

## RESEARCH ARTICLE

# Self-Localization via Circular Bluetooth 5.1 Antenna Array Receiver

**NUNO PAULINO**  AND **LUÍS M. PESSOA** , (Senior Member, IEEE)Faculty of Engineering, University of Porto, 4200-465 Porto, Portugal  
INESC TEC, Campus da FEUP, 4200-465 Porto, Portugal

Corresponding author: Nuno Paulino (nuno.m.paulino@inesctec.pt)

This work was supported by the ERDF—European Regional Development Fund through the Operational Programme for Competitiveness and Internationalization—COMPETE 2020 Programme under the Portugal 2020 Partnership Agreement through Project Continental Factory of the Future (FoF) under Grant POCI-01-0247-FEDER-047512.


**ABSTRACT** Future telecommunications aim to be ubiquitous and efficient, as widely deployed connectivity will allow for a variety of edge/fog based services. Challenges are numerous, e.g., spectrum overuse, energy efficiency, latency and bandwidth, battery life and computing power of edge devices. Addressing these challenges is key to compose the backbone for the future Internet-of-Things (IoT). Among IoT applications are Indoor Positioning System and indoor Real-Time-Location-Systems systems, which are needed where GPS is unviable. The Bluetooth Low Energy (BLE) 5.1 specification introduced Direction Finding to the protocol, allowing for BLE devices with antenna arrays to derive the Angle-of-Arrival (AoA) of transmissions. Well known algorithms for AoA calculation are computationally demanding, so recent works have addressed this, since the low-cost of BLE devices may provide efficient solutions for indoor localization. In this paper, we present a system topology and algorithms for self-localization where a receiver with an antenna array utilizes the AoAs from fixed battery powered beacons to self-localize, without a centralized system or wall-power infrastructure. We conduct two main experiments using a BLE receiver of our own design. Firstly, we validate the expected behaviour in an anechoic chamber, computing the AoA with an RMSE of  $10.7^\circ$ . Secondly, we conduct a test in an outdoor area of 12 by 12 meters using four beacons, and present pre-processing steps prior to computing the AoAs, followed by position estimations achieving a mean absolute error of 3.6 m for 21 map positions, with a minimum as low as 1.1 m.

**INDEX TERMS** BLE, bluetooth, wireless sensor networks, angle-of-arrival, signal processing, Internet of Things, self-localization, RTLS, IPS.

## I. INTRODUCTION

Telecommunications capabilities grow as a response to requirements of services and businesses. In turn, these new capabilities drive the innovation of more services benefiting from new technologies, ranging from device level up to the service/product level. In recent years, we can highlight the rise of several aspects which combined can drive future Internet-of-Things (IoT) applications. As the demand for the amount of data delivered to the end-user increases, wireless communications are faced with an over-crowded spectrum. Thus, 5G/6G solutions are advancing towards

higher frequency ranges [1], [2]. This enables the use of small scale antennas which, despite the device level challenges, opens up the possibility of small devices with antenna arrays [3], or the deployment of large arrays for Multiple-Input Multiple-Output (MIMO) communications. Antenna arrays allow for directional communications by controlling the individual antenna elements, performing beamforming for outgoing signals, or calculating the direction incoming signals. This can be key to proper use of the available spectrum and to improving power efficiency. Concurrently, in the edge domain, computing is continuing to trend towards fully featured System-on-Chip (SoC) solutions [4], [5], often now including Field-Programmable-Gate-Array (FPGA) components for algorithmic acceleration [6], [7],

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

enabling the implementation of more ambitious IoT applications. These factors have resulted in increased interest in antenna arrays for directional communications, which must be supported by the respective devices and algorithms for fast beamforming and Direction Finding (DF). Indoor localization is one possible application, and is the focus of this work. We evaluate a novel system topology and algorithms for self-localization based on Bluetooth Low Energy (BLE) and Angle-of-Arrival (AoA) via real-world experimental runs with a circular antenna array.

### A. INDOOR LOCALIZATION TECHNIQUES

Indoor localization cannot rely on GPS for triangulation, due to lack of line-of-sight. Therefore, existing solutions rely on other signal characteristics, such as the estimation of the distance of a transmitter by a receiver based on the Received Signal Strength Indicator (RSSI), the Time-of-Arrival (ToA) of a signal based on the propagation speed of electromagnetic waves and knowledge of the time of departure of a signal, or the Time-Difference-of-Arrival (TDoA), which is similar to ToA, but relies on multiple transmitters [8], [9].

For instance, a receiver may estimate the distance of a transmitter via the RSSI, if the original signal transmission power was known, and by relying on the Friis transmission equation which computes the decrease in power as a function of distance. However the RSSI is not entirely reliable as it can be affected by reflections, refraction, and the transmission power may vary from device to device either due to fabrication differences or depleting batteries. This influences approaches that perform fingerprinting of a space based on RSSI values, i.e., they map the RSSI expected from each fixed transmitter for every possible position of a mobile receiver in a space [10], [11]. However, an operational failure of a single transmitter will impact performance.

Use of ToA or TDoA relies on the time of travel of the signal. Specifically, ToA determines the distance between two devices by time-tagging the sent packets, and using the known constant speed of light. However, this requires a very precise synchronization of clock frequencies between both the receiver and transmitter, which can fail due to fabrication differences in clocks or clock drifts. For a minimum precision of 1 m, a synchronization error under 3.3 ns is required between receiver and transmitters [12], and in turn, very high frequency sampling is required to capture the time of flight of a signal with this accuracy. Instead, TDoA computes the difference in the arrival time of several signals. It is the technology employed by GPS, and does not require synchronization between receivers and transmitters. Instead, the transmitters are synchronized to simultaneously emit periodic signals. The TDoA amongst these signals produces a candidate area for the receiver. The distances involved in GPS localization are very high, as satellites may orbit as high as 20,000 km, and there is always line of sight between transmitters and receivers, so there is significant tolerance in the synchronization between transmitters. However, in indoor scenarios, the distances involved, especially

if precise localization is required, in turn impose a very high synchronization requirement between transmitters. This can be achieved with cabled installations, which are expensive, require tuning of cable lengths, and do not scale easily as a function of the target indoor area [9].

In contrast, the Angle-of-Arrival (AoA) does not require synchronization between receivers and transmitters, is not affected by variations in the RSSI, and functions for an arbitrary number of transmitters. As long as the signal is received, calculation of the AoA (computable by state-of-the-art algorithms presented in the next section), is possible. Even though the use of antenna arrays is required, several topologies are possible, depending on which device is fixed or mobile, and equipped with the antenna array. It is applicable for both Indoor Positioning System (IPS) or Real-Time-Location-Systems (RTLS) applications [13], and scales easily, with the localization accuracy improving with the number beacons (i.e., more data redundancy via more AoAs), while also being resilient to beacon failures. However, it suffers from the same effects as other technologies which are inherent to indoor scenarios, namely lack of line-of-sight, noise, and especially multi-path effects. In addition, the algorithms required to compute the AoA are computationally complex, and demand high processing capabilities. The following section briefly presents these state-of-the-art algorithms.

### B. ALGORITHMS FOR AoA CALCULATION

Use of the AoA is not a new technique, as the concept itself has been addressed for decades [14], [15]. The AoA is also commonly named Direction-of-Arrival (DoA) in literature. In [16] and [17], overviews accompanied by simulations are presented to compare existing state-of-art algorithms.

These algorithms include Multiple Signal Classification (MUSIC) [18], Root-MUSIC (RMUSIC) [19], a less computationally complex variant applicable only for Uniform Linear Arrays (ULAs), Estimation of Signal Parameters via Rotational Invariance Techniques (ESPRIT) [20], Space-Alternating Generalized EM (SAGE) [21], the Conventional Beam Former (CBF) (i.e., Bartlett beamformer) [22], and the Minimum Variance Distortionless Response (MVDR) method [23]. Different categorizations are used in literature [14], [16], [24], but major categories are spectral estimation methods (based on detecting the power maxima in the signal pseudo-spectrum) like MVDR, or decomposition into the signal and noise sub-spaces [25], such as MUSIC and ESPRIT.

These algorithms process the signal samples per array element to determine the AoA, even of multiple signals. Inversely, they can compute the stimuli to be applied to each element to focus the resulting radiation pattern in a specific direction (i.e., beamforming). Historically, the primary application areas of such directional communications were radar for aerial navigation in defense and space scenarios [26].

The MUSIC and ESPRIT algorithms are the most widely used for AoA calculation [27], [28], but have

some limitations. For example, performance degradation in the lack of line-of-sight, and the assumption of ideal orthogonality between the noise and sample subspaces, which does not hold in real world conditions (due to multi-path effects) [29]. Finally, the computational complexity of these algorithms is a major factor in allowing for faster response, and especially implementation in compute constrained devices [30], prompting efforts towards low complexity implementations of AoA calculation algorithms [31], [32], [33].

### C. LOCALIZATION VIA AoA & BLE

While these techniques, i.e., ToA, AoA, *etc.*, have been implemented on top of different technologies, e.g. Wi-Fi, Ultra-Wide-Band (UWB), and Radio Frequency (RF) identification, only relatively recently has the Bluetooth Special Interest Group (SIG) added DF capabilities to Bluetooth Low Energy (BLE) [13], [34], allowing for AoA/Angle-of-Departure (AoD) based solutions for this protocol.

There are two conventional approaches for angle-based localization proposed in the Bluetooth SIG specification for BLE [13], [34], depending on which device, i.e., the transmitter or the receiver, is equipped with an antenna array, and which device's location is being calculated.

In the former case, transmitters equipped with arrays emit a signal, which a mobile receiver with an omni-directional antenna samples at a known pattern. This allows the receiver to compute the AoD of multiple source signals, and in turn determine its position (aided by additional information regarding the space sent in the packet payloads). This allows for IPS solutions, where an arbitrary number of receiver devices with low-cost omni-directional antennas (e.g., smartphones) can determine their own location. Applications include navigation of indoor spaces such as shopping malls or airports [35], or high risk scenarios like mining [36].

If instead the receivers are equipped with arrays, they can each receive transmissions from omni-directional emitters, compute the respective AoAs, send this information to an upstream centralized system where a final position of the emitter is computed. The topology works for both IPS and RTLS. In the former case, the centralized system must send back the information to the potentially arbitrary number of emitters querying their own location. In the later case, the number of assets is typically pre-determined and equipped with low-cost omni-directional tags, and applications include tracking of patients or equipment within a hospital [37], [38], localization in emergency response scenarios [39], or pallets and stock items within warehouses [8], [40].

In both cases, it is presumed that the antenna array devices are non-mobile anchors that also act as gateways which connect to a centralized system. Therefore, ethernet and/or Wi-Fi and wall-power cabling infrastructure are also required, making it difficult to deploy such solutions in legacy locations. Also, the approaches are more efficient when the devices to locate are stationary, since the estimated positions will deviate from the true positions due to transmission, data

transfer, computation times, and calculation errors inherent to the AoA measurements and employed algorithms.

In summary, the advantages of AoA, allied with the low cost of BLE and its widespread availability, and the increased push towards edge computing, has cause renewed research efforts combining these two technologies [41], [42], [43].

### D. CONTRIBUTIONS

This paper consolidates two previous development stages. Firstly, in [44], we validated separate components of our approach, namely the positioning estimation via simulation. Secondly, in [45], we designed a BLE 5.1 receiver with an 8-element Uniform Circular Array (UCA), performed real-world data gathering of BLE packets, evaluated the quality of the attainable data and explored pre-processing steps to allow for improved phase difference profiles.

This present manuscript consolidates these two previous works by explaining the data pre-processing in greater detail, and then presenting actual AoA calculations, and respective positioning estimates with the above mentioned real-world data. In summary, in this paper, we:

- 1) propose a topology for self-localization receivers with antenna arrays;
- 2) describe our BLE 5.1 receiver design with an 8-element UCA and it's expected behaviour;
- 3) present the algorithm to compute AoA from a phase difference profile attained from the UCA;
- 4) present the algorithm to estimate a position from several AoAs, using a least-mean-squares method;
- 5) gather BLE packets in two experimental locations (anechoic chamber, and outdoor area);
- 6) perform data analysis on the achievable AoA and subsequent positioning estimation.

The rest of this paper is organized as follows: Section II summarizes related work on localization systems and AoA computation methods, Section III presents our approach, including system topology and design of the receiver device, Section IV explains our experimental setup, Sections V to VII present and analyse the experimental results, Section VIII briefly presents some additional preliminary experiments, and Section IX concludes the paper.

### II. RELATED WORK

Our approach for self-localization is fundamentally based on AoA. This is, however, only one of many existing technologies for localization [8], [46]. In this section we will primarily focus on AoA calculation approaches, not only for BLE, and briefly mention broader application contexts. Generally, the reviewed papers address methods for AoA computation that can benefit from a lower computational complexity relative to MUSIC [18] or ESPRIT [20], in order to allow for faster response in more compute constrained devices (e.g., edge computing for the IoT). When localization is employed for tracking of fast moving assets, fast computation of is even more critical, so hardware acceleration or especially machine learning techniques are vastly explored. Table 1 summarizes

the key points of the approaches most comparable to our own which are detailed further throughout the section.

### A. AoA AND LOCALIZATION

In [47], two pre-processing steps are analysed to improve the raw phase data sampled from a receiving array, before applying AoA calculation algorithms. Using two commercial devices, one transmitter and one receiver with a ULA with six elements, sets of 200 BLE packets per orientation, in a range of  $-90^\circ$  to  $90^\circ$  in steps of  $10^\circ$ , are stored. The setup is configured in an indoor location where the devices are 1 m off the ground and in line-of-sight, at a distance of 2 m from each other. The first pre-processing step is a non-linear recursive least squares method based on curve fitting of the raw data on an expected model of the data to observe. Secondly, a Kalman filter attempts to eliminate the effect of different oscillator frequencies between devices, which can introduce phase drift errors. Relative to MUSIC, the Root Mean Square Error (RMSE) is decreased by an average of  $3.9^\circ$ .

The work presented in [48] also applies pre-processing to the In-Phase/Quadrature (I/Q) samples, in three steps. The first are similar to [47], using a least mean squares method and a Kalman filter to address the same issues. The third step is a Gaussian filter method to compensate for AoA estimation errors due to the BLE channel used. The channel is known, as the approach relies on a connected mode, unlike most AoA implementations, where the transmitters employ advertising (i.e., broadcast mode). Using a pair of devices placed 1 m apart, where the receiver has a three-element ULA, the pre-processing improves the AoA calculation, with absolute errors under  $10^\circ$  within the range of  $-60^\circ$  to  $60^\circ$ . Like [47], for greater values of true AoA, i.e., cases where the incident wave is parallel to the array, the error increases significantly. This is a commonly observed behaviour in ULAs.

A Software Defined Radio (SDR) implementation on a Xilinx Zynq-7000 FPGA device is presented in [49], evaluating the performance of the MUSIC algorithm, with some portions of the operation performed via hardware modules implemented in the FPGA portion of the device, specifically, the control of the RF switch. As no BLE software stack is available for this device, the authors emulate the behaviour of several BLE protocol layers, and feed the FPGA with DF compliant packets for validation. The system's ARM processor (i.e., software side) receives the incoming I/Q samples as a data stream, extracts phase values, and performs pre-processing such as normalization, prior to executing the MUSIC algorithm. To the best of our understanding, the MUSIC algorithm is not itself accelerated in hardware. By using their own design of a microstrip antenna to compose a ULA with four elements, and a commercial radio module, the achieved RMSE is under  $5^\circ$ . The experimental setup consisted of one BLE beacon and the receiver, 3 m apart, with measurements taken for a range of  $-90^\circ$  to  $90^\circ$ .

