impairments facing protocol based communications, a blind receiver lacks any knowledge about the transmitted signal. Without an agreed upon preamble for synchronization and channel estimation and without knowing the type of modulation, a blind receiver has only access to a sequence of IQ samples representing unknown symbols. From this sequence, the receiver needs to figure out the symbol rate, the modulation type, the channel impulse response for equalization, and compensate for carrier and timing synchronization errors to recover the transmitted symbols. Additionally, the length of a blindly received signal is not determined apriori.

While it is easy to generate signals of different parameters and channel distortions, developing signal processing algorithms that jointly estimate all these parameters for blind decoding is more challenging. Many existing works have leveraged blind signal processing to address blind decoding and modulation classification [RYU14, KDG19, LW19]. However, these works address the estimation problems separately, thus, not benefiting from joint estimation, and many of them make assumptions impractical for a blind receiver like assuming synchronization.

Driven by the availability of training data, some of the estimation problems in blind decoding were separately addressed using deep learning; Center frequency and timing offset estimation neural networks were proposed in a non-blind context for QPSK signals [OKC17]. For modulation classification, deep neural networks were shown to outperform manually designed features in [ORC18]. However, posing blind decoding as a set of independent deep learning estimations has its drawbacks. It does not allow the networks to share common features and thus prevents them from performing joint estimation by learning the dependencies between the parameters present in the signal. It also denies each network benefiting from the other networks' outputs, which can partially recover the signal and reduce distortions. To avoid these limitations, another option is an end-to-end deep learning solution taking the signal as input and outputting the symbols.

Using a black box end-to-end neural network for blind decoding also has its drawbacks. Unlike modulation classification or estimation, blind decoding needs to be applied to the entire length of the signal and thus needs to be efficiently scalable. Deep learning can have a high computational complexity compared to digital signal processing (DSP) and lacks interpretation [BS19]. For efficient scaling, estimating the unknown parameters and relying on DSP for compensation can be more computationally efficient than relying on an end-to-end black box neural network. Additionally, interpretable outputs can easily be integrated with existing DSP techniques. The drawbacks of separate estimations and end-to-end solutions motivate for a model-based deep learning solution [HJW19], integrating both signal processing and deep learning in a design combining the benefits of both.

Another common limitation of neural networks used in many existing works addressing similar problems is the fixed input size. For instance, in [ORC18], it was shown that neural networks using longer signals can lead to improved modulation classification accuracy. However, the proposed networks can only process fixed length signals and network redesign and retraining is required to change the input length. Blind receivers do not have control over the received signal length and it is not practical to train a separate network for each possible signal length. To use fixed length networks on variable size inputs, one approach is to train them on short signals and divide longer signals into chunks and use averaging as proposed in [ZQC19]. Another approach, which provides more flexibility, is to design the neural networks that handle variable length signals as proposed in [RMG18, CB20]. However, it is not clear which approach yields a better performance.

In this chapter, we propose the Dual Path Network (DPN) for blind decoding and modulation classification. To overcome the limitations of separate estimations and end-to-end networks, the dual path network is designed as two paths; a signal path and a feature path. The signal path consists of linear signal processing operations where frequency offset, noise, and fading are compensated for sequentially. The feature path consists of neural networks that process the features extracted from the input and the compensated signals. These features are used to predict interpretable signal processing estimates and filter taps. Using this architecture, DPN benefits from recovered signals and can combine features from all stages. Once the estimates have been obtained, due to their signal processing interpretation, they can be integrated with existing signal processing techniques. By leveraging average pooling and recurrent networks to obtain the predictions, DPN can process variable length inputs. Our contributions can be summarized as follows

- We propose the Dual Path Network (DPN) architecture for blind signal decoding, which can process inputs of variable lengths. DPN design integrates signal processing with neural networks yielding interpretable outputs. The proposed architecture enables learning from recovered signals and allows features combining.

- We verify the benefit of learning from recovered signals and combining features along the signal path. To do that, we compare DPN to alternative architectures that lack access to either recovered signals or feature combining and show that DPN outperforms either architecture by up to 5% in modulation classification.

- For inference on variable length, we show that using DPN trained and tested with variable lengths provides lower estimation errors and up to 15% improvement in modulation classification compared to averaging the predictions calculated using the same network trained and tested only using a short fixed input size.

- We show that DPN provides lower estimation errors and a higher percentage of correctly decoded signals for BPSK and QPSK on the test dataset compared to an implemented blind signal processing algorithm.

6.2 Related Work

In section, we start by discussing existing signal processing approaches for blind decoding, then we survey the deep learning approaches used for problems that are part of blind decoding. Works that have considered deep learning demodulators are discussed last.

DSP approaches for Blind Decoding Blind decoding DSP approaches rely on recovering the symbols then analyzing them to identify the modulation types. In [RYU14], cyclostationary features were used for carrier frequency offset and symbol rate estimation to recover the symbols. Using the recovered symbols, a decision tree algorithm based on statistical tests for blind modulation classification was proposed. Assuming synchronization in an AWGN channel, joint modulation classification and symbol decoding were performed using two approaches based on a Bayesian framework and a minimax framework in [KDG19] . Under the assumption of perfect synchronization, in [LW19], an iterative approach was proposed for joint blind channel estimation, modulation classification, channel coding recognition, and data detection using a different approach for each task.

Deep Learning Approaches for Estimation and Modulation Classification Many deep learning approaches were proposed for problems relevant to blind decoding. Deep learning was proposed for carrier frequency offset and timing offset estimation and was compared to traditional estimators in [OKC17]. In [OCC16], deep learning from raw IQ samples was shown to outperform manually designed features in the problem of modulation classification. This result sparked a wide interest in developing novel modulation classification neural network architectures. In [ORC18], a residual neural network was proposed and the effects of fading and frequency offsets were studied. Recurrent neural networks were proposed for modulation classification in [HZX17] and were shown to outperform convolutional networks. A network combining both CNN and LSTM networks was proposed in [ZLW20]. In [VTB18], branch convolutional neural networks, consisting of a hierarchy of neural networks were used to classify 29 modulation types. A network design that combines shallow and high level features of the input signals was proposed in [NZH19]. In [CZH20], a cyclic connected CNN along with a bidirectional RNN were proposed. Instead of using IQ samples, the signals to be classified were represented as constellation images in [PJW19, HJG19, HCL19, DHH20]. Other works have focused on designing lightweight networks that reduce the number of parameters or the inference time [RJY19, RJY20, KV20, HGK20, ZWX20, LTZ19, CB20]. Modulation classification was also considered for multi-carrier signals in [XD19] and for MIMO systems in [LPW20, JWC20]. In [RMG18, WWF19], distributed modulation classification using multiple receivers to capture the same signal was proposed.

Deep learning was shown to be affected by synchronization errors, which motivated the development of custom signal processing layers. The effects of carrier frequency offsets and sampling offsets on modulation classification were studied in [HHM17]. It was shown that frequency offsets and sampling rate offsets degrade the classification accuracy with the former having a more pronounced effect. Driven by this observation, custom neural network layers that attempt to recover the signal before classification were proposed. A spatial transformer network was proposed in [MHC17] for timing recovery and was shown to improve accuracy at low oversampling. In [OPB16], a radio transformer network that can correct frequency offset and adjust the sampling rate of the signal was proposed. Similarly, a learnable distortion correction module for frequency and phase offset was proposed in [YSC18]. In [SHL20], a U-net network was proposed to reconstruct low SNR signals. In these works, the custom reconstruction layers were trained to improve the modulation classification results without the reconstruction ground truth, and hence are not guaranteed to recover the transmitted signals. Thus, the output of such networks can not be used for blind decoding.

Most of the neural network architectures in the literature were designed only to process signals with a fixed length input. To use a fixed length network on a signal with a different length, the network has to be redesigned and retrained. To extend a network to longer signals without retraining, in [ZQC19], the authors have proposed methods to combine the predictions applied on small chunks of a signal. These methods consist of averaging the predictions. Neural networks that can process variable input size signals were proposed by using LSTM in [RMG18] and by combining convolutional layers and average pooling in [CB20]. However, it is still not clear which network design yields a higher accuracy on signals of length unseen in training; a fixed length network with combined predictions or a variable length network.

Deep Learning Approaches for non blind Demodulation Several works have proposed using deep learning for demodulation when the transmitter signal parameters are known apriori. In [ZZC20], an end-to-end neural network demodulator was developed under the assumption of time synchronization between the transmitter and receiver. Several modulation types were supported and a network was trained for each type. For WiFi, neural networks were proposed to replace several signal processing blocks of a WiFi receiver in [ZDL20]. These blocks include channel estimation, phase error correction, sampling rate correction, and equalization and rely on the WiFi preambles to perform their tasks. A similar approach was proposed for 5G compliant waveforms in [HKH21]. Deep learning end-to-end communications systems, where both the transmitter and receivers are neural networks, were also proposed. In [DCH18] and [OKC16] autoencoders were proposed for SISO communications and in [OEC17] for MIMO. These approaches, however, rely on knowing or designing the transmitted signal and do not directly apply to blind decoding.

The dual path network was first introduced in our prior work [HDC20], which we extend in this work. In this work, DPN design and training process was improved as discussed later. A more thorough evaluation was performed to evaluate the effect of signal reconstruction and feature sharing among different stages. The blind decoding performance was compared with both blind and Genie signal processing approaches. An over-the-air capture was used to validate our results. DPN was evaluated on variable input lengths and different approaches for variable length training and inference were considered.

6.3 System Model and Problem Formulation

We start by describing the system model and the underlying assumptions of our work. A transmitter sends a vector of complex symbols $\mathbf{s} \in \mathbb{C}^{N_s}$ using modulation type M from a set of modulations $\mathcal{M}$. In the most general case, the transmitted signal $x(t)$ is determined by symbols $\mathbf{s}$ and symbol duration τ through a modulation specific mapping function $\mathcal{G}$ such that $x(t) = \mathcal{G}(\mathbf{s}, \tau)$. For a linear modulation type, each symbol s_i represents a mapping from bits to a constellation point, and the transmitted signal $x(t)$ is given by $x(t) = \sum_{i=1}^{N_s} s_i p(t - i\tau)$ where $p(t)$ is the pulse shaping filter. The signal is upconverted and transmitted over a multipath fading channel modeled with an impulse response $h(t)$ having a delay spread σ. At the receiver, the downconverted and sampled signal is modeled using the vector $\mathbf{y} \in \mathbb{C}^{N_r}$ with the k-th element given by

$$y[k] = e^{j2\pi(f_0 t_k + \phi_0)} \int_{-\infty}^{\infty} x(\gamma) h(t_k - \gamma) d\gamma + n(t_k) \tag{6.1}$$

where f_0 is the carrier frequency offset, ϕ_0 the phase offset, and $n(t)$ is the additive white Gaussian noise process with zero mean and power spectral density $N_0/2$. We assume that the receiver sampling rate is given by $\frac{1}{\tau_0}$. The sampling instance of the kth sample is given by $t_k = t_0 + k\tau_0$, where $\tau_0 \leq \tau$ and t_0 is the symbol timing offset. The length of the transmitted symbols N_s and the received IQ samples N_r are related using $N_r = N_s \left\lceil \frac{\tau}{\tau_0} \right\rceil$. Such signals can be captured using a narrowband receiver coarsely tuned to the center of a spectrum occupancy or using a wideband receiver followed by coarse band segmentation. Either way, we assume that the vector $\mathbf{y}$ contains only one signal and that the sampling satisfies the Nyquist criterion. The parameters f_0, ϕ_0, h are assumed to be constant along the duration of each signal.

To avoid dependence on hardware specifications that vary from one radio to the other, we normalize the previous parameters with respect to the sampling rate. This is done by using the frequency normalized to the sampling rate $\frac{f_0}{1/\tau_0}$, the number of samples per symbol $\frac{\tau}{\tau_0}$, and the normalized timing offset $\frac{t_0}{\tau_0}$. Without loss of generality, to simplify the notation, we

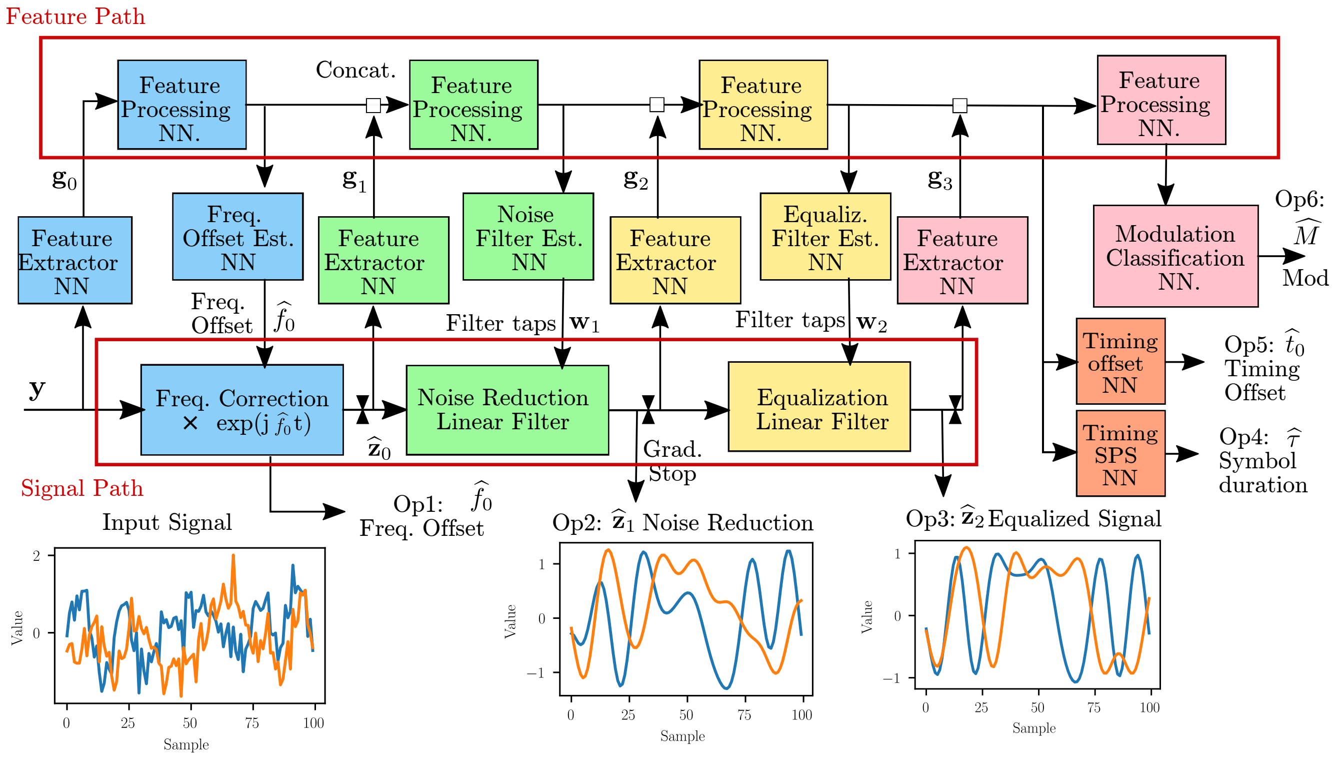


Figure 6.1: The Dual Path network consists of feature path and a signal path connected using neural networks (NN) for parameter estimation and feature extraction. A slice of an example input signal and expected outputs of the network are plotted.

consider $\tau_0 = 1$, which is equivalent to assuming a 1Hz sampling rate. Under this assumption, the number of samples per symbol τ becomes equal to the symbol duration.

The problem considered is described as follows: Given the vector $\mathbf{y}$, the receiver's objective is to identify the modulation type M among the set of modulations $\mathcal{M}$ and recover the transmitted symbols $\mathbf{s}$.

6.4 Dual Path Network (DPN)

The main ideas behind the dual path network design are: (1) the less the distortions, the better the predictions, (2) Estimating jointly is better than separately. If the receiver had access to the transmitted signal, $x(t)$, the considered problem would have been easier. Using this idea, to improve its predictions, DPN attempts to recover the signal $x(t)$. The signal recovery is inspired by existing DSP techniques, which rely on knowing the signal parameters (symbol rate, modulation type, etc.) [Har11, HEE16]. To overcome the ignorance of these parameters, the dual path network uses neural networks to predict the values for the compensation. These networks are designed to enable joint estimation by maximizing the feature sharing among them.

6.4.1 Dual Path Design

The proposed network consists of two paths: a signal path consisting of linear operations that gradually restore the input signal, and a feature path made of neural networks (NN) used to predict the parameters needed to recover the transmitted signal. The network is shown in Fig. 6.1. The signal path is described as follows: It starts by compensating for the frequency offset using the network's prediction $\widehat{f_0}$, to obtain the vector $\widehat{\mathbf{z}}_0$ defined as

$$\widehat{z}_0[k] = y[k]e^{-j2\pi\widehat{f}_0 k} \tag{6.2}$$

The noise of the signal is then attenuated using the predicted real filter taps $\mathbf{w}_1$ and generates the vector $\widehat{\mathbf{z}}_1$ defined as

$$\widehat{z}_1[k] = \widehat{z}_0[k] * w_1[k] \tag{6.3}$$

where $*$ denotes the linear convolution operator. At the end of the signal recovery, equalization is performed using the predicted complex filter taps $\mathbf{w}_2$, to generate the vector $\widehat{\mathbf{z}}_2$ defined as

$$\widehat{z}_2[k] = \widehat{z}_1[k] * w_2[k] \tag{6.4}$$

Note that due to linearity, both filtering stages can be combined, and a single set of predicted filters can perform both operations. However, we decided to keep the two filters separate similar to our prior work [HDC20] to make training easier and for the network to extract features from two recovered signals instead of one.

Feature Extractor NNs extract the features vectors $\mathbf{g}_0$, $\mathbf{g}_1$, $\mathbf{g}_2$, and $\mathbf{g}_3$ from the signals $\mathbf{y}$, $\widehat{\mathbf{z}}_0$, $\widehat{\mathbf{z}}_1$, and $\widehat{\mathbf{z}}_2$ respectively. Each of these feature vectors is concatenated with the processed previous features along the feature path, which consists of 4 cascaded Feature Processing neural networks (NNs) as shown in Fig. 6.1. The predicted parameters used by the signal path $\widehat{f}_0$, $\mathbf{w}_1$, and $\mathbf{w}_2$ along with predictions of the timing offset $\widehat{t}_0$, symbol duration $\widehat{\tau}$, and the modulation type $\widehat{M}$ are obtained by their own dedicated estimation neural networks. The layer by layer description of the networks is discussed later in Section 6.4.5. The recovery of the symbols $\mathbf{s}$ is performed using these parameters in post processing as described later in Section 6.5.2.

6.4.2 Design Motivation and Considerations

In this section, we explain the motivation behind DPN design. The sequential stage by stage design enables later stages to benefit from the output of earlier stages. The stage order is chosen based on the effect of the distortion on the signal time domain representation. A small frequency offset can significantly alter the signal in time domain even at a high SNR, so it was considered first. Frequency recovery is also typically the first stage in classical demodulators [Har11]. For limited fading spread, the noise has a more pronounced effect on the signal, and it was considered as the second stage. However, other stage orders are possible; for instance, in our prior work [HDC20], the noise reduction stage was first due to a different choice of training data. The training data used in DPN is discussed later in Section 6.4.4.

