Flying Among Stars: Jamming-resilient Channel Selection for UAVs through Aerial Constellations

Guillem Reus-Muns, Mithun Diddi, Chetna Singhal, Hanumant Singh and Kaushik Chowdhury

Abstract—Wireless communication between an unmanned aerial vehicle (UAV) and the ground base station is susceptible to adversarial jamming. In such situations, it is important for the UAV to indicate a new channel to the BS. This paper describes a method of creating spatial codes that map the chosen channel to the location of the UAVs in space, wherein the latter physically traverses the space from a given so called "constellation points" to another. These points create patterns in the sky, analogous to modulation constellations in classical wireless communications, and are detected at the BS through a millimeter-wave radar sensor. A constellation point represents a distinct n-bit field mapped to a specific channel, allowing simultaneous frequency switching at both ends without any RF transmissions. The main contributions of this paper are: (i) We conduct experimental studies to demonstrate how such constellations may be formed using COTS UAVs and mmWave sensors, (ii) We develop a theoretical framework that maps a desired constellation design to error and band switching time, including multi-user scenario-specific challenges, (iii) We compare our approach against current FHSS technology and (iv) We experimentally demonstrate jamming resilient communications and validate system goodput for links formed by UAV-mounted software defined radios.

Index Terms—UAV networks, Spectrum Access, Remote Sensing

1 Introduction

Unmanned aerial vehicles (UAVs) are utilized for military operations, surveillance, disaster management, telecommunications, monitoring, and cargo delivery [1]. All such roles require continuous control, navigation, communication and autonomy [2], requiring robust links between the UAV itself and the ground base station (BS) [3, 4] or between distributed UAVs [5]. The degradation in wireless links caused by adversarial actions, such as jamming or interference, has been widely studied for diverse applications, such as WSN [6] or wireless charging [7]. Moreover, UAVs have also been proposed as a solution for many challenges that cannot be solved by a single entity, i.e., natural disaster management, cellular network offloading, distributed aerial networks and Internet of Things (IoT). In such scenarios, cooperation among groups of UAVs or *swarms* working jointly towards a common goal has been widely studied for different applications in recent years [8, 9, 10, 11, 12].

•**Overview of the approach:** Fig. 1 shows a sample scenario where the RF link on channel X between the UAV and the BS is severed due to a jamming attack. The UAV selects a new channel Y for continuing the communication, but is unable to let the BS know of its choice owing to the active jammer. So, it uses an out-of-band control signaling method involving relaying channel information by moving between different spatial locations. Notice that X and Y could represent any available transmission band that the hardware of both the UAV and the BS could support (i.e. sub-6GHz, mmWave, etc), making this solution applicable to broadband jammers working in a certain band. Information conveying modulation constellations are used in classical wireless communications, and our approach attempts to map a similar concept into the UAV scenario. This creates a low-bandwidth control channel that is resilient to the

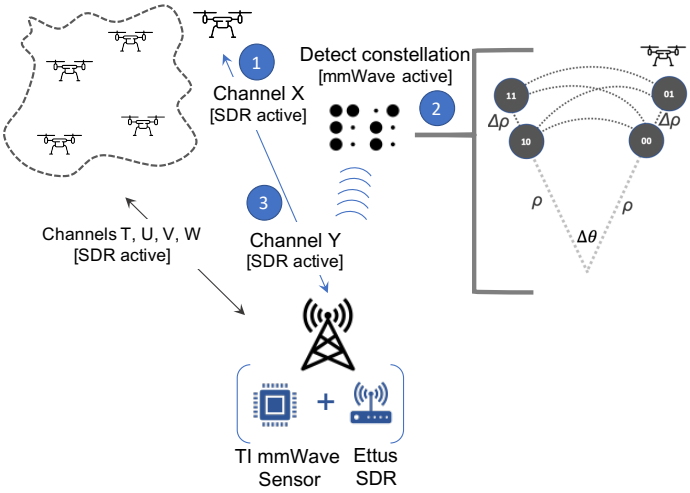


Fig. 1: UAV communicates with ground station on Channel X (step 1). When jammed, it moves through physical space to encode new channel information in a constellation, detected by mmWave sensor (step 2), expanded for 4-physical locations. The communication link switches to Channel Y free from jammer (step 3). Other UAVs communicate on different channels (T-W) and will follow the same logic (steps 1-3) if needed.

- *G. Reus-Muns, M. Diddi, H. Singh and K. Chowdhury are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115.*
  E-mail: {greusmuns, mdiddi}@coe.neu.edu, ha.singh@northeastern.edu, krc@ece.neu.edu
- *C. Singhal is with the Department of Electronics and Electrical Communication Engineering, Indian Institute of Technology Kharagpur, Kharagpur 721302, India.*
  E-mail: chetna@ece.iitkgp.ernet.in

ongoing jamming attack. Note that this approach would remain secure against a jammer equipped with its own localization technology since the location-channel mapping would be unknown on its side. Our approach relies on accurate localization of the UAV in 2-D space (in fact, any imprecision results in symbol error at the BS). While sensing-aided communications system have been explored in other works [13], for this paper we choose a single-chip Frequency-Modulated Continuous Wave (FMCW) mmWave radar.

•**Research challenges:** The idea of using spatial constellations raises many unique research challenges at the intersection of wireless communication and robotics. Firstly, based on an experimental study using a COTS mmWave radar TI IWR1642, we identify the regions where the sensor accuracy drops. This results is generating non-intuitive and irregular shapes for the resulting constellation. For instance, in Fig. 1, a regular QPSK modulation used in classical RF would have its points at the four vertices of a square, whereas our approach traces arcs in the sky for the same points. We answer the fundamental question of how these physical constellations scale and what forms they take as the number of bits required to represent additional information also changes.

With the available degrees of spatial freedom, we must also determine the separation between points, defined by Δ_ρ-$\Delta\theta$, which represent the symbol spacing between any consecutive constellation points along the ρ or θ polar coordinates axis, respectively. The need of using polar coordinates is explained later in this paper. Moreover, the problem of inter-point spacing has many non-intuitive elements. Since the UAV must physically move from one point to another, the separation between the constellation points may be minimized to reduce the travel time, and thus increase the information capacity. This is a distinction not present in classic information constellation designs, where the separation between symbols is always maximized to reduce the BER. However, simply bunching the points very close causes two problems: The natural hovering and instability during flight can move the UAV close to an incorrect location. It also decreases the ability of the ground-based mmWave sensor to resolve the UAV locations at these discrete points. Furthermore, we extend the feasibility of such spatial mapping concept to a multi-UAV scenario, where each UAV within a swarm can independently define its own constellation and choose its location based on its requirements. Thus, we address the challenge of designing adjacent constellations that maintain a minimum flight safety distance among UAVs.

Overall, designing such a physical constellation based control signaling method involves many unique interdisciplinary conditions at the intersection of robotics, communication and sensing.

In summary, the main contributions in this paper are:

1) We introduce the concept of spatial modulation constellation for UAVs and motivate its application as a method for frequency band selection for jamming resilience.
2) Through experimental traces and characterization of the mmWave sensor, we design a two-step clustering algorithm that is able to process the positional data and distinctly identify different UAVs with minimal impact of noise.
3) We extend our framework to a multi-UAV scenario while considering realistic implementation challenges. We quantify the number of UAV a single radar is able to track due to geometric constrains. In addition, we analyze the multi-target localization problem and test it with both real data as well and a large-scale simulation setup.
4) We design a constellation scheme for N=2 points in space, and propose generalization steps, by taking into account mmWave sensor performance and UAV flight limitations. The approach identifies the optimal separation distance that requires minimum movement for the UAV, while ensuring robustness in detection.
5) We compare our approach to Frequency-Hoppping Spread Spectrum (FHSS), a common anti-jamming technique currently employed in COTS UAVs.
6) We experimentally demonstrate the jamming resilience and implement our design on DJI M600 UAVs with Ettus B210 software defined radios for the case of two constellation points. Additional simulation results are provided for larger constellation sizes to demonstrate scalability.

2 Related Work

Traditional anti-jamming techniques typically consider power control or spread spectrum solutions. However, these approaches usually use the resources inefficiently and do not work well under dynamic spectrum environments [14, 15]. Frequency-Hopping Spread Spectrum has usually been proposed as an anti-jamming spread spectrum solution. For instance, [16] proposes Uncoordinated Frequency Hopping (UHF), an approach resilient to jamming that does not need agreement with the sender-receiver pair. The authors in [17] describes a method to bypass the need of pre-key establishment while [18] investigates ways for spectrum-efficient frequency hopping. In UAV networks, spread spectrum techniques have also been considered and analyzed as a jamming resilient solution [19]. Multiple works conclude that FHSS hopping sequences can be spoofed [20, 21] and their robustness is not guaranteed due to high-energy wideband jammers blocking the entire hopping space [22]. In addition, we note that all these methods require wideband spectrum, whereas our approach allows narrow-band, jamming free operation. Anti-jamming in UAV networks also face the problem of mobility which leads to highly dynamic spectrum usage for which multiple game-theoretic approaches have been proposed [23]. Moreover, some recent recent works have explored reinforcement learning techniques to counteract smart jammers in UAV networks [24, 25].

Also, sensing-aided communications are commonly proposed as a solution for challenges that classic communication systems cannot cope with. LIDAR for beam selection [26] or radar for mmWave beamtraining and tracking [13, 27] are examples proposed in the literature. Additionally, using motion and location for mapping certain modu-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

lation schemes was introduced in [28]. However, a position-based control channel fully established through accurate mmWave radar environment sensing and localization has never been proposed before.

Millimeter Wave multi-target radar detection is typically challenged by noise points and environment objects that need to be accurately filtered out in order to properly detect and localize the desired targets. Thus, clustering the UAV-located point clouds needs to be tackled, for which we propose Hi-DBSCAN$_p$. DBSCAN (Density Based Spatial Clustering of Applications With Noise) [29] is a clustering algorithm that has been modified numerous times for a wide variety of applications [30, 31, 32, 33]. We come up with our own DBSCAN modification which is able to cluster multiple UAV targets through a particular neighbor selection that considers the error distribution of the mmWave radar detection. In addition, we define a finer localization step to estimate the position of every UAV within each point cloud.

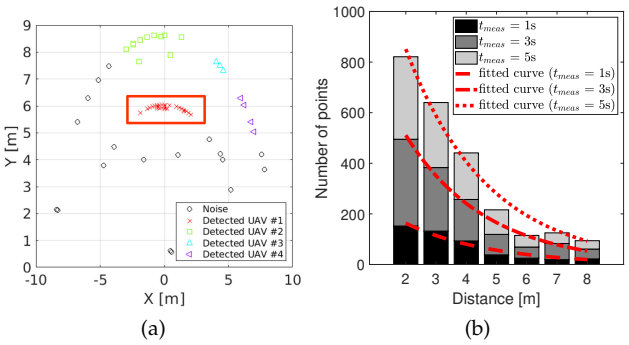
3 System Description

The system and scenario can be described in three major components (Base Station, UAV(s) and Jammer):

- Base Station: Ground-based system equipped with both a software defined radio (SDR) and a mmWave radar to establish communication and accurately localize the UAVs, respectively. It continuously tracks the UAVs location and selects a communication channel according to their position within the pre-established spatial constellations.
- UAV: Robotic system that establishes communication with the BS through its on-board SDR. Every UAV employs the spatial constellation method considering its spatial and communication capabilities. Every time a link is jammed, the UAV will hover to another constellation point to indicate the next frequency band of operation, which is detected by the BS using the mmWave radar.
- Jammer: Adversarial agent that actively tries to harm the communication link between the UAV(s) and the BS. We assume that the jammer is capable of interfering in multiple frequencies but it will not have the capability of jamming all the available bands. Notice that our channel selection approach can be effective against wideband jammers since each spatial constellation point can map to any band in the spectrum, from 6GHz to mmWave, if the hardware of a certain UAV-BS pair supports it. Additionally, the jammer can also be equipped with equivalent sensing capabilities to the BS, that enable accurate UAV localization and potentially learn the UAV position-frequency mapping. We discuss this in Sec. 7.4 and propose a solution based on pseudo-random permutations to counteract sensing-aided jammers.

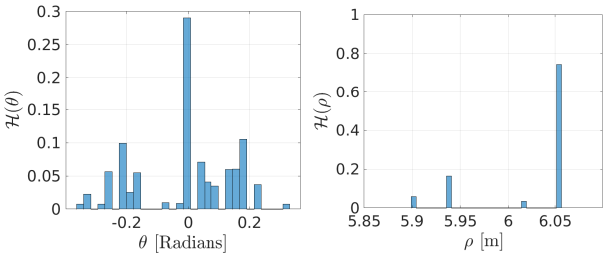
4 mmWave Sensing for UAV Localization

We use a Texas Instruments IWR1642 evaluation module, which is equipped with a 10.4x10.4mm mmWave sensor and employs FMCW radar technology. The sensor works in the 76-81GHz band with a chirp of up to 4GHz, and feeds real time location information to a laptop that analyzes the resulting point cloud. Also, we use a DJI Matrice M600 Pro UAV with access to the low level flight controller telemetry data. Furthermore, we integrated a real-time GPS kinematic solution, called DJI D-RTK, in the UAV for cm-level GPS accuracy, compared to variations in the scale of ± 1.5m in the horizontal plane otherwise.