The work in [35] focuses on indoor RTLS via AoA. The approach estimates both the azimuth and elevation angles,

by resorting to a dual channel arrangement of a UCA. The array contains six elements, where each is actually a pair of orthogonally polarized antennas. The array is complemented with an additional antenna at the center of the UCA, used as constant reference antenna for estimation of the frequency of the incoming signal, while switching is performed through the array elements. Similar to our previous approach for AoA based positioning and tracking [44], the authors apply a Kalman filter to mitigate the effects of a moving receiver, where the state vector of the filter is the current estimated 2D position and velocities of the receiver. Singular value decomposition methods are applied to the Kalman formulation to simplify matrix calculations and ensure that the state covariance matrix retains only positive values which the authors claim improves tracking accuracy. The experimental setup consists of a single receiver placed on a ceiling of an indoor location with an area of 6 m by 7 m, receiving transmissions from one omni-directional BLE tag. The antenna array was sampled in a circular pattern along its perimeter twice per transmission received. The authors characterize the positioning errors first with the beacon in static positions while also rotating it in place, and then perform a tracking test by traveling a trajectory along the room perimeter. The positioning error was above 0.5 m, with a maximum near 1 m.

A decentralized AoA calculation method based on ESPRIT is proposed in [50]. The authors highlight previous works on a decentralized variant of ESPRIT, d-ESPRIT, which circumvents issues such as noise or bandwidth limitations. However, in order to support faster calculation and tracking of a moving transmitter, a modification is proposed based on a network of sub-arrays followed by a consensus algorithm. The authors introduce a temporal component to the calculation of the covariance matrix of the received signal, in the form of a *forgetting factor*, which essentially combines eigenvalues of different sub-array elements throughout time allowing for an online application of the decentralized ESPRIT variant. For validation, a network of six identically oriented sub-arrays, each being a ULA with two elements, are used. By defining different sub-sets of the arrays, the consensus algorithm is applied, after a DoA is computed per sub-array, for a set of three different transmitters and 200 signal snapshots. The performance matches ESPRIT, while in turn requiring less computational power than d-ESPRIT. While tracking two moving sources, the RMSE of the AoA is  $1.1^\circ$ , surpassing a similar approach [55]. However, no details are given on the specific movement of the sources, or on positional calculation.

In [32], the focus is also given to implementation of AoA calculation methods that benefit from low complexity. As we did, the authors highlight the advantages of AoA over other techniques, but also that existing methods are not viable for IoT nodes or systems with limited computing power in order to achieve real time localization with AoA. While not focusing on BLE specifically, to the best of our understanding the authors propose simplifying the AoA calculation under

**TABLE 1. Summary of notable related works on AoA calculation and/or positioning (approaches do not report AoA or positioning results via the same metrics).**

| Work | Summary of Proposed Method                                 | Algorithm                        | Array    | #tx/#rx | Dist (m) | Experimental Summary                        | Range               | RMSE         |                     |
|------|--|----------------------------------|----------|---------|----------|---|---------------------|--------------|---------------------|
|      |  |                                  |          |         |          |   |                     | $\theta$ (°) | Pos (m)             |
| [47] | NLRSL <sup>1</sup> + Kalman filter                         | $\theta = f(\Delta\phi)^\dagger$ | ULA-6    | 1 / 1   | 2        | 1 position; 200 packets p/angle             | $\pm 90^\circ$      | 13*          | –                   |
| [48] | NLS <sup>2</sup> , Kalman and Gaussian filter <sup>3</sup> | $\theta = f(\Delta\phi)^\dagger$ | ULA-3    | 1 / 1   | 1        | 1 position; all BLE channels                | $\pm 60^\circ$      | –            | –                   |
| [49] | FPGA-controlled RF switch                                  | MUSIC                            | ULA-4    | 1 / 1   | 3        | azimuth motion (1° step)                    | $\pm 90^\circ$      | 5.0          | –                   |
| [35] | Matrix SVD; dual-channel UCA                               | MUSIC                            | UCA-6    | 1 / 1   | 3.5      | perimeter motion in 6x7m area               | (n.) <sup>6,*</sup> | –            | 0.2                 |
| [50] | Sub-arrays + consensus algorithm                           | d-ESPRIT                         | ULA-2    | 3 / 6   | 2        | 6 sub-arrays track 2 moving txs.            | 0–50°               | 1.1          | –                   |
| [32] | Multi-channel estimation + averaging                       | –                                | ULA-4    | 3 / 1   | 2.5–3.0  | azimuth motion ( $\approx 3.5^\circ$ step)  | $\pm 15^\circ$      | 1.5*         | –                   |
| [51] | Probability model of discrete space                        | ANGLE                            | Planar-8 | 1 / 3   | 2        | 10 positions in 6.5x9.5m area               | $\pm 90^\circ$      | –            | 0.57 <sup>*,*</sup> |
| [33] | FPGA LU/QR matrix decompositions                           | $\theta = f(\dots)^\ddagger$     | ULA-4    | 1 / 1   | 2        | runs at 20 arbitrary angles, 2 txs.         | N/A                 | 0.5          | –                   |
| [30] | FPGA calc. of phase diffs. + AoA                           | –                                | ULA-4    | 1 / 1   | 1–2      | azimuth motion (variable step)              | $\pm 10^\circ$      | 10*          | –                   |
| [52] | CNN <sup>4</sup> (6 layers, 2 outputs)                     | Regression                       | ULA-4    | 2 / 1   | –        | trained to estimate 2 AoAs <sup>8</sup>     | $\pm 60^\circ$      | 2.4          | –                   |
| [53] | 2 DNNs <sup>4</sup> (signal detect + sub-ranges)           | Classification                   | ULA-10   | 2 / 1   | –        | trained w/2 signals 2° apart                | $\pm 60^\circ$      | 0.25         | –                   |
| [54] | CNN Ensemble <sup>4</sup> (azimuth + elevation)            | Regression                       | UCA-9    | - / -   | –        | 5 CNN ensemble estimates 1 AoA <sup>5</sup> | $\pm 180^\circ$     | 0.6          | –                   |
| [29] | DNN (w/ single signal snapshot)                            | Regression                       | ULA-4    | 2 / 1   | –        | real-time runs on 2 SDR devices             | $\pm 90^\circ$      | 2.5          | –                   |
| Ours | Phase difference profile calculation                       | Profile phase                    | UCA-8    | 4 / 1   | 4–12     | 21 positions; 12x12m area                   | $\pm 180^\circ$     | 24           | 1.9                 |

<sup>1</sup>Non-linear recursive least squares (min.  $\phi$  between elements), <sup>2</sup>Non-linear least squares (I/Q curve fit), <sup>3</sup>channel compensation, <sup>†</sup> $\theta_k = \arcsin(\lambda\Delta\phi_k/2\pi d_k)$

<sup>4</sup>co-variance matrix as features, <sup>5</sup>10k generated signals w/ length 1024 at several SNRs, <sup>6</sup>360° azimuth and  $\pm 32^\circ$  elevation, <sup>\*</sup>derived from reported data

<sup>7</sup>for ANGLE-SS with 3 planar receivers, <sup>8</sup>500 generated signals w/ length 100 at several SNRs (5° step), <sup>‡</sup>derivation from eigen-values of LU or QR matrix

the assumption of systems based on a fixed number of transmitters. Each transmitter broadcasts on a fixed channel around a central carrier, separated by guard bands (in a way similar to BLE channels). The receiver sweeps the total bandwidth allotted for the system, and for its ULA, computes a final AoA per transmitter by averaging the AoA of each antenna pair. The approach is evaluated with 2 and 3 transmitters, and achieves comparable results to RMUSIC [19] and ESPRIT [20]. While the approach limits the system regarding the supported number of transmitters, and does not consider existing protocol concerns, it demonstrates that, for application specific scenarios, integration of AoA retrieval into the lower layers of the communications stack may lower the complexity of computation for constrained systems. The approach is further analysed in [56], where, similarly to our previous work [45], the effect of Gaussian noise is evaluated, along with oversampling and bit quantization (i.e., precision).

In [51] two novel algorithms for AoA calculation are presented. The first, ANGLE-SS, requires only a single signal snapshot. The second, ANGLE-SD, is comparable to MUSIC or ESPRIT, as it also relies on eigenvalue decomposition. However, either case relies on formulating a so called fixed probabilistic model. This is based on discretizing the localization space into a grid, so that a simpler formulation of the likelihood of the localization of the emitters can be derived. This first stage can then be fed as input to, ANGLE-SS, ANGLE-SD, or other methods, such as MUSIC, with which a comparison is performed. The ANGLE algorithms do not impose any particular array arrangement, e.g., they are applicable to UCA (unlike most of the presented cases), or other array element distributions, such as the planar array used for the experimental evaluations where 8 elements are placed non-homogeneously (i.e., not uniformly distributed). Experiments were conducted for a carrier frequency of 1.35 GHz, and a constant tone of 80 MHz, in an indoor office space of 6.5 m by 9.5 m, using one signal source, and three receivers. The transmitter was placed in 10 possible

positions of the map, and 100 transmissions were gathered per position. By first performing the discretization step, then employing ANGLE-SS, ANGLE-SD, or MUSIC (as well as other methods), results show that the respective maximum absolute localization errors are 3.2 m, 1.2 m, and 1.9 m.

## B. FPGA AND HARDWARE IMPLEMENTATIONS

In [33] an FPGA accelerated approach for AoA calculation is presented. Although also based on first computing the covariance matrix of the signal, two alternative matrix decomposition methods, LU decomposition and QR decomposition are implemented, that provide simpler matrix calculations during the processing pipeline, relative to MUSIC or ESPRIT. However, the authors point out that, even via hardware acceleration, one of the computing stages scales with the number of antenna elements, and may constitute a bottleneck for arrays with more than four antennas. The methods had been validated in simulation in previous works, but no real-time implementations had yet been presented. For a ULA with 4 elements, and by implementing the DoA calculation entirely in a Virtex-5 FPGA, the performance is tested for one and two signal sources. The array is designed to support a constant tone of 900 MHz. The AoA calculation pipeline for the target device operates between 40 MHz and approximately 65 MHz. Through simulation, the RMSE for two signal sources is approximately 0.02° for an Signal-to-Noise Ratio (SNR) of 0 dB, and the average absolute AoA error for two sources placed at four positions using a real-world setup is approximately 0.5° for an SNR of 10 dB, but the distance between receiver and sources is not reported. Further analysis of other types of matrix decomposition are performed by the same authors in [57], now including comparisons with FPGA implementations of MUSIC.

In [30] a very low level approach for AoA estimation is presented, which is based on a direct full hardware implementation of the calculations integrated into the receiver array. The approach is based on direct comparison of the

signal received by the two elements of the array at a time, by iteratively adjusting the phase of the normalized signal samples, until the signals match. The operations are performed in the digital domain, with a micro-controller unit adjusting the phase of one of two phase locked loops. For a ULA of 4 elements and a carrier frequency of 3.35 GHz, the AoA detection is tested for one source placed between 1 m to 2.2 m from the array. The source is moved parallel to the array, measuring the AoA from a range of  $\pm 10^\circ$ , with an absolute error of up to  $0.4^\circ$ . As the authors point out, the experiments assume some ideal conditions, but the capability to extract the AoA at the physical layer level promises viability due to its low-power cost, and because it is free from upper protocol layers (i.e., it is protocol independent and the adaptive beamforming could be left to the radio hardware itself), as previous works have also indicated [32].

In addition to these, other works have demonstrated that hardware accelerated implementations can compute the AoA quickly [58], [59], [60], even when relying on conventional matrix based methods (e.g., MUSIC or ESPRIT). The recent advances in FPGA technology coupled with the increased requirements and predicted uses for AoA justifies further research on hardware accelerated approaches.

### C. NEURAL NETWORK BASED METHODS

Yet another class of approaches for extracting channel characteristics, including AoA, include machine learning techniques, such as Neural Networks (NNs). The common motivator of these approaches is to outperform the state-of-the-art MUSIC and ESPRIT regarding computational complexity, accuracy, speed, and required power consumption. Although the number of works are numerous [29], [52], [53], [54], [61], [62], [63], [64], we present only some cases in deeper detail. Most commonly, the approaches rely on feeding the neural networks with the covariance matrix of the received signals from the antenna elements, with some topology variations.

In [52] the authors also recognize that AoA is promising for localization in indoor scenarios, with its major issue being the multi-path effect. A Convolutional Neural Network (CNN) approach is presented which estimates the AoA when orthogonal frequency division multiplexing is employed (i.e., multiple sub-carriers), while also subject to a multi-path effect. That is, the training and testing data is affected by reflections which are fed into the CNN model. The learned model then produces the AoAs of up to a maximum number of multiple path signals. The input features are the eigenvectors of the covariance matrix of the received signal samples. The simulations focus on a ULA with four elements, two signal paths, and 16 sub-carriers. Several levels of SNR are evaluated, but for 20 dB, the approach achieves a RMSE of only  $2.4^\circ$ , versus the  $17.4^\circ$  error attainable by MUSIC [18].

In [53] the authors address this with a sequence of two Deep Neural Networks (DNNs). The features fed to the DNNs are taken from the upper triangular portion of the signal correlation matrix of the array. The first stage is

responsible for reducing the size of the training sets, which then feeds a second stage composed of multiple DNNs, each corresponding to different physical subsets (i.e., antenna elements) of the ULA. The formulation is thus a classification problem, where the angular accuracy depends on the number of neurons between the two stages. For validation, the authors consider the effect of multiple (two or three) uncorrelated signals arriving at the same ULA containing 10 elements, and a synthetic dataset, where the AoA varied between  $-60^\circ$  and  $60^\circ$ , with DNN configurations providing an angular accuracy of  $1^\circ$ . The approach results in a RMSE roughly double that of MUSIC, but maintains a near constant inference time of the AoA regardless of the number of input data (15 ms), while MUSIC scales linearly with the signal length.

Estimation of both the azimuth and elevation AoAs are addressed in [54] via an ensemble of neural networks. The CNNs is fed with the covariance matrix values, and directly outputs the elevation angle (presumed greater than zero), and sine and cosine values for the azimuth angle, to disambiguate the quadrant. Like our work, and unlike most other works reviewed, the models target a UCA of antennas. For an array with nine elements, and training an ensemble of five different CNNs where each input signal is composed of 1024 snapshots, the approach can achieve a RMSE of  $0.59^\circ$ , requiring only 3.3 s to process 10 thousand test signals, while MUSIC requires close to a minute to process 100 signals.

In [29] a DNN system is evaluated which requires only a single temporal snapshot of the signal received by the array to estimate the AoAs from the multiple signal paths received (up to two). The authors note that, unlike the vast majority of other works which rely on synthetic datasets, they have collected several datasets with field experiments, using a four-element ULA, in both line-of-sight and non-line-of-sight conditions. The authors have made their implementation and datasets publicly available. The AoA accuracy is evaluated for a range of frequencies and modulations, using a hybrid classification and regression model implemented on a low-cost SDR device. The solution aims for low-power and low computational cost, so that RTLS is possible in constrained devices. An RMSE of  $2.5^\circ$  is achievable, and 60 samples per second can be processed on a Raspberry Pi 4, which is approximately  $7\times$  faster than MUSIC on the same device.

### D. MIMO AND MASSIVE ARRAYS

Finally, DF is only a part of a larger eco-system of future communications, as it constitutes a promising component to fully realize future 5G/6G communications through MIMO systems based on massive arrays. Directional communication would decrease interference, thus improving overall efficiency. Since the use of millimeter waves implies smaller antennas, future devices with smaller footprints could benefit from massive arrays to establish directional MIMO communications via fast beamforming [3].