Besides the 3 recovery stages, other DSP operations need to be performed to recover the symbols, however, there are design considerations that prevent integrating them into DPN. The first consideration is that all operations in DPN have to be differentiable to train it using gradient descent. The second consideration is that the output size needs to be determined only by the input size and should not depend on the network predictions. Although there are known techniques to train NN that violate this consideration (commonly used in natural language processing), this consideration makes training easier. The third consideration is that the network should not be penalized for phase ambiguity. For example, if a signal $x(t)$ is a valid BPSK signal, $-x(t)$ is also a valid BPSK signal. Since the network does not have sufficient information to determine the correct one, it should not be penalized.

These considerations make it hard to design a network that uses DSP modules to output the N_s symbols $\mathbf{s}$ for several reasons; (1) the relation between the signal length N_r and N_s depends on the number of samples per symbol τ, which is unknown and varies from one training sample to the other, (2) the number of possible values of each output symbol is also a variable that depends on the modulation order, (3) considering phase ambiguity, there are several possible valid values of the sequence symbols. So instead of designing a network that outputs $\mathbf{s}$ directly, we design a network that recovers the transmitted signal and estimates the parameters needed to decode $\mathbf{s}$, which is done in post-processing.

6.4.3 Design Merits and Drawbacks

The proposed DPN design combining neural networks and signal processing has several merits:

1. *Predicting using restored signals*: The less the distortions in the signal (higher SNR, smaller frequency offsets, etc.), the better the predictions that neural networks obtain [HHM17]. DPN design gradually restores the transmitted signal and uses the restored signal in the later predictions. This should lead to improved predictions.

2. *Incremental Feature Combining*: Each prediction made by the network has access to all features from the previous stages. For instance, the modulation classification NN, through the feature path can leverage all the previous features $\mathbf{g}_1$, $\mathbf{g}_2$, $\mathbf{g}_3$, and $\mathbf{g}_4$ not just $\mathbf{g}_4$. The combining enables any prediction NN block to reuse relevant features from the previous stages. It also works as a backup in case a signal reconstruction stage leads to degradation in the signal due to misprediction, which might occur at

low SNR.

3. *Compatibility with Existing DSP Methods*: The DPN outputs are frequency estimates, filter taps, and timing estimates. All these parameters have a clear interpretation and can be reused using existing signal processing techniques. For instance, if the received signal is too long, DPN can be applied to obtain estimates on a short portion of the signal. The estimates like frequency offset can be directly applied to the remaining signal. In case there exists a time varying frequency drift, a frequency offset tracking algorithm can be applied to DPN's estimate.

All three claimed merits are verified in the results section. To verify the first two merits, we consider two alternative designs of DPN, which are described later in Section 6.4.7.

The design of DPN also has several drawbacks. Since the signal path consists only of linear operations, DPN can only perform linear reconstruction of the signal, that is the reconstructed output $\widehat{\mathbf{z}}_2$ is obtained using linear operations on the input $\mathbf{y}$. This is in contrast with using a fully neural network design which can approximate non linear relations between its input and output. However, non linear distortions are typically mitigated in the radio hardware design, and communication systems are typically modeled using linear operations. The second drawback is that training DPN requires different types of data for training. DPN requires the true value of many signal parameters like frequency and timing offsets in addition to $\mathbf{y}$. However, these parameters are easy to obtain in a simulator. To train DPN with an over the air capture, a long preamble can be used to precede known transmitted signals. Using existing signal processing techniques, the channel, the frequency, and timing offsets can be estimated to obtain the required training data.

These previous merits and drawbacks are specific to the proposed design. DPN also inherits some of the merits and drawbacks of deep learning. DPN predicts using the weights calculated in training instead on relying on manually designed features. This data driven approach makes it relatively easy to support another modulation type for instance at the

Table 6.1: Loss Functions

Equation
$L_1 = \lvert f_0 - \widehat{f_0}\rvert$
$L_2 = \frac{1}{N_r}(\widehat{\mathbf{z}}_1^H\widehat{\mathbf{z}}_1 + \mathbf{z}_1^H\mathbf{z}_1 - 2\lvert\widehat{\mathbf{z}}_1^H\mathbf{z}_1\rvert)$
$L_3 = \frac{1}{N_r}(\widehat{\mathbf{z}}_2^H\widehat{\mathbf{z}}_2 + \mathbf{z}_2^H\mathbf{z}_2 - 2\lvert\widehat{\mathbf{z}}_2^H\mathbf{z}_2\rvert)$
$L_4 = (\widehat{t_0} - t_0)^2$
$L_5 = (\widehat{\tau} - \tau)^2$
$L_6 = \text{crossentropy}(M, \widehat{M})$

cost of lacking an interpretation of what is being learned.

6.4.4 Training Data and Method

DPN is trained using stochastic gradient descent. The labeled data used in training consists of received signals $\mathbf{y}$ along with their corresponding values f_0, $\mathbf{z}_1$, $\mathbf{z}_2$, t_0, τ, and M, where $\mathbf{z}_1$ is the noise free received signal

$$z_1[k] = \int_{-\infty}^{\infty} x(\gamma)h(t_k - \sigma)d\gamma \tag{6.5}$$

and $\mathbf{z}_2$ is the noise free and equalized signal

$$z_2[k] = x(t_k) \tag{6.6}$$

Since, we are including the values of f_0, $\mathbf{z}_1$, $\mathbf{z}_2$, the reconstruction stages are trained in a supervised manner. To evaluate the feasibility omitting the signal ground truth in blind decoding similar to [MHC17, OPB16, SHL20, YSC18], we consider an unsupervised-reconstruction alternative architecture as discussed later in Section 6.4.7. Other choices of the training data are possible. For instance, instead of using $\mathbf{z}_1$ and $\mathbf{z}_2$, we could have designed a low pass filter $\mathbf{w}_1$ and an equalization filter $\mathbf{w}_2$ and used them for training. Since there are many possible ways to design these filters, instead of forcing the network to a specific design, we opted for only including the desired outputs $\mathbf{z}_1$ and $\mathbf{z}_2$ in the training. Also instead of using f_0 for training, we could have used a reference signal with no offset similar to what we did in our prior work [HDC20], but explicitly using f_0 was found to provide a better reconstruction. Since we are providing the ground truth along the signal path, and to avoid confusing the estimation networks, gradients were prevented from propagating along the signal path according to the gradient stops shown in Fig. 6.1.

Since the outputs are heterogeneous, each output has its own loss function as defined in Table 6.1. For the modulation type, the classifier was trained using the categorical crossentropy loss. For the frequency and timing predictions, we used the mean absolute error and the mean square error respectively. A natural choice for $\mathbf{z}_1$ is the mean squared error (MSE) loss defined as $\frac{1}{N_r}\|\mathbf{z}_1 - \widehat{\mathbf{z}}_1\|^2 = \frac{1}{N_r}(\widehat{\mathbf{z}}_1^H\widehat{\mathbf{z}}_1 + \mathbf{z}_1^H\mathbf{z}_1 - 2\widehat{\mathbf{z}}_1^H\mathbf{z}_1)$, where $(\cdot)^H$ denotes the hermitian operator. However, this choice penalizes constant phase offsets, and thus violates our design consideration for phase ambiguity. To avoid penalizing constant phase offsets, we used the modified loss shown in Table 6.1 for L_2 and L_3. It is easy to verify that $\widehat{\mathbf{z}}_1$ and $\widehat{\mathbf{z}}_1 e^{j\phi}$ for any phase ϕ would yield the same loss. Note that the modified loss used in L_2 and L_3 is still sensitive to residual frequency offsets $(\widehat{f}_0 - f_0)$, which are time varying phase offsets.

To support the feature combining along the feature path, DPN is trained simultaneously as a single network with a loss $L = \sum_{k=1}^{6} c_k L_k$, for some weights c_k. This implies that the gradients for a given NN backpropagate from all the subsequent outputs. For instance, the weights of the first feature extractor (generating $\mathbf{g_0}$) are trained using gradients from all the outputs not just $\widehat{f}_0$. This makes the weights of each NN along the feature path optimized to minimize the total loss and not the loss of a specific output.

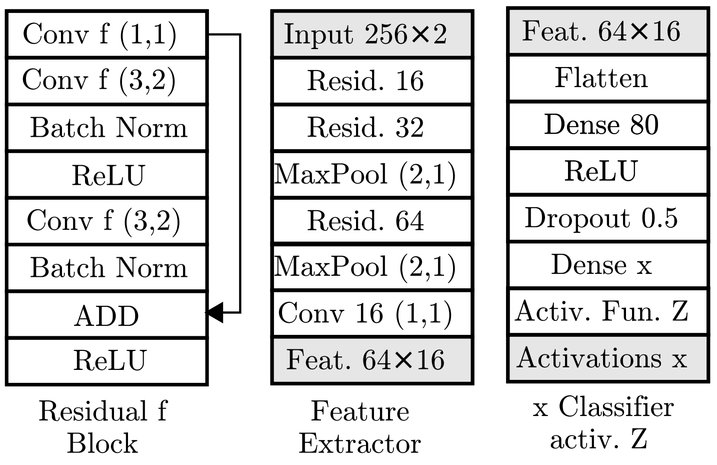


Figure 6.2: The layer by layer description of the NNs in DPN.

6.4.5 Scalable Neural Network Architecture

We start by describing the building blocks of the network. The entire network consists of a combination of convolutional networks and recurrent neural networks. This choice was not only based on their outstanding results in modulation classification [HZX17, ORC18], but also because their number of trainable parameters is independent from the input length [GBC16, Ch.10]. A convolutional layer is based on the convolution operation, in which the number of trainable filter weights is independent of the input size. Similarly, a recurrent neural network uses the same weights for any number of time steps [GBC16, Ch.10].

Then, we describe each network in details. The convolutional layers used in this work were arranged as residual blocks. Residual blocks improve the gradient flow during the training of deep networks [HZR15] and were proposed for modulation classification [ORC18]. The neural network blocks layer by layer descriptions are shown in Fig. 6.2. Both timing offset and symbol duration prediction used the same architecture referred to as Timing Est. The noise reduction filter used the Filter Est shown in the same figure. The fading filter used has complex taps, hence needs twice the number of real outputs. It was obtained by modifying the Filter Est block to have two 32 sample outputs by duplicating the layers following the last residual block. The modulation classification block consists of a gated recurrent unit

(GRU) network, which is a type of recurrent network followed by a dense layer.

To support variable length signals, all layers generating the network outputs either use global average pooling or recurrent neural networks. Global average pooling calculates the average along the time dimension. Hence, it does not have any trainable weights and is independent of its input time dimension. This is in contrast with most existing works, which rely on a dense network after the convolutions with the number of trainable weights dependent on the time dimension. The output of the recurrent network is obtained from the last time step and hence its output dimension is independent from the time dimension. The layers generating outputs independent of the time dimension are highlighted in red in Fig. 6.2.

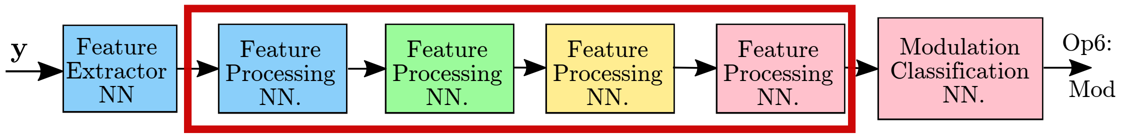
DPN has a total of 233,419 trainable parameters. This number of parameters is independent of the input length N_r. No matter how long the input is, the number of weights is constant. However, the fact that the same weights can be reused for the different lengths does not imply that the performance will generalize across different lengths. We will explore the effect of changing the signal input length on the performance in our evaluation.

6.4.6 DPN Implementation and Training Parameters

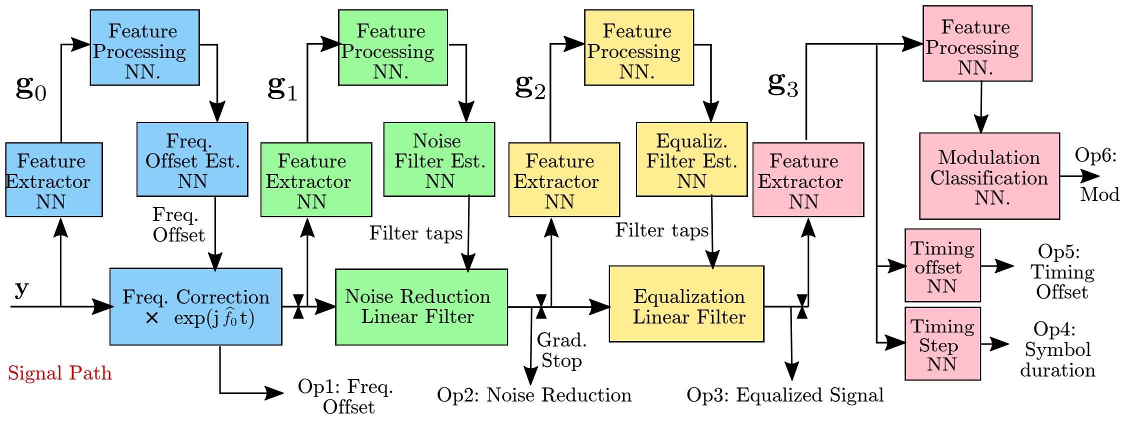
The code for DPN is publicly available[1]. DPN was implemented using the KERAS API for TensorFlow. It was trained for 100 epochs, with early stopping occurring if the validation loss did not improve for 10 consecutive epochs. The best weights according to the validation loss are retained. Note that early stopping is used to determine when the network weights have converged and not to avoid overfitting. Overfitting is avoided by using realtime signal generation as discussed later in Section 6.6.

The weights for the losses c_1 to c_6 were set to 500, 1, 5, 2.4e-4, 4.8e-4, 1.0 respectively. These values were chosen to account for the relative importance of the parameters and

[1]https://github.com/uclacores/dual_path_network



(a) Single Path



(b) Separate DPN (SepDPN)

Figure 6.3: Alternative DPN designs.

the different magnitudes of the losses which depend on the training labels values that are described later. For instance frequency offset loss (L_1) takes values in the range of 1e-2, and has a large impact on blind decoding and modulation classification and hence was assigned a large weight $c_1 = 500$. In contrast, the timing offset loss (L_4) takes values in the range of 1K (when using integer sample offsets) and hence was assigned a small weight $c_4 = 2.4e - 4$. The optimizer used for training is the ADAM optimizer with a learning rate of 0.001 and the gradients were clipped at a norm of 1.0 to avoid exploding gradients [PMB13].

6.4.7 Alternative Architectures

We consider two alternative designs for DPN to verify some of the claimed merits and a third one to evaluate the benefit of supervised reconstruction. The first design aims to quantify the impact of restoring the signal on modulation classification. This design omits the signal path entirely from DPN as shown in Fig. 6.3a and is referred to as Single Path. If restoring the signal improves predictions as claimed, DPN should outperform Single Path in modulation classification.

The second alternative design aims to verify the benefits of feature combining. To prevent feature combining, the feature path is disconnected as shown in Fig. 6.3b preventing feature combining. Since the signal path contains gradient stops, the gradients of one stage do not propagate to the previous stage and the stages are separate. This network referred to as Separate DPN (SepDPN) and is equivalent to training one stage, calculating the predictions, and using them to train the following stage. Hence, in this design, the features $\mathbf{g}_0$ are used exclusively for the prediction of $\widehat{f_0}$, and similarly for the later stages. If feature combining leads to an improvement in prediction as claimed, DPN should outperform SepDPN.

The third alternative considers the effect of training of the signal reconstruction stages without the ground truth similar to what was proposed in [MHC17, OPB16, SHL20, YSC18] for modulation classification. The design is similar to DPN design shown in Fig. 6.1 after omitting the ground truth values of f_0, $\mathbf{z}_1$, $\mathbf{z}_2$, t_0, and $\boldsymbol{\tau}$, leaving only the modulation type $\boldsymbol{M}$. To train the network, the gradient stops along the signal path were removed. The timing networks were also omitted because they cannot be trained without the ground truth. This alternative design, referred to as unsupervised DPN (unsupDPN), investigates whether unsupervised reconstruction is valid for blind decoding or not. It also investigates the impact of supervised reconstruction on modulation classification.

6.5 Signal Recovery and Symbol Decoding Algorithm

In this section, we discuss the benchmark algorithms for signal recovery used for comparison. We also discuss DPN postprocessing and symbol decoding.

6.5.1 Benchmark Signal Recovery Algorithms

To evaluate the performance of DPN in signal recovery, we contrast it with exiting DSP techniques. To that end, we consider two algorithms; the first one is a fully blind recovery algorithm based on signal processing, the second is a genie algorithm that assumes all the unknown signal parameters are provided by a genie. It is important to note that there exist many signal processing algorithms for recovery whether blind or genie. These algorithms have a tradeoff between complexity and performance. The chosen algorithms were selected to be non-iterative and to rely on well known DSP techniques.

6.5.1.1 Blind DSP Reference Algorithm (DSP Ref)

The reference blind algorithm used is based on the symbol recovery proposed in [RYU14]. It starts with band segmentation stage aiming to obtain an initial estimate of the signal center frequency and bandwidth. It is performed using the Welch power spectral density as described in Appendix D.1. The initial estimates are refined by detecting the signal's cyclostationary features as proposed in [RYU14, Sec. III-B]. The exact refinement procedures are described in Appendices D.2 and D.3 for center frequency and symbol rate respectively. After the estimation of τ and f_0, the timing offset t_0 is estimated using the Gardner symbol timing recovery algorithm [Gar86] as described in Appendix D.4. The constant modulus algorithm (CMA) is then applied for blind signal equalization [Shi12] as described in Appendix D.5.

6.5.1.2 Genie Algorithm

The genie algorithm assumes all the parameters which were used to generate the signal are known. It starts by correcting the frequency offset, then applying a low pass filter over its bandwidth, which are both assumed to be known. It uses the MMSE equalizer assuming that the channel is known as described in Appendix D.6. The symbol recovery is performed using the true τ and t_0 using the same procedures as DPN output, which are described in

Section 6.5.2.

6.5.2 DPN Post Processing for Signal Recovery

The signal output of DPN is given by $\widehat{\mathbf{z}} = \widehat{\mathbf{z}}_2$. This signal is already frequency compensated, noise reduced, and equalized by the network. The remaining processing that needs to be applied to this signal is the symbol recovery. This is done using the estimated timing offset $\widehat{t}_0$ and symbol duration $\widehat{\tau}$. To apply the timing recovery, the signal is interpolated by a factor of P, larger than any value of τ of interest, to get the signal $\widehat{\mathbf{z}}_I$. The vector $\widehat{\mathbf{z}}_I$ is padded at the start with $\widehat{t}_0 P$ zeros to correct for the time offset and then sampled every $P\widehat{\tau}$.

6.5.3 Symbol Decoding

The symbol decoding procedure is applied to the recovered symbols to evaluate the symbol error rate (SER) for the linear modulations. This same procedure is applied to the symbols calculated from all of the previous approaches (DPN, blind DSP, and genie). The symbols are decoded symbol per symbol, using a minimum distance euclidean receiver. A decision aided phase recovery loop uses the predicted symbol to correct the phase of the next symbol. This loop helps reduce the impact of any minor residual frequency errors. For evaluation purposes, the first symbol is assumed to be known and is used to estimate the phase to address phase ambiguity. The step-by-step procedure is described in Appendix D.7.

Although this approach is simple, its main disadvantage is that it propagates errors. A decision error in one symbol propagates to the remaining symbols. Since only the first symbol is used to estimate the phase in the evaluation, the initialization of the loop is subject to errors, which can propagate. More sophisticated receivers can be developed, however, we chose a simple receiver that would work for all linear modulation types. Since this receiver is common to all approaches, it should not create a bias in the evaluation.