(a) (b)

Fig. 2: (a) Measured point cloud with a UAV at position [0,6]m shows considerable noise and misclassification. However, after setting *MinPts* based on our analysis in Sec. 4.3, we successfully obtain a cluster of feasible points around the UAV's location (enclosed in the red bounding box). (b) Point cloud number of points for different t_{meas} and distance values.



(a) (b)

Fig. 3: Histogram for θ (a) and ρ (b) for the UAV point cloud in Fig. 2a. Both variables exhibit a peak which is leveraged for refining position estimation.

4.1 Static UAVs

Consider a UAV statically supported by a tripod, approximately 1m from the ground, and placed at the coordinate [0,6]m with respect to the origin [0,0] in the x-y plane, where the sensor is located. The UAV propellers are set to rotate at low rpm, and the sensor is configured to only detect moving objects. The sensor reports valid spatial coordinates (see red point cloud in Fig. 2a), but also many additional noise readings. Interestingly, the point cloud is not uniformly distributed around the target. Using polar coordinates, we see the sensor is more accurate in terms of distance from origin (say, ρ) rather than angle of the target *wrt* to origin (say, θ). Indeed, the histogram of the point cloud shown in Fig. 3 validates the comparatively greater uncertainty in localization accuracy with respect to θ over ρ.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

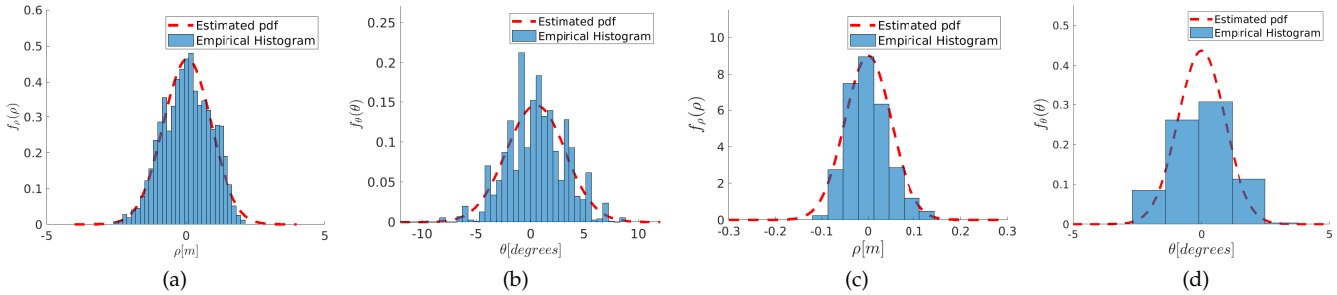


Fig. 4: mmWave radar observed UAV hovering probability density function for ρ (a) and θ (b) using standard GPS localization. Probability density function for ρ (c) and θ (d) using RTK localization.

This key insight is used for spacing the constellation points in our approach, which results in an asymmetric form of the constellation.

4.2 Hovering UAVs

When a UAV is set to operate a given point in 3-D space, it shows slight displacement over time in all three dimensions. We next determine if this unpredictable hovering motion can potentially result in the mmWave sensor mis-detecting the target constellation point. For the purpose of this work, we focus on a 2-D plane. We conducted an extensive data collection campaign using a mmWave radar while flying the UAV at coordinates with different ρ values (fixing $\theta = 0°$) from 2m to 12m in steps of 0.5m and 2 minutes per point. The same experiment is repeated for different θ coordinates, from $-32°$ to $32°$ in steps of $4°$, keeping ρ constant. The benefits of using D-RTK can be immediately seen in Fig. 4, giving an increased accuracy of both ρ and θ estimation, respectively, when the RTK is active in (c) and (d), versus using classical GPS in (a) and (b). Additionally, we leverage the fact that the distribution of the hovering error along both variables (ρ and θ) follow a Gaussian distribution in Sec. 6.

4.3 Accurate Real-time Localization

Our goal here is to accurately identify the constellation point from the data clouds obtained by the mmWave sensor. We use DBSCAN [34] as the starting point. DBSCAN groups together sets of points based on the region density, and at the same time, is able to detect outliers with low run-time overhead. In addition, it can be used in a wide range of cluster shapes (i.e. linear, concave, circular, etc.). We make two main contributions here: (i) we accurately initialize the parameters of the DBSCAN algorithm using measurement studies which are also seen in this section, and (ii) we reduce uncertainty with a novel *weighted histogram* approach to create Hi-DBSCAN, which increases accuracy over the stock algorithm.

•**DBSCAN tuning for UAV detection:** The DBSCAN algorithm has three main parameters: (i) ϵ, a measure of radius that defines the circular neighborhood around the true center of the UAV. Any measurement point within this circle is called as an ϵ-neighbor. (ii) `MinPts`, the minimum number of neighboring points a true UAV location should have in order to not be classified as noise. (iii) `Dist`, the maximum distance at which the mmWave sensor should detect measurement data. An inaccurate configuration of ϵ and `MinPts` would result in undesirable performance if not properly tunned [35]. The parameter ϵ can be trivially set from the dimensions of the UAV. Similarly, we can set the UAV flight boundaries to directly compute `Dist`. `MinPts`, on the other hand, is a function of the frame rate of the sensor (i.e., samples produced per second), the time over which the samples are collected (t_{meas}), and the total number of points obtained R, which include legitimate signal reflections from the UAV, along with noise and radar artifacts.

For a fixed frame rate, R increases linearly over time, as expected. Moreover, as we show in Fig.2b, R is also distance dependent. In order to characterize its dependency, we fit an exponential function curve to our data ($\mathcal{M} = ae^{bd}$, where d is distance and a and b are the parameters to be estimated). Fig. 2b shows the obtained fitting curves for different t_{meas} values. Thus, UAVs at different distances create different density point clouds. The lowest density areas give the lower bound on the performance of the clustering algorithm (i.e., if the UAV is detected accurately in low density areas, it is highly probable that will also be detected in more dense areas.). Thus, we define `MinPts` as $\texttt{MinPts} = \kappa\mathcal{M}(t_{meas}, \texttt{Dist})$, where κ is a scaling factor. In our experiments. We set κ to 0.5 under the assumption that at least half of the detected points are contributed by the UAV at a given location. Fig. 2a shows how properly tuning the `MinPts` parameter directly identifies the candidate points (in the red box) while *all* other external points are labelled as noise, unlike the case with a naive `MinPts` value, where the clustering predicts there are 3 extra UAVs.

•**Weighted histogram analysis:** DBSCAN outputs clusters, each containing a cloud of candidate UAV location points. However, our goal is to ultimately estimate a single location for the UAV. Thus, we define a second processing stage, which aims to refine the position estimation by performing a weighted histogram analysis. For every point cloud we make the following observation: (i) Points with higher received power are more likely to be the true location of the UAV, and (ii) Points in every cluster are more densely distributed around the true location. Thus, we define a weighted histogram, using the received power as the weights:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

$$\mathcal{H}_i = \frac{m_i}{\sum\limits_{m=1..M} p_m \times w_m} \tag{1}$$

$$m_i = \sum_{i \in B} p_i \times w_i \tag{2}$$

Where $\mathcal{H}_i$ is the weighted histogram, m_i is the sum of the weighted points at bin i, M is the number of points in the cluster, B defines the range of bin i and w_i is the weight applied to every point p_i. Consequently, the final estimated point is $\mathbf{d} = [\theta_e, \rho_e]$, where $\theta_e = \max_\theta \mathcal{H}(\theta)$ and $\rho_e = \max_\rho \mathcal{H}(\rho)$. In Fig. 3 we see how both $\mathcal{H}(\rho)$ and $\mathcal{H}(\theta)$ show a clear peak, which is known to match the UAV true location.

4.4 Revisiting DBSCAN

In the previous subsection, the design of Hi-DBSCAN as a two step clustering algorithm is explained. In this subsection, we take the legacy DBSCAN neighbor search and modify it taking into consideration the UAV radar detection characteristics. Also, we provide deeper insights on the hyper-parameter tuning within our theoretical formulation.

As mentioned in Sec. 4, the error is unevenly distributed along the polar coordinates. However, this property runs unexploited in the implementation of DBSCAN used previously, where the euclidean distance is used as the metric to group neighboring data points. Thus, we notice how the geometry of the radar-data clusters (Fig. 2a) does not match the geometry of the regions generated by an euclidean distance metric, which are circular with radius ϵ (Sec. 4.3). In order to cluster this kind of data, circular regions might be problematic because they cannot deal with the asymmetric nature of the radar error along the different polar coordinates. Thus, we propose to modify the distance metric used in DBSCAN such that it flexibly creates neighbourhoods along ρ and θ.

For instance, consider the case that DBSCAN is using the Euclidean distance to compute the neighbors of a certain data point. Then, all points at a distance $< \epsilon$ will be contained within the boundaries defined by a circular area of radius ϵ:

$$\mathcal{D}_e(\epsilon, x_c, y_c) = \{(x_i, y_i) \in \mathbb{R}^2, \forall i : \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} < \epsilon\} \tag{3}$$

where (x_c, y_c) are the Cartesian coordinates that define the center of the region. Following the same geometric interpretation, we define a new region that is bounded independently along the two polar axis ($\mathcal{P}(\epsilon_\rho, \epsilon_\theta)$), which can be defined as:

$$\mathcal{D}_p(\epsilon_\rho, \epsilon_\theta, \rho_c, \theta_c) = \{(\rho_i, \theta_i) \in \mathbb{R}^2, \forall i : |(\rho_i - \rho_c)| < \epsilon_\rho, |(\theta_i - \theta_c)| < \epsilon_\theta\} \tag{4}$$

where (ρ_c, θ_c) are the Polar coordinates that define the center of the region. Notice that while $\mathcal{D}_e$ is defined through a single parameter (ϵ), $\mathcal{D}_p$ is bounded by two different parameters ($\epsilon_\rho, \epsilon_\theta$)) along the ρ and θ coordinates respectively. From now on, we will use the notation DBSCAN$_e$ and DBSCAN$_p$ to refer to to the different DBSCAN implementations with region boundaries defined as $\mathcal{D}_e$ (3) and $\mathcal{D}_p$ (4) respectively. The prefix *Hi-* (Hi-DBSCAN$_e$) is used to indicate that the weighted histogram step in Sec. 4.3 is also used and consequently the algorithm outputs singular locations instead of point clouds.

Next, we fit the proposed modification into the rest of the already existing clustering framework. As mentioned in Sec. 4.2, each data cluster follows an independent Gaussian distribution along each Polar axis. Additionally, in Sec. 4.3, we show how `MinPts` can be expressed as a function of $\mathcal{M}(t_{meas}, \texttt{Dist})$, represented by the parameter κ. Then, we leverage these two properties to provide a theoretical formulation for showing the impact of `MinPts` (and in turn κ as of $\kappa\mathcal{M}(t_{meas}, \texttt{Dist})$), ϵ_θ and ϵ_ρ. First, considering $\texttt{MinPts} = \kappa\mathcal{M}(t_{meas}, \texttt{Dist})$, κ can be interpreted as a parametrization of the total point cloud precentage originated at a certain UAV location that will fit into the clustering region (i.e. $\mathcal{D}_p$), expressed as:

$$\kappa = \int_{-\epsilon_\rho}^{\epsilon_\rho} \int_{-\epsilon_\theta}^{\epsilon_\theta} p(\rho, \theta) d\rho d\theta \tag{5}$$

where $p(\rho, \theta)$ represents the 2D probability density distribution of the point cloud in the Polar coordinate axes. Given that ρ and θ are statistically independent and both follow a Gaussian distribution:

$$\begin{aligned} \kappa &= \int_{-\epsilon_\rho}^{\epsilon_\rho} p(\rho) d\rho \int_{-\epsilon_\theta}^{\epsilon_\theta} p(\theta) d\theta \\ &= \left(\mathcal{Q}\left(\frac{-\epsilon_\rho}{\sigma_\rho}\right) - \mathcal{Q}\left(\frac{\epsilon_\rho}{\sigma_\rho}\right)\right)\left(\mathcal{Q}\left(\frac{-\epsilon_\theta}{\sigma_\theta}\right) - \mathcal{Q}\left(\frac{\epsilon_\theta}{\sigma_\theta}\right)\right) \\ &= \left(1 - 2\mathcal{Q}\left(\frac{\epsilon_\rho}{\sigma_\rho}\right)\right)\left(1 - 2\mathcal{Q}\left(\frac{\epsilon_\theta}{\sigma_\theta}\right)\right) \end{aligned} \tag{6}$$

Then, for $\epsilon_\rho = \gamma\sigma_\rho$ and $\epsilon_\theta = \gamma\sigma_\theta$:

$$\texttt{MinPts} = (1 - 2\mathcal{Q}(\gamma))^2 \, \mathcal{M}(t_{meas}, \texttt{Dist}) \tag{7}$$

In the following sections, the performance improvement of Hi-DBSCAN$_p$ over Hi-DBSCAN$_e$ will we analyzed on both real and simulated data.