However, as [65] highlights, conventional methods for AoA estimation, e.g., MUSIC, ESPRIT, are even more

prohibitively complex for arrays with a massive number of elements. Instead, two DNN based methods are tested for the estimation of the AoA in massive MIMO systems. The two DNN presented model the entire massive MIMO system, including spatial features of the array, resulting in estimation of the DoA, which is then used to aid the channel estimation model. Similarly, in [66], a DNN is employed at the base station to derive channel features unknown to the base stations, among which the AoD at the source. Training is performed by using the measured AoA, signal strength, and relative delays of multiple paths, as seen by the receiver array. Four learning strategies are employed, demonstrating that the AoD is extracted as to allow efficient alignment of beams between sources and the base station, at low computational cost.

In [67] lens antennas are proposed to support AoA calculation in massive antenna arrays, along with an algorithm which determines the first arrival path, selects a subset of antennas of the array, and thereafter discards signal components belonging to reflections. The localization of a single source is evaluated, relying on a ULA of lens antennas, whose properties allow for the reduction of the total computational complexity required, since the signal focusing excites only a subset of antennas on the array. Again this demonstrates that AoA based localization capabilities should lower in the communication stack, to alleviate software processing. After the input data has been reduced by antenna selection, several state-of-the-art algorithms are employed [18], [19], [20] for a ULA of 100 elements. The proposed method computes the AoA with RMSE comparable to approaches without the use of lenses, for a range of  $-40^\circ$  to  $40^\circ$ .

### E. CLOSING REMARKS

The field of AoA estimation has experienced a burst of activity due to its mentioned foreseen applications. However, approaches often aim at different performance metrics, such as AoA estimation accuracy (under different conditions which are difficult to compare fairly), computation speed, or hardware cost. Furthermore, some works are simulation based, focusing on algorithmic components and relying on synthetic datasets, as real data sets are difficult to obtain. Further validation is required using real-world setups, as especially CNN/DNN based approaches can be prone to over-fitting. The interference of other unrelated signals within the band of interest is also a significant factor which requires further research. On the other hand, approaches that focus on real-world setups are limited in their scope, due to practical limitations in setting up a testing environment which is easy to scale and where all factors can be accounted for.

In this work, we present a self-localization approach for BLE devices, where we endeavour to explain the core topology, present and implement the algorithmic components, and collect a dataset with a real-world experimental setup using an in-house BLE receiver with an 8-element UCA.

### III. PROPOSED APPROACH

As mentioned in Section I, a typical AoA based topology for indoor localization is based on the use of receivers equipped with antenna arrays, which are fixed anchors connected to wall-power, ethernet, and may be supported by additional wireless protocols, like Wi-Fi. These receivers collect transmissions from devices wishing to know their location, send the respective AoA to a centralized system, which communicates the position back to the original device [13].

In contrast, we propose to invert this topology, using instead low-cost battery powered omni-directional beacons as the fixed anchors, and the devices with antenna arrays are instead mounted on mobile receivers. Specifically, the beacons are BLE tags operating in broadcast, i.e., connectionless mode. By receiving DF enabled packets from the beacons, the mobile receivers can triangulate their position using the AoA of each received transmission, and the known absolute fixed position of the identified beacon.

While the former approach is apt for localization where the number of mobile assets are unknown, our topology instead aims to target industrial scenarios, such as factories, warehouses, shipping ports, or similar. In such cases, the number of devices which wish to self-locate is assumed to be known (e.g., forklifts or other vehicles), and is low in comparison with the number of fixed antenna array anchors that would otherwise need to be installed at a higher cost throughout the entire area of interest if relying on the conventional approach.

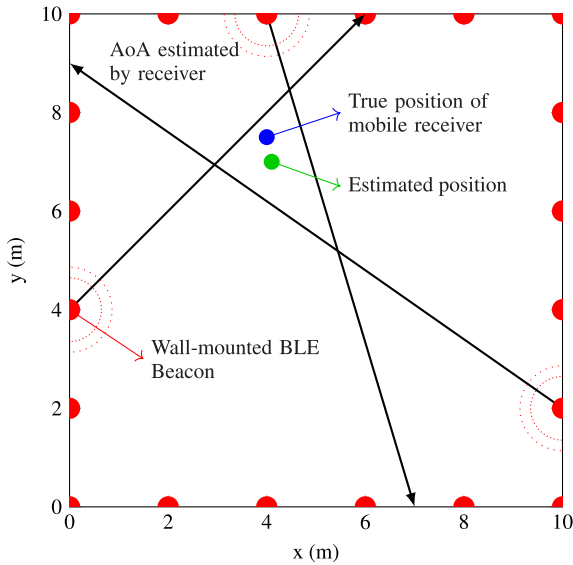
Instead, our topology can be deployed in legacy locations without wall-power or ethernet infrastructure to support the antenna array anchors (i.e., gateways), accelerating progress towards Industry 4.0 applications. Allied with the low-cost of BLE beacons, the topology is scalable in two ways. Firstly, if greater localization accuracy is desired, additional beacons can be placed throughout the area of interest, providing more data for position estimation. This also means that, unlike fingerprinting based approaches for instance, there is resilience to beacon failures (e.g., beacons exhausting their battery or malfunctioning). Secondly, additional self-locating devices can be added to the system without imposing additional computing requirements, as there is no centralized system.

In this paper, we do not yet evaluate the tracking of the receiver in motion (although we have done this via simulation [44]). Instead we focus on the first stage of determining the real-world data quality attainable, the resulting AoA accuracy, and subsequent position estimation error for our BLE 5.1 UCA at several static positions.

Our results show that multiple packets from the same beacon while the receiver is static are currently required to achieve good localization. We analyze these results and comment on potential future solutions to the pinpointed issues. In spite of this, we have shown that localization is achievable, even with a small number of beacons, and

believe that this is a contribution towards enabling self-driving factory vehicles based on wireless indoor localization technologies.

We now explain the top level view of this topology, showing how localization can be realized, the core concepts of AoA calculation based on utilizing antenna arrays, the basics of BLE's for support of AoA based DF, and the design of our UCA and algorithms for AoA and position estimation.



**FIGURE 1.** Principle of our localization topology (illustrative 10x10m area) based on mobile receivers with antenna arrays, and fixed omni-directional beacons (in red). The receiver computes the AoA of a received transmission, represented as the vectors originating from the beacon position. Several vectors (three or more) are combined to produce a candidate area.

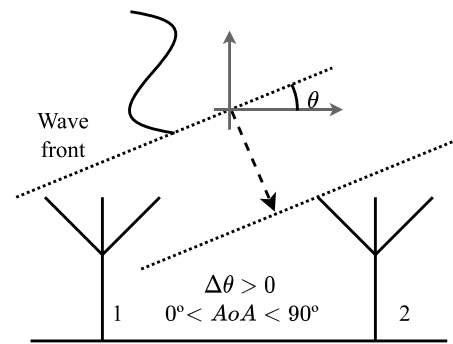
**A. SYSTEM TOPOLOGY**

Figure 1 illustrates the core of the proposed concept. The figure represents a top down view of an area within we wish to compute the location of the receivers. The red dots are the wall mounted beacons (which could be placed at any locations within the space). By receiving transmissions from these beacons, the respective AoAs can be combined into the receiver's location. In an ideal scenario, two vectors would be sufficient to determine a correct intersection point (i.e., real location). However, due to noise, two or more are required to form, at least, a candidate area for the receiver's location. The signal processing involved is later explained in Section III-C.

A number of parameters can be varied in this topology, which influence the final positioning solution, as we have previously evaluated [44]. For instance the size of the area of interest, the number of beacons placed and their locations (i.e., beacon density), the periodicity of beacon broadcasting, strategies like accumulating multiple packets from the same beacon to estimate a more accurate average AoA, how many AoAs should be used to estimate a position, and the specific AoA calculation method used (especially in regards to the computational time required versus accuracy). An adaptive

model that takes all such parameters into consideration will likely be the best solution. For example, if odometry reports a low velocity, then an adaptive algorithm can buffer a higher number of packets per beacon, assuming that the inherent error on the average AoA is consequently lower.

In this paper, we utilize only four beacons, one primary experimental area, and we do not yet advance to motion tracking, in favor of first evaluating the AoA computation accuracy for a low complexity method, and the resulting position estimations. The experimental setup and respective experiments are explained in detail in Section IV.



**FIGURE 2.** Illustration on how the Angle-of-Arrival of an incident wave front implies a difference in travel time between antennas, and therefore a phase difference for the same instant in time.

**B. KEY CONCEPTS ON ANGLE-OF-ARRIVAL (AoA)**

Figure 2 illustrates the key concepts for extraction of AoA via multiple antennas. A single pair of antennas can be used to determine the AoA of a wavefront. As the angle varies, the distance the wave must travel (at the constant speed of light), varies accordingly. By knowing the physical distance separating the antennas, and by determining the time difference of arrival between the pair, simple trigonometry can be used to derive the angle.

In order to easily determine the time difference, the method relies on the fact that the sampled wave is a so called *constant tone*. By knowing the constant angular frequency  $\omega$  of this tone, the relationship between time  $t$  and phase  $\theta$  can easily be established as per Equation (1),

$$\theta = t\omega \tag{1}$$

This holds for any value of  $\omega$  since the speed of light,  $c$ , is constant. Therefore, for a pair of antennas,  $\Delta\theta = \theta_1 - \theta_2$ , can be used to extract the incident angle as per Equation (2),

$$\phi = \arccos\left(\frac{\Delta\theta\lambda}{2\pi d}\right) \tag{2}$$

The physical spacing  $d$  between antennas must be less than half of the wavelength of the incident wavefront,  $\lambda$ , to prevent ambiguities in the phase value. Ideally, for a ULA, the AoA for each antenna neighbour pair can be computed, and a final value could be produced via an average. Since conditions are



not ideal, e.g., noise and multi-path effects, the previously mentioned state-of-the-art algorithms emerged.

In addition, although easier to implement, ULAs suffer from left-right ambiguities regarding the true AoA [68], [69]. However, in typical approaches the arrays are fixed gateways attached to wall structures, as we have noted, therefore this ambiguity does not constitute a problem since there is a fixed reference orientation. However, we target a mobile antenna array which may rotate around its center point. Therefore a UCA solves this ambiguity, as it allows for a 360° range, and provides data redundancy due to inherent symmetry, as explained in the next section and in Section IV-C.

Circular arrays are not novel [15], [70], but have experienced newfound interest due to the aforementioned growth in edge, IoT, and localization applications [43], [71], [72]. Since BLE recently implemented DF features, UCAs are now candidates for application relying on this protocol [41], [43].

1) DIRECTION FINDING SUPPORT IN BLE 5.1

The DF features added to the BLE protocol include the required Constant Tone Extension (CTE), which is appended to an ordinary BLE packet [13]. The tone is implemented by sending a sequence of digital ones over the carrier, effectively producing a constant frequency offset which depends on the data rate. For BLE, the rate can be 1 or 2 Mbits<sup>-1</sup>, respectively resulting in a CTE with a frequency of 250 kHz or 500 kHz [73]. If a packet is DF-enabled, then certain parameters must be configured. The primary parameters are the length of the CTE, the switching period, the sampling period, and the data rate. The CTE has minimum duration of 16 μs and a maximum of 160 μs, configurable in steps of 8 μs.

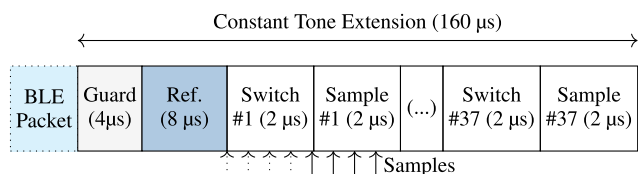


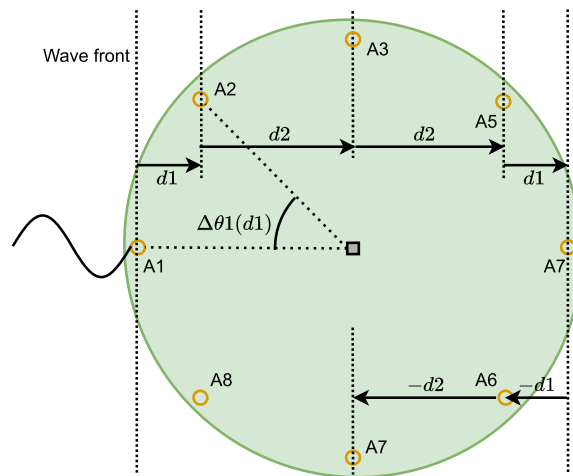
FIGURE 3. Simplified view of CTE extension to a conventional BLE Packet. (CTE length of 160 μs, switch period of 4 μs, and sample rate of 0.5 μs).

The components of the CTE extension are shown in Figure 3. The first 4 μs constitute a guard period where no sampling is performed, followed by a reference period of 8 μs. During this period, a total of eight I/Q samples are retrieved, that the receiver can use to determine the exact frequency of the received signal, since the carrier frequency changes according to which of the 40 available BLE channels is being used (from 2.40 GHz to 2.48 GHz).

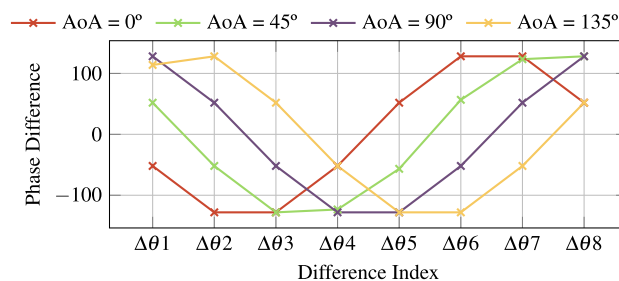
The rest of the CTE is occupied by an alternating sequence of one switch slot and one sample slot, which repeat at the rate defined by the switching period. These slot pairs have the same duration, which may be 1 μs or 2 μs. The switch slot is the time reserved to allow for the device (receiver or transmitter) to change the electrical connection to the target

antenna, using an RF switch. This switch must therefore be able to respond within the selected switch slot duration. The sample slot assumes the connection is stable, and I/Q samples are retrieved at the rate defined by the sample period.

Processing past this point depends on the antenna array arrangement, i.e., the samples retrieved must be processed at user-level based on the relationship of physical placement of the antennas, the configured sampling pattern, and the timestamp at which the respective samples are retrieved. The following section details this for our 8-element UCA.



(a) Diagram of our circular antenna array design, illustrating antenna placement and relationships between distance travelled by the wave-front and the phase differences between neighbour antennas.



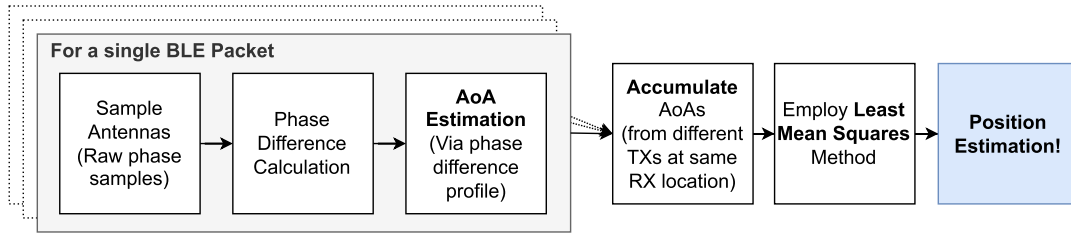
(b) Phase difference profiles for all 8 neighbour pairs, for four incident angles (using generated phase sample data).

FIGURE 4. Uniform circular antenna array design (Figure 4a) and resulting profile of sampled phase differences (Figure 4b). Due to the intrinsic symmetry, the distances and phase differences will be complementary, i.e., the resulting eight phase differences will form a sinusoid profile as a function of the AoA.