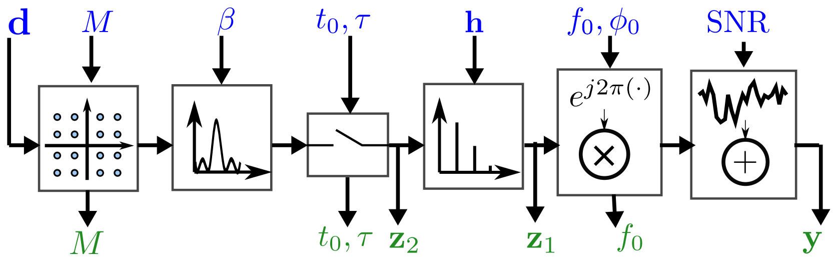


Figure 6.4: Flow graph for generating samples showing on top the input parameters and the bottom the outputs used for training.

6.6 Data Generation and USRP Capture

To train and test DPN, we generated a dataset consisting of signals with different data, modulation types, symbol rates, timing and frequency offsets, phase, channel impulses, and SNRs. While public datasets exist for modulation classification, they do not contain all the required signal labels, and hence are not suitable for blind decoding. In the dataset, each signal is generated according to the flow graph shown in Fig. 6.4. Random data $\mathbf{d}$ is generated and modulated using modulation type M selected from the set of single carrier modulations $\mathcal{M}$. If M is a linear modulation, the output is upsampled by an integer factor N_{up} and pulse shaped with a root-raised-cosine filter with a roll-off factor β. To simulate timing offsets, the first $t_0 N_{up}$ (rounded to the nearest integer) samples are removed and the signal is downsampled by a factor of $\lfloor N_{up}/\tau \rfloor$. While removing $t_0 N_{up}$ might make the first symbol unrecoverable, it is more realistic for a blind system as it emulates capturing an ongoing transmission. Since the number of removed samples and the downsampling are integers, the number of possible values of t_0 and τ is discrete and depends on N_{up} and the range of values of τ. Multipath fading is simulated using convolution with random complex fading taps having a delay spread σ. Then frequency and phase offsets, f_0 and ϕ_0, are applied, and Gaussian noise is added to model different SNRs.

The dataset consists of signals generated with parameters uniformly sampled from the ranges shown in Table 6.2. The upsampling factor was chosen to be equal to $N_{up} = 64$, making

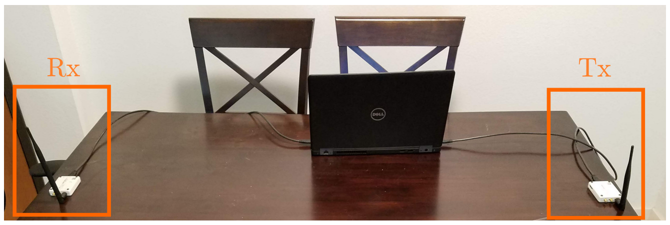


Figure 6.5: Over-the-air capture setup using USRPs B205 mini.

the number of possible values of t_0 and τ equal to 64 and 14, respectively. The channel $\mathbf{h}$ has 3 non random zero complex taps having a Rayleigh magnitude and uniform phase. The 3 taps are spaced uniformly within the fading spread. Since single carrier modulations are more commonly used in narrowband systems with limited fading, the fading spread is assumed to be limited within one symbol duration. Unless otherwise stated, the signal length is given by $N_r = 1024$. Note that in Table 6.2, the assumption of a sampling duration of 1 second is only made to normalize parameters and simplify the notations as stated in the system model. For a 20MHz sampling rate (50ns duration), for instance, all durations in the table should be multiplied by 50ns and the frequency offset by 20MHz.

Typically, a fixed dataset is used in training, and data augmentation is performed to avoid overfitting. Since our dataset is generated using simulations, instead of fixing the size of the training data and using augmentations, we generated the signals in real-time during training. Real-time signal generation, while eliminating overfitting, requires optimizing the generator execution time to avoid slowing down the training significantly. For training DPN, in each epoch 800K new signals are generated, the total training data for 100 epochs is 80M signals. However, note that DPN can work with a smaller fixed dataset with appropriate regularization and data augmentation similar to any other network. Also, all the networks used for comparison in this work used real-time sample generation. The validation and test sets contain a fixed 100K samples. Both sets use the same parameters given in Table 6.2, except that the test set SNR was discretized from 0 to 20dB at 5dB steps for a more convenient result visualization.

To validate DPN's performance beyond simulations, we collected an over-the-air capture using software-defined-radios (SDRs). The capture was performed indoor using two USRP B205 mini SDRs [Ett] in a line-of-sight channel as shown in Fig. 6.5 using a center frequency of 915MHz a sampling rate of 1M samples per second ($\tau_0 = 1\mu s$). The receiver center frequency was intentionally offset by 5KHz (0.005Hz at $\tau_0 = 1$) to emulate the lack of prior agreement, in addition to the offsets due to the USRP oscillator accuracy having a rang e of ±2ppm [Ett] (±1.83KHz at 915MHz). The Tx USRP sent 100K signals following the same modulation types and normalized symbol durations (samples per symbol) in Table 6.2 over a period of 3 minutes. The Rx USRP captured the signals, which were isolated and matched to the corresponding transmitted signals in post processing. This process was repeated three times with different signal amplitudes to obtain 3 captures with estimated SNRs equal to 8,14, and 20dB. The SNR was estimated by measuring the received signal power and dividing it with measured power with no active transmissions. Hence, the SNR does not account for quantization noise, which is more significant for smaller signal amplitudes. The received signals, transmitted symbols, and the modulation types form the over-the-air test dataset at a given SNR.

6.7 Results

6.7.1 Signal Recovery

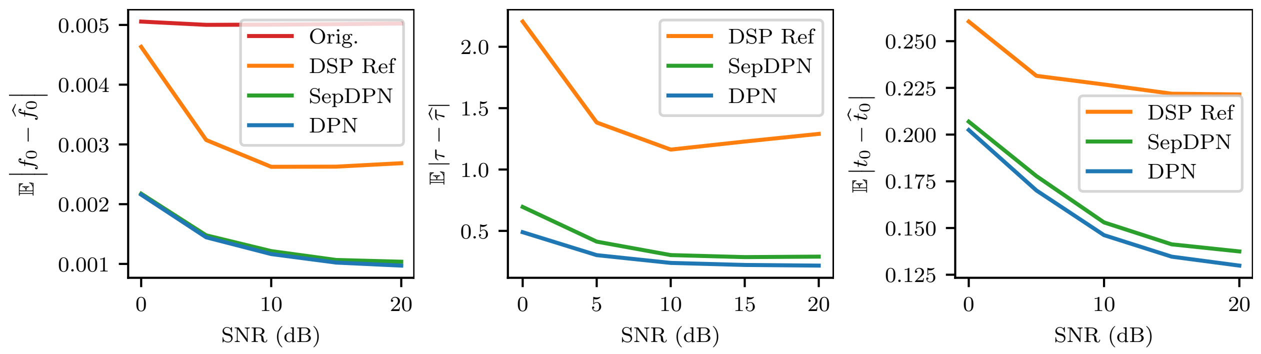
We start by evaluating the performance of DPN, SepDPN, and DSP Ref in recovering the transmitted signal. While signal recovery is not our main objective in this chapter, it helps verify the merits of DPN and explain the blind decoding and modulation classification results. We start by evaluating the mean absolute error for the frequency offset $\widehat{f_0}$, timing offset $\widehat{t_0}$, and the symbol duration $\widehat{\tau}$ as a function of the SNR. The mean absolute error for $\widehat{f_0}$ is calculated using $\mathbb{E}|f_0 - \widehat{f_0}|$ on the test dataset where $\mathbb{E}$ denotes the expectation.

The results are shown in Fig. 6.6. In Fig. 6.6a, we show the original frequency offset

Table 6.2: Dataset Description

Desc.	Par.	Range
Modulation Types	M	{OOK, ASK4, ASK8, OOK, ASK4, ASK8, BPSK, QPSK, PSK8, PSK16, PSK32, APSK16, APSK32, APSK64, QAM16, QAM32, QAM64, GMSK, CPFSK}
Sampling Period	τ_0	1
Frequency Offset	f_0	$[-0.01, 0.01]$
SNR (dB)	SNR	$[0, 20]$
Timing Offset	t_0	$[0, 1]$
Symbol Duration	τ	$[4, 16]$
Pulseshape rolloff	β	$\{0.15, 0.25, 0.35\}$
Phase Offset	ϕ_0	$[0, 2\pi]$
Delay Spread	σ	$[0,\tau]$

(Orig.), which is the mean absolute value of a uniform variable from [-0.01,0.01] as stated in Table 6.2. We can see that for all methods the estimation error decreases as the SNR gets larger and that the residual frequency offset is lower than the original offset. Comparing the two variations of DPN, both give almost identical frequency estimation errors. This is expected since both approaches for the first frequency offset stage use only the features from the input signal. The curve for unsupDPN was omitted as it provided values above 55, which are obviously erroneous. Since unsupDPN is trained to reduce the modulation classification loss, the predicted frequency is an arbitrary value that reduces this loss with



(a) Carrier Frequency Offset (b) Symbol duration (c) Timing Offset

Figure 6.6: The mean absolute error of DPN in parameter estimation.

no signal processing meaning.

The DSP Ref algorithm using the parameters in the Appendices gives a higher estimation error than DPN. It is important, however, to note that DPN has the advantage of having prior information about the signals from the training data, unlike DSP Ref. The purpose of comparing to the blind DSP algorithm is just to provide a baseline reference for comparison. Each approach is considered in the way it is typically implemented and we do not claim to provide an absolute comparison between DSP algorithms in general and deep learning. This trend of results between DSP Ref and the DPN variations continues for the symbol duration and timing offset estimation. It is noteworthy that DSP Ref uses the symbol duration in the calculation of the timing offset. A mistake in the former will lead to a mistake in the latter, which would lead to a high symbol error rate in the same signal.

Then, we consider the results for the later stages of DPN; symbol duration in Fig. 6.6b and timing offset in Fig. 6.6c. We see that the proposed DPN with feature sharing performs better than the separately trained DPN, in contrast to the first stage's where the results were close. This is attributed to the fact the symbol duration and timing estimation networks of the proposed DPN, due to being at the end of the network, benefit from all the extracted features. SepDPN, on the other hand, only uses the features from the last stage. This shows

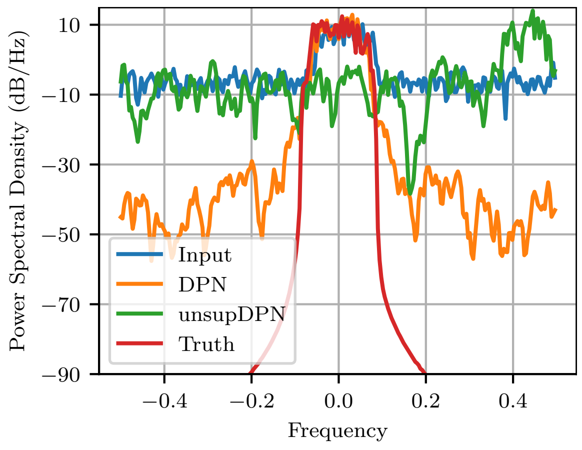


Figure 6.7: The frequency representation of an example input signal, the signal recovered by DPN, and the ground truth.

that feature sharing can lead to an improvement in estimation.

The fact that the predicted estimations are close to the true values makes them reusable by signal processing algorithms. For instance, the frequency offset obtained can be used as an initial value for any carrier frequency offset tracking algorithm [Har11]. The symbol duration and offset can be used to initialize a symbol timing recovery algorithm [Har11]. This can be useful in scenarios where a long signal is received at a high sampling rate and processing the entire signal with neural networks is not computationally feasible.

Lastly, we show the recovery results on the example signal visualized in Fig. 6.1. For the BPSK signal having $\tau = 8$ and an SNR = 5dB and fading spread $\sigma = 0.3$, in Fig. 6.7, we show the power spectral density (PSD) of the input $\mathbf{y}$, the recovered signal $\widehat{\mathbf{z}}$ from DPN, unsupDPN, and the true clean signal $\mathbf{z}$. These signals in time domain are visually hard to compare because of phase offsets, that is why we used the PSD. From that Figure, we can see the filter taps predicted by DPN attenuated the noise by over 25dB and made the output more similar to the truth. This shows that using only training data, DPN can predict filters that partially recover the transmitted signals. In contrast, the recovered signal from unsupDPN is arbitrary and hence has no value in blind decoding. This shows the necessity

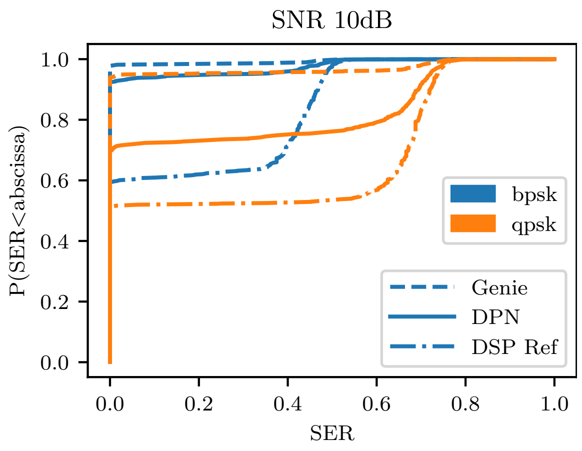


Figure 6.8: The CDF of the symbol error rate (SER) calculated for each signal at SNR=10dB of different approaches.

of including the ground truth for blind decoding. Instead of evaluating the signals $\widehat{\mathbf{z}}_1$ and $\widehat{\mathbf{z}}_2$ over the entire dataset, which is not straight forward due to residual frequency errors, we move on to the symbol decoding evaluation. Symbol decoding implicitly evaluates the recovery stages and is our objective in this chapter.

6.7.2 Blind Symbol Decoding

Then, we evaluate DPN performance in blind symbol decoding. Blindly decoding symbols is challenging since the signal parameters are not known by the blind receiver and no known preambles are assumed. Even at a high SNR, synchronization errors can lead to a high symbol error rate. Additionally, the symbol decoding procedure used in this work is relatively simple and not optimized for any specific modulation type. All these factors make obtaining a low symbol error rate (SER) as expected in a typical communication system impractical. A completely blind receiver also would not expect to obtain the same performance as a receiver with protocol knowledge.

For these reasons, we limit our results in this section to BPSK and QPSK modulations. We calculate the symbol error rate (SER) for each signal and plot the CDF of SER over

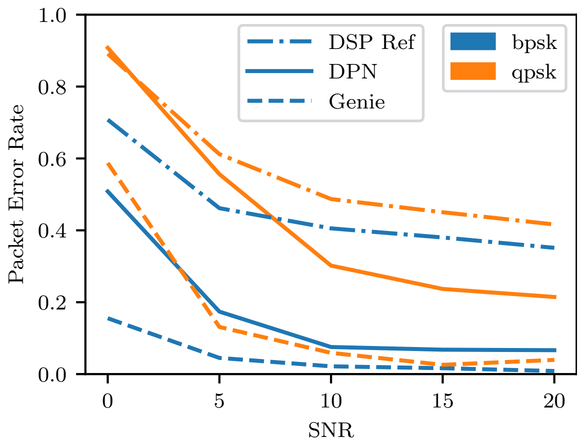


Figure 6.9: The packet error rate of different approaches plotted against SNR.

all the test signals when using DPN, Genie, and DSP Ref for signal recovery in Fig. 6.8 at SNR = 10dB. DPN obtains lower SER for a larger fraction of the signals compared to Ref DSP. This follows from the fact that DPN had better results in signal recovery. As expected Genie performs better than DPN since it has access all the signal and channel parameters. Since BPSK is a lower order modulation, it can tolerate a larger synchronization errors and the gap between Genie and DPN is smaller for BPSK than QPSK. Notice that the CDF has a jump near SER 0.5 for BPSK and SER 0.75 for QPSK. These values are equivalent to a random guess (0.75 and not 0.5 for QPSK since we are using SER and not bit error rate). This jump is due to either a significant mistake in parameter estimation or an error in one symbol propagating to the remaining symbols in a given signal. From these results, we see that each signal is either decoded almost entirely correctly or not.

Based on this observation, we consider each signal to be a packet and consider the packet error rate (PER), equivalent to P(SER=0), as a function of SNR for the same modulation types in Fig. 6.9. The results follow the same trend with Genie having the lowest SER followed by DPN and then DSP Ref. Since this curve is for packet error and there are between 64 and 256 symbols per packet, a random guess would yield a PER close to one. Despite that the problem of blind symbol decoding is challenging, DPN performance, while not great in absolute terms, is better than the blind DSP Ref and the PER gap between it

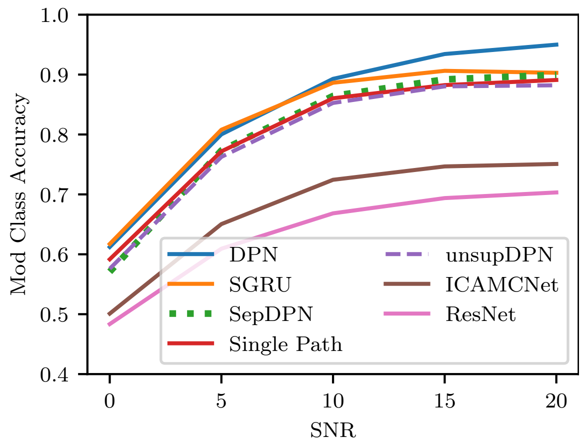


Figure 6.10: The modulation classification of DPN, its alternatives SepDPN, and Single Path, and networks from the literature.

and Genie gets as low as 5% for BPSK.

6.7.3 Modulation Classification

In this section, we consider the modulation classification performance of DPN and its variations (SepDPN, UnsupDPN, and Single Path). We also consider networks from the literature, namely the Stacked GRU (SGRU) [HZX17] and the ResNet [ORC18], and ICAMCNet [HGK20]. SGRU consists of recurrent networks and ResNet and ICAMCNet are fully convolutional networks. All networks considered are trained with real-time sample generation using the same procedures as DPN. Note that SGRU, ResNet, Single Path and unsupDPN use only the modulation type as a label in contrast to DPN and SepDPN that additionally use the frequency offset, noise reduced signal, equalized signals, and timing information. For fairness, the modulation-only networks were allowed up to 4 times the training epochs and up to 4 times the training data as DPN (up to 400 epochs and 320M training signals with early stopping).

The results are shown in Fig 6.10. The two networks giving the highest accuracy are DPN and SGRU. At SNR less than 10dB, both approaches yield almost the same accuracy.