5 UAV Swarm Detection

In this section, we extend the UAV detection and data clustering approach developed in Sec. 4 to a multi-UAV scenario. While the single user scenario is developed in the previous section, we extend the spatial coding concept to a swarm of coordinated UAVs. First, we discuss the geometrical limitations due to space constraints as well as radar capabilities. Second, we study our detection algorithm applied to large-scale UAV deployments through simulation, as well as using real radar data for a smaller number of UAVs.

5.1 Single-radar maximum swarm users

Consider a UAV swarm sensed by a single BS, wherein each UAV simultaneously communicates its preferred channel by independently traveling to its matching constellation point location. The first problem that we tackle is determining the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

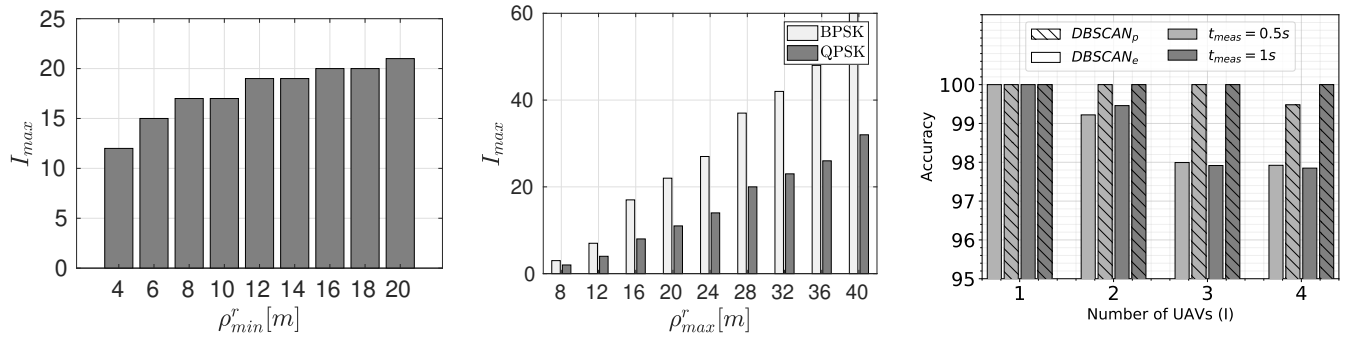


Fig. 5: (a) I_{max} increases with ρ^r_{min} due to the dependency in (10), (b) ρ^r_{max} impact on I_{max} for BPSK-like and QPSK-like constellations, (c) Clustering accuracy comparison between $DBSCAN_e$ and $DBSCAN_p$.

TABLE 1: List of parameters values

Symbol	Definition
$\Delta_{\rho/\theta}$	Inter-symbol separation along the ρ/θ axis
$\mathcal{M}$	Radar number of points estimating function
κ	Scaling Factor of $\mathcal{M}$
MinPts	DBSCAN minimum points to form a cluster
$\mathcal{H}_i$	Weighted histogram in bin *i*
m_i	Sum of the weighted points at bin *i*
B	Range of bin *i*
A_ρ	Total space occupied for a single UAV along ρ axis
A_θ	Total space occupied for a single UAV along θ axis
Δ_s	UAV safety distance
$\theta^r_{min/max}$	Minimum/maximum boundary of the radar along the θ axis
$\rho^r_{min/max}$	Minimum/maximum boundary of the radar along the ρ axis
$\mathcal{F}$	Area bounded by the field of view of the radar
$I^{max}_{\rho/\theta}$	Maximum number of UAVs with $A_{\rho/\theta}$ that can fit in $\mathcal{F}$ along the ρ/θ axis respectively
I^{max}	Total maximum number of UAVs that can fit in $\mathcal{F}$
$\mathbf{p}$	UAV location modeled as a Random Variable
$\mathbf{s}_i$	Polar coordinates for symbol *i*
$\mathcal{R}_i$	Region defined by $\mathbf{s}_i$
P^n_e	Probability of error of symbol *n*
$\mathcal{T}$	Average travel time

maximum swarm size I^{max}, which depends upon the angular range of the mmWave sensor as well as the constellation size.

We start by defining the extremities of the constellation generated for a given UAV in polar coordinates, defined in terms of A_ρ and A_θ as follows,

$$A_\rho = \rho_{max} - \rho_{min} + \Delta_{s_\rho} \tag{8}$$

$$A_\theta(\rho) = \theta_{max} - \theta_{min} + \Delta_{s_\theta}(\rho) \tag{9}$$

where $\rho_{max/min}$ and $\theta_{max/min}$ are the highest/lowest values for a given UAV constellation, along radial and angular directions, respectively. In a practical deployment, we also need to maintain a minimum distance between UAVs to ensure a safe buffer zone during flight. Thus, we define Δ_{s_θ/s_ρ} to represent this buffer space along the θ/ρ axis, respectively. We note that Δ_{s_θ} is defined as a function of ρ. This is due the fact that the minimum in-flight separation is defined in terms of absolute distance and the angular occupation is relative to the observation point. Such dependency can be expressed as:

$$\Delta_{s_\theta}(\rho) = 2\arcsin\frac{\Delta_{s_\rho}}{4\rho} \tag{10}$$

While $\rho/\theta_{max/min}$ are defined by the constellation, we assume Δ_s to be constant and independent of each UAV capabilities. Next, we define the *Field of View* (FoV) of the radar, $\mathcal{F}$, as:

$$\mathcal{F} = \begin{cases} \theta^r_{min} < \theta < \theta^r_{max} \\ \rho^r_{min} < \rho < \rho^r_{max} \end{cases} \tag{11}$$

where θ^r and ρ^r correspond to the operational boundaries of the radar along angular and radial directions, respectively. Then, assuming A_ρ and $A_\theta(\rho)$ are constant for any given member of the swarm, the maximum number of UAVs that can fit in the FoV of a sensor is:

$$I^{max}_\rho = \left\lfloor \frac{\rho^r_{max} - \rho^r_{min}}{A_\rho} \right\rfloor \tag{12}$$

$$I^{max}_\theta(\rho) = \left\lfloor \frac{\theta^r_{max} - \theta^r_{min}}{A_\theta(\rho)} \right\rfloor \tag{13}$$

If $I^{max}_\theta(\rho)$ were independent of ρ, I^{max} would simply be the product of I^{max}_ρ and I^{max}_θ. However, we first find the lower bound as the worse case $A_\theta(\rho)\Big|_{\rho^r_{min}}$. This can be obtained in closed form as:

$$I^{max}_{min} = I^{max}_\rho I^{max}_\theta(\rho^r_{min}) \tag{14}$$

Furthermore, the maximum I^{max} is obtained from:

$$I^{max} = \sum_{l=0}^{I^{max}_\rho - 1} I^{max}_\theta(\rho^r_{min} + lA_\rho) \tag{15}$$

where A_θ is computed every $(\rho^r_{min} + lA_\rho)$. In Fig. 5a we analyze the dependency of I^{max} over ρ^r_{min} for a BPSK-like constellation (2 symbols). Here, $\rho_{max} = \rho_{min} + 1.5$, $\Delta_s = 1m$ ($A_\rho = 2.5m$) and $A_\theta = 9 + \Delta_{s_\theta}(\rho)$. Although $\rho^r_{max} - \rho^r_{min}$ is kept constant, it can be observed how I_{max} increases with ρ^r_{min}. This is can be easily justified considering that Δ_{s_θ} decreases with ρ (Eq. 10) which in turns makes $A_\theta(\rho)$

Algorithm 1: Hi-DBSCAN$_p$

Input: $D, t_{meas}, fps, \epsilon_\rho, \epsilon_\theta$;
$MinPts = \kappa\mathcal{M}(t_{meas}, d_{max})$; ▷ Adaptive MinPts
$\boldsymbol{C} = \text{DBSCAN}_p(D, MinPts, \epsilon_\rho, \epsilon_\theta)$; ▷ $\boldsymbol{C}$ contains input data D clustered in N clusters
for $n \leftarrow 1$ **to** N **do**
  $D_n = D \in C_n$;
  Compute $\mathcal{H}^n_\rho(D_n)$;
  $\rho^n_e = \max(\mathcal{H}^n_\rho)$;
  Compute $\mathcal{H}^n_\theta(D_n)$;
  $\theta^n_e = \max(\mathcal{H}^n_\theta)$
Return: $\boldsymbol{\rho_e}, \boldsymbol{\theta_e}$; ▷ $\boldsymbol{\rho_e}$, $\boldsymbol{\theta_e}$ are vectors containing the N estimated polar coordinates for each UAV location.

decrease and both $I^{max}_\theta(\rho)$ and I^{max} increase (Eq. 9, 13, 15). For comparison purposes, in Fig. 5b we show I_{max} for two different constellation sizes and different radar field of view sizes ($\mathcal{F}$) along the ρ axis, where ρ^r_{min} is fixed and ρ^r_{max} takes different values.

5.2 Multi-UAV clustering

While the above section describes the theoretical maximum number of possible UAVs that can be accommodated in the FoV of the mmWave radar, there is also the additional concern of being able to isolate point clouds originating from multiple different UAV targets.

To address this, we examine the performance of the clustering algorithm in Sec. 4.4. Algorithm. 1 shows the pseudo-code for Hi-DBSCAN$_p$ as it applied to multi-UAV case. Notice that Hi-DBSCAN$_e$ would run the same logic, with DBSCAN$_e$ instead of DBSCAN$_p$. First, we create a dataset of up to 4 flying UAVs to empirically demonstrate successful detection and localization of multiple UAVs. Second, we compare the performance of the two different Hi-DBSCAN implementations, quantifying the meaningful performance improvement introduced by the modification described in Sec. 4.4. Finally, we use the formulation in the previous subsection to test the multi-UAV clustering large-scale performance through simulation.

5.2.1 Experiments

Our validation study consists of flying 1-4 DJI M100 UAVs while simultaneously collecting data with the TI IWR1642 mmWave sensor on the ground. To ensure in-flight safety, we set the UAVs at a minimum flying distance of 2m. Then, we use Hi-DBSCAN$_{e/p}$ to post process the data and compare its performance for different distance metrics (Eq. (3)-(4)) and number of UAVs. Also, we show how our adaptive method to set `MinPts` detailed in Sec.4.4 works efficiently in case of multiple UAVs.

In Fig. 5c, we show the statistical accuracy of Hi-DBSCAN$_{e/p}$ for generating distinctly separable point-clouds and then accurately mapping each cloud to a specific UAV location. We run this analysis for two t_{meas} values and set `MinPts` as explained in previous sections. For each case, we compare the number of obtained clusters with the ground truth (number of UAVs flying). It can be observed how the accuracy obtained with Hi-DBSCAN$_p$ is superior to Hi-DBSCAN$_e$ for all different scenarios. As mentioned in Sec. 4.4, this is due to the modification on the region definition (Eq. 4) taking into consideration the radar error distribution enhances the UAV clustering performance. Hi-DBSCAN$_p$ achieves 100% accuracy with $t_{meas} = 1s$ for any I. Reducing t_{meas} reduces the accuracy since the accumulated samples that pass through the clustering algorithm are reduced, which increases uncertainty. However, for Hi-DBSCAN$_p$ and $t_{meas} = 0.5s$ we only see a reduction of $\approx 0.5\%$ in the I=4 case. The presented results help verify the proposed theoretical formulation introduced in Sec.4.4. Thus, this experiment on a real deployment with varying number of UAVs shows how Hi-DBSCAN$_p$ permits extending the spatial constellation to a multi-UAV scenario. We test Hi-DBSCAN$_{e/p}$ performance on larger scale deployments through simulation in Sec. 5.2.2.

TABLE 2: Simulation parameters

Parameters	Value
Δ_s	1m
$\rho_{max} - \rho_{min}$	1m
$\theta_{max} - \theta_{min}$	18^o
θ^r_{max}	35^o
θ^r_{min}	-35^o
ρmax^r	$4m$
ρmin^r	$15m$
A_ρ	2m
A_θ	$18^o + 2 * \arcsin \frac{\Delta_s}{4*\rho}$

•**Multi-Target Hovering Probability Distribution:** As mentioned in Sec. 4.2, the spatial probability distribution of the UAV hovering motion follows a Gaussian distribution. We leverage this in Sec. 5.2.2 for simulation, as well as in Sec. 6, for a probability error analysis. We next extend the previous result for a single target in Fig. 6a, where we plot the obtained PDF along the ρ axis for three UAVs flying simultaneously. We observe a Gaussian fit for every empirical distribution for this multi-UAV scenario as well. We observe similar results along the θ axis. This suggests that our signal-target analysis can be extended for a multi-UAV scenario, where independent Gaussian distributions are obtained for every individual target.