2) BLE 5.1 DIRECTION FINDING WITH 8-ELEMENT UCA

In Figure 4a shows the simplified circuit board design (with components and tracks omitted), which takes advantage of the described DF features. The center of the board contains an RF switch integrated circuit, used to select one of the eight antennas which is then sampled by a single radio device during the CTE. The antennas are evenly spaced, and sampled clockwise (this is configured via the embedded software).

As explained, the phase values that are sampled directly relate to the distance the wavefront must travel between adjacent antennas, and this distance must be less than half



**FIGURE 5. Complete processing pipeline of the proposed approach. Multiple BLE packets are received by a single mobile device, and conjunction of the respective AoAs allows for self localization. The approach has been previously validated in simulation with real-world AoA data [44].**

of the wavelength to prevent phase difference ambiguities. In this design, each antenna is therefore separated by a distance of 4.56 cm, less than half of the wavelength of the central frequency for BLE, 2.4 GHz, which is 12.5 cm. Considering this, and the minimum and maximum AoAs of  $0^\circ$  and  $90^\circ$ , for a single antenna pair, we may plug  $\lambda = 0.125$  and  $d = 4.56 \times 10^{-3}$  into Equation (2), we obtain:

$$0^\circ = \arccos\left(\frac{0.125}{0.0456} \frac{\Delta\theta}{2\pi}\right) \Leftrightarrow \Delta\theta = 0^\circ \quad (3)$$

$$90^\circ = \arccos\left(\frac{0.125}{0.0456} \frac{\Delta\theta}{2\pi}\right) \Leftrightarrow \Delta\theta \approx 131.5^\circ \quad (4)$$

That is, the maximum phase difference which can be plugged into Equation (2) is of approximately  $\pm 131.5^\circ$  ( $\pm 2.30$  rad), otherwise a value greater than  $\pm 1$  will be passed to the arc-cosine. However, we do not directly calculate one AoA per antenna pair using this formulation, and instead rely on the *phase difference profile* to attain a single final AoA. This profile is composed by the sequence of the eight phase differences ( $\Delta\theta_1, \Delta\theta_2$ , etc), and is itself a sinusoid due to the UCA symmetry. For example, the distance  $dI$  travelled between the first pair,  $A1-A2$ , equals the distance between  $A7-A6$ . That is, each of the eight phase difference values has an analogous value of an opposing sign occurring in its complementary pair. This results in phase difference profiles as shown in Figure 4b, where the profile suffers a horizontal displacement as a function of the AoA.

However, although we do not directly calculate one AoA per antenna pair, values greater than  $\pm 131.5^\circ$  still introduce issues. These values appear due to noise, and since a modulus is applied to constrain the profile to  $\pm 180^\circ$ , this introduces ambiguity by distorting the profiles. The AoA calculation algorithm we employed in these experiments was designed to be fast and simple, but is subject to such effects as it expects stable profiles. In Section VI-A we explain a processing step to heuristically attempt to address this (see Figure 14).

The waves shown in Figure 4b are ideal, since they are computed with phase samples produced synthetically using a dataset generator, based on the physical model of the board, the known frequency of the CTE for BLE, and the expected observable phase at each antenna at several time steps, as a function of the AoA. We used this to incrementally validate our algorithms, including the phase difference calculation already shown, and the angle estimation algorithm, both

detailed in the following section. We also use this to compare the expected phase difference profiles with those obtained in the experimental campaigns, (e.g., Figures 17 and 18).

### C. SIGNAL PROCESSING PIPELINE

Figure 5 illustrates the complete signal processing pipeline of our approach. The first three steps are the processing relative to a single BLE packet. Firstly, the receiver controls the RF switch to sample all antennas in a circular pattern. Each sequence of 8 samples corresponds to one antenna, and the length of the CTE allows for each antenna to be sampled four times, i.e., four sampling rotations of the board. This leads to further data redundancy (i.e., more than one phase difference per antenna pair), beyond the intrinsic profile symmetry.

From the raw phase samples, we then compute the phase difference profiles shown previously, and from the values of the phase difference per antenna pair (i.e., displacement of the profile), we extract the AoA. These two algorithms are explained in detail below. The receiver must accumulate at least three AoAs from multiple beacons to produce a candidate area, as explained in Section III-A. At this point in the processing pipeline, policy variations could be studied to account for the topology parameters in the aforementioned section, e.g., accumulating multiple packets from the same beacon to produce a single AoA, or averaging individual AoAs from the same beacon (either case would suffer for effects due to receiver velocity). In this paper, we will apply the former to attain viable AoAs, due to the quality of the phase profiles attainable per packet, show in Section IV-C.

The last three steps of the pipeline had already been validated via simulation (where AoA values were real-world data sampled from a commercial antenna array) [44]. The viability of phase difference calculation using our own design was recently performed in [45], focusing on the first two steps of the pipeline. This paper further analyses this data by presenting AoA and position estimations.

#### 1) PHASE DIFFERENCE CALCULATION

In a generic scenario, the calculation of phase differences between antennas would require sampling two antennas at once (i.e., at the same instant in time). Since this is not possible using only one radio receiver, RF switches are used to retrieve samples at a known pattern, and post-processing is

required to account for the variation in phase implied by the different time instants of phase sampling [13].

As explained in Section III-B, the first 8  $\mu\text{s}$  are reference samples used to fine tune calculations by deriving the exact carrier frequency. In [74], these samples are instead used to establish a linear progression model of the phase at the reference antenna, estimating phase values at future time instants to compute the phase differences to other antennas.

In our case, the switching period is set to 4  $\mu\text{s}$ , and since the 250 kHz frequency of the CTE implies that the phase varies 90° per 1  $\mu\text{s}$ , then each  $i$ -th phase sample from an antenna will have an equivalent value after 4  $\mu\text{s}$ . Consequently, we discard the reference samples, and directly compute the phase difference between two neighbour antennas using each  $i$ -th pair of samples. Also, we consider only advertising mode (only 3 BLE channels are used), and do not currently explore the suggested fine tuning based on channel frequency.

---

**Algorithm 1:** Calculation of Phase Difference Between Antenna Pairs for One Packet With  $N$  Phase Samples. Eight Antennas and Three Phase Samples per Antenna Are Assumed

---

**Data:** Array of  $p_i$  phase samples, of size  $N$   
**Result:** Array of  $pd_i$  phase differences, of size  $M$

```

1 nrSamplesPerAnt ← 3;
2 nrAntennas ← 8;
3 N ← (4 × nrSamplesPerAnt × nrAntennas);
4 p ← p(1, . . . , N) // Cut to 4 full rotations
5 d[N − nrSamplesPerAnt] ← zeros // Pair diffs. (3 per pair)
6 // Computes phase differences between antenna pairs
7 for i ← 0 to N do
8   for j ← i to j + 3 do
9     // One difference per sample j of each neighbour
10    d_i ← p_j − p_{j+3};
11   i ← i + 3;
12 // Apply bounding loop to individual differences
13 for i ← 0 to N − nrSamplesPerAnt do
14   if d_i < −180 then
15     d_i ← d_i + 360;
16   if d_i > 180 then
17     d_i ← d_i − 360;
18 // Compute final phase difference from 3 differences per pair
19 M ← N ÷ nrSamplesPerAnt;
20 pd[M] ← zeros // Final phase differences
21 i ← 0;
22 for j ← 0 to M do
23   pd_j ← mean(p_i, p_{i+1}, p_{i+2});
24   i ← i + 3;

```

---

In [45] we presented an earlier version of the phase calculation algorithm. The phase samples retrieved by our micro-controller were bound to  $\pm 180^\circ$ , meaning that the sequence of phase samples per antenna did not necessarily increase monotonically due to this wrapping effect. In a way similar to [74], we first reversed this wrapping effect by iterative comparison of the phase value of sequential samples, then calculated the  $N$  phase differences per antenna

pair, produced an average, and finally reduced the result to a range of  $\pm 180^\circ$ . However, the unwrapping phase and several modulus steps introduced additional ambiguities and was more computationally complex overall, versus the current version shown in Algorithm 1 (an implementation is presented in Listing 1, in Appendix A). Remember that in our configuration 8 samples are retrieved per antenna, totaling 304 samples for the entire CTE. However, 4 correspond to the switching period and are thus discarded from the input. Also, an earlier analysis in the anechoic chamber determined that the last sample during the sampling period is also affected by the start of the switching behaviour [45], so we discard it. The input array  $p_i$  thus assumes 3 samples per antenna, i.e., 96 samples per packet.

Array  $d$  holds 3 phase differences per antenna pair, given we have 3 samples per antenna, with each  $i$ -th sample spaced by 4  $\mu\text{s}$  (i.e.,  $360^\circ$ ). That is, for all useful samples  $N$  (line 9), we compute the difference to its corresponding sample in the next antenna (i.e.,  $p_j$  and  $p_{j+3}$ ). Afterwards, we apply a bound to the each computed phase difference (line 17). Finally, we compute  $M = 32$  average phase differences.

The procedure is applicable for any number of antennas and samples per antenna, but relies on having each  $i$ -th of two neighbour antennas separated by 4  $\mu\text{s}$  (i.e.,  $360^\circ$ ), as explained, to simplify calculations.

## 2) ANGLE-OF-ARRIVAL ESTIMATION

In Section III-B we illustrated how the AoA can be derived for a single pair of antennas, and further elaborated on how a UCA could provide the AoA, in the azimuth plane, without ambiguity in the range of 0 to  $360^\circ$ . Although 4 antennas spaced by  $90^\circ$  would be the minimum to allow for this, utilizing 8 allows for greater accuracy due to additional data.

One approach to compute a final AoA would be to compute eight individual angles of arrival per antenna pair, and combine this information. As we explained, we opted to explore a different approach and directly utilize the phase difference profiles to attain the AoA, by determining the horizontal displacement of the phase difference profile (as shown in Figure 4b). Specifically, we determine the displacement profiles relative to a reference profile that corresponds to the orientation established as the AoA of  $0^\circ$ .

Considering that the x-axis in the phase difference profile represents phase difference indices (i.e.,  $\Delta\theta 1$  is the difference for the first antenna pair), and that there are 8 differences, then the range from  $0^\circ$  to  $360^\circ$  is contained within this discrete range of 0 to 8. That is, the wave translates as a function of the incident angle and repeats every 8 x-axis values. Therefore, an offset of  $8 \div 360$  on this axis is equivalent to  $1^\circ$  of AoA variation, as per Equation (5),

$$\phi = \frac{X_c - X_{\text{zero}}}{8 \div 360} \quad (5)$$

where  $X_c$  is the value on the x-axis where the  $0^\circ$  reference wave intersects with  $y = 0^\circ$  ( $x = 4.5$ ), and  $X_{\text{zero}}$  is the analogous value for the profile of the AoA,  $\phi$ , to be computed.

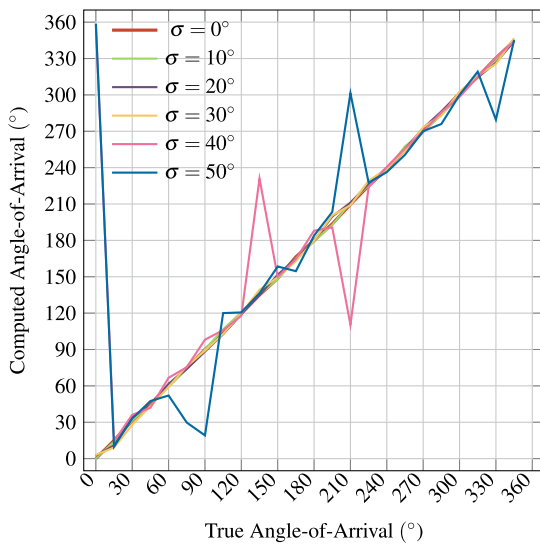
**Algorithm 2:** Angle-of-Arrival Estimation Based on Comparison With the Shift Relative to a Reference Profile for 0°

```

Data: Array of  $pd_i$  phase differences, and array size  $N$ 
Result: Angle-of-Arrival Estimation,  $angle$ 
1  $zRef \leftarrow 4.5;$  //  $xx$  value for AoA = 0° reference
2  $dDeg \leftarrow 8/360;$  // Shift along  $xx$  axis per deg. of AoA
3  $angle \leftarrow 0;$  // Estimated AoA
4 for  $i \leftarrow 1$  to  $N$  do
5   if  $pd_{i-1} < pd_i$  then
6     break
7  $time \leftarrow i - 1;$ 
8  $time \leftarrow time + \frac{-pd_{i-1}}{pd_i - pd_{i-1}};$  // Find exact intersection value
9 if  $time < zRef$  then
10   $time \leftarrow time + 8;$ 
11  $crossDiff \leftarrow time - zRef;$ 
12  $angle = \frac{crossDiff}{dDeg};$  // Estimated Angle-of-Arrival

```

Algorithm 2 illustrates the pseudo-code for this approach (an implementation can be found in Appendix A). Note that this algorithm only finds the first intersection with the y-axis. In this regard, there are two alternatives that can be explored to test if better results are attainable. Firstly, the algorithm could search for all four intersections (given the four rotations), produce four AoAs, and produce the average (eventually discarding any outliers). Alternatively, we can reduce the four rotations to an average profile with only eight values, i.e., take each phase difference value per antenna pair, and produce eight averages. This results in a profile with only 8 points, much like the example illustrated Figure 4b. In this paper, we employ the latter approach, and rely on the first ascending intersection located in the profile.



**FIGURE 6.** Output of the algorithm explained in Algorithm 2 when it is applied to phase difference profiles resulting from synthetically generated raw phase data with Gaussian noise where  $\mu = 0$ , for different standard deviations,  $\sigma$ .

Figure 6 illustrates the output of this algorithm when it is fed with synthetic raw phase data that we produce using the

mentioned dataset generator. In this case, we produce phase difference profiles corresponding to the angles from 0° up to 360°, with a step of 15°. We apply Gaussian noise to the samples, with  $\mu = 0$  and for  $\sigma$  from 0° up to 50°.

It is noticeable that up to 30° of noise, the approach seems resilient to any significant deviations from the true angle. Note that different but analogous plots could be produced by again generating one profile per angle, per noise value. This is merely to demonstrate that the algorithm, as presented (of low computational complexity), produces the correct AoA in the presence of clean data. If we produced a large number of profiles from synthetic data, produced the AoAs, and plotted the resulting average, all cases would trend towards the true AoA, as the noise has a mean value of  $\mu = 0$ .

Having shown how this algorithmic component of the approach works, we now briefly explain how this data can be used for position estimation. Also, as Section IV-C will show, the quality of data attainable in a real-world scenario suffers from significant noise. Thus, we will explain the additional algorithms required to extract higher quality data, in order to afterwards compute the AoA and position estimations.

3) POSITION ESTIMATION

The receiver position is estimated by combining AoAs from multiple sources, which is the core concept of our approach, reversing the conventional AoA topology for RTLS. Briefly, two or more AoAs, from fixed-position omni-directional beacons, are used to determine a point of intersection. Ideally, for a receiver in an immobile state, two vectors (i.e., AoAs) would suffice to determine an intersection in a known map. However, due to sampling noise, the intersection of only two AoAs can generate high positioning errors. Collecting additional AoAs also does not directly result in a single intersection point, as all measurements are subject to error, therefore producing only a candidate area for the position.

Additionally, although this is out of the scope of this paper, this issue is exacerbated by considering a moving receiver, since only one AoA can be received/computed at a time. Once multiple AoAs have been accumulated, each will correspond to a different true position of the receiver. Methods to address this are topic of future work.