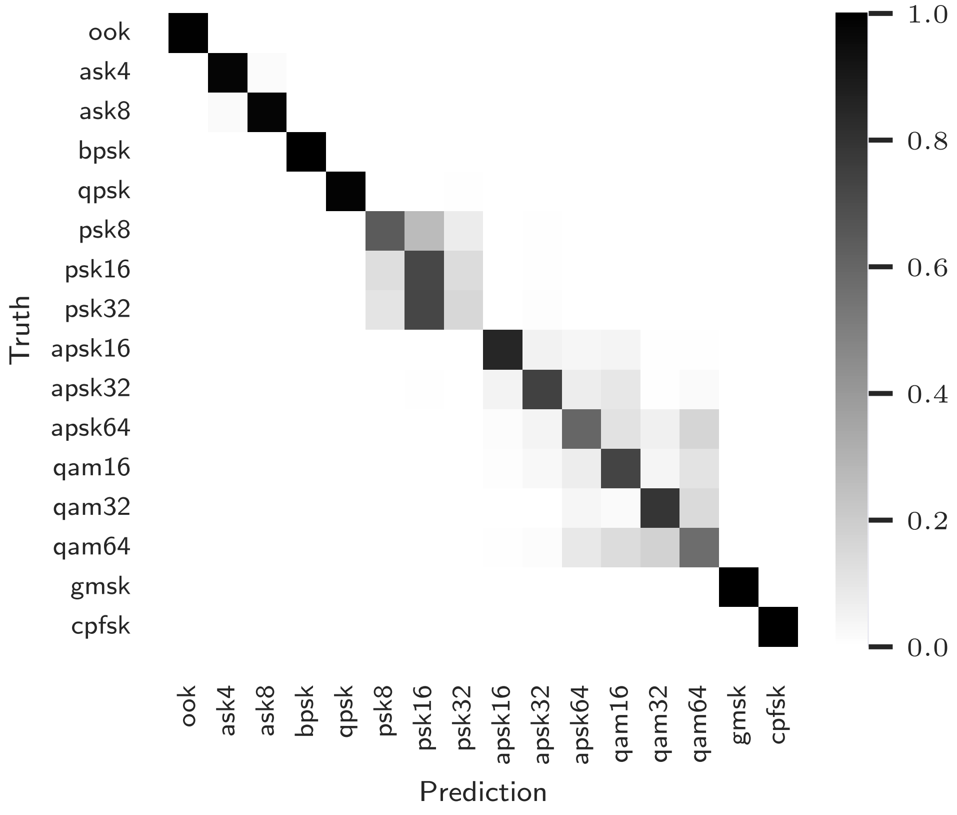


Figure 6.11: The confusion matrix of DPN at SNR of 5dB.

For SNRs larger than 10 dB, SGRU performance starts to saturate and DPN performance improves surpassing SGRU by up to 5%. This is explained by considering the signal recovery performance. At higher SNR, the signal recovery stages perform better, providing signals with less distortions for the modulation classification. Signals with less distortions result in better predictions. At lower SNR, the improvements to the signal along the signal path are more limited and do not lead to significant gains compared to SGRU. Even though DPN and SGRU attain the same accuracy below 10dB, DPN can be used for blind decoding, and is about twice faster in inference compared to SGRU as discussed later. The fully convolutional networks ICAMCNet and ResNet gave a lower accuracy than SGRU and DPN that use recurrent networks for modulation classification for this dataset having variable samples per symbol. In Fig. 6.11, we show the confusion matrix of DPN at 5dB SNR, from which we see that the confusion occurs mostly in high order modulations of the same

Table 6.3: Impact of each stage on Mod. Class. Accuracy

Zeroed Loss	None	L_1	L_2	L_3	L_4	L_5
Mod. Class.	83.7%	80.5%	81.6%	83.4%	83.11%	82.8%

type. Note that many improved deep learning layers and architectures were proposed for modulation classification in the literature. Our focus in this work is on combining DSP with deep learning and not just using improved deep learning modules for higher classification accuracy. Improved deep learning modules can easily be integrated within the dual path architecture. Hence, we focus more on interpreting DPN's design performance by comparing it with its alternative architectures.

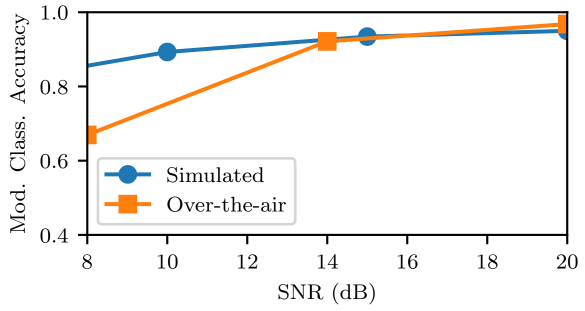
To better understand the factors leading to DPN's results, we consider its variations. First, we see that DPN outperforms Single Path. Since Single Path consists of the exact deep-learning-based feature path of DPN, the improved results of DPN are not solely attributed to the design of the feature path. The DSP signal recovery leads to a significant accuracy improvement. To further understand the impact of each recovery stage individually on the classification, several instances of DPN were trained while setting the weight of one of the losses L_1 to L_5 to zero. The obtained accuracies are shown in Table III. From this table, we see that eliminating the frequency correction by zeroing L_1 has the largest impact on classification, since the dataset uses high order PSK signal which are sensitive to phase shifts due to frequency errors. On the other hand, the timing offset loss L_4 has the smallest impact because of its limited impact on the signal when using more than 4 samples per symbol. Comparing DPN to SepDPN, which lacks features sharing, DPN is better at identifying the signal modulation type. Since the only advantage DPN has over SepDPN is the connection of the feature path, feature sharing that occurs along this path also contributes to the improved performance. Hence, by leveraging signal reconstruction and feature sharing, DPN design improves modulation classification by up to 5%. For unsupDPN, we see that the

Table 6.4: Training and Inference Times

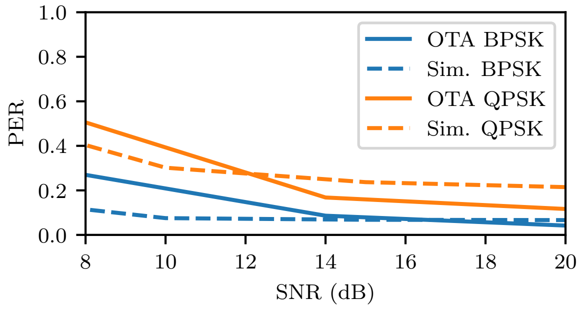
Net.	Training (s/epoch)	Inference (μs/signal)
DPN	578.82	736
SGRU	339.22	1530
ResNet	214.52	168
ICAMCNet	227.60	135

accuracy is also about 5% less than DPN. This result shows that including the reconstruction ground truth has the added benefit of improving the modulation classification. The similarity between the results of unsupDPN and Single Path indicates that unsupDPN was not able to leverage the reconstruction modules to improve the accuracy.

Lastly, the training and inference times of the neural networks are shown in Table IV. The training time is calculated as the average of training time per epoch. The inference time is calculated by averaging the time to make predictions per signal over the test set. The server used for calculations has an Intel Core i9-9920X (12 Cores, 3.50 GHz), an RTX 2080Ti GPU, and 128GB of RAM. From Table IV, DPN is slower to train than the other networks, since it has 6 outputs and thus requires more slow memory operations to transfer the data to the GPU for training. Note that since we are using real-time signal generation, the training time includes the overhead signal generation. In terms of inference time, DPN is slower than the fully convolutional networks (ResNet, ICAMCNet), however it is about twice faster than the SGRU. Recurrent neural networks (like SGRU and the modulation classification NN of DPN) have data dependencies among their different units, which makes them slower compared to convolutional networks. DPN consists mostly of convolutional layers except for one GRU layer much smaller that those in SGRU, and hence is faster.



(a) Modulation Classification

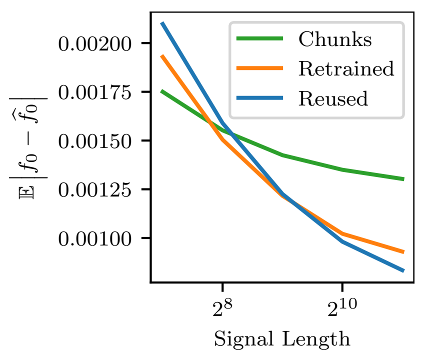


(b) PER

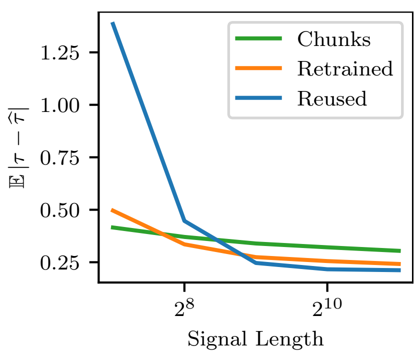
Figure 6.12: Comparison between simulation and over-the-air results.

6.7.4 Over-the-Air USRP Evaluation

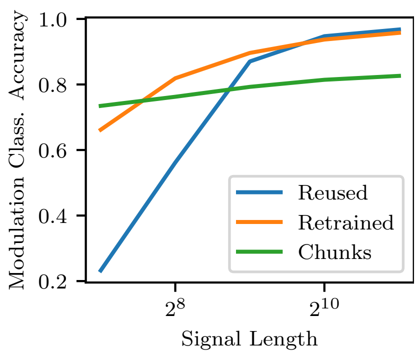
To validate our signal model, we evaluate DPN, trained with the simulated data, using the over-the-air test dataset without any retraining. The obtained results are shown for modulation classification in Fig. 6.12a and packet error rate in Fig. 6.12b. From these Figures, we see that the OTA results are close to the simulated datasets for the 14dB and 20dB captures. This result verifies our signal model and DPN's performance at this range of SNR. At 8dB SNR, DPN results degrade with lower modulation accuracy and higher PER. This degradation is attributed to quantization errors, which become more significant in weaker signals and has not been accounted for in the SNR estimation, along with other hardware imperfections (amplifier non-linearity, IQ imbalance, etc). These imperfections, which are typically not modeled, are known to impact RF deep learning classifiers [CHH20]. While retraining using the OTA capture can improve performance on this specific radio capture [ORC18], it does not guarantee that the results will generalize to different radio hardware or deployment environment and thus we do not consider it. A more in depth study of the performance trade-offs of using simulated and over-the-air data for training was performed in [CHH20].



(a) Frequency Offset



(b) Symbol duration



(c) Modulation Classification

Figure 6.13: Comparison of different approaches to scale DPN; dividing the signal into chunks, reusing the weights trained for $N_r = 1024$, retraining using different input lengths.

6.7.5 Variable Length Evaluation

As discussed earlier, DPN architecture supports using variable input lengths using the same weights. Our previous results were only for signals of length $N_r = 1024$. In this section, we evaluate the ability of DPN to generalize to lengths different than the training signals. We consider the lengths $N_r \in \{128, 256, 512, 1024, 2048\}$ for testing with 10000 signals generated at each length using a 20dB SNR. Three approaches are considered; in the first approach, we reuse the same weights θ_1 trained on signals having $N_r = 1024$ without any modifications. In the second approach, we retrain the network starting from θ_1 on signals of different lengths to obtain the weights θ_2. The retraining was done for 20 epochs such that the signals in each training batch had a length chosen randomly within the set $\{192, 384, 768, 1536\}$, which does not overlap with the test lengths. The third approach, which works for fixed input networks, consists of training a network on the smallest input length, dividing longer signals into chunks, and averaging the predictions of each chunk as proposed in [ZQC19]. For that approach, we train DPN from scratch using signals of length 128 and predict exclusively using that length. For signals longer than 128, predictions are applied on chunks of 128 and the results are averaged.

The results for prediction errors and modulation classification are shown in Fig. 6.13. For all approaches, as expected the larger the signal length, the lower the estimation error and the higher the classification accuracy. Looking at the modulation classification results in Fig. 6.13c, reusing the weights seems to generalize well on lengths larger than 512 without any modification. For shorter lengths, however, the performance drops significantly below 60%. Retraining the weights on shorter and longer signal lengths, although different from the testing ones, does improve the performance on shorter signals as seen by looking at the retrained curve in the same Figure. This retraining, however, comes at the cost of slightly degraded performance on longer signals. The relative trend of results between retrained and reused is the same for frequency offset and symbol duration estimation. This indicates that DPN learns signal features that generalize to unseen lengths. By retraining, the weights are adjusted to improve performance on shorter signals at the expense of longer signals.

Now, looking at the curve for dividing the signal into chunks, we see that it only outperforms both the other approaches when $N_r = 128$, which is the length it was trained on. Other than that length, it provides higher estimation errors and a lower modulation classification accuracy by up to 15%. This shows that this approach is not the best way to use DPN predictions on long signals. Since the outputs $\widehat{f_0}$ and $\widehat{\tau}$ are calculated using average pooling, the degradation is not due to the averaging operation itself but to the weights learned in training. When using only short signals in training, the learned weights extract features leveraging only short signal durations neglecting features spanning longer periods. These results highlight the benefit of DPN's design that can work using different input lengths.

6.8 Summary

In this chapter, we proposed the dual path network (DPN) for blind decoding and modulation classification. DPN's design consists of three stages of signal recovery connected to form a signal path made of DSP operations and a feature path consisting of neural networks. This design enables features to be shared, enabling improved estimates in the later stages and a 5% improvement in modulation classification compared to a network separately trained. Extracting features from the recovered signals provides a 5% improvement in modulation classification compared to a similar network not recovering the signal. The estimation results of DPN are shown to outperform a reference blind DSP algorithm based on cyclostationary features. These improved estimates make it yield lower packet error rates compared to the reference algorithm. Due to the signal processing inspired design, the output filter taps and estimates are compatible with existing signal processing approaches. Using an over-the-air capture, we validated DPN results at SNRs above 14dB. By relying on average pooling and recurrent neural networks, DPN can process variable length inputs, which are shown to provide better estimates than dividing the input into short chunks and averaging.

CHAPTER 7

Conclusion

7.1 Summary of Contributions

In this dissertation, we developed approaches to improve the spectrum and energy efficiencies for UAV swarms. We also addressed some security considerations facing wireless devices.

In Chapter 2, we optimized the placements of a UAV swarm to maximize the MIMO backhaul capacity starting from given swarm initial positions. We mathematically defined a set of UAV placements that orthogonalize the channel and maximize the MIMO capacity. The problem of minimizing the distance traveled to reach a placement in this set was formulated. An offline centralized solution was developed by relaxing the problem and decomposing it into two convex problems which were solved iteratively using block coordinate descent. We also proposed FF as a distributed iterative solution to this problem. FF requires sharing channel estimates only between neighbors and we derived the conditions for its convergence. Using numerical simulation, we have shown its robustness under channel and UAV induced disturbances. Upper bounds for the distance that UAVs need to travel using the centralized solution and force field were derived and numerically verified. Our approaches were shown to provide significant sum rate improvements while requiring only bounded displacements. The gains from our approach were shown to remain significant as we transition to the massive MIMO regime with far more ground station antennas than UAVs.

In Chapter 3, we developed and verified a mathematical framework to model the BF performance for a destination-led BF protocol. The BF gains distribution was approximated by a gamma distribution assuming a zero mean normally distributed combining phase errors and the proposed distribution was verified using simulations. The effect of the pre-BF SNR and the preamble lengths on the combining phase error was derived for our choice of estimators. Using software-defined radios, in a lab, we experimentally verified the predictions of our BF framework. The BF radios were mounted on UAVs and were shown to exceed 80% of the ideal BF gains despite the low coherence time channel. The proposed framework can be used to design BF systems for a given deployment as illustrated by two example scenarios.

In Chapter 4, we proposed guided distributed beamforming as an approach for mobile radios sharing a LOS channel to make their signals coherently combine at a remote destination, without feedback from the destination and without having a strict requirement on location information. We have derived and verified using simulations a geometric criterion for the placement of the radios, to bound the phase errors leading to degraded beamforming gains at the destination. The proposed approach was implemented using software defined radios. The results show a 9 dB SNR improvement in the beamforming direction when using 4 radios, with a weaker gain in other directions as predicted using simulations. Thus, our approach enables an extended range of communications towards a distant destination not providing feedback.

In Chapter 5, we have considered the problem of transmitter authorization using RF fingerprints captured from raw IQ samples. Since this problem has been scarcely investigated in the wireless domain, we performed a comprehensive evaluation of the most prominent machine learning approaches from the openset recognition and the anomaly detection literature, as applied to our problem definition. The dependence of the evaluation results on the choice of transmitters was demonstrated and a simple strategy was proposed to reduce it. We have also shown that the performance of a given neural network model on closed set classification is not an indicator of its performance in outlier detection, indicating the need for architectures specifically designed for this problem. Also, we demonstrated that minor change in network architecture and data labeling can lead to a significantly different approaches. Using a known outlier set was proposed and was shown to improve the outlier detection accuracy. While classification based OvA gives the highest accuracy in most cases, it is outperformed by reconstruction based AutoEnc for small number of authorized.

In Chapter 6, we proposed the dual path network (DPN) for blind decoding and modulation classification. DPN's design consists of three stages of signal recovery connected to form a signal path made of DSP operations and a feature path consisting of neural networks. This design enables features to be shared, enabling improved estimates in the later stages and a 5% improvement in modulation classification compared to a network separately trained. Extracting features from the recovered signals provides a 5% improvement in modulation classification compared to a similar network not recovering the signal. The estimation results of DPN are shown to outperform a reference blind DSP algorithm based on cyclostationary features. These improved estimates make it yield lower packet error rates compared to the reference algorithm. Due to the signal processing inspired design, the output filter taps and estimates are compatible with existing signal processing approaches. Using an over-the-air capture, we validated DPN results at SNRs above 14dB. By relying on average pooling and recurrent neural networks, DPN can process variable length inputs, which are shown to provide better estimates than dividing the input into short chunks and averaging.

7.2 Future Work

In Chapter 2, we only considered the problem of minimizing the distance to attain the capacity maximizing positions. While distance is a metric suitable for many applications, the approaches can be generalized to minimize any arbitrary objective functions specified by the deployment task. These positions can be optimized to minimize these arbitrary objective functions using learning based approaches like reinforcement learning.

In Chapter 3, even though we only considered a specific BF protocol and only two example scenarios, the proposed framework can support many protocol variations and use cases. For the protocol, the framework is applicable for any other choice of estimators as long as their phase variance can be expressed mathematically. As for the scenarios, heuristics can easily be developed to optimize over a combination of the SNR, preamble lengths, and the number of BF slaves, enabling the framework to adapt to many different deployment scenarios.

For the guided BF proposed in Chapter 4, we assumed that the feedback between the guide and the followers is ideal. However, in practice, that is not the case. In addition to phase errors due to the guide-follower geometry, there are errors due to the feedback. As guided BF gets deployed on UAVs, the channel coherence time and the guide-follower signaling will have a larger impact on BF gains. Further analysis is needed to quantify the impact of these errors.

In Chapter 5, we only focused on the problem of rejecting unseen transmitters, while classifying known transmitters. There are other interesting RF identification problems which require identifying the unauthorized transmitter. One possible approach to tackle this problem is using the output from the feature extractors to build a database of fingeprints. By comparing new signals with entries in the database, we can classify among unauthorized transmitters and quickly add new transmitters to the authorized list without retraining.

While Chapter 6 has only considered blind symbol decoding for single carrier signal, the dual path concept can be extended to other problems in wireless communications leveraging deep learning like non-blind symbol decoding or interference cancellation. It can also be extended to multicarrier signals.