5.2.2 Simulation

We performed limited real-world testing (using 4 UAVs) on the performance of Hi-DBSCAN earlier. In this section we extend the radar detection and UAV hovering analysis with real hardware in Sec. 4 and 5, respectively, to create a UAV detection simulator. First, we define the radar area of operation $\mathcal{F}$ (11) given by the FoV. Next, we compute I^{max} (15) given the area each UAV will occupy given a spatial constellation. As mentioned in Sec. 5.1, we assume A_ρ and $A_\theta(\rho)$ to be constant for any UAV in the swarm. In addition, the I^{max} UAVs are placed along $\mathcal{F}$ respecting the $A_\rho / A_\theta(\rho)$ spacing limitations. In order to simulate the radar-like data centered at each UAV location, we take the statistics in Fig. 4 to generate multiple point clouds.

Moreover, we analyze the impact of $\sigma_{\rho/\theta}$ on Hi-DBSCAN accuracy (Fig. 6b-6c) using the simulation parameters listed

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

in Table. 2. We observe how the accuracy only decreases for considerably high $\sigma_{\rho/\theta}$ values. When compared with the results obtained in the characterization in Fig. 4c-4d, ($\sigma_\rho \approx 0.05m$; $\sigma_\theta \approx 0.9^o$) we see how Hi-DBSCAN can cope with instabilities found in real deployments. Additionally, we also compare the performance of Hi-DBSCAN$_e$ and Hi-DBSCAN$_p$. Again, we observe how Hi-DBSCAN$_p$ is more robust to high sparse clusters, ($\sigma_\rho > 0.55$; $\sigma_\theta > 4.5$), whereas Hi-DBSCAN$_e$'s accuracy reduces drastically.

6 Constellation Location Error Probability

The unpredictable hovering introduces errors in the localization process. We model the location of each UAV as a random variable defined as $\mathbf{p} = (\theta, \rho)$ using the polar coordinate system. As discussed in Sec. 4.2, the UAV movement along each axis follows a Gaussian distribution. In addition, in Sec. 5 we showed how this property also applies for a multi-UAV scenario. Assuming statistical independence, we next discuss the means and variances of these variables: Let $\mathbf{s}_i = (s_{\theta_i}, s_{\rho_i})$ be the polar coordinates for a given symbol i in a 2D plane. In an ideal case, the UAV location exactly overlaps with a given symbol location, or at least, exhibits a mean location $\boldsymbol{\mu}_i$ equal to the constellation point, i.e., $\boldsymbol{\mu}_i = \mathbf{s}_i$. Furthermore, let $\boldsymbol{\sigma}_i = (\sigma_{\theta_i}, \sigma_{\rho_i})$ denote the vector of standard deviations that defines the precision with which a given UAV hovers around certain symbol coordinates. The lower the $\boldsymbol{\sigma}_i$, the more stable the UAV is. Since the UAV's position $\mathbf{p}_i$ on any axis is assumed to be an independent Gaussian random variable, while transmitting symbol i, this can be expressed as:

$$\mathbf{p}_i = \mathcal{N}(\boldsymbol{s}_i, \boldsymbol{\sigma}_i\mathcal{I}), \qquad \forall i \in \{1, 2, ..., N\} \tag{16}$$

where N stands for the number of different symbols in the constellation. There is a symbol error anytime the hovering displaces the UAV out of its feasible symbol region. Without loss of generality, each symbol region $\mathcal{R}_i$ is defined as:

$$\mathcal{R}_i = \begin{cases} \alpha_i^l < \theta < \alpha_i^u \\ \beta_i^l < \rho < \beta_i^u \end{cases} \tag{17}$$

where α and β define the bounds on each of the axis and the l and u respectively stand for the *lower* and *upper* bound that define the $\mathcal{R}_i$ limits. While we provide a detailed analysis for the case of N=$\{2\}$ next, higher order constellation sizes are not included due to space constraints, though they follow very similar steps. As the constellation design is not only dependent on N but Δ_ρ and Δ_θ, a general symbol error probability is hard to derive. However, in Sec. 6.2 we develop a generic framework that accounts for the number of neighbors of each symbol in order to compute the average probability of error.

6.1 Analysis for N=2

The constellation can have two possible configurations, where both symbols are placed either along the ρ or the θ axis. In order to avoid redundancy, we derive the expression for the θ case only, where $s_\theta = \{-\frac{\Delta_\theta}{2}, \frac{\Delta_\theta}{2}\}$. Here, we define the probability of error as:

$$P_e = \sum_{i=\{1,2\}} P(\mathbf{p}_i \notin \mathcal{R}_i) = \sum_{i=\{1,2\}} P(\beta_i^l > \theta_i > \beta_i^u)P(s_i) \tag{18}$$

$$= \sum_{i=\{1,2\}} P(\beta_i^l > s_{\theta_i} + \varepsilon_{\theta_i} > \beta_i^u)P(s_i) \tag{19}$$

We reformulate $\mathbf{p}_i$ as the addition of a deterministic value ($\mathbf{s}_i$) and a random component ($\boldsymbol{\varepsilon} = \mathcal{N}(0, \boldsymbol{\sigma}_i\mathcal{I})$). As the symbols are equally probable with same error probabilities:

$$P_e = \frac{1}{2}2P(\varepsilon_{\theta_i} > \beta_i^u - s_{\theta_i}) = \mathcal{Q}\left(\frac{\beta_1^l - s_{\theta_1}}{\sigma_{\theta_1}}\right) \tag{20}$$

Where $\mathcal{Q}(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp\frac{-t^2}{2} dt$ is adopted to simplify the expression. Finally, considering s as the constellation point centered at $\theta = 0$, then $\beta_1^l = 0$, $P_e = \mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_{\theta_1}}\right)$.

Similarly, if both symbols were placed along the ρ axis $P_e = \mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_{\rho_1}}\right)$. This result matches with the BPSK probability of error through an AWGN channel, as expected.

6.2 Generalized formulation

We derived the error probability for the constellation size (N=2) in Sec. 6.1. We next tabulate the error probability as a function of the number of neighbors a given constellation point has along ρ axis (n_ρ) and θ axis (n_θ). Also, a given symbol is considered as the neighbor of another one if it is placed at a distance Δ_ρ - Δ_θ along the ρ - θ axis respectively. We define the probability of error in terms of n_ρ and n_θ because this allows us to derive an expression for every single symbol in the constellation and then compute total P_e as the average. Table. 3 shows $P_e(n_\theta, n_\rho)$ for every possible (n_θ, n_ρ) pair.

n_θ	n_ρ	$P_e(n_\theta, n_\rho)$
2	2	$1 - \left\{\left[1 - 2\mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_{\theta_1}}\right)\right]\left[1 - 2\mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_{\rho_1}}\right)\right]\right\}$
2	1	$1 - \left\{\left[1 - 2\mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_{\theta_1}}\right)\right]\left[1 - \mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_{\rho_1}}\right)\right]\right\}$
2	0	$2\mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_{\theta_1}}\right)$
1	2	$1 - \left\{\left[1 - \mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_{\rho_1}}\right)\right]\left[1 - 2\mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_{\theta_1}}\right)\right]\right\}$
1	1	$1 - \left\{\left[1 - \mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_\theta}\right)\right]\left[1 - \mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_\rho}\right)\right]\right\}$
1	0	$\mathcal{Q}\left(\frac{\Delta_\theta/2}{\sigma_{\theta_1}}\right)$
0	2	$2\mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_{\rho_1}}\right)$
0	1	$\mathcal{Q}\left(\frac{\Delta_\rho/2}{\sigma_{\rho_1}}\right)$

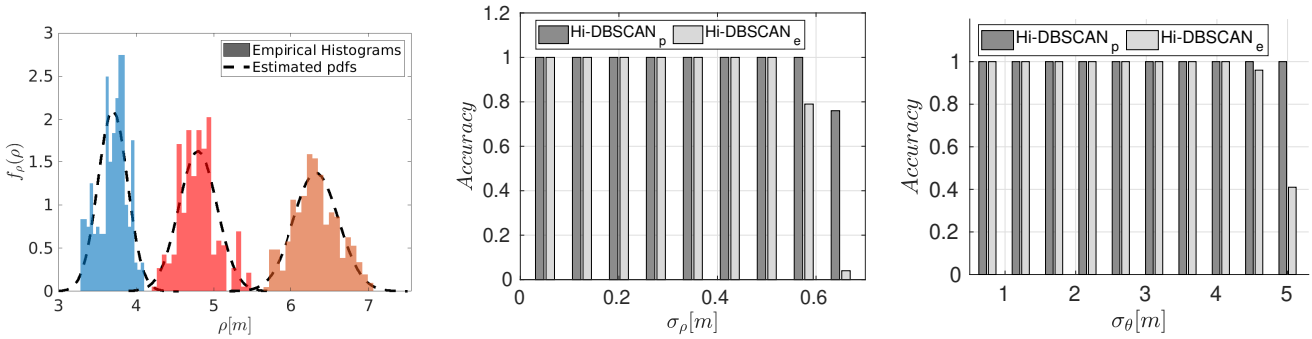
TABLE 3: P_e expressions for different (n_θ, n_ρ) pairs.

Given a constellation setup and the probability of error for every symbol, the total probability of error for any constellation can be expressed as:

$$P_e = \frac{1}{N}\sum_{n=1}^{N} P_e^n(n_\theta, n_\rho) \tag{21}$$

where P_e^n represents the P_e for symbol n. Notice how every expression in Table. 3 depends explicitly on Δ_ρ and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing



(a) (b) (c)

Fig. 6: Multi-UAV probability density functions (I=3) (a). Hi-DBSCAN accuracy under different uncertainty regimes along the ρ-axis (b) and the θ-axis (c).

Δ_θ. This will be relevant in the following section, where the constellation design is explained. As mentioned earlier, we model the location of a UAV in a multi-UAV scenario as a random variable that follows an independent random distribution. Our analysis is applicable to any multi-UAV scenario, where every UAV has an independent value of P_e based on its own hovering performance.

7 Creating Efficient Constellations

In this section, we describe in detail how to design the spatial constellations for a given set of I UAVs. The optimal set of constellation coordinates for each UAV is composed of a set of N points (see Fig. 13), which we refer to as $\mathbf{p}_i^*$, $1 \leq i \leq I$. We aim to minimize the average travel time ($\mathcal{T}$), which occurs when the inter-symbol distance is also minimized. The parameters Δ_ρ and Δ_θ are selected from the analysis in Sec. 6 using the generalization of error probability obtained in Eq. 21 and Table. 3. As mentioned above, the error directly depends on Δ_ρ and Δ_θ. Then, both of these parameters are set so that P_e is kept below a certain threshold ξ. Note that different UAVs may have different flying performances, defined by the Δ_{ρ_i}-Δ_{θ_i} pair, as well as different probability of error demands, defined by ξ_i. Thus, the problem can be formulated as:

$$\min_{\mathbf{P}^*} \quad \sum_{i=1}^{I} \mathcal{T}_i(N, \Delta_{\rho_i}, \Delta_{\theta_i}) \quad \forall I \leq I^{max} \tag{22a}$$

$$\text{such that: } P_e \leq \xi_i, 1 \leq i \leq I, \tag{22b}$$

$$\Gamma = 1 \tag{22c}$$

$$\text{where, } \Gamma = \left\{ \begin{array}{ll} 1, & \text{if } d_{i,j}^{min} > \Delta_s \text{ and } (\rho_i, \theta_i) \in \mathcal{F} \\ 0, & \text{otherwise} \end{array} \right\}$$

$$\text{with, } i \neq j, 1 \leq i \leq I, 1 \leq j \leq I.$$

where $\mathbf{P}^* = \{\mathbf{p_1^*}, \mathbf{p_2^*}, ..., \mathbf{p_I^*}\}$ is the set of optimal constellations that minimize the travelling time for each UAV, represented by $\mathcal{T}_i$. The constraint (22b) ensures that the symbol error rate for every constellation is upper bounded by the threshold ξ_i. The constraint (22c), where $d_{i,j}^{min}$ represents the shortest distance between any point in constellation i and j, prevents neighboring UAV constellations to get closer than Δ_s, as well as ensuring that every constellation point is within the radar FoV region $\mathcal{F}$.

From Sec. 4.1, the mmWave sensing accuracy is not equally distributed along both the polar coordinate axes. For instance, considering Fig. 13, the optimal constellation for N=8, uses 4 values in the ρ axis while only 2 along θ axis. If we analyze this asymptotically, the optimal design may require placing all the N points along one of the axis. Then, we define a grid of L=N^2 elements which represent the solution space for our problem. In a multi-UAV scenario, if any of the L locations overlaps with another constellation from another UAV, those will not be considered in order to avoid potential collisions, as defined in ((22c)). For a given starting point ($\boldsymbol{p_c} = [\theta_c, \rho_c]$), the grid range is defined as $\{[\theta_c - \frac{\Delta_\theta(N-1)}{2}, \theta_c + \frac{\Delta_\theta(N-1)}{2}], [\rho_c, \rho_c + \Delta_\rho(N-1)]\}$ (see Fig. 13). The optimal constellation ($\mathbf{p}^*$) is the set of N elements out of L for which its average distance is minimized, i.e., from (22a). To solve this, we propose (i) an exhaustive search within a reduced constellation candidates, where finding the reduced set is equivalent to solving the combination sum algorithm, and (ii) a reduced complexity heuristic algorithm for faster in-flight constellation calculation. Moreover, we show how the approach of exploring only the reduced set narrows the complexity drastically in comparison to the brute-force exploration. Additionally, we compare the performance of our heuristics to the exhaustive search approach and analyze their complexities.