In this paper we evaluate only the quality of the phase data, and the respective AoAs, to determine a positioning error of an immobile receiver. To do this, we rely on a state-of-the-art method [75], which is formulated as shown in Equations (6) to (8). Each  $n_j$  represents a normalized (i.e., unitary magnitude) vector associated with an AoA, whose origin is the location,  $a_j$ , of the respective beacon, such that:

$$n_j = [x_j, y_j]^T, \quad \|n_j\| = 1 \tag{6}$$

$$R = \sum_{j=1}^K c_j(I - n_j n_j^T), \quad q = \sum_{j=1}^K c_j(I - n_j n_j^T) a_j \tag{7}$$

$$R \cdot p = q. \tag{8}$$

This method computes a position estimation such that the sum of the distances from that estimated point to all edges

of the polygon defining the candidate area is minimum. We previously validated in it simulation [44], including tracking of a moving receiver. However, the simulation relied on random sampling of real-world AoA data retrieved from commercial antenna array boards. As future work, we will re-evaluate the simulation, relying on our own phase sample data-sets as a starting point, and implement the phase difference and AoA calculation algorithms we have presented.

In Section IV-C we present the AoA quality for a total of 21 map points, and respective position estimations, using one receiver and four beacons. We estimate the position using all four AoAs, and then using only three AoAs, and compare the difference in accuracy.

#### IV. EXPERIMENTAL SETUP

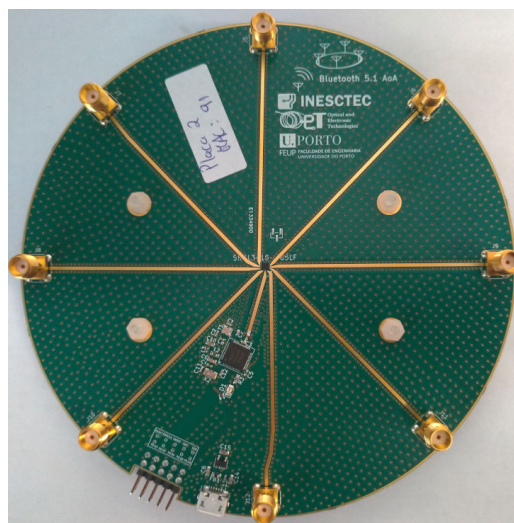
In this work we wished to evaluate the viability of self-localization by relying on the features introduced in the BLE standard for DF, as of version 5.0 [34]. Specifically, the introduction of features allowing for calculation of the AoA of a transmission, by a receiver. We have so far focused on system topology and algorithmic aspects of our work. We now present BLE receiver design, its general hardware and software characteristics and how they allow for gathering the phase data required, the setup used for data gathering runs, and the respective experimental evaluations we conducted.

Our exploration covered all three main components required the test our system topology, which is explained below. These components are a BLE 5.1 antenna array device of our own design, the configuration for acquisition of data (and acquisition of data for experimental evaluation), and the processing of the data to arrive at self-localization.

We fabricated the receiver design, and implemented the embedded software. We analysed if the AoA can be correctly computed in a controlled scenario, and then evaluated the quality of the phase profile data in a real-world scenario. This data required further off-board processing, which we will present, to obtain usable AoAs in order to finally evaluate the localization accuracy.

##### A. BLE 5.1 CIRCULAR ANTENNA ARRAY DESIGN

Figure 7 shows one unit of the fabricated design, without antennas mounted. The board is 13 cm in diameter, fabricated in a 1 mm thick FR-4 substrate. All antennas are equally spaced, i.e., at  $45^\circ$  steps. The main component is the Nordic Semiconductor nRF52811 micro-controller [73]. Since this device contains only one radio receiver (i.e., one radio pin), the integrated circuit at the center is an SP8T RF switch rated up to 6 GHz, a SKY13418-485LF switch [76], which is controlled via software to establish the desired sampling pattern. The antenna tracks from each SMA connector to the switch have the same length, and a width of 0.85 mm. Regarding interfaces, a 10-pin J-Link interface connector is visible, which is used for programming the nRF52811. Five pins are left as GPIO pins, two of which are used to establish UART communication. The micro-USB connector supplies



**FIGURE 7.** One unit of the fabricated board design, without mounted antennas. In our experiments, we mount all 8 antennas on one board, to serve as receiver, and mount a single antenna on four additional boards, programmed as beacons.

only power, followed by a 5 V to 2.5 V regulator [77], that in turn powers the nRF52811 and the RF switch.

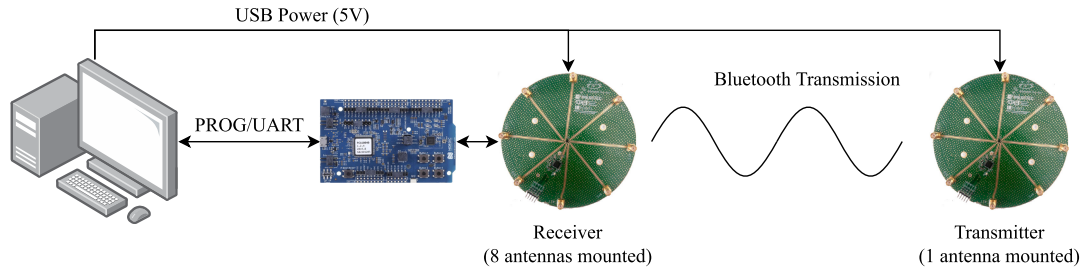
The micro-controller is programmed in C, using Nordic's proprietary software development kit, which already includes the so called *soft-device* which implements the BLE protocol stack. This includes control registers for DF, which by a one-time configuration at device boot sets the CTE length, the sampling and switching periods, and the sampling pattern. The pattern is fed into the RF switch via three GPIO pins.

For all our experiments we utilized a CTE length of  $160 \mu\text{s}$ , a switching period of  $4 \mu\text{s}$ , a sample rate of  $500 \text{ ns}$ , and configure a circular sampling pattern along the board perimeter (i.e.,  $A1$ ,  $A2$ , etc). Given the configured values, the embedded software samples each antenna four times, i.e., four sampling rotations. This leads to a total of 32 phase differences, where we have four differences per antenna pair.

##### B. HARDWARE AND DATA GATHERING SETUP

In addition to the UCA design itself, we utilized a Nordic nRF52840 Development Kit [78] for programming and to act as a pass-through for serial communication. Figure 8 illustrates the connections between system components for data gathering, namely, phase samples from the received packets, which we then process offline. The setup is applicable to all experimental campaigns, described in the next section. Appendix C contains additional photographs of the setup of device connections.

The host computer (laptop) communicates with the development kit via USB to program the target board with either the beacon or receiver software. An additional USB cable from the laptop powers the receiver, and the development kit and receiver communicate using the free GPIO pins in the J-Link pin-header. The boards programmed as beacons are



**FIGURE 8.** General data gathering setup for all experiments. The embedded software in the receiver transmits the phase samples per received packet to the workstation.

powered by battery packs with 4 AA batteries in series, but with a USB output regulated to 5 V.

As mentioned, the nRF52811 is programmed in C, via the described development kit setup. Therefore, we can utilize the same design as either a receiver, or as a beacon. For our experiments, there is one receiver, and we program the four remaining units with different embedded software which transmits a CTE enabled packet, tagged with a beacon identifier, and set the RF switch to one track, where a single antenna is mounted on the board. All beacons use the same antenna for transmission. The behaviour of the beacon and receiver software, as it executes in the nRF52811 micro-controllers, can be summarized by the simplified pseudo-code shown in Algorithm 3 and Algorithm 4, respectively.

**Algorithm 3:** Simplified Pseudo-Code of Behaviour of Embedded Software for Beacon Device

**Result:** Periodic BLE 5.1 Transmissions for Direction Finding

```

1 Function main () is
2   // Set RF switch pin drive and select one output antenna
3   controlPinConfig ();
4   // Configure TX power to 4dBm, 1Mbit
5   // data rate, and CTE to 160us
6   radioConfigureTX ();
7   while true do
8     // Send packet where payload is this beacon ID
9     sendPacket (beaconID);
10    // Random delay between 0 and 250 millisecond
11    delay (randTime ());

```

The beacon code, summarized in Algorithm 3, simply configures the radio device to enable the DF features, transmission power and data rate, and sets the CTE duration to 160  $\mu$ s, which is the maximum allowed by the protocol. It then constantly broadcasts packets whose only payload is the beacon ID (we employed the first 2 bytes of the MAC address), followed by a random delay between 0 ms and 250 ms. This prevents any undesired periodicity in transmissions which, when using multiple beacons (i.e., Section VI), could result in packets from some beacons to never be received.

The receiver code is more complex, and can receive commands to change switching and sample rates, perform

**Algorithm 4:** Simplified pseudo-code of behaviour of embedded software for receiver device

**Result:** Reception of BLE 5.1 Packets for Direction Finding, With UART Transmission of Phase Samples

```

1 uint32_t samples[2000] // Shared memory for Mag-Phase
2 uint8_t bOut[5] // Output UART Bytes
3 uint8_t rxPacket[2] // Shared memory for packet payload
4 Function processSamples () is
5   // "getAmount" returns number of retrieved samples
6   // (first 8 reference antenna samples are not used)
7   for i ← 8 to getAmount () by 8 do
8     uint16_t phase // Get best 3 samples (2 bytes each)
9     for j ← 0 to 3 by 1 do
10      phase ← samples[i + 4 + j] & 0x0000FFFF;
11      bOut[0] ← (phase & 0x00FF);
12      bOut[1] ← (phase & 0xFF00) >> 8;
13      sendUART (&bOut);
14 Function main () is
15   // Set RF switch pin drive
16   controlPinConfig ();
17   // Configure CTE to 160us, switch period to 4us,
18   // sampling period to 0.5us, and a
19   // circular sampling pattern
20   radioConfigureRX ();
21   while true do
22     // Blocking wait for radio event; "samples"
23     // is filled by radio during CTE at defined pattern
24     receivePacket (&samples);
25     // First send ID of beacon to workstation
26     bOut ← rxPacket;
27     sendUART (&bOut);
28     // Transmit phase samples via UART
29     processSamples ();

```

a soft-reset, enable/disable UART transmission, or control the amount of samples sent to the host laptop. For brevity, Algorithm 4 shows a simplified version focusing on packet reception and processing.

Firstly, the pins which will drive the RF switch are configured, and the DF parameters are set. Namely, the CTE duration must be equal to those sent by the beacons, the switching and sample periods are set, and finally the

circular switching pattern is configured by writing the desired sequence to a shift-register built-in to the radio hardware.

Note that, although the switching pattern is automatically applied during CTE reception, the nRF52811's radio does not directly compute phase differences, since it has no direct knowledge of the switching pattern and, especially, no knowledge of the physical layout of the antenna array being used. That is, the DF features are not natively supported by hardware at this point. It is the user level software that must parse the phase samples and know which and how many samples correspond to each antenna, and perform calculations based on the array used. As mentioned previously, future devices may benefit from full hardware integration of DF computation to allow for faster response, for cases where the array configuration is known [30], [32].

After configuration, line 27 is a blocking wait for a packet reception. When this event occurs, the radio fills a shared data array in memory, *samples*, with all samples that are received. The radio can deliver phase samples either in magnitude-phase format or in I/Q format. We use the former format and directly handle the radio's provided phase values. Each sample is saved as a 16 bit integer with a fixed-point value range of  $[-201, 201]$ . After a packet reception, the beacon ID is first sent to the computer. The procedure in line 7 then processes all magnitude and phase samples retrieved. The magnitude value is not currently used, and we transmit only the phase samples to a computer. The first 8 samples correspond to the reference period specified by the BLE 5.1 protocol, which we discard. For each eight samples retrieved for each antenna, the first four are sampled during the RF switching behaviour, and therefore are non-utilizable combinations of the RF signals of multiple antennas. For reasons explained in Section III-C, we utilize only three of the samples retrieved during the sample slot. This results in the aforementioned 96 samples per BLE packet, for the default configurations. The UART baud rate used was  $11.52 \text{ Kbits}^{-1}$ .

### C. EXPERIMENTAL EVALUATIONS

Using our board design, algorithms, and data gathering setup, we perform several experimental evaluations with data gathered in two environments. Firstly, in Section V, we verify the functional correctness of the design's operation using data collected in an anechoic chamber. Specifically, we confirm that phase profiles are achievable as expected, and analyse their coherency between received packets, followed by calculation of AoA and respective error versus the true angle.

The remaining experiments focus on data collected in an outdoor test area. In Section VI, we gather DF enabled packets from multiple beacons for a set of positions in this area. We then evaluate the quality and coherency of the resulting profiles (Section VI-A), which prove to be significantly noisy. Consequently, we introduce additional processing to attain cleaner phase profiles per map position (Section VI-B), and verify if the resulting post-processed profiles match those expected for the respective physical

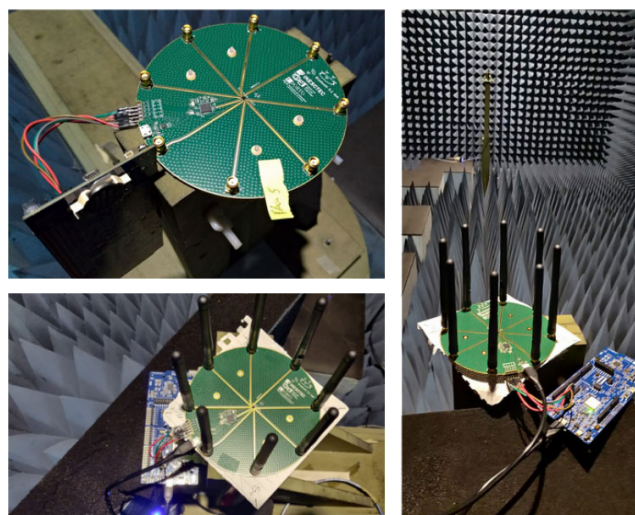
map positions, and that the profiles remain consistent for the same relative receiver-beacon orientation regardless of distance (Section VI-C).

In Section VII, we compare the achievable AoA estimates with and without these post-processing steps, present their error relative to the true angles, and utilize the best AoA values to compute position estimations. We estimate the receiver positions for two scenarios: using all four available AoAs per map position to produce a candidate area, or using only three AoAs, via multiple runs of randomly sampling, demonstrating a degradation in performance.

Finally, in Section VIII, we present some preliminary experiments regarding positioning accuracy in an indoor area, a hardware accelerated implementation of AoA calculation, and use of NN to derive the AoA.

## V. EXPERIMENT A: VALIDATION IN CHAMBER

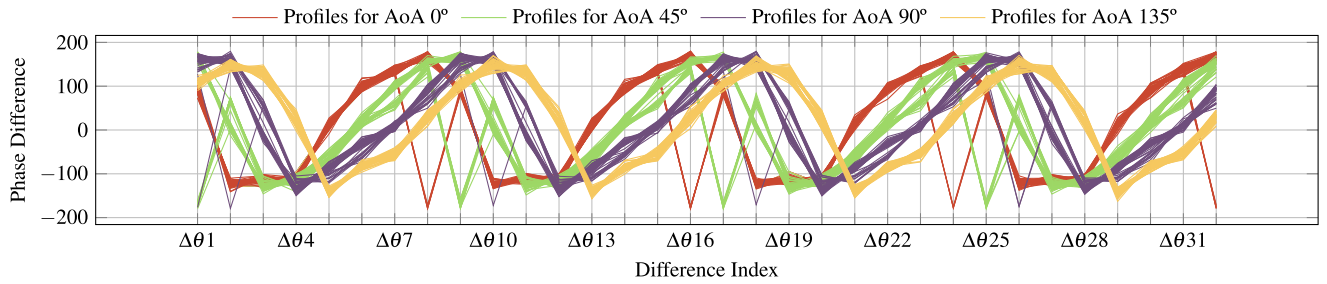
In Section III we demonstrated that, if the phase difference profiles can be extracted as per the model in Figure 4, then the AoA is derivable, as per Algorithm 2. We now verify if our fabricated antenna array receiver and embedded software produce phase data in accordance with the expected model, firstly by computing the phase profiles, followed by the AoA.