APPENDIX A

Appendix for Chapter 1

A.1 Proof of Lemma 1

Let us define the scaled x and z translations, $x''_n = \frac{x_n}{y_n}$ and $z''_n = \frac{z_n}{y_n}$. We start by assuming that the solution is found on a uniform grid with respect of the scaled variables with dimensions M_x and M_z. The separation of the UAVs along this grid in the x and z planes is given by e_x and e_z, such that we can rewrite $x''_n = i_c e_x$ and $z''_n = j_c e_x$ for some integers $i_c \in \{0, \cdots, M_x - 1\}, j_c \in \{0, \cdots, M_z - 1\}$. Our objective, hence, becomes calculating the value of e_x and e_z. Starting from (2.11), we get

$$\mathbf{h}_l^H \mathbf{h}_k = \sum_{n=0}^{N-1} \exp\left(\frac{-j2\pi}{\lambda}\left((-i_l + i_k)d_x x''_n + (-j_l + j_k)d_z z''_n\right)\right) \tag{A.1}$$

$$= \sum_{i_c=0}^{M_x-1}\sum_{j_c=0}^{M_z-1} \exp\left(\frac{-j2\pi}{\lambda}\left((-i_l + i_k)i_c d_x e_x \ \ +(-j_l + j_k)j_c d_z e_z\right)\right) \tag{A.2}$$

$$= \sum_{i_c=0}^{M_x-1} \exp\left(\frac{-j2\pi}{\lambda}\left((i_k - i_l)i_c d_x e_x\right)\right) \cdot \sum_{j_c=0}^{M_z-1} \exp\left(\frac{-j2\pi}{\lambda}\left((j_k - j_l)j_c d_z e_z\right)\right) \tag{A.3}$$

In (A.2), the summation over UAVs was rewritten as a summation over the x and z UAV grid positions. As is evident from equation (A.3), this summation is a product of two geometric sums and can therefore be simplified to

$$\frac{\sin\left(\frac{\pi M_x(i_k-i_l)d_x e_x}{\lambda}\right)}{\sin\left(\frac{\pi(i_k-i_l)d_x e_x}{\lambda}\right)}\frac{\sin\left(\frac{\pi M_x(j_k-j_l)d_z e_z}{\lambda}\right)}{\sin\left(\frac{\pi(j_k-j_l)d_z e_z}{\lambda}\right)} = 0 \tag{A.4}$$

where the summation is set to 0 because of the orthogonality condition defined in (2.8). The orthogonality is achieved when $e_x = \frac{\lambda}{M_x d_x}$ and $e_z = \frac{\lambda}{M_z d_z}$. If we set y_n to be constant for all UAVs, we get the same condition derived for the optimal design of a parallel planar uniform rectangular arrays (URA) derived in [BOO07b, Lar05]. Hence, orthogonality is achieved when $x_n = i_n \frac{\lambda y_n}{M_x d_x}$ and $z_n = j_n \frac{\lambda y_n}{M_z d_z}$, where $n = i_n M_z + j_n$.

A.2 Proof of Proposition 1

The objective of (2.17) is monotonically increasing with respect to $(x'_{m,n})^2$. This objective is minimized by minimizing $(x'_{m,n})^2$. To prove that the optimal $f_{m,n}$ is within the set $\{-1, 0\}$, we show that any value outside this set will correspond to a larger value of $(x'_{m,n})^2$.

Given that $0 \le \tilde{x}_{m,n} \le S_x \epsilon_n$ and that $-\frac{1}{2}S_x\epsilon_n \le \delta_x S_x \epsilon_n \le \frac{1}{2}S_x\epsilon_n$ from (2.23), we get

$$-\frac{1}{2}S_x\epsilon_n \le \tilde{x}_{m,n} + \delta_x S_x \epsilon_n \le \frac{3}{2}S_x\epsilon_n \tag{A.5}$$

So, for any value of δ_x and $\tilde{x}_{m,n}$, using (2.25) the optimal value $\hat{f}_{m,n}$ can be calculated using

$$\begin{aligned}\hat{f}_{m,n} &= \underset{f_{m,n}\in\mathbb{Z}}{\operatorname{argmin}}(\tilde{x}_{m,n} + f_{m,n}S_x\epsilon_n + \delta_x S_x\epsilon_n)^2 \\ &= \begin{cases} 0 & -\frac{1}{2}S_x\epsilon_n \le \tilde{x}_{m,n} + \delta_x S_x\epsilon_n < \frac{1}{2}S_x\epsilon_n \\ -1 & \frac{1}{2}S_x\epsilon_n \le \tilde{x}_{m,n} + \delta_x S_x\epsilon_n \le \frac{3}{2}S_x\epsilon_n \end{cases}\end{aligned} \tag{A.6}$$

By substituting $\hat{f}_{m,n}$ to calculate the absolute translation $(\hat{x}'_{m,n})^2 = \min_{f_{m,n}\in\mathbb{Z}}(x'_{m,n})^2$, we find that it is bounded by $(\hat{x}'_{m,n}) \le (\frac{S_x\epsilon_n}{2})^2$. If $f_{m,n}$ is outside the set $\{-1, 0\}$, we get a larger translation such that $(x'_{m,n})^2 \ge \left(\frac{1}{2}S_x\epsilon_n\right)^2$. Hence, the optimal value of $f_{m,n}$ has to be within $\{-1, 0\}$ and is given by (2.27).

A.3 Proof of Proposition 2

In the proof of Proposition 1, we showed that $(\hat{x}'_{m,n})^2 \leq (\frac{S_x \epsilon_n}{2})^2$ and similarly $(\hat{z}'_{m,n})^2 \leq (\frac{S_z \epsilon_n}{2})^2$. This holds for any value of the remaining variables. Hence, the translation made by UAV n is upper bounded by $\frac{\sqrt{S_x^2+S_z^2}}{2}\epsilon_n$.

A.4 Proof of Lemma 4

If K_p is sufficiently small, the phase unwrapping given by (2.38) retains the linearity of the measurements. Assuming UAV $n-1$ is fixed, x_{n-1} is constant across iterations, and we get

$$\begin{aligned} e_n[k] &= 2\pi \left(\frac{x_n[k]}{\epsilon_n} - \frac{x_{n-1}}{\epsilon_{n-1}} \right) \frac{1}{S_x} - \psi_n \\ &= 2\pi \left(\frac{x_n[k-1] - K_p e_n[k-1]}{\epsilon_n} - \frac{x_{n-1}}{\epsilon_{n-1}} \right) \frac{1}{S_x} - \psi_n \\ &= -2\pi \frac{K_p e_n[k-1]}{\epsilon_n S_x} + 2\pi \left(\frac{x_n[k-1]}{\epsilon_n} - \frac{x_{n-1}}{\epsilon_{n-1}} \right) \frac{1}{S_x} - \psi_n \\ &= \left(1 - \frac{K_p 2\pi}{\epsilon_n S_x} \right) e_n[k-1] \end{aligned} \tag{A.7}$$

If $0 < K_p < \frac{\epsilon_n S_x}{2\pi}$, the error will decrease in each iteration, and hence it will converge to zero. However, to avoid phase wrap errors when using (2.38) we need to guarantee that any transition does not exceed π, which is realized when $0 < K_p < \frac{\epsilon_n S_x}{4\pi}$

APPENDIX B

Appendix for Chapter 2

B.1 Proof of Proposition 4

The variance error of the KF output is given by $p_{k|k}$ and we want to calculate its value. Substituting (3.17) into (3.21), we get $p_{k+1|k} = \frac{rp_{k|k-1}}{p_{k|k-1}+r} + q$. At steady state $p_{k+1|k} = p$ for all k and we get a simple form of the algebraic Riccati equation $p = \frac{rp}{p+r} + q$ [Ber00]. Solving the equation, we get $p_{k+1|k} = p = \frac{q+q\sqrt{1+4\frac{r}{q}}}{2}$. Using (3.21), we get $\sigma_{fk}^2 = p_{k|k} = \frac{-q+q\sqrt{1+4\frac{r}{q}}}{2}$.

B.2 Proof of Proposition 5

$$G = \frac{1}{N}\left|\sum_{n=1}^{N} e^{j\phi_n^e}\right|^2 = \frac{1}{N}\sum_{n=1}^{N} e^{j\phi_n^e}\sum_{m=1}^{N} e^{-j\phi_m^e} = 1 + \frac{2}{N}\sum_{m=1}^{N}\sum_{n=m+1}^{N}\cos(\phi_n^e - \phi_m^e) \tag{B.1}$$

Using the fact that for a zero mean Gaussian RV x, $\mathbb{E}\{\cos x\} = e^{-\mathrm{var}\{x\}/2}$ [RBM12b], we get

$$\mathbb{E}\{G\} = 1 + (N-1)e^{-\sigma_e^2} \tag{B.2}$$

$$\mathrm{var}\{G\} = \frac{4}{N^2}\mathrm{var}\{\sum_{m=1}^{N}\sum_{i=m+1}^{N}\cos(\phi_i^e - \phi_m^e)\} \tag{B.3}$$

$$= \frac{4}{N^2}\sum_{m=1}^{N}\sum_{i=m+1}^{N}\mathrm{var}\{cos(\phi_i^e - \phi_m^e)\} + \frac{8}{N^2}\sum_{m=1}^{N}\sum_{i=m+1}^{N}\sum_{p=i+1}^{N}\mathrm{cov}\{\cos(\phi_i^e - \phi_m^e), \cos(\phi_m^e - \phi_p^e)\} \tag{B.4}$$

$$= \frac{(N-1)}{N}(e^{-\sigma_e^2} - 1)^2\left((e^{-\sigma_e^2} - 1)^2 + 2Ne^{-\sigma_e^2})\right) \tag{B.5}$$

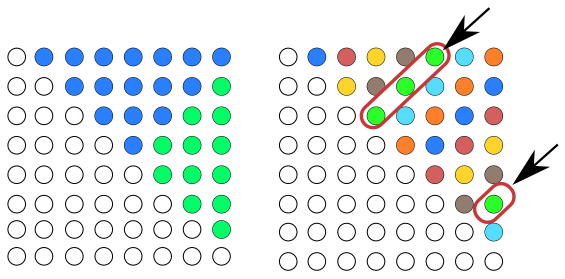


Figure B.1: Summation order for the matrix $\mathbf{X}$.

where $\text{cov}\{x, y\}$ denotes the covariance of RVs x, y. Line (B.4) was obtained using the fact that $\text{var}\{\sum_{m=1}^{M} x_m\} = \sum_{m=1}^{M} \text{var}\{x_m\} + 2\sum_{m=1}^{M}\sum_{n=m+1}^{N} \text{cov}\{x_m, x_n\}$ for any correlated M RVs x_m and by simplifying the summations. Line (B.5) uses the fact that for a zero mean Gaussian RV $\text{var}\{\cos x\} = \frac{1}{2}(e^{-\text{var}\{x\}-1})^2$ [RBM12b] and using that $\text{cov}\{\cos(\phi_i^e - \phi_m^e), \cos(\phi_m^e - \phi_p^e)\} = 0.5e^{-3\sigma_e^2} + 0.5e^{-\sigma_e^2} - e^{-2\sigma_e^2}$ as can be shown using the Gaussian RV relations from [RBM12b], the definition of covariance, and some trigonometric identities.

B.3 Proof of Proposition 6

We start be considering the simplified definition of G from (B.1). We rewrite the elements of the summation as the $N \times N$ matrix $\mathbf{X}$, such that its element $\mathbf{X}_{m,n} = \frac{2}{N}\cos(\phi_m - \phi_n)$. This yields $G = 1 + \sum_{m=1}^{N}\sum_{n=m+1}^{N} \mathbf{X}_{m,n}$. The summation is over the upper diagonal elements of the matrix. Our objective is to rewrite the inner sum as independent RVs of length proportional to N to invoke the central limit theory (CLT). To achieve that, we must avoid reusing the same value of ϕ_n^e in the inner sum, that is, the inner sum elements should have unique column and row indices.

$$\sum_{m=1}^{N}\sum_{n=m+1}^{N}\mathbf{X}_{m,n}=\sum_{n=2}^{N}\sum_{m=1}^{n-1}\mathbf{X}_{m,n} \tag{B.6}$$

$$=\sum_{n=2}^{N}\sum_{m=1}^{\min(n-1,N-n+1)}\mathbf{X}_{m,n}+\sum_{n=N/2+1}^{N}\sum_{m=N-n+1}^{n-1}\mathbf{X}_{m,n} \tag{B.7}$$

$$=\sum_{n=2}^{N}\left(\sum_{m=1}^{\lfloor n/2\rfloor}\mathbf{X}_{m,n-m+1}+\sum_{m=1}^{\lfloor (N+1-n)/2\rfloor}\mathbf{X}_{N+1-(n-m+1),(N+1)-m}\right) \tag{B.8}$$

The summation in (B.6) rewrites the equation from column wise to row wise. In Line (B.7), we split the elements of the summation at the upward diagonal as illustrated in the first image of Fig. B.1 for an 8×8 matrix. In Line (B.8),the inner summation is rewritten as two summations over the upward diagonal elements as shown in different colors in the second image of Fig. B.1. From (B.8), each element of the inner summation consists of about $N/2$ terms[1] and none of the terms have common rows or columns, thus consist of independent RVs. We can rewrite the inner sum as the RV b_n as follows

$$b_n=\sum_{m=1}^{\lfloor n/2\rfloor}\mathbf{X}_{m,n-m+1}+\sum_{m=1}^{\lfloor (N+1-n)/2\rfloor}\mathbf{X}_{N+1-(n-m+1),(N+1)-m} \tag{B.9}$$

The variable b_n consists of identical independent RVs. Hence, for large N, the distribution of b_n converges to a Gaussian distribution. Lastly, we can rewrite G as

$$G=1+\sum_{n=2}^{N}b_n \tag{B.10}$$

The variables b_n are correlated Gaussian RVs, hence their sum is Gaussian. This proves that for large N, G is Gaussian and its mean and variance are given by Proposition 5.

B.4 Proof of Proposition 7

We start this proof by considering the case of small σ_e^2 and then discuss the case of large N. Since ϕ_n are zero mean and assuming small σ_e^2, $\phi_m-\phi_n$ is typically small and we can use

[1]For odd N, the number of elements is either $N/2$ or $N/2-1$. This difference is insignificant for large N

the Taylor expansion of cosine around zero to simplify $\mathbf{X}_{m,n}$ (as defined in Appendix B.3) as $\mathbf{X}_{m,n} \approx \frac{2}{N}\left(1 - \frac{(\phi_m-\phi_n)^2}{2}\right)$. Then, we can rewrite (B.9) as $b_n = 2\frac{s_n}{N} - \chi_n$ where $s_n = \lfloor n/2 \rfloor + \lfloor (N+1-n)/2 \rfloor$ is the number of elements of b_n and $\chi_n = \frac{1}{N}\sum_{r=1}^{s_n}(\phi_{m_r} - \phi_{n_r})^2$ with m_r and n_r corresponding to the indexes from (B.9). The summation in χ_n is over independent zero mean Gaussian RVs that are squared, hence χ_n follows the Chi-squared distribution. We can rewrite G as

$$G = 1 + \frac{2}{N}\frac{N(N-1)}{2} - \sum_{n=2}^{N}\chi_n = N - X_\gamma \tag{B.11}$$

where $X_\gamma = \sum_{n=2}^{N}\chi_n$ is the sum of correlated Chi-squared RVs. The distribution of the summation of correlated Chi-squared RVs can be obtained using the Gamma distribution [GR83]. The shape K and scale θ parametrization of the resulting Gamma distribution can be calculated to realize the mean and variance of X_γ [Fer19]. Using the mean and variance of G from Proposition 5, we get the following equations for the mean and variance respectively

$$K\theta = N - 1 - (N-1)e^{-\sigma_e^2} \tag{B.12}$$

$$K\theta^2 = \frac{(N-1)}{N}(1-e^{-\sigma_e^2})^2\left((1-e^{-\sigma_e^2})^2 + 2Ne^{-\sigma_e^2}\right) \tag{B.13}$$

Solving these two equations, we get the values of K and θ in (3.29) and (3.30). This proof is based on the assumption that σ_e^2 is small. For large values of N, K becomes large, and the Gamma distribution converges to a Gaussian distribution with mean $K\theta$ and variance $K\theta^2$ [Das10], which is the true distribution of G as shown in Proposition 6.

APPENDIX C

Appendix for Chapter 4

For each approach, we describe how the ROC curve is calculated. We also state how a specific threshold is chosen to calculate the outlier detection accuracy along with choices of hyperparameters. Since the evaluation is to be done over multiple realizations, manual tuning is not possible and we provide a systematic way to set these values.

C.1 Discriminator (Disc)

In Disc, we only have one threshold to make a decision. Ideally, we want the threshold to be as low as possible without falsely rejecting authorized transmitters. This can be done by adapting the threshold to tightly fit the predictions of authorized signals in the training set. We follow the approach proposed in [SXL17], where the predicted output of the sigmoid for the correctly classified authorized training samples $\bar{z}_0$ (having labels equal to 0) is concatenated with its negative $-\bar{z}_0$ (to make the distribution symmetric around zero) and fit to a Gaussian distribution having mean 0. The standard deviation σ of these samples is calculated and a threshold of 3σ would allow the majority of authorized transmitters to be accepted. To deal with degenerate cases having large standard deviation, the threshold is set to $\gamma = \min(0.5, 3\sigma)$ in practice. As for obtaining the ROC curve to calculate the AUC, the value of γ is scanned from 0 to 1.

C.2 One Vs All (OvA)

OvA has $|\mathcal{A}|$ thresholds given by $\boldsymbol{\gamma}$. While it is possible to use one common threshold, we use multiple thresholds designed according to the same method of Gaussian fitting used in Disc to calculate the accuracy as it yields better results. As for obtaining the ROC curve for the AUC calculation, we consider one single threshold γ scanned from 0 to 1 such that $\boldsymbol{\gamma} = \gamma\mathbf{1}$.

C.3 OpenMax (OpMx)

As for the parameters, the tail size used to calculate the Weibull distribution is $\tau = 10$, $\alpha = \min(\lfloor|\mathcal{A}|/3\rfloor, 5)$, and ϵ was chosen to be the 95% quantile of the maximum activation in the training data. To obtain these values, we started by considering the values proposed in [BB15] which was optimized for their dataset. However the performance of these values varied drastically as we varied the authorized set and from one realization to the other. After running several experiments and analyzing the activation values, we found that these parameters gave the best performance.

C.4 AutoEncoder (AutoEnc)

We chose γ to be the 90% quantile of the mean squared error of the training data. The ROC curves used for calculating the AUC are obtained by scanning the value of γ from 0 to the maximum MSE.

APPENDIX D

Appendix for Chapter 5

D.1 Band Segmentation

It is applied in two stages using Welch power spectral density (PSD), which consists of dividing the signal into overlapping segments, calculating the squared magnitude FFT of each segment, and averaging them. In the first stage, PSD uses an FFT of size N_1. The occupied frequency bins are detected using a threshold T, with the edges of the occupied frequency bins given by b_1 and b_2. The center frequency and bandwidth are calculated using $\frac{b_1+b_2}{2}$ and $\frac{b_2-b_1}{2}$, respectively. The center frequency offset is corrected and a low pass filter is applied to the signal to attenuate the noise. In the second stage, PSD is applied again with a larger FFT of size $N_2 > N_1$ to yield a higher resolution frequency and bandwidth estimate using the same procedure. Other than the FFT size, the Welch power spectral used the default parameters in the python SciPy library [VGO20]. We used $N_1 = 64$ and $N_2 = 256$ and $T = 2N_0$.

D.2 Fine Carrier Frequency Estimation

The estimation of the carrier frequency is performed by detecting cyclostationary features at $\alpha = 4f_0$ [RYU14]. Using the initial frequency estimate, a search window $\mathcal{W}_{f_0}$ is calculated and the estimated $4f_0$ is calculated on the input signal $z[k]$ using

$$\max_{\alpha_i \in \mathcal{W}_{f_0}} \left| \sum_{k=0}^{N_r-1} z[k]^4 e^{-j2\pi\alpha_i k\tau_0} \right| \tag{D.1}$$

Given that the coarse frequency estimate from the prior stage was given by f_{0c}, the window $\mathcal{W}_{f_0}$ consisted of 100 samples within $4f_0 + \pm 0.001$.