7.1 Exhaustive exploration on feasible constellations

A brute-force solution that would find the best constellation after considering all the options is not feasible due to its complexity. A total of $\binom{N^2}{N}$ different constellations would need to be considered, which represents an intractable solution. In order to reduce the solution complexity, only the constellations with symbols spaced by $\Delta_{\rho/\theta}$ will be considered. For example, for a constellation with 4 symbols ($N = 4$), a brute-force method would compare a total of $\binom{16}{4} = 1820$ possible constellation combinations. Instead, we reduce the constellation search space to a feasible set consisting of only 5 unique constellations: (4, 0),(3, 1),(2, 2),(2, 1, 1),(1, 1, 1, 1). Every constellation is expressed as the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

number of constellation points within each row, where all constellations point have the same ρ value, of the $N \times N$ grid. For instance, a 2-by-2 QPSK-like constellation with constellation point separation $\Delta_{\rho/\theta}$ will be referred as $(2, 2)$. Notice that all the excluded constellation candidates will have at least a pair of symbols separated by more than the minimum symbol separation $\Delta_{\rho/\theta}$, increasing the average constellation distance. For instance, a constellation with its 4 symbols placed at every vertex of the $N \times N$ grid would never provide the shortest average distance constellation.

For generalization purposes, we define the set of constellation candidates of size N with minimum inter-symbol distance as C_N. Notice that in the previous example, the elements in the described set represent all the possible ways to express $N = 4$ as a sum of positive integers. This is expected since the sum of constellation points placed at every row should be equal to the constellation size N. Thus, the problem of finding C_N is equivalent to finding all the possible ways to arrange the N constellations points within the N rows of an $N \times N$ grid. This is analogous to solving the combination sum problem with a target N. The combination sum problem finds all the unique combinations of non-ascending positive integers that sum to a certain target value [36]. The total number of constellations of size N ($|C_N|$) can be counted as:

$$|C_N| = \sum_{i=1}^{N} p_i(N) \tag{23}$$

$$p_i(N) = p_i(N - i) + p_{i-1}(N - 1) \tag{24}$$

where $p_i(N)$ is the number of ways in which an integer N can be expressed as the sum of exactly i positive integers. $p_i(N)$ does not present a close form formula and is typically computed following the recursion in Eq. 24

Then, given that computing the average distance for every single constellation requires $\frac{N^2-N}{2}$ operations, the overall complexity becomes $\mathcal{O}(N^2 \times |C_N|)$. As mentioned before, $|C_N|$ does not present a closed form formula [37]. However, in Fig. 7b we show that for the range of interest ($N \leq 64$), the complexity fits a polynomial fit of degree 4 (N^4). Then, the complexity can be approximated to $\mathcal{O}(L^3)$. In contrast, notice that the brute-force search presents a complexity of $\mathcal{O}(L \times \binom{L}{N})$. In Fig. 7a, we numerically show a comparison for the number of cases that a brute-force search and our combination sum approach require. It can be observed that the number of cases considered is multiple orders of magnitude.

7.2 Heuristic exploration

In order to further reduce the complexity of the exhaustive search presented above, we present an heuristic algorithm that works under the assumption that the inter-symbol distance is reduced if the set of points closer to $\boldsymbol{p_c}$ are picked. Then, it will only be required to compute the distance from $\boldsymbol{p_c}$ to the L points in the $N \times N$ grid to select the N closest ones. As only one set of distances needs to be calculated, the complexity is reduced to L operations ($\mathcal{O}(L)$). Thus, the algorithm complexity has been reduced from ($\mathcal{O}(L^3)$) to ($\mathcal{O}(L)$). We complete the analysis in Sec. 11 by comparing the average distance constellation obtained through both methods.

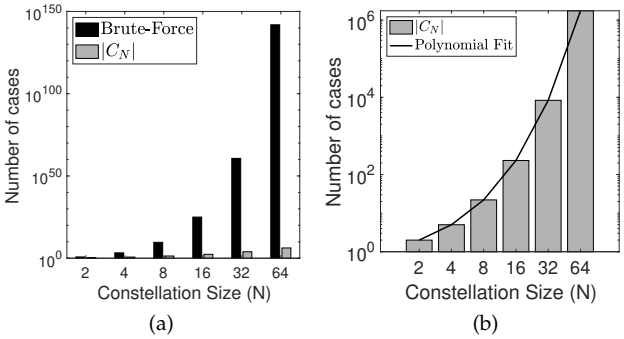


Fig. 7: Number of cases comparison for different constellation search algorithms (a). Although $|C_N|$ does not have a closed form solution, the numerical values for the range of interests in this paper fits a polynomial curve of order 4.

7.3 Multi-UAV constellations

The complexity of computing multiple constellations will depend on the number of UAVs as well as each constellation size. Assuming that all UAVs employ the same constellation size, the complexity for the combinatorial and heuristic exploration would be $\mathcal{O}(IL^3)$ and $\mathcal{O}(IL)$ respectively. However, the formulation throughput this paper has been defined so that each UAV can accommodate different constellation size independently. In that case, the complexity will be mostly impacted by higher order modulations (L_{hm}). Then, the complexity for the exhaustive exploration can be expressed as $\mathcal{O}(I_{hm}L_{hm}^3)$. In contrast, the heuristic complexity follows a $\mathcal{O}(I_{hm}L_{hm})$ complexity.

7.4 Pseudo-Random Symbol Mapping

Every spatial constellation point is directly mapped into a specific frequency band. We formally define such mapping as a bijective function $\mathbf{M} : \{0,1\}^M \longrightarrow F$, which takes as an input an M-bit representation of a constellation symbol (P) and returns a certain frequency band from the set F, that has a total of N values, one for each center frequency. While the mapping $\mathbf{M}$ is assumed to be known at both ends of the communication link, it must be kept private to the jammer at all times. Otherwise, a jammer equipped with equivalent sensing capabilities, could easily infer the future transmission band and tune its disruptive actions accordingly. Similarly, even if $\mathbf{M}$ was not known a priori by the attacker, the linear mapping could easily be learned by the jammer if it were kept constant in time. Thus, an scheme that will update $\mathbf{M}$ in time on both the BS and UAV seems necessary to stop jammers from being able to learn a constant mapping between the constellation points and available channels. To do so, we propose a method that periodically pseudo-randomly permutates the original linear mapping $\mathbf{M}$. In particular, we extend from prior works in block-ciphers, which generate pseudo-random permutations uniquely identifiable through a fixed key k [38, 39]. We formally introduce such mapping as:

$$\mathbf{M} : \{0,1\}^M \times \{0,1\}^k \longrightarrow F \quad F = \{f_1, ..., f_N\}; f_n \in \mathbb{R} \tag{25}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

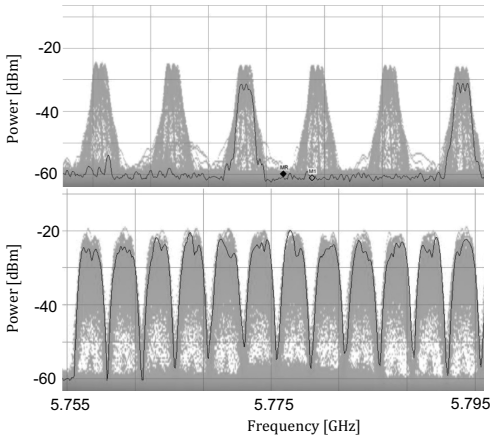


Fig. 8: FHSS pattern captured from a DJI M600 RC (top) and a DJI Mavic Pro 2 (bottom).

where $\mathbf{M}$ is a function that takes an M-bit representation of a constellation point (P) and a k-bit key (K). For every key $K \in \{0,1\}^k$, there is a mapping $\mathbf{M}_K : \{0,1\}^M \longrightarrow F$ that is a pseudo-random permutation of any other mapping with a different key K. The key K will incrementally be updated on both ends of the communication link every time the UAV needs to move into another constellation point ($K_{t+1} = K_t + 1$). Thus, every time the BS detects a new spatial constellation point, the key K will be incremented by a unitary value. Notice that through this method, the mapping $\mathbf{M}$ will remain unknown to the jammer and the ability to sense a certain UAV location will provide no additional information.

8 UAV Transmission Signal Analysis

For completeness of this work, we analyze the RF technology employed on commercial devices in both uplink and downlink. To do so, we used commercial DJI projects as well as a Tektronix RSA507A spectrum analyzer for proper data visualization.

8.1 Uplink

Commercial devices employ FHSS as a robust RC to UAV (uplink) communication solution. For instance, in Fig. 8 we show the RF spectrum of the captured transmissions from two different DJI controllers in the 5.725-5.825GHz band. The RC uses a spread spectrum of 70MHz with hops of 5MHz (M600) and 2MHz (Mavic Pro 2). The 2.4GHz band is also utilized, where a bandwidth of up to 80MHz is used and a minimum hopping distance of 2MHz between subcarriers. This numerical values will be of relevance for the comparison conducted in the following subsections.

8.2 Downlink

UAVs typically communicate periodically with the RC in order to report telemetry data or battery level among others. In the most recent years, most UAVs also dispose of video streaming capabilities which require high throughput and low latency links. Based on the application requirements, such transmissions may vary in bandwidth and

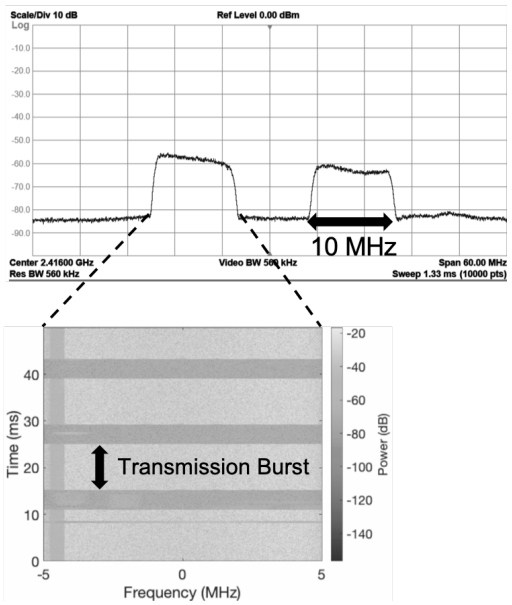


Fig. 9: Downlink transmission of two DJI M600 UAVs (top). Spectrogram visualization of the downlink bursty transmissions (bottom).

transmission periodicity. In this subsection, we analyze the downlink of a DJI M600. These devices transmit in the ISM band to communicate with the RC. More specifically, the 2.401GHz-2.481GHz band is divided into eight 10MHz channels, where each UAVs picks one of them based on their proprietary interference analysis protocol. In Fig. 9, we show a spectrum visualization of two DJI 600 transmissions. It can be observed that every different UAV selects a different transmission band to avoid mutual interference. Moreover, we also observed that the UAV accesses the medium at a fixed periodicity (≈50Hz), as we show in the spectrogram in Fig. 9.

9 Proposed Anti-jamming Approach Vs. FHSS

In this section, we compare our frequency switching approach to FHSS, which is commonly presented as a jamming resilient solution, as discussed in Sec. 8.1. Also, we analyze under what scenarios our method is superior to the current technology used in COTS devices.

9.1 Latency formulation

We analytically show how our approach of switching the transmission to another band results in reduced overhead compared to FHSS under partial jamming conditions. First, we define R_{SC} as the bitrate of a single carrier transmission with bandwidth B_{SC}. R_{SC} represents the achievable bit rate for every available channel that is mapped to a certain constellation point. Similarly, we define the FHSS bitrate R_{FHSS} as a fraction of R_{SC}:

$$R_{\text{FHSS}} = R_{SC} \frac{B_{\text{FHSS}}}{B_{SC}} \tag{26}$$

where B_{SC} is the bandwidth used for a single carrier transmission and B_{FHSS} is the transmission bandwidth for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

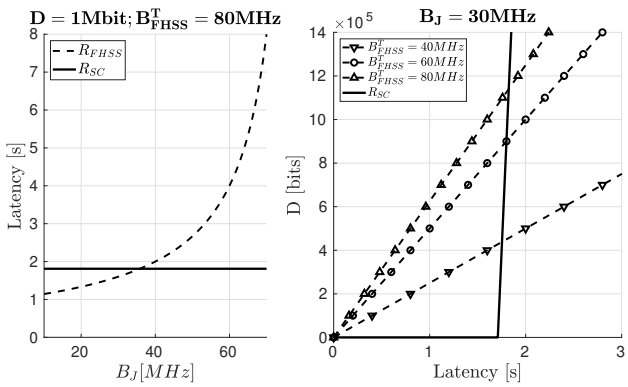


Fig. 10: Latency to transmit a stream of 1 Mbit of data versus B_J (a). Data transmitted over time for different B^T_{FHSS} (b). For this simulation, we set R_{SC}=10Mbps; B_{SC}=10MHz; B_{FHSS}=1MHz.