**FIGURE 9.** Setup for retrieval of packets in anechoic chamber. The receiver is mounted on a rotating platform, and a single beacon is placed 5 m away in a static mount. 100 packets are retrieved for a full rotation, with a step of  $15^\circ$ .

### A. PHASE DIFFERENCE CALCULATION

The physical setup in the anechoic chamber is shown in Figure 9. The receiver is mounted on a revolving platform, while a single transmitter is mounted on a static support. We collect 100 packets per orientation, while rotating the receiver in  $15^\circ$  steps. The beacon and receiver are approximately 5 m apart. The phase samples were retrieved into the computer, which was within the chamber with all wireless connectivity disabled (i.e., Wi-Fi and Bluetooth), to prevent interference.



**FIGURE 10.** Superposition of phase difference profiles for four incident angles. Per angle, 100 packets are retrieved, and Algorithm 1 is applied, and consistent (i.e., overlapping) profiles are observed per angle.

Figure 10 shows the resulting profiles for four example orientations. As explained, we can compute up to 32 phase differences, due to the CTE duration and sample rate. Since the orientation is constant during sampling, the phase difference profile itself is expected to be periodic, which is the observed behaviour. Additionally, the consistency of the profiles is confirmed. Each plotted sinusoidal wave is actually the superposition of 100 profiles, i.e., per angle, each of the 100 packets produces very similar results. Additionally, we also confirm the horizontal displacement of the phase profile wave, as a function of the relative orientation of the devices.

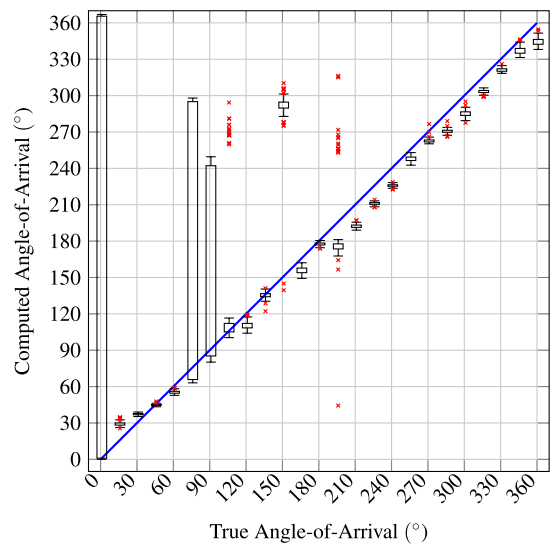
Finally, given a 15° step, and the 100 packets collected per step, we can compute 25 mean profiles for a full rotation. That is, we can compute the mean value of each phase difference (e.g.,  $\Delta\theta_1$ ), for a given orientation, and the respective standard deviation. For brevity, to give a general idea of the precision by computing, we present only the average standard deviation of all of the 32 phase differences for all 25 profiles, which is of 15°.

**B. ANGLE-OF-ARRIVAL CALCULATION**

We have shown that the phase difference profiles can be calculated as we presented, and that they behave as expected, shifting as a function of the relative orientation of the devices. We now compute the resulting AoA and respective accuracy, Algorithm 2, which we have already demonstrated produces the expected AoA when fed with ideal data, in Section III.

Figure 11 shows a plot similar to Figure 6. However, we now present the distribution of the computed AoAs as box plots, again starting from 0° up to 360° with a step of 15°. We observe that the mean of each box plot closely matches the true angle, shown as a solid line. Outliers for each case are shown in red. When considering all boxplots, outliers represent 11.8 % of all 2500 computed AoAs.

There are however some notable deviations from the true angle for some test cases. The case for 0° is easiest to explain: any small error will lead to an AoA of, e.g., 358°, which in truth is only an error of -2°. We did not however post-process the data to account for this specific case for purposes of showing the plot in Figure 11. Regarding the other cases showing significant deviations, at present, we can only attribute this to an erroneous alignment of the



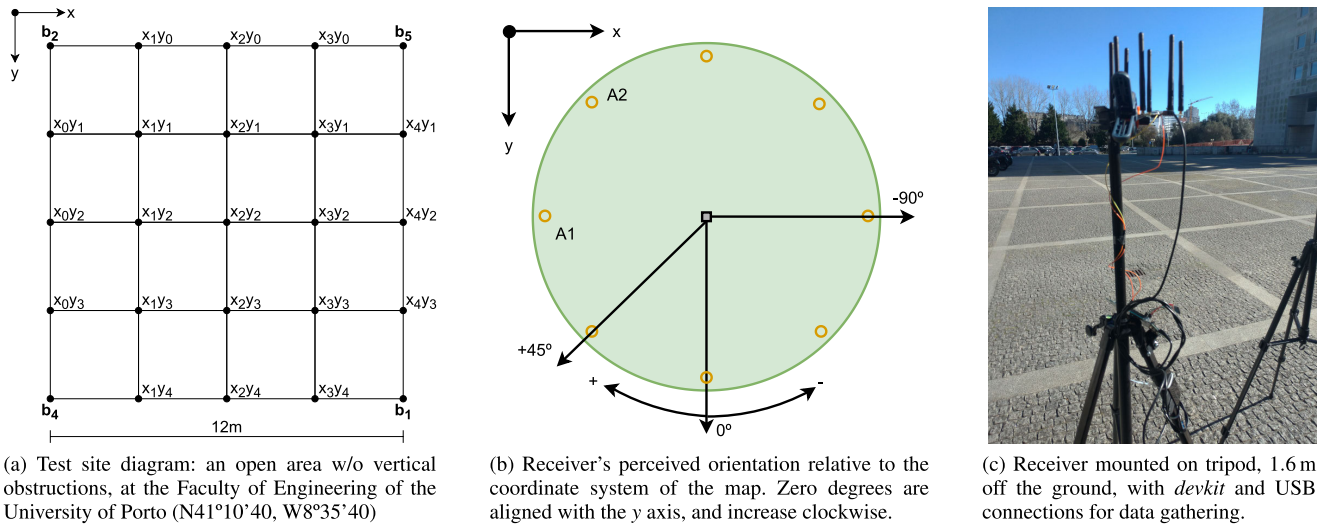
**FIGURE 11.** True angles vs. respective 100 samples of measured angle for our 8 antenna design in the anechoic chamber.

boards during setup (e.g., a true angle of 150° results in a consistent estimate of approximately 280°), or influence of other wireless devices mistakenly left within the chamber, or an improperly closed chamber door. We can identify these outliers as the orientations for 75°, 90°, 105°, 150°, and 190°.

Regarding the RMSE, we first correct the aforementioned deviation for the special case of the 0° bin (e.g. by applying full rotations such that samples like 358° become -2° as per the example). Considering the 15° step, and the 100 packets per step, we now have 2500 AoA values, for which we can compute a RMSE of 45°. However, this calculation takes into account the aforementioned outliers. When not taking these into consideration, the RMSE of the remaining 2000 computed AoAs versus the true AoA is 10.7°.

Regardless, the presented algorithms produce considerably accurate values in the presence of ideal data. Most cases produce boxplots with few outliers, and the mean is only slightly deviated from the true value. Additionally, this deviation appears to be a near constant offset for all bins. This could be corrected by adjusting the value of the reference wave, explained in Algorithm 2, leading to a lower RMSE. In fact, the average of this offset, i.e. the average of the center





**FIGURE 12.** Diagrammatic representation of the test area (Figure 12a), the receiver's established reference for zero degrees relative to the map (Figure 12b), and photo of tripod assembly of receiver (Figure 12c). The beacons are mounted on equal tripods in the vertices of the map.

of each boxplot to its true value, is  $9.15^\circ$ , with a standard deviation of  $5.2^\circ$ . Regardless, the objective was to illustrate that, in a controlled environment, our receiver and embedded software produce phase values which, when processed by the presented algorithms, compute AoAs which are coherent with the groundtruth. The next section evaluates the phase difference profile attainable instead in an outdoor area.

**VI. EXPERIMENT B: VALIDATION IN OUTDOOR AREA**

After functional validation of the device and algorithms in controlled conditions, we proceeded with gathering data in the outdoor area, using a setup with four beacons, and one receiver. The objective was to evaluate the quality of the phase difference profiles, per packet, and to advance towards AoA calculation and position estimation. Figure 12 illustrates the essential components of the setup. Additional photos of the setup can be found in Appendix C.

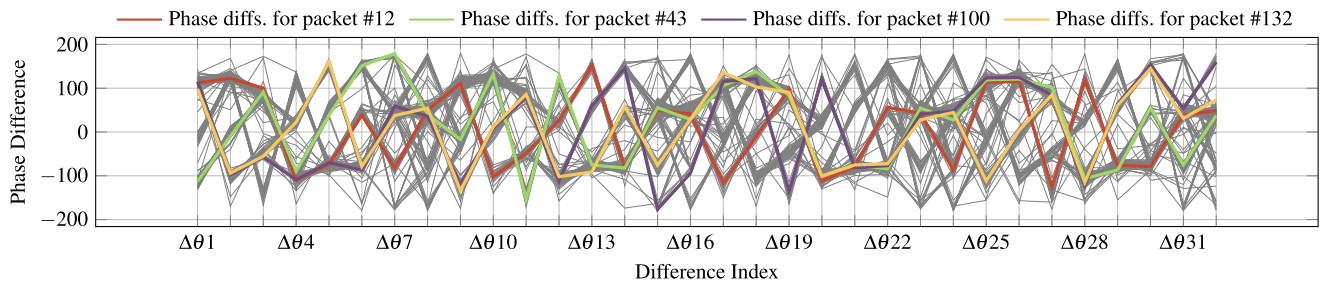
Figure 12a is a diagram of the aerial view of the test area. It is an exterior location with a flat rugged cement floor with no line-of-sight obstructions between the various devices. The area is of 12 m by 12 m, itself located in a larger outdoor lot, with a total open unobstructed space of  $1.5 \text{ km}^2$ . Figure 12c illustrates the receiver mounted on one of the tripods. All devices were placed on top of equal tripods, approximately 1.6 m from the ground (to attempt to avoid interference due to reflections from the ground surface). While the four beacon tripods only support the beacon and a battery power supply, the receiver tripod also holds the development kit, which is necessary for communication with the data collection laptop, as it was used in this setup as a UART-USB communication bridge. Devices such as smartphones or the laptop itself were put in flight mode, to reduce the influence on the final results, but the environment remained affected by any other Wi-Fi networks or devices outside our experimental control. Detailed photos of the area and setup are shown in Figure 25, in Appendix C.

**TABLE 2.** Number of received packets from each beacon for each map position. a total of 600 packets were collected per position.

| Map Position   | Number of Received BLE Packets |                |                |                |
|----------------|--------------------------------|----------------|----------------|----------------|
|                | $b_2 (x_0y_0)$                 | $b_5 (x_4y_0)$ | $b_1 (x_4y_4)$ | $b_4 (x_0y_4)$ |
| $x_0y_1$       | 252                            | 152            | 62             | 134            |
| $x_0y_2$       | 182                            | 195            | 89             | 134            |
| $x_0y_3$       | 140                            | 140            | 83             | 237            |
| $x_1y_0$       | 235                            | 164            | 70             | 131            |
| $x_1y_1$       | 199                            | 195            | 4              | 202            |
| $x_1y_2$       | 153                            | 201            | 54             | 192            |
| $x_1y_3$       | 187                            | 129            | 93             | 191            |
| $x_1y_4$       | 155                            | 167            | 110            | 168            |
| $x_2y_0$       | 226                            | 133            | 84             | 157            |
| $x_2y_1$       | 153                            | 192            | 110            | 145            |
| $x_2y_2$       | 169                            | 205            | 51             | 175            |
| $x_2y_3$       | 200                            | 136            | 94             | 170            |
| $x_2y_4$       | 149                            | 153            | 94             | 204            |
| $x_3y_0$       | 165                            | 177            | 84             | 174            |
| $x_3y_1$       | 122                            | 190            | 104            | 184            |
| $x_3y_2$       | 142                            | 228            | 91             | 139            |
| $x_3y_3$       | 167                            | 150            | 102            | 181            |
| $x_3y_4$       | 146                            | 197            | 96             | 161            |
| $x_4y_1$       | 130                            | 219            | 101            | 150            |
| $x_4y_2$       | 181                            | 159            | 92             | 168            |
| $x_4y_3$       | 123                            | 197            | 121            | 159            |
| <b>Average</b> | 170                            | 175            | 85             | 169            |

Figure 12b represents what we considered to be the  $0^\circ$  orientation for the receiver relative to the coordinate system of the map (also shown in Figure 12a). The angle increases clockwise, and decreases counterclockwise; calculations are made considering a range from  $\pm 180^\circ$ . The receiver was placed at each position on the map such that its  $0^\circ$  orientation is the same for all positions, namely along the y-axis. The four tripods with the boards configured as beacons were placed at the vertices of the area, while the receiver was moved to all 21 locations annotated with coordinate names. For each position we started the data gathering run from the laptop, stopping once 600 packets were collected.

We gathered numerous packets in order to verify if the phase difference profiles of all packets, for each



**FIGURE 13.** All phase difference profiles for the total of 170 packets sent by beacon  $b_2$  received at position  $x_2y_2$ . We have highlighted four random packets to demonstrate that, although a primary component is observable (as a thicker superposition of grey lines), no single packet follows this profile.

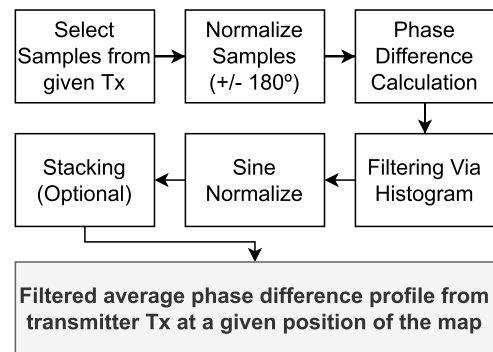
position and considering each specific beacon, would remain as consistent as observed in Section V (e.g., that the packets received from beacon  $b_2$  at position  $x_1y_1$  would produce, ideally, equal profiles). Table 2 is a listing of the number of packets received per transmitter, at each location of the map. The table shows that beacon  $b_1$  originates significantly less packets than the remaining beacons, regardless of receiver distance. We presently can not determine the root cause of this, but note that the quality of the phase profiles attainable for this beacon (after processing steps we explain below) is inferior relative to the others (Appendix B provides further examples). So this may be attributed to a faultier assembly for this beacon leading to a lower transmission power.

**A. QUALITY OF PHASE DIFFERENCE PROFILES**

We compute all phase profiles for all packets received, but show only the packets from one map position as an example in Figure 13. Each grey line represents the phase difference profile for once received packet. In this specific case, there were 170 packets received at position  $x_2y_2$ , the center of the map, from beacon  $b_2$ . Unlike the anechoic chamber results, it is noticeable that, although there appears to be a superposition of a significant number of profiles (manifesting as a thicker grey sinusoidal profile), the result is very noisy.

We first speculated that the receiver was capturing the primary transmission, while some of the stored data corresponded to reflections. From the observable thicker grey line we extrapolated that, by post-processing, we could discard the packets that produced profiles that did not match the bulk of the cleaner (i.e., stable) profiles. However, after some attempts at this post-processing, we concluded that no such subset of clean profiles exists. We can demonstrate this by highlighting four arbitrary profiles in Figure 13. Notice that none displays the clean sinusoidal profile (with four periods).