D.3 Fine Symbol Rate Estimation

The single carrier modulations used in the evaluation have a cylostationnary feature at $\alpha = 1/\tau$, from which τ can be detected [RYU14]. Using the coarse bandwidth estimate, a search window $\mathcal{W}_\tau$ is calculated and the estimated $1/\tau$ is calculated on the input signal $z[k]$ using

$$\max_{\alpha_i \in \mathcal{W}_\tau} \left| \sum_{k=0}^{N_r-1} |z[k]|^2 e^{-j2\pi\alpha_i k\tau_0} \right| \tag{D.2}$$

Given that the coarse bandwidth estimate from the prior stage was given by $1/\tau_c$, the window $\mathcal{W}_\tau$ consisted of 100 samples between $\frac{0.85}{\tau_c}$ and $\frac{1.15}{\tau_c}$.

D.4 Symbol Timing Offset Estimation

The input signal $z[k]$ is first interpolated by a factor of P to obtain $z_I[k]$. An error is calculated using [Gar86]

$$e[k] = (z_I[k - P/2] - z_I[k + P/2])z_I[k]^* \tag{D.3}$$

where $(\cdot)^*$ denotes the conjugate. The error vector is divided into windows of size P, which are averaged across time. The timing offset index is given by the index of the zero down crossing. We used $P = 64$.

D.5 Blind Equalization

The CMA is an iterative blind equalization algorithm. At step m, it generates $\mathbf{w}(m)$ using stochastic gradient descent as follows [Shi12]

$$\mathbf{w}(m) = \mathbf{w}(m-1) - \mu g(|g|^2 - 1)\mathbf{r}(m)^* \tag{D.4}$$

where μ is the learning rate, $g(m) = \mathbf{w}(m-1)^T \mathbf{r}(m)$, and $\mathbf{r}(m)$ is a slice of the input signal $z[k]$ starting with index m and of the same length as the filter $\mathbf{w}$. We used $\mu = 10^{-4}$ and an equalization filter of length 20.

D.6 Genie Equalization

Let $h[n]$ be the channel taps having Fourier transform $H[k]$. The frequency domain representation of the equalized signal $\widehat{\mathbf{Z}}$ is given by $\widehat{Z}[k] = \frac{Z[k]H^*[k]}{H^*[k]H[k]+N_0}$ where $\mathbf{Z}$ is the FFT of the input signal. The equalized signal is obtain using the inverse FFT of $\widehat{\mathbf{Z}}$.

D.7 Symbol Decoding

Let $\hat{\mathbf{s}}$ denote the recovered symbols before the hard decision, $\mathbf{s}$ the true symbols, $\mathbf{c}_M$ be the constellation of linear modulation M. The filtered phase error at the k-th symbol is denoted by $e_f[k]$. Using the known the first symbol, we set $e_f[0] = \hat{s}[0]s[0]^*$. We start by calculating the constellation index of the predicted symbol $s_b[k]$ by finding the minimizer of the Euclidean distance using $s_b[k] = \operatorname{argmin} \left|\hat{s}[k]e^{je_f[k]} - \mathbf{c}_M\right|$, thus making the decoded symbol $s_d[k] = c_m[s_b[k]]$. The phase error $e[k]$ is calculated using the received and the decoded symbol using $e[k] = \arctan(\hat{s}[k]d[k]^*)$, which is then filtered according to $e_f[k] = e_f[k] + \alpha e[k]$ for some constant α. The symbol error rate is calculated by comparing the decoded symbols $\mathbf{s}_d$ with the true symbols $\mathbf{s}$. The symbol by symbol comparison is limited to the shortest of both if their lengths are different due to timing errors. We used $\alpha = 0.5$.

REFERENCES

[AAP19] Ioannis Agadakos, Nikolaos Agadakos, Jason Polakis, and Mohamed R. Amer. "Deep Complex Networks for Protocol-Agnostic Radio Frequency Device Fingerprinting in the Wild." *arXiv:1909.08703 [cs, eess]*, September 2019.

[AEY18] M. Alzenad, A. El-Keyi, and H. Yanikomeroglu. "3-D Placement of an Unmanned Aerial Vehicle Base Station for Maximum Coverage of Users With Different QoS Requirements." *IEEE Wireless Communications Letters*, **7**(1):38–41, February 2018.

[ARD20] Amani Al-Shawabka, Francesco Restuccia, Salvatore D'Oro, Tong Jian, Bruno Costa Rendon, Nasim Soltani, Jennifer Dy, Kaushik Chowdhury, Stratis Ioannidis, and Tommaso Melodia. "Exposing the Fingerprint: Dissecting the Impact of the Wireless Channel on Radio Fingerprinting." *Proc. of IEEE Conference on Computer Communications (INFOCOM)*, p. 10, 2020.

[AV09] Mohammed F. A. Ahmed and Sergiy A. Vorobyov. "Collaborative Beamforming for Wireless Sensor Networks with Gaussian Distributed Sensor Nodes." *IEEE Transactions on Wireless Communications*, **8**(2):638–643, February 2009.

[BB15] Abhijit Bendale and Terrance Boult. "Towards Open Set Deep Networks." *arXiv:1511.06233 [cs]*, November 2015.

[BBG08] Vladimir Brik, Suman Banerjee, Marco Gruteser, and Sangho Oh. "Wireless Device Identification with Radiometric Signatures." In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, pp. 116–127. ACM, 2008.

[Ber99] Dimitri P. Bertsekas. *Nonlinear Programming.* Athena scientific Belmont, 1999.

[Ber00] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control: Vol. 1.* Athena scientific Belmont, 2000.

[BGD19] Gianmarco Baldini, Raimondo Giuliani, and Franc Dimc. "Physical Layer Authentication of Internet of Things Wireless Devices Using Convolutional Neural Networks and Recurrence Plots." *Internet Technology Letters*, **2**(2):e81, 2019.

[BGG19] G. Baldini, C. Gentile, R. Giuliani, and G. Steri. "Comparison of Techniques for Radiometric Identification Based on Deep Convolutional Neural Networks." *Electronics Letters*, **55**(2):90–92, 2019.

[BOO07a] F. Bohagen, P. Orten, and G. E. Oien. "Design of Optimal High-Rank Line-of-Sight MIMO Channels." *IEEE Transactions on Wireless Communications*, **6**(4):1420–1425, April 2007.

[BOO07b] Frode Bohagen, Paal Orten, and Geir Oien. "Optimal Design of Uniform Rectangular Antenna Arrays for Strong Line-of-Sight MIMO Channels." *EURASIP J. Wirel. Commun. Netw.*, **2007**(2):12–12, January 2007.

[BPB19] Lorenzo Bertizzolo, Michele Polese, Leonardo Bonati, Abhimanyu Gosain, Michele Zorzi, and Tommaso Melodia. "mmBAC: Location-Aided mmWave Backhaul Management for UAV-Based Aerial Cells." In *Proceedings of the 3rd ACM Workshop on Millimeter-Wave Networks and Sensing Systems*, mmNets'19, pp. 7–12, New York, NY, USA, October 2019. Association for Computing Machinery.

[BS19] Alexios Balatsoukas-Stimming and Christoph Studer. "Deep Unfolding for Communications Systems: A Survey and Some New Directions." *arXiv:1906.05774 [cs, eess, math]*, June 2019.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge university press, 2004.

[CB20] Thomas Courtat and Hélion du Mas des Bourboux. "A Light Neural Network for Modulation Detection under Impairments." *arXiv:2003.12260 [cs, eess, stat]*, March 2020.

[CC19] Raghavendra Chalapathy and Sanjay Chawla. "Deep Learning for Anomaly Detection: A Survey." *arXiv:1901.03407 [cs, stat]*, January 2019.

[CDL18] P. Chandhar, D. Danev, and E. G. Larsson. "Massive MIMO for Communications With Drone Swarms." *IEEE Transactions on Wireless Communications*, **17**(3):1604–1629, March 2018.

[CDS18] Baibhab Chatterjee, Debayan Das, and Shreyas Sen. "RF-PUF: IoT Security Enhancement through Authentication of Wireless Nodes Using in-Situ Machine Learning." In *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 205–208, Washington, DC, April 2018. IEEE.

[CFP18] Yousra Chakkour, Herman Fernández, Vicent M. Rodrigo Peñarrocha, Lorenzo Rubio, and Juan Reig. "Coherence Time and Doppler Spread Analysis of the V2V Channel in Highway and Urban Environments." In *2018 IEEE International Symposium on Antennas and Propagation USNC/URSI National Radio Science Meeting*, pp. 373–374, July 2018.

[CG20] Junting Chen and David Gesbert. "Efficient Local Map Search Algorithms for the Placement of Flying Relays." *IEEE Transactions on Wireless Communications*, **19**(2):1305–1319, February 2020.

[CGM20] Metehan Cekic, Soorya Gopalakrishnan, and Upamanyu Madhow. "Robust Wireless Fingerprinting: Generalizing Across Space and Time." *arXiv:2002.10791 [cs, eess, stat]*, February 2020.

[CHH20] William H. Clark IV, Steven Hauser, William C. Headley, and Alan J. Michaels. "Training Data Augmentation for Deep Learning RF Systems." *arXiv:2010.00178 [cs, eess]*, September 2020.

[CS03] Liyu Cao and Howard M. Schwartz. "Exponential Convergence of the Kalman Filter Based Parameter Estimation Algorithm." *International Journal of Adaptive Control and Signal Processing*, **17**(10):763–783, 2003.

[CZH20] Shiyao Chen, Yan Zhang, Zunwen He, Jinbo Nie, and Wancheng Zhang. "A Novel Attention Cooperative Framework for Automatic Modulation Recognition." *IEEE Access*, **8**:15673–15686, 2020.

[Das10] Anirban DasGupta. *Normal Approximations and the Central Limit Theorem*, pp. 213–242. Springer, New York, NY, 2010.

[DB16] Steven Diamond and Stephen Boyd. "CVXPY: A Python-embedded modeling language for convex optimization." *Journal of Machine Learning Research*, **17**(83):1–5, 2016.

[DC09] Boris Danev and Srdjan Capkun. "Transient-Based Identification of Wireless Sensor Nodes." In *2009 International Conference on Information Processing in Sensor Networks*, pp. 25–36, April 2009.

[DCB13] Alexander Domahidi, Eric Chu, and Stephen Boyd. "ECOS: An SOCP Solver for Embedded Systems." In *2013 European Control Conference (ECC)*, pp. 3071–3076, Zurich, July 2013. IEEE.

[DCH18] S. Dörner, S. Cammerer, J. Hoydis, and S. t Brink. "Deep Learning Based Communication Over the Air." *IEEE Journal of Selected Topics in Signal Processing*, **12**(1):132–143, February 2018.

[DHH20] Van-Sang Doan, Thien Huynh-The, Cam-Hao Hua, Quoc-Viet Pham, and Dong-Seong Kim. "Learning Constellation Map with Deep CNN for Accurate Modulation Recognition." *arXiv:2009.02026 [cs, eess, math]*, September 2020.

[DWW18] L. Ding, S. Wang, F. Wang, and W. Zhang. "Specific Emitter Identification via Convolutional Neural Networks." *IEEE Communications Letters*, **22**(12):2591–2594, December 2018.

[EGE03] Jeremy Elson, Lewis Girod, and Deborah Estrin. "Fine-Grained Network Time Synchronization Using Reference Broadcasts." *ACM SIGOPS Operating Systems Review*, **36**(SI):147–163, December 2003.

[EK72] Jack Edmonds and Richard M. Karp. "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems." *Journal of the ACM*, **19**(2):248–264, April 1972.

[ENC17] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz. "Help from the Sky: Leveraging UAVs for Disaster Management." *IEEE Pervasive Computing*, **16**(1):24–32, Jan.-Mar. 2017.

[Ett] Ettus Research. "USRP B205mini-i." https://www.ettus.com/all-products/usrp-b205mini-i/.

[Fed21] Federal Aviation Administration. "FAA Aerospace Forecast Fiscal Years 2021–2041." 2021.

[Fer19] Alberto Ferrari. "A Note on Sum and Difference of Correlated Chi-Squared Variables." *arXiv:1906.09982 [math, stat]*, June 2019.

[FIG18] A. Fouda, A. S. Ibrahim, I. Guvenc, and M. Ghosh. "UAV-Based In-Band Integrated Access and Backhaul for 5G Communications." In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pp. 1–5, August 2018.

[FIG19] Abdurrahman Fouda, Ahmed S. Ibrahim, Ísmail Güvenç, and Monisha Ghosh. "Interference Management in UAV-Assisted Integrated Access and Backhaul Cellular Networks." *IEEE Access*, **7**:104553–104566, 2019.

[FT13] Mikael Fallgren and Bogdan Timus. "Scenarios, Requirements and KPIs for 5G Mobile and Wireless System." Technical Report ICT-317669-METIS/D1.1, Mobile and wireless communications Enablers for the Twenty-twenty Information Society, 2013.

[FWH18] He Fang, Xianbin Wang, and Lajos Hanzo. "Learning-Aided Physical Layer Authentication as an Intelligent Process." *arXiv:1808.02456 [cs]*, August 2018.

[Gar86] Floyd M. Gardner. "A BPSK/QPSK Timing-Error Detector for Sampled Receivers." *IEEE Transactions on Communications*, **34**:423–429, 1986.

[GB14] Michael Grant and Stephen Boyd. *CVX: Matlab Software for Disciplined Convex Programming, Version 2.1.* March 2014.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016.

[GCM19] Soorya Gopalakrishnan, Metehan Cekic, and Upamanyu Madhow. "Robust Wireless Fingerprinting via Complex-Valued Neural Networks." *arXiv:1905.09388 [cs, eess, stat]*, May 2019.

[GCX20] Zhifang Gu, He Chen, Pingping Xu, Yonghui Li, and Branka Vucetic. “Physical Layer Authentication for Non-Coherent Massive SIMO-Based Industrial IoT Communications.” *arXiv:2001.07315 [eess]*, January 2020.

[GGD13] Nikhil Gulati, Rachel Greenstadt, Kapil R. Dandekar, and John M. Walsh. “GMM Based Semi-Supervised Learning for Channel-Based Authentication Scheme.” In *2013 IEEE 78th Vehicular Technology Conference (VTC Fall)*, pp. 1–6, September 2013.

[GGG18] Giovanni Geraci, Adrian Garcia-Rodriguez, Lorenzo Galati Giordano, David Lopez-Perez, and Emil Bjoernson. “Supporting UAV Cellular Communications through Massive MIMO.” In *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, May 2018.

[GGL19] Adrian Garcia-Rodriguez, Giovanni Geraci, David Lopez-Perez, Lorenzo Galati Giordano, Ming Ding, and Emil Bjornson. “The Essential Guide to Realizing 5G-Connected UAVs with Massive MIMO.” *IEEE Communications Magazine*, **57**(12):84–90, December 2019.

[GHC19] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. “Recent Advances in Open Set Recognition: A Survey.” *arXiv:1811.08581 [cs, stat]*, July 2019.

[GK17] Farid Golnaraghi and Benjamin C. Kuo. *Automatic Control Systems.* McGraw-Hill Education, 2017.

[GNU] GNU Radio Website. “GNU Radio.” https://gnuradio.org/.

[Gol05] Andrea Goldsmith. *Wireless Communications.* Cambridge university press, 2005.

[GPY20] Jemin George, Anjaly Parayil, Cemal Tugrul Yilmaz, Bethany L. Allik, He Bai, and Aranya Chakrabortty. “Multi-Agent Coordination for Distributed Transmit Beamforming.” In *2020 American Control Conference (ACC)*, pp. 144–149, July 2020.

[GR83] N. H. Gordon and P. F. Ramig. “Cumulative Distribution Function of the Sum of Correlated Chi— Squared Random Variables.” *Journal of Statistical Computation and Simulation*, **17**(1):1–9, January 1983.

[GSK05] M. Guillaud, D.T.M. Slock, and R. Knopp. “A Practical Method for Wireless Channel Reciprocity Exploitation through Relative Calibration.” In *Proceedings of the Eighth International Symposium on Signal Processing and Its Applications, 2005.*, volume 1, pp. 403–406, August 2005.

[GWJ19] Andrey Gritsenko, Zifeng Wang, Tong Jian, Jennifer Dy, Kaushik Chowdhury, and Stratis Ioannidis. “Finding a ‘New’ Needle in the Haystack: Unseen Radio Detection in Large Populations Using Deep Learning.” In *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–10, Newark, NJ, USA, November 2019. IEEE.

[GYP20] Jemin George, Cemal Tugrul Yilmaz, Anjaly Parayil, and Aranya Chakrabortty. “A Model-Free Approach to Distributed Transmit Beamforming.” In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5170–5174, May 2020.

[HAR] HARDKERNEL. “ODROID-XU4 – ODROID.” https://www.hardkernel.com/shop/odroid-xu4/.

[Har11] Fred Harris. “Let’s Assume the System Is Synchronized.” In Ramjee Prasad, Sudhir Dixit, Richard van Nee, and Tero Ojanpera, editors, *Globalization of Mobile and Wireless Communications*, pp. 311–325. Springer Netherlands, Dordrecht, 2011.

[HC17] Ghaith Hattab and Danijela Cabric. “Energy-Efficient Massive Cellular IoT Shared Spectrum Access via Mobile Data Aggregators.” In *Wireless and Mobile Computing, Networking and Communications (WiMob),*, pp. 1–6. IEEE, 2017.

[HC19] S. S. Hanna and D. Cabric. “Deep Learning Based Transmitter Identification Using Power Amplifier Nonlinearity.” In *2019 International Conference on Computing, Networking and Communications (ICNC)*, pp. 674–680, February 2019.

[HC21] Samer Hanna and Danijela Cabric. “Distributed Transmit Beamforming: Design and Demonstration from the Lab to UAVs.” *arXiv:2110.13804 [eess]*, October 2021.

[HCL19] S. Huang, L. Chai, Z. Li, D. Zhang, Y. Yao, Y. Zhang, and Z. Feng. “Automatic Modulation Classification Using Compressive Convolutional Neural Network.” *IEEE Access*, **7**:79636–79643, 2019.

[HDC20] Samer Hanna, Chris Dick, and Danijela Cabric. “Combining Deep Learning and Linear Processing for Modulation Classification and Symbol Decoding.” *arXiv:2006.00729 [eess]*, June 2020.

[HDC21] Samer Hanna, Chris Dick, and Danijela Cabric. “Signal Processing Based Deep Learning for Blind Symbol Decoding and Modulation Classification.” *arXiv:2106.10543 [cs, eess]*, June 2021.

[HEE16] S. S. Hanna, A. A. El-Sherif, and M. Y. ElNainay. “Maximizing USRP N210 SDR Transfer Rate by Offloading Modulation to the On-Board FPGA.” In

2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), pp. 110–115, October 2016.

[HGK20] Ade Pitra Hermawan, Rizki Rivai Ginanjar, Dong-Seong Kim, and Jae-Min Lee. “CNN-Based Automatic Modulation Classification for Beyond 5G Communications.” *IEEE Communications Letters*, pp. 1–1, 2020.