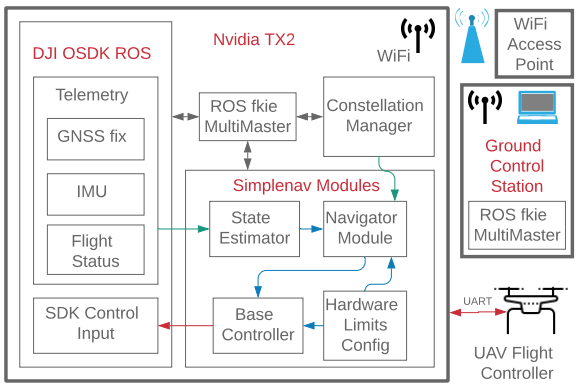


Fig. 11: UAV System Architecture

a single sub-carrier of a FHSS system. Notice that B_{FHSS} defines the frequency separation between to subsequent FHSS hops and not the overall used spread spectrum, which we define as B^T_{FHSS}. Next, consider the attack model in Fig. 1, where the UAV and the BS switch to another channel if the ongoing communication link is jammed. Here, we explore whether using an spread spectrum technique such as FHSS is provides better performance compared to the proposed approach. In the latter case, the transmission rate of the jammed link (R^J_{SC}) reduces to zero, whereas the FHSS will see its throughput reduced based on what percentage of B_{FHSS} is under severe interference. The effective FHSS rate (R^J_{FHSS}) under an interferer jammer with bandwidth B_J can be expressed as:

$$R^J_{\text{FHSS}} = R_{SC}\frac{B_{\text{FHSS}}}{B_{SC}}\left(1-\frac{B_J}{B^T_{\text{FHSS}}}\right) \tag{27}$$

Next, we formulate the latency incurred to transmit a certain bitlength in both approaches. In addition, we find a threshold in terms of $\mathcal{T}$ for which our approach is superior over FHSS. Under active jamming conditions, consider a data packet with length D. Then, the latency for each method is expressed as:

$$L_{SC} = \mathcal{T} + \frac{D}{R_{SC}} \tag{28}$$

$$L_{\text{FHSS}} = \frac{D}{R^J_{\text{FHSS}}} \tag{29}$$

where the first term in L_{SC} represents the overhead introduced by the UAV travelling from one constellation point to another and the second term is the transmission time at a rate of R_{SC}. L_{FHSS} shows the transmission at the FHSS jamming-reduced rate (Eq. 27). Then, we analyze under what conditions our frequency switching approach provides a better response the the jamming attack ($L_{SC} < L_{\text{FHSS}}$). From (26-29), we obtain the following inequality:

$$\mathcal{T} \leq \frac{D}{R_{SC}}\left(\frac{B_{SC}}{B_{\text{FHSS}}\left(1-\frac{B_J}{B^T_{\text{FHSS}}}\right)}-1\right) \tag{30}$$

which indicates the condition that $\mathcal{T}$ needs to fulfill such that our approach provides a superior better performance compared to FHSS. We derive this relationship terms of $\mathcal{T}$, since it is the metric that quantifies the overhead introduced by our channel switching approach. The inflexion point where the equality in (30) holds corresponds to the intersections between R_{SC} and R_{FHSS} in Fig. 10. While the $\mathcal{T}$ values in Fig. 10 are for a constellation with 2 points ($N = 2$) and have been obtained from the system implementation described in Sec. 10, notice that (30) generally describes the inequality in terms of the different communication parameters.

Furthermore, in Fig. 10a we analyze the impact of B_J under the requirement of D=1Mbits. It can be observed that R_{SC} is constant throughout all B_J values. This is justified due to the fact that the overhead of switching into a different channel is dictated by the UAV movement into another spatial constellation point ($\mathcal{T}$). However, the latency remains constant with B_J since the communication resumes in a jamming-free channel. In contrast, the FHSS latency (R_{FHSS}) increases with B_J, given that the $\left(1-\frac{B_J}{B^T_{\text{FHSS}}}\right)$ term in Eq. 27 approaches 0 for increasing B_J. In addition, in Fig. 10b we show how R_{SC} remains zero for a duration of $\mathcal{T}$ but achieves a higher bit rate compared to FHSS when the transmission starts. Also, we note that the overhead of $\mathcal{T}$ is only present while the transmission moves into a jamming-free channel, achieving a constant latency of $\frac{D}{R_{SC}}$ afterwards.

10 UAV SYSTEM IMPLEMENTATION

10.1 Hardware Overview

We integrated Nvidia TX2 with Connecttech Orbitty carrier board to act as an on-board computer on commercial off-the-shelf (COTS) developer friendly DJI drones (M600 Pro, M100). Robotic Operating System (ROS) was employed in order to establish interaction with the UAV and be able to send control inputs. ROS is a middleware used for message passing in a publish-subscribe pattern and it hosts a suite of software frameworks [40, 41]. The communication between the control ground station (Ubuntu 16.04 laptop) and the UAV-mounted computer is established through a common WiFi access points working on the 2.4GHz ISM band. Additionally, we integrated DJI real-time kinematic (D-RTK) GPS

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

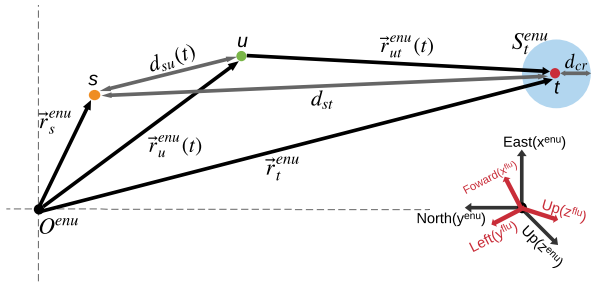


Fig. 12: Waypoint navigation diagram. *s* represents the initial UAV location, *t* be target waypoint and *u* is UAV location at time, t.

solution with an M600 pro (Fig. 4). D-RTK enhances both navigation and position lock during the hovering state. This was done for the hovering and radar accuracy comparing purposes in Fig. 4.

10.2 Software Architecture Overview

As mentioned above, ROS was employed for sending control messages. While single ROS master implementations can successfully handle multiple machines in a LAN, its centralized approach is prone to errors. To overcome this, we used the fkie-Multimaster, which is a decentralized implementation where each host runs its own master and all masters are synchronized. In order to ensure the system level time synchronization, the ground control station acts as an NTP time server.

While a few 3D navigation packages for aerial vehicles have been developed, they are mostly designed for open source hardware and open source software protocols, and would not fit our requirements. For that purpose, we implemented our own modular 3D navigation software stack capable of working with DJI ROS SDK, which we will refer as Simplenav. An overview of our UAV control system implementation is shown in Fig. 11. Simplenav capabilities, among others, are telemetry data fusion for 6-degree of freedom state at 100Hz (State Estimator), upsampling control signals from 10Hz to 50Hz (Base Controller) and set GPS location to constellation points (Constellation Manager in coordination with Navigator Module). Further controller details and PID implementation details will be discussed in the following subsection.

10.3 Waypoint Navigation Overview

We use the Universal Transverse Mercator (UTM) projection of the GPS fix in East-North-Up (ENU) based coordinate system as localization aid. For that, we define the UAV take off location as the ENU coordinate origin, O^{enu}. This allows us to define the necessary position vectors and distance metrics for the UAV starting point (*s*), the UAV location at time, *t* be (*u*) and the target waypoint (*t*), which can be observed in Fig.12 and mathematically expressed below:

$$\begin{aligned} d_{st} &= |\vec{r}_s^{enu} - \vec{r}_t^{enu}| \\ d_{su}(t) &= |\vec{r}_u^{enu}(t) - \vec{r}_s^{enu}| \\ \vec{r}_{ut}^{enu}(t) &= \vec{r}_t^{enu} - \vec{r}_u^{enu}(t) \\ d_{ut}(t) &= |\vec{r}_{ut}^{enu}(t)| \end{aligned} \tag{31}$$

The tracking error $\vec{r}_{ut}^{enu}(t)$ is computed in ENU frame. Thus, it needs to be projected into the UAV body frame Forward-Left-Up (FLU) ($\vec{e}_{ut}^{flu}(t)$). The projection is computed according to:

$$\vec{e}_{ut}^{flu}(t) = \begin{bmatrix} \cos\Psi_{yaw}(t) & \sin\Psi_{yaw}(t) & 0 \\ -\sin\Psi_{yaw}(t) & \cos\Psi_{yaw}(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{r}_{ut}^{enu}(t) \tag{32}$$

where $\Psi_{yaw}(t)$ is the yaw angle with respect to the East direction in ENU coordinate system.

10.4 Controller Formulation

As mentioned in Sec. 7 (Eq. 22a), we aim to minimize the travel time ($\mathcal{T}$) between constellation points under realistic kinematic constraints. For commodity, we define:

$$\begin{aligned} t_{cr} &= t_{acc} + t_{const\_vel} + t_{dec} \\ \mathcal{T} &= t_{cr} + t_{os} \end{aligned} \tag{33}$$

where t_{cr} is the travel time from the start location until the UAV enters the capture radius (d_{cr}) of the destination, t_{os} is the overshooting error correction time and t_{acc}, t_{const_vel} and t_{dec} are the acceleration, constant velocity and deceleration travel times respectively.

Next, the error of the UAV when travelling to a certain waypoint also impacts the problem formulation and has to be bounded (Eq. 22b) to properly design the spacial constellations. This error will be referred to as residual error (e_{res}) and is defined as the euclidean distance between the destination constellation point and the UAV's location after UAV considers it has reached it. The capture radius (d_{cr}) as defined in Fig. 12 is a design parameter that plays major role in the ultimately achieved e_{res} of the controller. Generally, d_{cr} is determined mainly on factors like frequency of localization sensor (RTK GPS at 5Hz), frequency of control cycle (10Hz) and velocity saturation limits of the UAV.

Typically, PID controllers are tuned to optimize one of the two mentioned metrics, $\mathcal{T}$ or e_{res}, which is to prioritize low residual error in detriment of higher travel time, or vice-versa, higher residual error when low travel time is prioritized. In this work, we choose to minimize the travel time and characterize the error (Fig. 4) afterwards in order to design the spatial constellations accordingly. We propose a two fold process to minimize $\mathcal{T}$ with maximum residual error constraint (e_{res}^{max}):

1) Tune the PID controller to minimize t_{cr} with overshoot error not exceeding e_{res}^{max}.
2) Minimize t_{os} by relaxing criterion to determine success in reaching a waypoint.

10.4.1 Tuning Process Overview

We manually tuned our PID controller online since DJI low-level Flight controller (FC) is closed sourced. As mentioned in the previous section, the objective is two-fold, minimize $\mathcal{T}$ while keeping e_{res} under a certain threshold (e_{res}^{max}). First, the PID controller was tuned to minimize t_{cr}. Second, we tuned the controller saturation limits to $u_{min} = 0.15\,\mathrm{m/s}$ and $u_{max} = 5\,\mathrm{m/s}$ in order to achieve $e_{res}^{max} = \pm 0.20\,\mathrm{m}$ with $d_{cr} = 0.15\,\mathrm{m}$. The tuned proportional, integral and derivative constants are $k_p = 0.6$, $k_I = 0.05$ and $k_d = 0.12$ respectively.

10.4.2 Criteria to Determine Success in Reaching a Waypoint

Typically, the following expression would be used in order to decide whether a UAV has reached a destination location.

$$d_{ut}(t) <= d_{cr} \tag{34}$$

However, experimental exploration after parameter tuning revealed that the overhead of t_{os} over $\mathcal{T}$ was considerably high. In order to mitigate this problem we proposed a relaxation in the criteria used to determine if a UAV has reached its destination, expressed as the following:

$$d_{su}(t) > d_{st} + d_{cr} \tag{35}$$

The new destination point reaching criterion assumes that if the UAV overshoots, it will have mostly been in the direction defined by $\vec{r}_{st}^{enu}$. Then, the proposed modification eliminates the overhead introduced by t_{os} in exchange for potentially higher residual errors that can be compensated with the proper design of the spatial constellations. Thus, if either of the criteria defined by Eq. (34) or Eq. (35) is fulfilled, the controller will consider that the UAV has reached the destination point.