We surmised that the noticeable thicker grey line (observable for all beacons at all map positions), was the result of a superposition of most frequent phase differences per phase difference index (e.g.,  $\Delta\theta_1$ , etc), regardless of the originating profile (i.e., packet). To address this, we implemented a filtering process explained in the next section, to attain a clean



**FIGURE 14.** Process applied to an entire dataset (i.e., 600 packets for a particular map location) to produce a single filtered phase difference profile, as seen by the receiver for the specified beacon Tx.

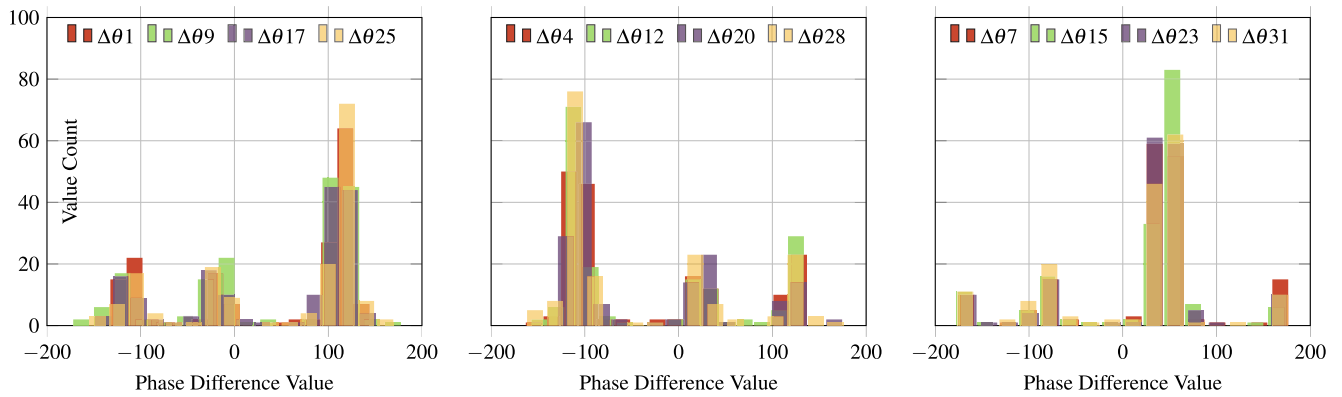
profile per map position, per beacon, to allow us to continue the evaluation of AoA calculation and position estimation.

**B. FILTERING PRIMARY PACKETS**

The filtering process is shown in Figure 14, and is based on computing a cleaner profile by determining the most frequent value of each  $\Delta\theta$ . We apply the process to all retrieved datasets, where one dataset corresponds to all packets received at a specific position of the map. Specifically, we extract four clean profiles, one per beacon, per map position.

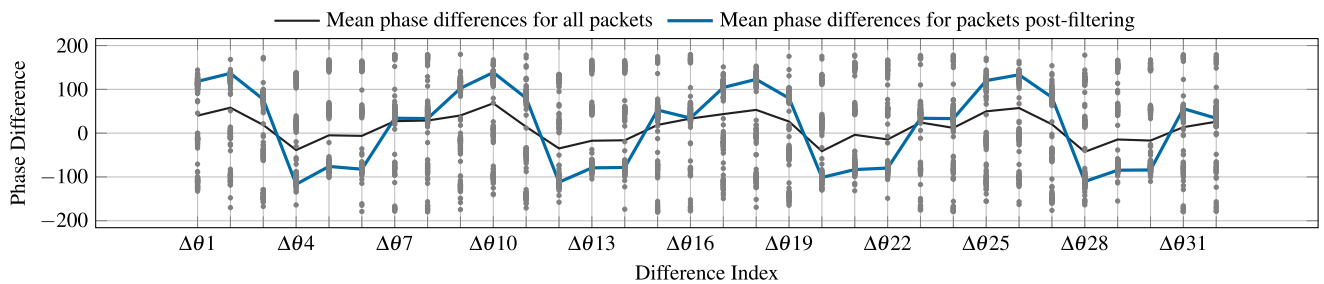
Firstly, the subset of packets from one beacon is selected from the dataset, followed by a normalization step which converts the range of  $[-201, 201]$  given by the nRF52811 into a range of  $[-180, 180]$ . Algorithm 1 is then applied to each packet (i.e., each row of 96 phase samples that were kept), followed by a filtering via histogram process. This process takes all computed phase profiles for the given dataset (e.g., the 170 profiles show in Figure 13), and per phase difference, computes a histogram with a chosen number of bins (for our results, we adopted 16 bins). Per phase difference, that is, from  $\Delta\theta_1$  to  $\Delta\theta_{32}$ , the most frequent value of the histogram is kept, and a final phase profile is produced.

Figure 15 shows a detailed example of the histograms produced. As mentioned, we keep the most frequent value per phase difference, therefore, we require 32 histograms. For brevity, the figure shows only the histograms produced for the



(a) histogram for phase difference values for the 1st antenna pair, for all four sampling rotations (b) histogram for phase difference values for the 4th antenna pair, for all four sampling rotations (c) histogram for phase difference values for the 7th antenna pair, for all four sampling rotations

**FIGURE 15.** Three example histograms (for 16 bins) of the values of the computed phase differences, for all 170 packets from beacon  $b_2$  received at position  $x_2y_2$ . Each plot shows the most frequent value for a particular phase difference (e.g.,  $\Delta\theta_1$ ), and its analogous differences for the next three sampling rotations.



**FIGURE 16.** Cleaner phase difference profile at position  $x_2y_2$ , for the packets from  $b_2$ , obtained via the histogram-based process. The phase differences shown in Figure 13 are now plotted as grey marks, illustrating the values in the respective histograms. The average of all profiles is shown in black, while in blue, the clean profile is obtained by taking only the most frequently occurring value for each  $\Delta\theta$ .

first, fourth, and seventh antenna pairs, for all the 170 profiles illustrated in Figure 13.

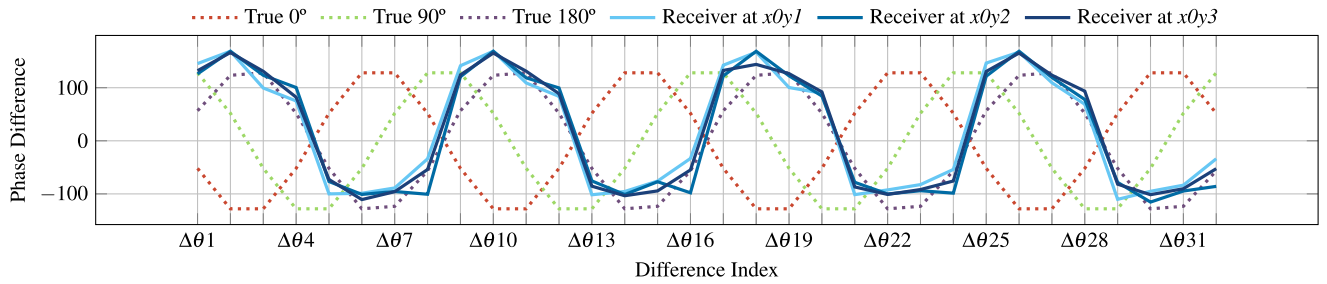
These histograms correspond therefore only to the packets received at position  $x_2y_2$  from beacon  $b_2$ , but they demonstrate that there is a primary component, and other components that, although consistent in their occurrence, likely belong to reflections (e.g., a consistent ground reflection) or periodic transmissions originated by other sources.

Additionally, since we perform four sampling rotations, we can actually compute four phase differences per antenna pair. That is,  $\Delta\theta_1$  should be equivalent to  $\Delta\theta_9$  in Figure 15a, for example. We separated the plotting in this way to also verify that, during the CTE itself, there is no significant drift effect due to clock frequency differences between beacon and receiver, which would result in difference phase difference values for the same antenna pair during the CTE. However, the consistent overlapping of the four different histograms per phase difference shows that this is not the case. Therefore, taking the most frequently occurring value in each histogram, we produce a single profile from the entire dataset of received packets per position per beacon.

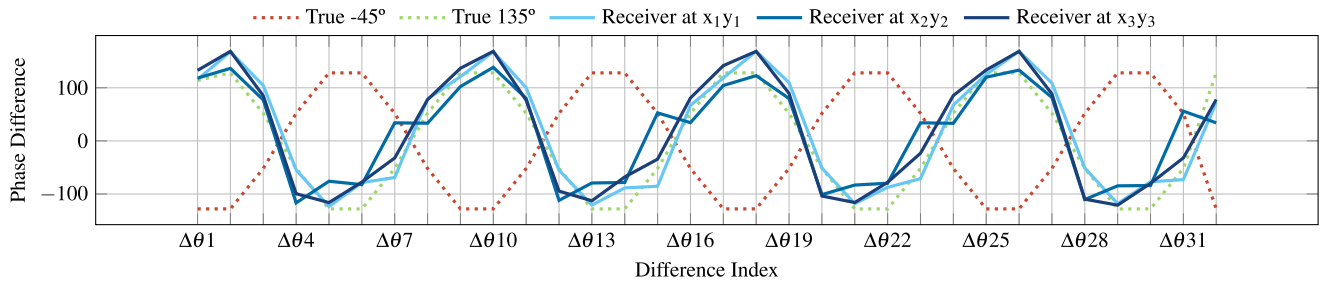
This profile is further processed by the *Sine Normalize* step which heuristically attempting to recover a more consistently sinusoidal profile. This step was introduced due to the

ambiguity of certain phase differences along a profile. That is, due to noise in the original phase samples, a given difference may overflow or underflow, e.g., a serial phase difference of  $175^\circ$  can result in a value of  $-15^\circ$  due to the accumulation of errors, and the application steps of the modulus to the phase differences in order to limit its value to the range of  $\pm 180^\circ$ . This step attempts to correct these errors, based on comparing each phase difference with the values and sign of the previous and next differences. We first check if a phase difference,  $\Delta\theta_i$ , value exceeds  $\pm 140^\circ$ , and then if both absolute differences to the previous,  $\Delta\theta_{i-1}$ , and next,  $\Delta\theta_{i+1}$ , phase difference are greater than  $70^\circ$ . If so, and if the sign of the mean of  $\Delta\theta_{i-1}$  and  $\Delta\theta_{i+1}$  is opposite to the sign of  $\Delta\theta_i$ , an overflow likely occurred, and we correct it with a sign inversion.

Finally, an optional *stacking* step consists of superimposing the various repetitions of the profile (e.g., each sampling rotation), producing a final profile with fewer periods. Specifically, we can stack the first two periods and produce a final profile with 16 differences, or stack all four periods and produce a final profile only 8 phase differences. For clarity, we note that all AoA values computed in further sections utilize the former case, and locate the first intersection of the profile where the phase equals zero, as per Algorithm 2.



(a) Profile observed by the receiver for the packets from  $b_2$  (post-filtering), for the 3 positions corresponding to the border between  $b_2$  and  $b_4$  (true  $0^\circ$ )



(b) Profile observed by the receiver for the packets from  $b_2$  (post-filtering), for the 3 positions corresponding to a diagonal between  $b_2$  and  $b_1$  (true  $135^\circ$ )

**FIGURE 17.** Two illustrations of how the phase difference profile remains consistent for different distances between receiver and a particular beacon, for the same relative orientation between the pair. Also shown are the ground-truth profiles, showing a correspondence with the observed behaviour.

Applying this step to the filtered profiles has little effect, since the four periods generated by the histogram step are already very consistent. However, this type of averaging may be one method to leverage the mentioned data redundancy provided by a long CTE, and could provide future improvements, when a much lower number of packets from a given beacon for a given position is available. This is likely to be the case considering our final application scenario of a moving receiver, where the complexity is further increased considering that, even when storing multiple packets from the same beacon, each will likely correspond to a different receiver position at the time of reception. Potential future algorithmic strategies are discussed later in Section VIII-D.

The resulting filtered profile for the same map position and beacon is shown in Figure 16. The 170 grey profiles are now shown only as grey marks per phase difference, where the clusters of values become apparent. These correspond to values with higher counts in the histograms, composing the resulting filtered profile shown in blue. In contrast, the black plot shows the average of all individually computed phase profiles. Although a sinusoidal behaviour is still observable (since the primary component is dominant), its amplitude is much lower, as the other reflective components act as destructive noise. Similar clean profiles can be achieved for all positions of the map for all beacons (examples shown in Appendix B). The next section shows that these profiles are consistent relative to the position of the receiver on the map and its orientation to the four beacons.

### C. PHASE PROFILE COHERENCY ANALYSIS

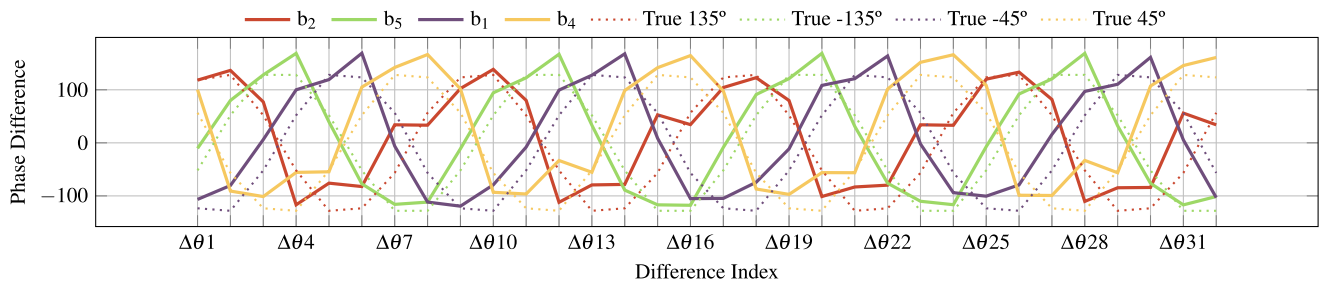
We have shown how to extract better quality phase difference profiles for each position of the map, using the 600 packets

we collected per position. The successful extraction of a primary component for all packets received, from a particular beacon for a given position, shows that the profile shifts as a function of relative orientation, therefore demonstrating the same behaviour we had verified within the anechoic chamber.

However, we also wish to verify that the phase profiles are equal for map positions where the relative orientation between the receiver and a particular beacon are the same (always under the assumption that the receivers' relative  $0^\circ$  is oriented along the y-axis). Since the tests in the anechoic chamber were performed for a single distance, and that the four overlapping sampling rotations shown as an example in Figure 15 correspond to a single position, we do this to confirm that the profiles did not suffer secondary horizontal displacements as a function of the distance, due to effects such as clock drift.

Figure 17 shows this for two possible examples. Note that the profiles shown in blue are the post-filtered averages using all available packets, computed as explained in the previous section. Figure 17a shows three expected ideal profiles as dotted lines, thus establishing a ground-truth. They assume the same  $0^\circ$  orientation along the y-axis, and thus represent what the receiver would observe if a beacon was placed in front ( $0^\circ$ ), to its right ( $90^\circ$ ) or at its rear ( $180^\circ$  or  $-180^\circ$ ). The solid lines are the computed profiles for positions  $x_0y_1$ ,  $x_0y_2$ , and  $x_0y_3$  (i.e., the border between  $b_2$  and  $b_4$ ), which are consistent among themselves and match the expected profile corresponding to an AoA of  $180^\circ$ .

An analogous case is shown in Figure 17b, this time for the three map positions that compose the diagonal between  $b_2$  and  $b_1$  (top-left to bottom-right). Referring again to Figure 12b, the ground-truths of  $-45^\circ$  and  $135^\circ$  are shown,



**FIGURE 18.** Phase difference profiles after filtering, between the receiver placed at the center of the test area and all beacons. The four solid waves display phase differences of 90°, e.g., between  $b_1$  and  $b_4$ , which is the expected behaviour given the physical disposition of the boards. The dotted plots correspond to the respective expected groundtruth profiles for these orientations at this location.

and the computed profiles for the three positions for the packets from  $b_2$  are consistent regardless of position, and match the ground-truth. If the profile for  $b_1$  was shown, it would overlap with the opposing profile corresponding to  $-45^\circ$ .