[HH03] B. Hassibi and B.M. Hochwald. “How Much Training Is Needed in Multiple-Antenna Wireless Links?” *IEEE Transactions on Information Theory*, **49**(4):951–963, April 2003.

[HHM17] S. C. Hauser, W. C. Headley, and A. J. Michaels. “Signal Detection Effects on Deep Neural Networks Utilizing Raw IQ for Modulation Classification.” In *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, pp. 121–127, October 2017.

[HJG19] S. Huang, Y. Jiang, Y. Gao, Z. Feng, and P. Zhang. “Automatic Modulation Classification Using Contrastive Fully Convolutional Network.” *IEEE Wireless Communications Letters*, pp. 1–1, 2019.

[HJW19] Hengtao He, Shi Jin, Chao-Kai Wen, Feifei Gao, Geoffrey Ye Li, and Zongben Xu. “Model-Driven Deep Learning for Physical Layer Communications.” *IEEE Wireless Communications*, **26**(5):77–83, October 2019.

[HK03] T. Haustein and U. Kruger. “Smart Geometrical Antenna Design Exploiting the LOS Component to Enhance a MIMO System Based on Rayleigh-Fading in Indoor Scenarios.” In *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003.*, volume 2, pp. 1144–1148 vol.2, September 2003.

[HKC20a] S. Hanna, S. Karunaratne, and D. Cabric. “Open Set Wireless Transmitter Authorization: Deep Learning Approaches and Dataset Considerations.” *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2020.

[HKC20b] Samer Hanna, Samurdhi Karunaratne, and Danijela Cabric. “Deep Learning Approaches for Open Set Wireless Transmitter Authorization.” In *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, May 2020.

[HKC21a] Samer Hanna, Enes Krijestorac, and Danijela Cabric. “Feedback Free Distributed Transmit Beamforming Using Guided Directionality.” *arXiv:2108.01837 [eess]*, August 2021.

[HKC21b] Samer Hanna, Enes Krijestorac, and Danijela Cabric. “UAV Swarm Position Optimization for High Capacity MIMO Backhaul.” *IEEE Journal on Selected Areas in Communications*, pp. 3006–3021, 2021.

[HKH21] Mikko Honkala, Dani Korpi, and Janne M. J. Huttunen. "DeepRx: Fully Convolutional Deep Learning Receiver." *arXiv:2005.01494 [cs, eess]*, January 2021.

[HWW12] J. Huang, P. Wang, and Q. Wan. "Collaborative Beamforming for Wireless Sensor Networks with Arbitrary Distributed Sensors." *IEEE Communications Letters*, **16**(7):1118–1120, July 2012.

[HYC19] S. Hanna, H. Yan, and D. Cabric. "Distributed UAV Placement Optimization for Cooperative Line-of-Sight MIMO Communications." In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4619–4623, May 2019.

[HYM16] S. Hayat, E. Yanmaz, and R. Muzaffar. "Survey on Unmanned Aerial Vehicle Networks for Civil Applications: A Communications Viewpoint." *IEEE Communications Surveys Tutorials*, **18**(4):2624–2661, Fourthquarter 2016.

[HYW20] H. Huang, Y. Yang, H. Wang, Z. Ding, H. Sari, and F. Adachi. "Deep Reinforcement Learning for UAV Navigation Through Massive MIMO Technique." *IEEE Transactions on Vehicular Technology*, **69**(1):1117–1121, January 2020.

[HZR15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition." *arXiv:1512.03385 [cs]*, December 2015.

[HZX17] D. Hong, Z. Zhang, and X. Xu. "Automatic Modulation Classification Using Recurrent Neural Networks." In *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pp. 695–700, December 2017.

[IP08] D. R. Brown III and H. V. Poor. "Time-Slotted Round-Trip Carrier Synchronization for Distributed Beamforming." *IEEE Transactions on Signal Processing*, **56**(11):5630–5643, November 2008.

[IQM13] A. T. Irish, F. Quitin, U. Madhow, and M. Rodwell. "Achieving Multiple Degrees of Freedom in Long-Range Mm-Wave MIMO Channels Using Randomly Distributed Relays." In *2013 Asilomar Conference on Signals, Systems and Computers*, pp. 722–727, November 2013.

[JOA18] H. Jafari, O. Omotere, D. Adesina, H. Wu, and L. Qian. "IoT Devices Fingerprinting Using Deep Learning." In *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, pp. 1–9, October 2018.

[JRL17] S. Jayaprakasam, S. K. A. Rahim, and C. Y. Leow. "Distributed and Collaborative Beamforming in Wireless Sensor Networks: Classifications, Trends, and Research Directions." *IEEE Communications Surveys Tutorials*, **19**(4):2092–2116, Fourthquarter 2017.

[JWC20] Weiheng Jiang, Xiaogang Wu, Bolin Chen, Wenjiang Feng, and Yi Jin. "Time-Frequency Analysis Based Blind Modulation Classification for Multiple-Antenna Systems." *arXiv:2004.00378 [cs, eess, stat]*, April 2020.

[KAL19] Dmitry Kramarev, Ishtiaq Ahmad, Kelvin Layton, Marc Lavenant, Hidayat Soetiyono, Gottfried Lechner, Hajime Suzuki, Ian Grivell, and Stephen Leak. "Event-Triggered Synchronization for Mobile Distributed Transmit Beamforming." In *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, pp. 343–348, November 2019.

[KBB19] Fekher Khelifi, Abbas Bradai, Abderrahim Benslimane, Priyanka Rawat, and Mohamed Atri. "A Survey of Localization Systems in Internet of Things." *Mobile Networks and Applications*, **24**(3):761–785, June 2019.

[KCZ18] A. A. Khuwaja, Y. Chen, N. Zhao, M. Alouini, and P. Dobbins. "A Survey of Channel Modeling for UAV Communications." *IEEE Communications Surveys Tutorials*, **20**(4):2804–2821, fourth quarter 2018.

[KDG19] Ertan Kazikli, Berkan Dulek, and Sinan Gezici. "Optimal Joint Modulation Classification and Symbol Decoding." *IEEE Transactions on Wireless Communications*, **18**(5):2623–2638, May 2019.

[KDS20] Justin Kong, Fikadu T. Dagefu, and Brian M. Sadler. "Simultaneous Beamforming and Nullforming for Covert Wireless Communications." In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pp. 1–6, May 2020.

[KHC19] Enes Krijestorac, Samer Hanna, and Danijela Cabric. "UAV Access Point Placement for Connectivity to a User with Unknown Location Using Deep RL." In *2019 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, December 2019.

[KN00] Samuel Kotz and Saralees Nadarajah. *Extreme Value Distributions: Theory and Applications*. World Scientific, 2000.

[KSH07] A. Knopp, R. T. Schwarz, C. A. Hofmann, M. Chouayakh, and B. Lankl. "Measurements on the Impact of Sparse Multipath Components on the LOS MIMO Channel Capacity." In *2007 4th International Symposium on Wireless Communication Systems*, pp. 55–60, October 2007.

[Kuh55] H. W. Kuhn. "The Hungarian Method for the Assignment Problem." *Naval Research Logistics Quarterly*, **2**(1-2):83–97, 1955.

[KV20] Ziqi Ke and Haris Vikalo. "Real-Time Radio Technology and Modulation Classification via an LSTM Auto-Encoder." *arXiv:2011.08295 [cs, eess]*, November 2020.

[Lar05] P. Larsson. “Lattice Array Receiver and Sender for Spatially Orthonormal MIMO Communication.” In *2005 IEEE 61st Vehicular Technology Conference*, volume 1, pp. 192–196 Vol. 1, May 2005.

[LCT09] S. Liu, Y. Chen, W. Trappe, and L. J. Greenstein. “ALDO: An Anomaly Detection Framework for Dynamic Spectrum Access Networks.” In *IEEE INFOCOM 2009 - The 28th Conference on Computer Communications*, pp. 675–683, Rio De Janeiro, Brazil, April 2009. IEEE.

[LCW19] Chuan-Chi Lai, Chun-Ting Chen, and Li-Chun Wang. “On-Demand Density-Aware UAV Base Station 3D Placement for Arbitrarily Distributed Users with Guaranteed Data Rates.” *IEEE Wireless Communications Letters*, **8**(3):913–916, June 2019.

[LET14] Erik G. Larsson, Ove Edfors, Fredrik Tufvesson, and Thomas L. Marzetta. “Massive MIMO for next Generation Wireless Systems.” *IEEE Communications Magazine*, **52**(2):186–195, February 2014.

[LFS19] Shuang Liang, Zhiyi Fang, Geng Sun, Yanheng Liu, Xiaohui Zhao, Guannan Qu, Ying Zhang, and Victor CM Leung. “JSSA: Joint Sidelobe Suppression Approach for Collaborative Beamforming in Wireless Sensor Networks.” *IEEE Access*, **7**:151803–151817, 2019.

[LGS18] Stephen Leak, Ian Grivell, Hajime Suzuki, Chang Kyung Sung, Mark Hedley, Gottfried Lechner, Marc Lavenant, Hidayat Soetiyono, and Dmitry Kramarev. “Distributed Transmit Beamforming Expanding the Capacity and Range of Tactical Communications.” In *2018 Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, November 2018.

[LPW20] Lei Li, Qihang Peng, and Jun Wang. “Deep Architectures for Modulation Recognition with Multiple Receive Antennas.” *arXiv:2008.06720 [cs, eess, math]*, August 2020.

[LRP73] Gerald W. Lank, Irving S. Reed, and Gerald E. Pollon. “A Semicoherent Detection and Doppler Estimation Statistic.” *IEEE Transactions on Aerospace and Electronic Systems*, **AES-9**(2):151–165, March 1973.

[LTZ19] kaisheng Liao, Guanhong Tao, Yi Zhong, Yaping Zhang, and Zhenghong Zhang. “Sequential Convolutional Recurrent Neural Networks for Fast Automatic Modulation Classification.” *The Astrophysical Journal*, **878**(1):6, June 2019.

[LW19] Yu Liu and Fanggang Wang. “Blind Channel Estimation and Data Detection with Unknown Modulation and Coding Scheme.” *arXiv:1909.11306 [cs, eess, math]*, September 2019.

[LWL19] Chang Liu, Jie Wang, Xuemeng Liu, and Ying-Chang Liang. "Deep CM-CNN for Spectrum Sensing in Cognitive Radio." *IEEE Journal on Selected Areas in Communications*, **37**(10):2306–2321, October 2019.

[LWN20] Chang Liu, Zhiqiang Wei, Derrick Wing Kwan Ng, Jinhong Yuan, and Ying-Chang Liang. "Deep Transfer Learning for Signal Detection in Ambient Backscatter Communications." *arXiv:2009.05231 [cs, eess, math]*, September 2020.

[LZZ18] Liang Liu, Shuowen Zhang, and Rui Zhang. "CoMP in the Sky: UAV Placement and Movement Optimization for Multi-User Communications." *arXiv:1802.10371 [cs, math]*, February 2018.

[MBM07] R. Mudumbai, G. Barriac, and U. Madhow. "On the Feasibility of Distributed Beamforming in Wireless Networks." *Transactions on Wireless Communications*, **6**(5):1754–1763, May 2007.

[MBM09] R. Mudumbai, D.R. Brown, U. Madhow, and H.V. Poor. "Distributed Transmit Beamforming: Challenges and Recent Progress." *IEEE Communications Magazine*, **47**(2):102–110, February 2009.

[MBM19] Subhramoy Mohanti, Carlos Bocanegra, Jason Meyer, Gokhan Secinti, Mithun Diddi, Hanumant Singh, and Kaushik Chowdhury. "AirBeam: Experimental Demonstration of Distributed Beamforming by a Swarm of UAVs." In *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 162–170, November 2019.

[MCH19] Cyrille Morin, Leonardo Cardoso, Jakob Hoydis, Jean-Marie Gorce, and Thibaud Vial. "Transmitter Classification With Supervised Deep Learning." *arXiv:1905.07923 [cs, eess]*, May 2019.

[MHC17] M. Mirmohammadsadeghi, S. S. Hanna, and D. Cabric. "Modulation Classification Using Convolutional Neural Networks and Spatial Transformer Networks." In *2017 51st Asilomar Conference on Signals, Systems, and Computers*, pp. 936–939, October 2017.

[MHM10] R. Mudumbai, J. Hespanha, U. Madhow, and G. Barriac. "Distributed Transmit Beamforming Using Feedback Control." *IEEE Transactions on Information Theory*, **56**(1):411–426, January 2010.

[MM18] A. Muralidharan and Y. Mostofi. "Energy Optimal Distributed Beamforming Using Unmanned Vehicles." *IEEE Transactions on Control of Network Systems*, **5**(4):1529–1540, December 2018.

[MNC18] À O. Martínez, J. Ø Nielsen, E. De Carvalho, and P. Popovski. "An Experimental Study of Massive MIMO Properties in 5G Scenarios." *IEEE Transactions on Antennas and Propagation*, **66**(12):7206–7215, December 2018.

[MRS18] K. Merchant, S. Revay, G. Stantchev, and B. Nousain. "Deep Learning for RF Device Fingerprinting in Cognitive Communication Networks." *IEEE Journal of Selected Topics in Signal Processing*, **12**(1):160–167, February 2018.

[MS17] David W. Matolak and Ruoyu Sun. "Air–Ground Channel Characterization for Unmanned Aircraft Systems—Part III: The Suburban and Near-Urban Environments." *IEEE Transactions on Vehicular Technology*, **66**(8):6607–6618, August 2017.

[MSB17] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, and Merouane Debbah. "Communications and Control for Wireless Drone-Based Antenna Array." *arXiv:1712.10291 [cs, eess, math]*, December 2017.

[MSB18] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, Young-Han Nam, and Merouane Debbah. "A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems." *arXiv:1803.00680 [cs, math]*, March 2018.

[MSB19] Mohammad Mozaffari, Walid Saad, Mehdi Bennis, Young-Han Nam, and Mérouane Debbah. "A Tutorial on UAVs for Wireless Networks: Applications, Challenges, and Open Problems." *IEEE Communications Surveys Tutorials*, **21**(3):2334–2360, third quarter 2019.

[NBC19] Nataliia Neshenko, Elias Bou-Harb, Jorge Crichigno, Georges Kaddoum, and Nasir Ghani. "Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations." *IEEE Communications Surveys Tutorials*, **21**(3):2702–2733, thirdquarter 2019.

[NLM14] Hien Quoc Ngo, Erik G. Larsson, and Thomas L. Marzetta. "Aspects of Favorable Propagation in Massive MIMO." In *2014 22nd European Signal Processing Conference (EUSIPCO)*, pp. 76–80, September 2014.

[NZH11] N. T. Nguyen, G. Zheng, Z. Han, and R. Zheng. "Device Fingerprinting to Enhance Wireless Security Using Nonparametric Bayesian Method." In *2011 Proceedings IEEE INFOCOM*, pp. 1404–1412, April 2011.

[NZH19] Jinbo Nie, Yan Zhang, Zunwen He, Shiyao Chen, Shouliang Gong, and Wancheng Zhang. "Deep Hierarchical Network for Automatic Modulation Classification." *IEEE Access*, **7**:94604–94613, 2019.

[OCC16] Timothy J. O'Shea, Johnathan Corgan, and T. Charles Clancy. "Convolutional Radio Modulation Recognition Networks." *arXiv:1602.04105 [cs]*, February 2016.

[OEC17] Timothy J. O'Shea, Tugba Erpek, and T. Charles Clancy. "Deep Learning Based MIMO Communications." *arXiv:1707.07980 [cs, math]*, July 2017.

[OKC16] T. J. O'Shea, K. Karra, and T. C. Clancy. "Learning to Communicate: Channel Auto-Encoders, Domain Specific Regularizers, and Attention." In *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 223–228, December 2016.

[OKC17] Timothy J. O'Shea, Kiran Karra, and T. Charles Clancy. "Learning Approximate Neural Estimators for Wireless Channel State Information." *arXiv:1707.06260 [cs]*, July 2017.

[OMP05] H. Ochiai, P. Mitran, H.V. Poor, and V. Tarokh. "Collaborative Beamforming for Distributed Wireless Ad Hoc Sensor Networks." *IEEE Transactions on Signal Processing*, **53**(11):4110–4124, November 2005.

[OPB16] Timothy J. O'Shea, Latha Pemula, Dhruv Batra, and T. Charles Clancy. "Radio Transformer Networks: Attention Models for Learning to Synchronize in Wireless Systems." *arXiv:1605.00716 [cs]*, May 2016.

[ORC17] Timothy J. O'Shea, Tamoghna Roy, and T. Charles Clancy. "Over the Air Deep Learning Based Radio Signal Classification." *arXiv:1712.04578 [cs, eess]*, December 2017.

[ORC18] T. J. O'Shea, T. Roy, and T. C. Clancy. "Over-the-Air Deep Learning Based Radio Signal Classification." *IEEE Journal of Selected Topics in Signal Processing*, **12**(1):168–179, February 2018.

[PHN20] A. Pogue, S. Hanna, A. Nichols, X. Chen, D. Cabric, and A. Mehta. "Path Planning Under MIMO Network Constraints for Throughput Enhancement in Multi-Robot Data Aggregation Tasks." In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11824–11830, October 2020.

[PHZ19] L. Peng, A. Hu, J. Zhang, Y. Jiang, J. Yu, and Y. Yan. "Design of a Hybrid RF Fingerprint Extraction and Device Classification Scheme." *IEEE Internet of Things Journal*, **6**(1):349–360, February 2019.

[PJW19] Shengliang Peng, Hanyu Jiang, Huaxia Wang, Hathal Alwageed, Yu Zhou, Marjan Mazrouei Sebdani, and Yu-Dong Yao. "Modulation Classification Based on Signal Constellation Diagrams and Deep Learning." *IEEE Transactions on Neural Networks and Learning Systems*, **30**(3):718–727, March 2019.

[PMB13] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. "On the Difficulty of Training Recurrent Neural Networks." In *Proceedings of the 30th International*

Conference on International Conference on Machine Learning - Volume 28, ICML'13, pp. III–1310–III–1318, Atlanta, GA, USA, June 2013. PMLR.

[PMG16] Ben Peiffer, Raghu Mudumbai, Sairam Goguri, Anton Kruger, and Soura Dasgupta. "Experimental Demonstration of Retrodirective Beamforming from a Fully Wireless Distributed Array." In *MILCOM 2016 - 2016 IEEE Military Communications Conference*, pp. 442–447. IEEE, November 2016.

[POP18] J. Pokorny, A. Ometov, P. Pascual, C. Baquero, P. Masek, A. Pyattaev, A. Garcia, C. Castillo, S. Andreev, J. Hosek, and Y. Koucheryavy. "Concept Design and Performance Evaluation of UAV-Based Backhaul Link with Antenna Steering." *Journal of Communications and Networks*, **20**(5):473–483, October 2018.

[PSY21] Shengliang Peng, Shujun Sun, and Yu-Dong Yao. "A Survey of Modulation Classification Using Deep Learning: Signal Representation and Data Preprocessing." *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–19, 2021.

[PW00] Florian A. Potra and Stephen J. Wright. "Interior-Point Methods." *Journal of Computational and Applied Mathematics*, **124**(1):281–302, December 2000.

[PYP19] Y. Pan, S. Yang, H. Peng, T. Li, and W. Wang. "Specific Emitter Identification Based on Deep Residual Networks." *IEEE Access*, **7**:54425–54434, 2019.

[QMR12] F. Quitin, U. Madhow, M. M. U. Rahman, and R. Mudumbai. "Demonstrating Distributed Transmit Beamforming with Software-Defined Radios." In *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp. 1–3, June 2012.