10.4.3 Implementation Overview

Here, we present final implementation details of the PID controller. As mentioned in Sec. 10.3, the UAV takes inputs from the DJI ROS SDK in FLU coordinate format. Accordingly, the unbounded estimated velocity along a FLU axis ($\vec{u_e}^{flu}(t)\cdot\hat{n}$) and yaw rate ($\vec{\omega_e}(t)$) PID updates are estimated as expressed in Eq. (36) and Eq. (37) respectively.

$$\begin{aligned}\vec{u_e}^{flu}(t)\cdot\hat{n} =& k_p\, \vec{e_{ut}}^{flu}(t)\cdot\hat{n} + k_d\frac{d\vec{e_{ut}}^{flu}(t)\cdot\hat{n}}{dt} + \\ & k_I\int \vec{e_{ut}}^{flu}(t)\cdot\hat{n}\,dt|_{\hat{n}\in\hat{i},\hat{j},\hat{k}}\end{aligned} \tag{36}$$

$$\vec{\omega_e}(t) = k_p\, \vec{e_{yaw}}(t) + k_d\frac{d\vec{e_{yaw}}(t)}{dt} + k_I\int \vec{e_{yaw}}(t)\,dt \tag{37}$$

The process of bounding $\vec{u_e}^{flu}(t)$ within saturation limits(u_{max}, u_{min}) (Sec. 10.4.1) to compute the commanded velocity ($\vec{u_c}^{flu}(t)$) can be expressed as:

$$\vec{u_c}^{flu}(t)\cdot\hat{n} = \begin{cases}\text{sign}(\vec{u_e}^{flu}(t)\cdot\hat{n})\, u_{min}, & \text{if} \quad |\vec{u_e}^{flu}(t)\cdot\hat{n}| \le u_{min} \\ \vec{u_e}^{flu}(t)\cdot\hat{n}\ \cdot u_{max}, & \text{if} \quad |\vec{u_e}^{flu}(t)\cdot\hat{n},| \ge u_{max} \\ \vec{u_e}^{flu}(t)\cdot\hat{n}, & \text{otherwise}\end{cases}$$
$$\text{where,} \quad \hat{n}\in\hat{i},\hat{j},\hat{k} \tag{38}$$

However, the above introduced formulation does not actively tackle the PID integral windup problem [42], which might cause velocity saturation resulting in excessive overshoot. A variant of the anti-windup strategy described in [42] is implemented to constrain the cumulative error to a maximum threshold, ${C_e}^{max}$, limiting the PID integral component to attain a maximum of $k_I\cdot{C_e}^{max}$. This solution with a lower ${C_e}^{max}$ ensures the cumulative error to reach a maximum threshold within a few control cycles, thereby avoiding controller saturation. Lowering ${C_e}^{max}$ increases the sensibility of the controller to change in the direction of the error, which is commonly observed during the overshoot correction(settling) phase. The integral term of the PID formulation (Eq. (36)) after introducing this anti-windup strategy, $\vec{u_I}$ can be described as the following:

$$\vec{u_I}(t)\cdot\hat{n} = \begin{cases}k_I\cdot\text{sign}(C_e(t))\cdot{C_e}^{max}, & \text{if, } |C_e(t)| \ge {C_e}^{max} \\ k_I\cdot C_e(t), & \text{otherwise}\end{cases} \tag{39}$$

where,

$$C_e(t) = \int_0^t \vec{e_{ut}}^{flu}(t)\cdot\hat{n}\,\mathrm{d}t|_{\hat{n}\in\hat{i},\hat{j},\hat{k}} \tag{40}$$

In our implementation, we set a threshold on the cumulative error, ${C_e}^{max}$ to $\pm5\,\mathrm{m}$, such that the maximum value of the integral component ($= \left|0.25\,\mathrm{m\,s^{-1}}\right|$) is closer to the controller saturation velocity lower bound.

Finally, notice that the system architecture and PID controller described in this section applies to a multi-UAV scenario as well, where each UAV would run its own PID controller independently and all drones would be centrally connected to the same ground control station.

10.5 Estimation of Travel Time for simulation

Following the nomenclature in Sec. 10.3, we define $\vec{u_a}(t)$ as the real velocity the UAV experiences under external disturbances. In our PID implementation, the estimated velocity $\vec{u_e}^{flu}(t)\cdot\hat{n}$ is calculated independently along each FLU axis. In this work, in order to compute the estimated travel time $\mathcal{T}$ in simulation, we use the PID controller implementation in Sec. 10.4.3 while considering equivalent kinematic constraints of the UAV. Here, we assumed that the commanded velocity ($\vec{u_c}^{flu}(t)\cdot\hat{n}$) and actual velocity ($\vec{u_a}(t+dt)\cdot\hat{n}$) will be proportional at all times and they will never have opposite directions. We represent such relationship as $\vec{u_a}\cdot\hat{n} = \lambda\cdot\vec{uc}\cdot\hat{n}$, where the scaling factor λ is a positive constant that compensates $\vec{u_c}\cdot\hat{n}$ for the combined effect of the external disturbances and the FC response. Additionally, since the PID integral component quickly saturates to ${C_e}^{max}$, the term $\vec{u_I}\cdot\hat{n}$ in (Eq. 39) will be set to $k_I\cdot\text{sign}(C_e(t))\cdot{C_e}^{max}$ for the entire travel time. Notice that this also takes into account the relaxation criteria defined in Sec. 10.4.2.

The parameter λ is typically bounded considering imposed kinematic constraints of controller and realistic UAV velocities. Additionally, λ varies in time based on t_{acc}, t_{const_vel} and t_{dec}. However, given the PID implementation details and a short constellation point separation in our scenario, we observe that: i) $t_{dec} \gg t_{acc}$ and ii) $t_{const\_vel} = 0$. Thus, in simulation, we assume a constant λ for the entire travel time.

The travel time spent along a FLU axis is derived from our PID implementation in Eq. 36 after considering the above described assumptions. The bounded travel time ($\mathcal{T}_n$) along a FLU axis is estimated using the waypoint reaching criterion in Sec. 10.4.2 ($\vec{r_{su}}^{flu}(t)\cdot\hat{n} = \vec{r_{st}}\cdot\hat{n} - d_{cr}$), and can be expressed as:

$$\mathcal{T}_n = -\frac{\frac{1}{\lambda}+k_d}{k_p}\cdot\ln\left(1 - \frac{\vec{r_{st}}^{flu}\cdot\hat{n} - d_{cr}}{\vec{r_{st}}^{flu}\cdot\hat{n} + \frac{k_I}{k_P}\cdot C_{max}}\right); \hat{n}\in\hat{i},\hat{j},\hat{k} \tag{41}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

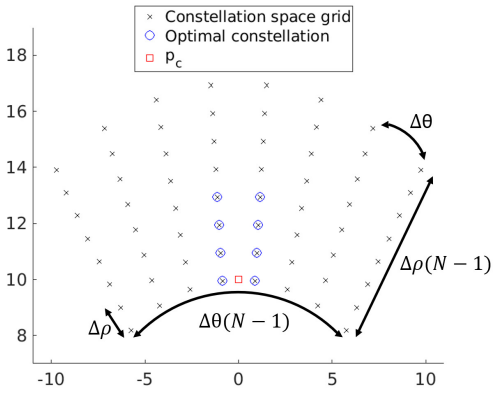


Fig. 13: Spatial constellation

The total travel time $\mathcal{T}$ to reach $r_t^{\vec{enu}}$ (Eq. 33) is computed as the maximum of travel times $\mathcal{T}_n$ along each FLU axis.

$$\mathcal{T} = \max_{\forall i \in x,y,z} \mathcal{T}_n \tag{42}$$

11 Performance Evaluation

11.1 Simulation results

In Fig. 14, we compare the average distance for the exhaustive search and the heuristically derived constellation points, and we see a near-perfect agreement comparing both approaches. Fig. 15 analyzes the impact on $\mathcal{T}$ of Δ_θ and Δ_ρ, as well as the constellation size. Whenever Δ_θ or Δ_ρ increases, so does $\mathcal{T}$. This is expected since the UAV takes more time for traversing longer inter-symbol distances. The travel time is computed considering a PID controller that chooses the velocity of the UAV based on the distance to its target. We set $k_p = 0.6$, $k_d = 0.12$ and $k_i = 0.05$, which represent for the *proportional*, *derivative* and *integral* PID constants, respectively. All the simulations are conducted in MATLAB.

11.2 Experimental results

We assume the problem involves selecting one of two channels centered at 900MHz and 905MHz, and so, we set

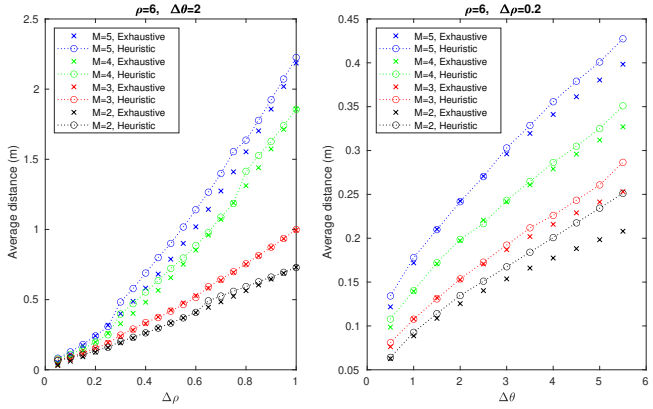


Fig. 14: Performance of heuristic with respect to exhaustive exploration.

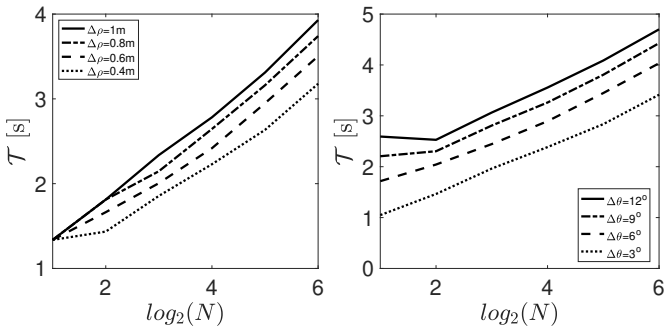


Fig. 15: Travel time vs $log_2(N)$ for different Δ_ρ and Δ_θ values. Both plots are for $\rho = 5m$ and $\lambda = 1$ (Eq. 41). On the left, Δ_θ is fixed to 5^o. On the right, Δ_ρ is fixed to 0.8m.

N=2. For this experiment, we set Δ_ρ and Δ_θ such that $P_e \approx 0$. This is achieved by choosing $\Delta_{\rho/\theta} \gg \sigma_{\rho/\theta}$. For measurements with RTK (Sec. 4.2) we find $\sigma_\rho \approx 0.05m$ and $\sigma_\theta \approx 0.9^o$. Thus, we pick the quotient $\frac{\Delta_{\rho/\theta}}{\sigma_{\rho/\theta}}$ to be $\approx$ 13dB, which results in $\Delta_\rho = 1m$ and $\Delta_\theta = 18^o$. The exhaustive search constellation for these values results in two symbols along the ρ axis. For $\rho = 5m$, these points are at $s_1 = [0, 5]$ and $s_2 = [0, 6]$, where $s_i = [\theta_i, \rho_i]$ represent the coordinates for symbol i. Then, we mount one Ettus B210 on a DJI M600 that acts as a transmitter. Another B210 and a TI IWR1642 radar is connected to a BS running Linux, which also executes the clustering algorithm from Sec. 4.3. The UAV location is an input for switching the center frequency of the B210 SDR. Finally, a third B210 on the ground emulates a jammer by transmitting at high power. Fig. 16 shows how goodput evolves over the jamming attack, and how position-based information relaying allows the link to recover via channel switching. Prior to the jamming (1), the average goodput is 100% with near-perfect data decoding. However, when the jamming attack begins, the receiver is not able to decode the received data (2). Then, the UAV switches the transmission to a new channel and moves to a new constellation point ($\rho = 6 \rightarrow \rho = 5$) in order to communicate it to the BS. This movement is sensed and the data transmitted in the new jamming-free channel is decoded at the BS.

Finally, in Fig. 17 we show the experimental and theoretical average travel time $\mathcal{T}$. We see that our model is more accurate for larger ($\geq N$) constellation sizes. This is because the theoretical model does not account for wind, which makes the UAV take longer time to converge to its next location. Larger constellation sizes are analyzed via simulations in the previous subsection. (Fig. 15).

12 Feasibility of the Approach

In this section, we discuss the feasibility of the approach presented in this paper. While there are current solutions that already solve the problem of channel selection under high interference conditions i.e. Bluetooth. [43], our approach proposes a general solution that considers limitations of UAV deployments and at the same time leverages from their mobility. UAVs are generally power constrained and tend to operate hundreds of meters away from their targets. Thus, spectrum exploration, specially on a high

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2021.3102883, IEEE Transactions on Mobile Computing

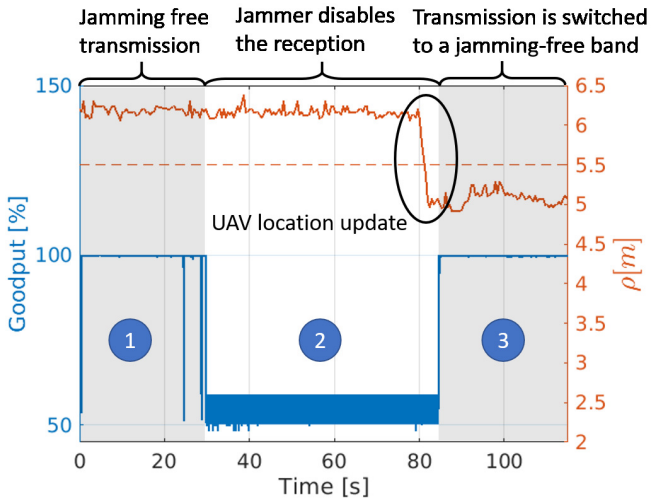


Fig. 16: UAV and BS communicate on a certain channel (1) which is jammed right after (2). Then, the UAV moves to a new location within its constellation (from $\rho = 6$ to $\rho = 5$) to inform the BS the communication will switch to a new band. In (3), the transmission resumes in the new band.