[QRM13] 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 Transactions on Wireless Communications*, **12**(3):1418–1428, March 2013.

[RBM12a] Muhammad M. Rahman, Henry E. Baidoo-Williams, Raghuraman Mudumbai, and Soura Dasgupta. "Fully Wireless Implementation of Distributed Beamforming on a Software-Defined Radio Platform." In *Proceedings of the 11th International Conference on Information Processing in Sensor Networks*, IPSN '12, pp. 305–316, New York, NY, USA, 2012. ACM.

[RBM12b] D. Richard Brown, Patrick Bidigare, and Upamanyu Madhow. "Receiver-Coordinated Distributed Transmit Beamforming with Kinematic Tracking." In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5209–5212, March 2012.

[RDA19] Francesco Restuccia, Salvatore D'Oro, Amani Al-Shawabka, Mauro Belgiovine, Luca Angioloni, Stratis Ioannidis, Kaushik Chowdhury, and Tommaso Melodia. "DeepRadioID: Real-Time Channel-Resilient Optimization of Deep Learning-Based Radio Fingerprinting Algorithms." *arXiv:1904.07623 [cs]*, April 2019.

[RIG16] N. Rupasinghe, A. S. Ibrahim, and I. Guvenc. "Optimum Hovering Locations with Angular Domain User Separation for Cooperative UAV Networks." In *2016 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, December 2016.

[RJY19] Sharan Ramjee, Shengtai Ju, Diyu Yang, Xiaoyu Liu, Aly El Gamal, and Yonina C. Eldar. "Fast Deep Learning for Automatic Modulation Classification." *arXiv:1901.05850 [cs, eess, stat]*, January 2019.

[RJY20] Sharan Ramjee, Shengtai Ju, Diyu Yang, Xiaoyu Liu, Aly El Gamal, and Yonina C. Eldar. "Ensemble Wrapper Subsampling for Deep Modulation Classification." *arXiv:2005.04586 [cs, eess, stat]*, May 2020.

[RMC19] Debashri Roy, Tathagata Mukherjee, Mainak Chatterjee, Erik Blasch, and Eduardo Pasiliao. "RFAL: Adversarial Learning for RF Transmitter Identification and Classification." *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2019.

[RMG18] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin. "Deep Learning Models for Wireless Signal Classification With Distributed Low-Cost Spectrum Sensors." *IEEE Transactions on Cognitive Communications and Networking*, **4**(3):433–445, September 2018.

[RML19] Sreeraj Rajendran, Wannes Meert, Vincent Lenders, and Sofie Pollin. "Unsupervised Wireless Spectrum Anomaly Detection With Interpretable Features." *IEEE Transactions on Cognitive Communications and Networking*, **5**(3):637–647, September 2019.

[RPB18] Y. Ren, L. Peng, W. Bai, and J. Yu. "A Practical Study Of Channel Influence On Radio Frequency Fingerprint Features." In *2018 IEEE International Conference on Electronics and Communication Engineering (ICECE)*, pp. 1–7, December 2018.

[RSC12] S. U. Rehman, K. Sowerby, and C. Coghill. "Analysis of Receiver Front End on the Performance of RF Fingerprinting." In *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)*, pp. 2494–2499, September 2012.

[RSI18] S. Riyaz, K. Sankhe, S. Ioannidis, and K. Chowdhury. "Deep Learning Convolutional Neural Networks for Radio Identification." *IEEE Communications Magazine*, **56**(9):146–152, September 2018.

[RSO05] Dipankar Raychaudhuri, Ivan Seskar, Max Ott, Sachin Ganu, Kishore Ramachandran, Haris Kremo, Robert Siracusa, Hang Liu, and Manpreet Singh. "Overview of the ORBIT Radio Grid Testbed for Evaluation of Next-Generation Wireless Network Protocols." In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 3, pp. 1664–1669. IEEE, 2005.

[RYU14] Eric Rebeiz, Fang-Li Yuan, Paulo Urriza, Dejan Marković, and Danijela Cabric. "Energy-Efficient Processor for Blind Signal Classification in Cognitive Radio Networks." *IEEE Transactions on Circuits and Systems I: Regular Papers*, **61**(2):587–599, 2014.

[SAB13] George Sklivanitis, Konstantinos Alexandris, and Aggelos Bletsas. "Testbed for Non-Coherent Zero-Feedback Distributed Beamforming." In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2563–2567, May 2013.

[SB11] George Sklivanitis and Aggelos Bletsas. "Testing Zero-Feedback Distributed Beamforming with a Low-Cost SDR Testbed." In *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pp. 104–108, November 2011.

[SBG19] Linda Senigagliesi, Marco Baldi, and Ennio Gambi. "Statistical and Machine Learning-Based Decision Techniques for Physical Layer Authentication." *arXiv:1909.07969 [cs]*, September 2019.

[SBS20] Georgy Skorobogatov, Cristina Barrado, and Esther Salamí. "Multiple UAV Systems: A Survey." *Unmanned Systems*, **08**(02):149–169, April 2020.

[SBZ18] Kunal Sankhe, Mauro Belgiovine, Fan Zhou, Shamnaz Riyaz, Stratis Ioannidis, and Kaushik Chowdhury. "ORACLE: Optimized Radio clAssification through Convolutional neuraL nEtworks." *arXiv:1812.01124 [cs, eess]*, December 2018.

[SBZ19] Kunal Sankhe, Mauro Belgiovine, Fan Zhou, Luca Angioloni, Francesco Restuccia, Salvatore D'Oro, Tommaso Melodia, Stratis Ioannidis, and Kaushik Chowdhury. "No Radio Left Behind: Radio Fingerprinting Through Deep Learning of Physical-Layer Hardware Impairments." *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2019.

[SEH19] Samer Hanna, Enes Krijestorac, Han Yan, and Danijela Cabric. "UAV Swarms as Amplify-and-Forward MIMO Relays." In *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, July 2019.

[Shi12] Qinghua Shi. "Blind Equalization and Characteristic Function Based Robust Modulation Recognition." In *2012 14th International Conference on Advanced Communication Technology (ICACT)*, pp. 660–664, February 2012.

[SHL20] Xiaolei Shang, Honglin Hu, Xiaoqiang Li, Tianheng Xu, and Ting Zhou. “Dive into Deep Learning Based Automatic Modulation Classification: A Disentangled Approach.” *IEEE Access*, pp. 1–1, 2020.

[SLC19] Geng Sun, Yanheng Liu, Zhaoyu Chen, Aimin Wang, Ying Zhang, Daxin Tian, and Victor CM Leung. “Energy Efficient Collaborative Beamforming for Reducing Sidelobe in Wireless Sensor Networks.” *IEEE Transactions on Mobile Computing*, 2019.

[SMG13] W. Su, J. D. Matyjas, M. J. Gans, and S. Batalama. “Maximum Achievable Capacity in Airborne MIMO Communications with Arbitrary Alignments of Linear Transceiver Antenna Arrays.” *IEEE Transactions on Wireless Communications*, **12**(11):5584–5593, November 2013.

[SRG15] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini. “Security, Privacy and Trust in Internet of Things: The Road Ahead.” *Computer Networks*, **76**:146–164, January 2015.

[SSA19] Hazim Shakhatreh, Ahmad Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. “Unmanned Aerial Vehicles: A Survey on Civil Applications and Key Research Challenges.” *IEEE Access*, **7**:48572–48634, 2019.

[SWK09] Yanmin Sun, Andrew K. C. Wong, and Mohamed S. Kamel. “CLASSIFICATION OF IMBALANCED DATA: A REVIEW.” *International Journal of Pattern Recognition and Artificial Intelligence*, **23**(04):687–719, June 2009.

[SXL17] Lei Shu, Hu Xu, and Bing Liu. “DOC: Deep Open Classification of Text Documents.” *arXiv:1709.08716 [cs]*, September 2017.

[TKF20] Andrea Toma, Ali Krayani, Muhammad Farrukh, Haoran Qi, Lucio Marcenaro, Yue Gao, and Carlo S. Regazzoni. “AI-Based Abnormality Detection at the PHY-Layer of Cognitive Radio by Learning Generative Models.” *IEEE Transactions on Cognitive Communications and Networking*, **6**(1):21–34, March 2020.

[TL98] N. A. Thacker and A. Lacey. “Tutorial: The Kalman Filter.” *Imaging Science and Biomedical Engineering Division, Medical School, University of Manchester*, p. 61, 1998.

[TMD20] Angelo Trotta, Ufuk Muncuk, Marco Di Felice, and Kaushik R. Chowdhury. “Persistent Crowd Tracking Using Unmanned AerIal Vehicle Swarms: A Novel Framework for Energy and Mobility Management.” *IEEE Vehicular Technology Magazine*, **15**(2):96–103, June 2020.

[TMG20] N. Tafintsev, D. Moltchanov, M. Gerasimenko, M. Gapeyenko, J. Zhu, S. Yeh, N. Himayat, S. Andreev, Y. Koucheryavy, and M. Valkama. "Aerial Access and Backhaul in mmWave B5G Systems: Performance Dynamics and Optimization." *IEEE Communications Magazine*, **58**(2):93–99, February 2020.

[TO21] Berke Tezergil and Ertan Onur. "Wireless Backhaul in 5G and Beyond: Issues, Challenges and Opportunities." *arXiv:2103.08234 [cs]*, March 2021.

[Tre85] S. Tretter. "Estimating the Frequency of a Noisy Sinusoid by Linear Regression (Corresp.)." *IEEE Transactions on Information Theory*, **31**(6):832–835, November 1985.

[TV05] David Tse and Pramod Viswanath. *Fundamentals of Wireless Communication.* Cambridge university press, 2005.

[VGO20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python." *Nature Methods*, **17**:261–272, 2020.

[VSD18] Evgenii Vinogradov, Hazem Sallouha, Sibren De Bast, Mohammad Mahdi Azari, and Sofie Pollin. "Tutorial on UAV: A Blue Sky View on Wireless Communication." *Journal of Mobile Multimedia*, **14**(4):395–468, 2018.

[VTB18] G. Vanhoy, N. Thurston, A. Burger, J. Breckenridge, and T. Bose. "Hierarchical Modulation Classification Using Deep Learning." In *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, pp. 20–25, October 2018.

[VVN16] Tien Dang Vo-Huu, Triet Dang Vo-Huu, and Guevara Noubir. "Fingerprinting Wi-Fi Devices Using Software Defined Radios." In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks - WiSec '16*, pp. 3–14, Darmstadt, Germany, 2016. ACM Press.

[WC02] Brad Williams and Tracy Camp. "Comparison of Broadcasting Techniques for Mobile Ad Hoc Networks." In *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing*, MobiHoc '02, pp. 194–205, New York, NY, USA, 2002. ACM.

[WFC19] Xuanxuan Wang, Wei Feng, Yunfei Chen, and Ning Ge. "UAV Swarm-Enabled Aerial CoMP: A Physical Layer Security Perspective." *arXiv:1905.05449 [cs, eess, math]*, May 2019.

[WFK18] Qingyang Wu, Carlos Feres, Daniel Kuzmenko, Ding Zhi, Zhou Yu, Xin Liu, and Xiaoguang ‘Leo’ Liu. “Deep Learning Based RF Fingerprinting for Device Identification and Wireless Security.” *Electronics Letters*, **54**(24):1405–1407, November 2018.

[WHH16] Xianbin Wang, Peng Hao, and Lajos Hanzo. “Physical-Layer Authentication for Wireless Security Enhancement: Current Challenges and Future Developments.” *IEEE Communications Magazine*, **54**(6):152–158, June 2016.

[WHM18] Lauren J. Wong, William C. Headley, and Alan J. Michaels. “Emitter Identification Using CNN IQ Imbalance Estimators.” *arXiv:1808.02369 [eess]*, August 2018.

[WKS17] Andreas Weinand, Michael Karrenbauer, Raja Sattiraju, and Hans D. Schotten. “Application of Machine Learning for Channel Based Message Authentication in Mission Critical Machine Type Communication.” *arXiv:1711.05088 [cs, eess]*, November 2017.

[WSP16] W. Wang, Z. Sun, S. Piao, B. Zhu, and K. Ren. “Wireless Physical-Layer Identification: Modeling and Validation.” *IEEE Transactions on Information Forensics and Security*, **11**(9):2091–2106, September 2016.

[WWF19] Lauren J. Wong, Parker White, Michael Fowler, and William C. Headley. “Distributed Automatic Modulation Classification with Compressed Data.” In *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, pp. 299–304, November 2019.

[WXZ21] Qingqing Wu, Jie Xu, Yong Zeng, Derrick Wing Kwan Ng, Naofal Al-Dhahir, Robert Schober, and A. Lee Swindlehurst. “A Comprehensive Overview on 5G-and-Beyond Networks With UAVs: From Communications to Sensing and Intelligence.” *IEEE Journal on Selected Areas in Communications*, **39**(10):2912–2945, October 2021.

[XD19] Tongyang Xu and Izzat Darwazeh. “Deep Learning for Over-the-Air Non-Orthogonal Signal Classification.” *arXiv:1911.06174 [cs, eess]*, November 2019.

[XGM08] L. Xiao, L. J. Greenstein, N. B. Mandayam, and W. Trappe. “Using the Physical Layer for Wireless Authentication in Time-Variant Channels.” *IEEE Transactions on Wireless Communications*, **7**(7):2571–2579, July 2008.

[XSN18] Dongfang Xu, Yan Sun, Derrick Wing Kwan Ng, and Robert Schober. “Robust Resource Allocation for UAV Systems with UAV Jittering and User Location Uncertainty.” *arXiv:1809.03706 [cs, math]*, September 2018.

[XSN19] Dongfang Xu, Yan Sun, Derrick Wing Kwan Ng, and Robert Schober. "Multiuser MISO UAV Communications in Uncertain Environments with No-Fly Zones: Robust Trajectory and Resource Allocation Design." *arXiv:1905.10731 [cs, eess, math]*, May 2019.

[XZ18] Ning Xie and Shengli Zhang. "Blind Authentication at the Physical Layer Under Time-Varying Fading Channels." *IEEE Journal on Selected Areas in Communications*, **36**(7):1465–1479, July 2018.

[XZS16] Qiang Xu, Rong Zheng, Walid Saad, and Zhu Han. "Device Fingerprinting in Wireless Networks: Challenges and Opportunities." *IEEE Communications Surveys Tutorials*, **18**(1):94–104, Firstquarter 2016.

[YBH17] K. Youssef, Louis-S. Bouchard, K. Z. Haigh, H. Krovi, J. Silovsky, and C. P. Vander Valk. "Machine Learning Approach to RF Transmitter Identification." *arXiv:1711.01559 [cs, eess, stat]*, November 2017.

[YHB19] H. Yan, S. Hanna, K. Balke, R. Gupta, and D. Cabric. "Software Defined Radio Implementation of Carrier and Timing Synchronization for Distributed Arrays." In *2019 IEEE Aerospace Conference*, pp. 1–12, March 2019.

[YHL19] J. Yu, A. Hu, G. Li, and L. Peng. "A Robust RF Fingerprinting Approach Using Multi-Sampling Convolutional Neural Network." *IEEE Internet of Things Journal*, pp. 1–1, 2019.

[YHP16] J. Yu, A. Hu, and L. Peng. "Blind DCTF-Based Estimation of Carrier Frequency Offset for RF Fingerprint Extraction." In *2016 8th International Conference on Wireless Communications Signal Processing (WCSP)*, pp. 1–6, October 2016.

[YHZ19] Jiabao Yu, Aiqun Hu, Fen Zhou, Yuexiu Xing, Yi Yu, Guyue Li, and Linning Peng. "Radio Frequency Fingerprint Identification Based on Denoising Autoencoders." *arXiv:1907.08809 [cs, eess]*, July 2019.

[YNF19] M. Youssef, C. A. Nour, J. Farah, and C. Douillard. "Backhaul-Constrained Resource Allocation and 3D Placement for UAV-Enabled Networks." In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pp. 1–7, September 2019.

[YP02] Yung-Szu Tu and G. J. Pottie. "Coherent Cooperative Transmission from Multiple Adjacent Antennas to a Distant Stationary Antenna through AWGN Channels." In *Vehicular Technology Conference. IEEE 55th Vehicular Technology Conference. VTC Spring 2002 (Cat. No.02CH37367)*, volume 1, pp. 130–134 vol.1, May 2002.

[YSC18] Kumar Yashashwi, Amit Sethi, and Prasanna Chaporkar. "A Learnable Distortion Correction Module for Modulation Recognition." *arXiv:1803.01319 [eess]*, March 2018.

[YSK19] Ryota Yoshihashi, Wen Shao, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. "Classification-Reconstruction Learning for Open-Set Recognition." In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4011–4020, June 2019.

[ZAG09] Keyvan Zarifi, Sofiene Affes, and Ali Ghrayeb. "Distributed Beamforming for Wireless Sensor Networks with Random Node Location." In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2261–2264, April 2009.

[ZDL20] Yi Zhang, Akash Doshi, Rob Liston, Wai-tian Tan, Xiaoqing Zhu, Jeffrey G. Andrews, and Robert W. Heath. "DeepWiPHY: Deep Learning-Based Receiver Design and Dataset for IEEE 802.11ax Systems." *arXiv:2010.09268 [cs, eess]*, October 2020.

[ZHL19] Xinyu Zhou, Aiqun Hu, Guyue Li, Linning Peng, Yuexiu Xing, and Jiabao Yu. "Design of a Robust RF Fingerprint Generation and Classification Scheme for Practical Device Identification." In *2019 IEEE Conference on Communications and Network Security (CNS)*, pp. 196–204, June 2019.

[ZHL21] X. Zhou, A. Hu, G. Li, L. Peng, Y. Xing, and J. Yu. "A Robust Radio Frequency Fingerprint Extraction Scheme for Practical Device Recognition." *IEEE Internet of Things Journal*, pp. 1–1, 2021.

[ZLW20] Z. Zhang, H. Luo, C. Wang, C. Gan, and Y. Xiang. "Automatic Modulation Classification Using CNN-LSTM Based Dual-Stream Structure." *IEEE Transactions on Vehicular Technology*, **69**(11):13521–13531, November 2020.

[ZQC19] Shilian Zheng, Peihan Qi, Shichuan Chen, and Xiaoniu Yang. "Fusion Methods for CNN-Based Automatic Modulation Classification." *IEEE Access*, **7**:66496–66504, 2019.

[ZTJ19] Pinchang Zhang, Tarik Taleb, Xiaohong Jiang, and Bin Wu. "Physical Layer Authentication for Massive MIMO Systems with Hardware Impairments." *IEEE Transactions on Wireless Communications*, pp. 1–1, 2019.

[ZWX20] Hao Zhang, Yan Wang, Lingwei Xu, T. Aaron Gulliver, and Conghui Cao. "Automatic Modulation Classification Using a Deep Multi-Stream Neural Network." *IEEE Access*, **8**:43888–43897, 2020.

[ZZ17] Y. Zeng and R. Zhang. "Energy-Efficient UAV Communication With Trajectory Optimization." *IEEE Transactions on Wireless Communications*, **16**(6):3747–3760, June 2017.

[ZZC20] Shilian Zheng, Xiaoyu Zhou, Shichuan Chen, Peihan Qi, and Xiaoniu Yang. "DemodNet: Learning Soft Demodulation from Hard Information Using Convolutional Neural Network." *arXiv:2011.11337 [eess]*, November 2020.