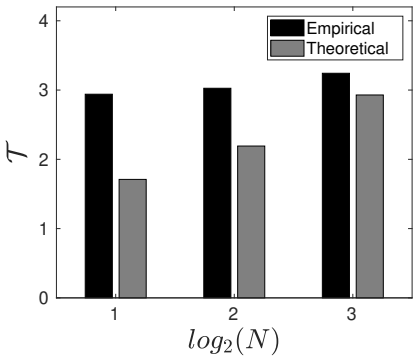


Fig. 17: Average travel time comparison.

number of available bands, would be a burden for such deployments. Furthermore, in scenarios where UAVs work cooperatively in a shared spectrum [44] and their wish to establish interference-free links with a same BS, our approach would be a feasible solution towards safe and robust dynamic channel communication between the UAVs and the ground-based station. Also, in the recent increase of research interest in Millimeter-Wave (mmWave), multiple UAV applications have been investigated and have already shown successful results. However, due to the extremely high attenuation in the mmWave band due to atmospheric absorption, spectrum exploration would be prohibitive in such scenarios as well. To sum up, our approach allows the UAV to deterministically indicate on what frequency the communication will happen, while the overhead introduced due to the travel time between constellations points is not negligible, such capability is of great advantage specially when wide spectrum bands are available in different frequency ranges.

13 Conclusion

In this work, we present an interdisciplinary paradigm of position based modulation using UAVs which can convey information by creating spatial codes. As a use case, we show this method can be used for channel selection in jamming situations. In addition, we show and analyze the feasibility of the multi-UAV case, tackling the clustering, space limitation and probability of error derivations challenges. Next, we compare our method to FHSS, a widely used transmission technique under broadband jamming attacks, and identify the conditions under which of these two methods might be preferred. Our work is driven by experimental characterization of localization error caused by hovering UAVs, and errors introduced by COTS mmWave sensor. Finally, we experimentally demonstrate how our system can be used to overcome jamming using a DJI M600, Ettus B210 SDRs and a TI IWR1642 mmWave sensor.

Acknowledgments

This work is supported by the Office of Naval Research under grant N000141612651.

References

[1] M. Mozaffari, W. Saad, M. Bennis, Y. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *CoRR*, vol. abs/1803.00680, 2018.

[2] J. Wang, C. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Vehicular Technology Magazine*, vol. 12, no. 3, pp. 73–82, Sep. 2017.

[3] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Optimal transport theory for power-efficient deployment of unmanned aerial vehicles," *CoRR*, vol. abs/1602.01532, 2016.

[4] S. H. Seo, J. I. Choi, and J. Song, "Secure utilization of beacons and UAVs in emergency response systems for building fire hazard," *Sensors (Basel)*, vol. 17, no. 10, pp. 1–21, Sep. 2017.

[5] A. Trotta, M. D. Felice, F. Montori, K. R. Chowdhury, and L. Bononi, "Joint coverage, connectivity, and charging strategies for distributed uav networks," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 883–900, Aug 2018.

[6] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2005, pp. 46–57.

[7] M. Y. Naderi, K. R. Chowdhury, S. Basagni, W. Heinzelman, S. De, and S. Jana, "Experimental study of concurrent data and wireless energy transfer for sensor networks," in *2014 IEEE Global Communications Conference*, Dec 2014, pp. 2543–2549.

[8] G. Secinti, A. Trotta, S. Mohanti, M. Di Felice, and K. R. Chowdhury, "Focus: Fog computing in uas software-defined mesh networks," in *IEEE Trans. on Transactions on Intelligent Transportation Systems*, 2019.

[9] S. Kim, H. Oh, J. Suk, and A. Tsourdos, "Coordinated trajectory planning for efficient communication relay using multiple uavs," *Control Engineering Practice*, vol. 29, pp. 42–49, 2014.

[10] I. Rubin and R. Zhang, "Placement of uavs as communication relays aiding mobile ad hoc wireless networks," in *MILCOM 2007-IEEE Military Communications Conference*. IEEE, 2007, pp. 1–7.

[11] Y. Zeng, R. Zhang, and T. J. Lim, "Throughput maximization for uav-enabled mobile relaying systems," *IEEE Transactions on Communications*, vol. 64, no. 12, pp. 4983–4996, 2016.

[12] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging uavs for disaster management," *IEEE Pervasive Computing*, vol. 16, no. 1, pp. 24–32, 2017.

[13] G. R. Muns, K. V. Mishra, C. B. Guerra, Y. C. Eldar, and K. R. Chowdhury, "Beam alignment and tracking for autonomous vehicular communication using ieee 802.11 ad-based radar," in *2019 IEEE INFOCOM Workshop*, May 2019.

[14] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," *Proceedings of the IEEE*, vol. 104, no. 9, pp. 1727–1765, 2016.

[15] Y. Xu, G. Ren, J. Chen, X. Zhang, L. Jia, and L. Kong, "Interference-aware cooperative anti-jamming distributed channel selection in uav communication networks," *Applied Sciences*, vol. 8, no. 10, p. 1911, 2018.

[16] M. Strasser, C. Pöpper, and S. Čapkun, "Efficient uncoordinated fhss anti-jamming communication," in *Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*. ACM, 2009, pp. 207–218.

[17] E.-K. Lee, S. Y. Oh, and M. Gerla, "Randomized channel hopping scheme for anti-jamming communication," in *2010 IFIP Wireless Days*. IEEE, 2010, pp. 1–5.

[18] L. Zhang, H. Wang, and T. Li, "Anti-jamming message-driven frequency hopping—part i: System design," *IEEE Transactions on Wireless Communications*, vol. 12, no. 1, pp. 70–79, 2012.

[19] X. Shi, C. Yang, W. Xie, C. Liang, Z. Shi, and J. Chen, "Anti-drone system with multiple surveillance technologies: Architecture, implementation, and challenges," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 68–74, 2018.

[20] H. Shin, K. Choi, Y. Park, J. Choi, and Y. Kim, "Security analysis of fhss-type drone controller," in *International Workshop on Information Security Applications*. Springer, 2015, pp. 240–253.

[21] K. Pärlin, M. M. Alam, and Y. Le Moullec, "Jamming of uav remote control systems using software defined radio," in *2018 International Conference on Military Communications and Information Systems (ICMCIS)*. IEEE, 2018, pp. 1–6.

[22] H.-B. Kil, J.-S. Lee, and E.-R. Jeong, "Analysis of frequency hopping signals in commercial drones," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 19, pp. 2015–2024, 2018.

[23] Y. Xu, G. Ren, J. Chen, Y. Luo, L. Jia, X. Liu, Y. Yang, and Y. Xu, "A one-leader multi-follower bayesian-stackelberg game for anti-jamming transmission in uav communication networks," *IEEE Access*, vol. 6, pp. 21 697–21 709, 2018.

[24] X. Lu, L. Xiao, C. Dai, and H. Dai, "Uav-aided cellular communications with deep reinforcement learning against jamming," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 48–53, 2020.

[25] L. Xiao, X. Lu, D. Xu, Y. Tang, L. Wang, and W. Zhuang, "Uav relay in vanets against smart jamming with reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4087–4097, 2018.

[26] A. Klautau, N. González-Prelcic, and R. W. Heath, "Lidar data for deep learning-based mmwave beam-selection," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 909–912, 2019.

[27] N. González-Prelcic, R. Méndez-Rial, and R. W. Heath, "Radar aided beam alignment in mmwave v2i communications supporting antenna diversity," in *2016 Information Theory and Applications Workshop (ITA)*. IEEE, 2016, pp. 1–7.

[28] E. Natalizio and V. Loscrí, "Controlled mobility in mobile sensor networks: advantages, issues and challenges," *Telecommunication Systems*, vol. 52, no. 4, pp. 2411–2418, 2013.

[29] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.

[30] D. Birant and A. Kut, "St-dbscan: An algorithm for clustering spatial–temporal data," *Data & Knowledge Engineering*, vol. 60, no. 1, pp. 208–221, 2007.

[31] Y. He, H. Tan, W. Luo, S. Feng, and J. Fan, "Mr-dbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data," *Frontiers of Computer Science*, vol. 8, no. 1, pp. 83–99, 2014.

[32] B. Borah and D. Bhattacharyya, "An improved sampling-based dbscan for large spatial databases," in *International conference on intelligent sensing and information processing, 2004. proceedings of.* IEEE, 2004, pp. 92–96.

[33] J. Shen, X. Hao, Z. Liang, Y. Liu, W. Wang, and L. Shao, "Real-time superpixel segmentation by dbscan clustering algorithm," *IEEE transactions on image processing*, vol. 25, no. 12, pp. 5933–5942, 2016.

[34] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." *Kdd*, vol. 96, no. 34, pp. 226–231, 1996.

[35] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Dbscan revisited, revisited: why and how you should (still) use dbscan," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 3, p. 19, 2017.

[36] R. P. Stanley, "What is enumerative combinatorics?" in *Enumerative combinatorics*. Springer, 1986, pp. 1–63.

[37] K. P. Bogart, *Combinatorics through guided discovery*. American Institute of Mathematics Edition: FirstPreTeXtEdition Website: http . . . , 2017.

[38] T. W. Cusick and P. Stanica, *Cryptographic Boolean functions and applications*. Academic Press, 2017.

[39] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC press, 2018.

[40] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system." [Online]. Available: https://www.ros.org

[41] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

[42] C. Bohn and D. P. Atherton, "An analysis package comparing pid anti-windup strategies," *IEEE Control Systems Magazine*, vol. 15, no. 2, pp. 34–40, 1995.

[43] J. C. Haartsen, "Bluetooth radio system," *Wiley Encyclopedia of Telecommunications*, 2003.

[44] I. Y. Abualhaol and M. M. Matalgah, "Throughput optimization of cooperative uavs using adaptive channel assignment," in *IEEE Wireless Communications and Networking Conference, 2006. WCNC 2006.*, vol. 4. IEEE, 2006, pp. 2279–2284.

Guillem Reus-Muns received the B.Sc. degree in telecommunications engineering from the Polytechnic University of Catalonia (UPC-BarcelonaTech). He joined Northeastern University, USA, in 2018, where he got his M.Sc. in electrical and computer engineering and is currently working towards his Ph.D. His current interests include mobile communications, networked robotics, machine learning for wireless communications and spectrum access.

Mithun Diddi received his B.Tech. degree in Mechatronics Engineering from SRM University, India in 2015. He received his Master's degree in Mechatronics Engineering from Northeastern University, Boston in 2017. He is currently a Ph.D candidate at Field Robotics Lab, Institute for Experiential Robotics at Northeastern University. His research interests include system integration, perception and all-day autonomy for aerial robots.

Chetna Singhal (S'13-M'15) received the B.Eng. in Electronics and Telecommunications from the University of Pune, India in 2008, M.Tech. degree in Computer Technology from the Indian Institute of Technology (IIT) Delhi in 2010, and Ph.D. degree also from IIT Delhi in 2015. She worked in IBM Software Lab, New Delhi, as a Software Engineer in 2010 for a year. She is currently an Assistant Professor with the Department of Electronics and Electrical Communication Engineering, IIT Kharagpur since 2015. Her research interests are in next generation heterogeneous wireless networks, with emphasis on cross-layer optimization, adaptive multimedia services, energy efficiency, and resource allocation.

Hanumant Singh) is a Professor at Northeastern University. He received his Ph.D. from the MIT WHOI Joint Program in 1995 after which he worked on the Staff at Woods Hole Oceanographic Institution until 2016 when he joined Northeastern. His group has designed and built the Seabed AUV, as well as the Jetyak Autonomous Surface Vehicle dozens of which are in use for scientific and academic research across the globe. He has participated in 60 expeditions in all of the world's oceans in support of Marine Geology, Marine Biology, Deep Water Archaeology, Chemical Oceanography, Polar Studies, and Coral Reef Ecology. In collaboration with his students his awards include the ICRA Best Student Paper Award, the Sung Fu Memorial Best IEEE Transactions on Robotics Paper Award and Best Paper Awards at the Oceans Conference and at AGU.

Kaushik Roy Chowdhury (M'09-SM'15) received the M.S. degree from the University of Cincinnati in 2006, and the Ph.D. degree from the Georgia Institute of Technology in 2009. He was an Assistant Professor from 2009 to 2015 at Northeastern University, where he is now an Associate Professor with the Electrical and Computer Engineering Department. He was a winner of the Presidential Early Career Award for Scientists and Engineers (PECASE) in 2017, ONR Director of Research Early Career Award in 2016 and the NSF CAREER Award in 2015. His current research interests include deep learning for wireless sensing and spectrum access, networked robotics, wireless RF energy harvesting/transfer and IoT applications for intra/on-body communication.