These two cases demonstrate that, regardless of distance, the same profile is observed for the same relative orientation. However, we also need to verify that, if outside the chamber, the profile of the phase differences shifts as expected as well. The center of the map is the best position for this verification, as the profiles must be complementary (i.e., evenly spaced by  $90^\circ$ ) given the physical disposition of the beacons, and that the map is square. Figure 18 shows the filtered profiles for the four beacons at this position. That is, each line represents the same filtering result show in Figure 16, for each beacon.

This set of analyses confirms that, although this pre-processing is required, the phase difference profiles behave as expected. Thus, we can use this offline processed data to compute the respective AoAs, evaluate the respective accuracy, followed by the same analysis for the position estimations.

### VII. EXPERIMENT C: AoA AND POSITION ESTIMATIONS

Using the clean phase profiles computed in the previous stage, whose consistency was verified per position of the map, we can advance to employing Algorithm 2 to compute the respective AoAs, and compare them to the true angle. The true angles per position of the map from the point of view of the receiver, from each beacon, can be easily obtained by trigonometry. Note that the computed AoAs (and respective errors to the true AoA) are influenced by how precise the positioning of the receiver was on the ground location during data collecting (i.e., since the receiver was manually placed, we cannot ensure its precise orientation according to the established  $0^\circ$  reference). That is, the actual AoA errors may be lower than those presented, as we only have an approximate ground-truth.

#### A. AoA ESTIMATIONS IN OUTDOOR AREA

We compute four AoAs per position, one per beacon, using the respective filtered profile, and compare them to the respective true values. We calculate the error as the smallest

difference to the true angle, since the error can be either clockwise or counter-clockwise, sometimes implying sign changes when the AoA is close to  $\pm 180^\circ$ . For example, if a computed and true AoAs are  $168.0^\circ$  and  $-170^\circ$ , the actual error is  $12^\circ$  rather than  $328^\circ$ . Each individual AoA error is computed by:

$$\epsilon_\phi = |((t_\theta - e_\theta + 180) \% 360) - 180| \quad (9)$$

Table 3 summarizes this analysis, where each true and computed AoA are shown for all map positions, for each beacon. The data from the highlighted row ( $x_1y_2$ ) is used as an illustrative example of positioning in the following section. Individual errors per position and beacon are shown, as computed by Equation (9). The last column also shows the average error per position (the average error of the four AoAs). We discuss the effect of distance on the AoA accuracy, the effect on applying the filtering process on the accuracy, and conclude with overall performance metrics.

We observe that the AoA error does not increase as a function of distance to the receiver. For example, considering the AoAs computed for  $b_2$  for the three positions along the side connecting to  $b_4$ , the side connecting to  $b_5$ , and the diagonal towards  $b_1$ , some positions at a closer distance actually result in higher AoA error. This is in fact consistent with the previous examples of profiles along, for example, the diagonal between  $b_2$  and  $b_1$  in Figure 17b. The profiles for the closest ( $x_1y_1$ ) and farthest positions ( $x_3y_3$ ) show a greater overlap relative to the middle position ( $x_2y_2$ ), where the first ascending intersection with the y-axis occurs at a prior point. Since our AoA estimation relies on finding this point, we can observe the effect on the respective AoA errors for the three positions from closest to farthest:  $0.5^\circ$ ,  $35.8^\circ$ , and  $1.1^\circ$ .

That is, the AoA estimation is affected by between which  $\Delta\theta$  indexes the intersection occurs. As we estimate  $X_{zero}$  via linear interpolation, the magnitudes of the phase differences corresponding to these discrete indexes affect the result. However, improvements do not necessarily occur for all cases (at least for this dataset), since the intersection point can remain the same if the two magnitudes used for interpolation increase/decrease by the same value. Using Figure 16 again as an example, the intersections occur at the same places for the filtered profile and the mean.

**TABLE 3.** Comparison between the true AoA and the measured AoA, for all map positions and beacons. The *Result* columns of the AoAs were computed using the filtered profiles generated by the process shown in Figure 14. The average absolute error, standard deviation, and RMSE shown are relative to these values, but the same metrics for the AoAs computed using the mean of all profiles is also shown. The highlighted row is used as an illustrative example in Figure 19.

| AoA ( $\phi$ ) ( $^\circ$ ) |                           | From $b_2$   |             |                   | From $b_5$    |            |                   | From $b_1$   |            |                   | From $b_4$  |            |                   | $\mu(\varepsilon\phi)$ |
|-----------------------------|---------------------------|--------------|-------------|-------------------|---------------|------------|-------------------|--------------|------------|-------------------|-------------|------------|-------------------|------------------------|
| Receiver Position           |                           | True         | Result      | $\varepsilon\phi$ | True          | Result     | $\varepsilon\phi$ | True         | Result     | $\varepsilon\phi$ | True        | Result     | $\varepsilon\phi$ |                        |
| $x_0y_1$                    | -180.0                    | 168.0        | 12.0        | -104.0            | -106.7        | 2.7        | -53.1             | -115.7       | 62.6       | 0.0               | -37.8       | 37.8       | 28.8              |                        |
| $x_0y_2$                    | -180.0                    | 177.8        | 2.2         | -116.6            | -109.1        | 7.5        | -63.4             | -86.9        | 23.5       | 0.0               | -30.7       | 30.7       | 16.0              |                        |
| $x_0y_3$                    | -180.0                    | 172.7        | 7.3         | -126.9            | -88.5         | 38.4       | -76.0             | -95.6        | 19.6       | 0.0               | -17.7       | 17.7       | 20.8              |                        |
| $x_1y_0$                    | 90.0                      | 94.1         | 4.1         | -90.0             | -57.7         | 32.3       | -36.9             | -32.0        | 4.9        | 14.0              | 38.4        | 24.4       | 16.4              |                        |
| $x_1y_1$                    | 135.0                     | 134.5        | 0.5         | -108.4            | -122.0        | 13.6       | -45.0             | -40.7        | 4.3        | 18.4              | 66.8        | 48.4       | 16.7              |                        |
| <b><math>x_1y_2</math></b>  | <b>153.4</b>              | <b>173.0</b> | <b>19.6</b> | <b>-123.7</b>     | <b>-122.9</b> | <b>0.8</b> | <b>-56.3</b>      | <b>-63.9</b> | <b>7.6</b> | <b>26.6</b>       | <b>26.9</b> | <b>0.3</b> | <b>7.1</b>        |                        |
| $x_1y_3$                    | 161.6                     | 93.9         | 67.7        | -135.0            | -157.4        | 22.4       | -71.6             | -105.3       | 33.7       | 45.0              | 24.8        | 20.2       | 36.0              |                        |
| $x_1y_4$                    | 166.0                     | 136.7        | 29.3        | -143.1            | -112.0        | 31.1       | -90.0             | -83.8        | 6.2        | 90.0              | 57.4        | 32.6       | 24.8              |                        |
| $x_2y_0$                    | 90.0                      | 95.6         | 5.6         | -90.0             | -72.8         | 17.2       | -26.6             | -14.4        | 12.2       | 26.6              | 28.3        | 1.7        | 9.2               |                        |
| $x_2y_1$                    | 116.6                     | 125.4        | 8.8         | -116.6            | -116.4        | 0.2        | -33.7             | -46.6        | 12.9       | 33.7              | 53.9        | 20.2       | 10.5              |                        |
| $x_2y_2$                    | 135.0                     | 99.2         | 35.8        | -135.0            | -153.0        | 18.0       | -45.0             | -66.4        | 21.4       | 45.0              | 38.2        | 6.8        | 20.5              |                        |
| $x_2y_3$                    | 146.3                     | 131.1        | 15.2        | -146.3            | -157.4        | 11.1       | -63.4             | -79.6        | 16.2       | 63.4              | 59.7        | 3.7        | 11.6              |                        |
| $x_2y_4$                    | 153.4                     | 103.3        | 50.1        | -153.4            | 144.5         | 62.1       | -90.0             | -96.7        | 6.7        | 90.0              | 54.2        | 35.8       | 38.7              |                        |
| $x_3y_0$                    | 90.0                      | 90.6         | 0.6         | -90.0             | -65.1         | 24.9       | -14.0             | -6.6         | 7.4        | 36.9              | 38.8        | 1.9        | 8.7               |                        |
| $x_3y_1$                    | 108.4                     | 100.4        | 8.0         | -135.0            | -128.9        | 6.1        | -18.4             | -5.8         | 12.6       | 45.0              | 67.7        | 22.7       | 12.4              |                        |
| $x_3y_2$                    | 123.7                     | 122.0        | 1.0         | -153.4            | -109.9        | 43.5       | -26.6             | 5.5          | 32.1       | 56.3              | 72.3        | 16.0       | 23.2              |                        |
| $x_3y_3$                    | 135.0                     | 123.9        | 11.1        | -161.6            | -157.3        | 4.3        | -45.0             | -58.2        | 13.2       | 71.6              | 66.0        | 5.6        | 8.5               |                        |
| $x_3y_4$                    | 143.1                     | 107.5        | 35.6        | -166.0            | -142.0        | 24.0       | -90.0             | -87.5        | 2.5        | 90.0              | 43.9        | 46.1       | 27.1              |                        |
| $x_4y_1$                    | 104.0                     | 95.3         | 8.7         | -180.0            | -154.2        | 25.8       | 0.0               | 17.1         | 17.1       | 53.1              | 12.4        | 40.7       | 23.1              |                        |
| $x_4y_2$                    | 116.6                     | 80.2         | 36.4        | -180.0            | 147.2         | 32.8       | 0.0               | -38.9        | 38.9       | 63.4              | 39.3        | 24.1       | 33.1              |                        |
| $x_4y_3$                    | 126.9                     | 107.2        | 19.7        | -180.0            | -162.1        | 17.9       | 0.0               | 1.6          | 1.6        | 76.0              | 86.0        | 10.0       | 12.3              |                        |
| Filtered Profile            | $\mu(\varepsilon\phi)$    |              | 18.1        |                   |               | 20.8       |                   |              | 17.0       |                   |             |            | 21.3              |                        |
|                             | $\sigma(\varepsilon\phi)$ |              | 18.1        |                   |               | 15.7       |                   |              | 14.8       |                   |             |            | 15.1              |                        |
|                             | RMSE                      |              | 25.3        |                   |               | 25.9       |                   |              | 22.3       |                   |             |            | 25.9              |                        |
| Mean Profile                | $\mu(\varepsilon\phi)$    |              | 46.5        |                   |               | 24.0       |                   |              | 17.3       |                   |             |            | 30.4              |                        |
|                             | $\sigma(\varepsilon\phi)$ |              | 54.8        |                   |               | 22.9       |                   |              | 13.2       |                   |             |            | 36.7              |                        |
|                             | RMSE                      |              | 70.9        |                   |               | 32.8       |                   |              | 21.6       |                   |             |            | 46.9              |                        |

We considered including an analysis of the effect of using subsets of each dataset per position, i.e., using only  $N = [300, 150, \dots]$  packets for filtering. However, multiple runs using a uniform sampling of the datasets (to prevent bias) would produce histograms similar to those in Figure 15, where the same values would be the mode. Thus the resulting profiles would match the ones currently computed. The main effect of the filtering process is increasing the magnitude of each phase difference, meaning that improved AoA accuracy will only be attained by exploiting this data, such as curve fitting methods or NN approaches. We present some preliminary results for the latter case in Section VIII. Even so, the filtering produces significant improvements even for the current algorithm, as some unfiltered cases suffer from intersections at the incorrect indexes due to noise (e.g., Figure 21a), as the following global metrics demonstrate.

The last six rows of Table 3 show three metrics: the average and standard deviation of the error, and the RMSE. The values in the rows under the label *Filtered Profile* are relative to the values shown in the table. For brevity, we omit the individual AoAs and errors computed using the straightforward mean of all profiles, and show only the same metrics in the three rows, to establish a comparison. Considering the metrics for the *Filtered Profile* cases, there is little variation between beacons, suggesting no significant operational differences between beacons, and no asymmetric noise interference in the test area. When comparing to the metrics attained when computing the AoA without filtering, the overall improvement in performance is evident. Finally,

we compute the overall RMSE using all true and computed AoAs (i.e., a total of 84, given 21 positions and 4 beacons). For the filtered and unfiltered cases, the respective values are 24.8° and 46.8°. The RMSE is considerable relative to state of the art approaches such as MUSIC, but as the next section shows near meter accuracy is still achievable.

## B. POSITION ESTIMATIONS IN OUTDOOR AREA

Using the best AoA values for each map position (produced from the filtered phase profiles), we employed the least squares method — Equations (6) to (8) — to determine, although currently offline, the positioning accuracy of the approach. Note that since the receiver was manually placed, its actual true position is only approximate to the coordinate points of the diagram, analogous to how the true orientation of the receiver was approximate to 0° reference of the coordinate system. Since the position estimation is itself dependent on the AoAs, there is a compound error effect that is difficult to quantify. Future experiments may rely on an auxiliary positioning system, e.g., UWB to provide a more accurate ground-truth position for comparison.

To evaluate the effect of the number of available AoAs on the positioning accuracy, we estimate all 21 positions for two cases: firstly using all four available AoAs per position, and secondly using only three AoAs. We first intended to use only the three AoAs corresponding to the three beacons closest to the receiver, assuming that the farthest beacon would produce the worst AoA estimate. However, due to the symmetry of the map (specifically the test locations) and

**TABLE 4.** True and estimated positions and errors for two estimation cases: case #1 uses all four AoAs received by each beacon at each position, case #2 performs 100 runs per position, using only three AoAs randomly selected from any of the four beacons. Both cases use the AoAs produced by the post-filtered profiles (i.e., filtered average) using all 600 packets per map position.

| Receiver Position          | True Position |            | Estimation Case #1 (Using 4 AoAs) |            |                    | Estimation Case #2 (Using 3 AoAs - 100 runs per pos.) |              |            |            |
|----------------------------|---------------|------------|-----------------------------------|------------|--------------------|---|--------------|------------|------------|
|                            | y             | x          | Estimated Position                | Error (m)  | Estimated Position | $\mu$ Error (m)                                       | $\sigma$ (m) |            |            |
| $x_0y_1$                   | 3.0           | 0.0        | 10.4                              | 1.3        | 7.5                | 10.7  | 1.4          | 8.1        | 3.9        |
| $x_0y_2$                   | 6.0           | 0.0        | 8.3                               | -1.5       | 2.7                | 8.1   | -1.8         | 3.3        | 1.5        |
| $x_0y_3$                   | 9.0           | 0.0        | 6.6                               | 0.0        | 2.4                | 6.7   | -0.0         | 4.8        | 2.6        |
| $x_1y_0$                   | 0.0           | 3.0        | 0.4                               | 7.8        | 4.9                | 0.1   | 7.8          | 4.9        | 1.7        |
| $x_1y_1$                   | 3.0           | 3.0        | 6.4                               | 6.6        | 4.9                | 6.2   | 6.4          | 4.7        | 1.9        |
| $x_1y_2$                   | 6.0           | 3.0        | 7.1                               | 1.5        | 1.8                | 7.2   | 1.5          | 1.9        | 0.4        |
| $x_1y_3$                   | 9.0           | 3.0        | 6.1                               | 7.5        | 5.3                | 7.3   | 6.9          | 7.4        | 2.6        |
| $x_1y_4$                   | 12.0          | 3.0        | 7.4                               | 4.7        | 4.9                | 7.4   | 3.7          | 5.7        | 1.0        |
| $x_2y_0$                   | 0.0           | 6.0        | -0.6                              | 7.9        | 1.9                | -0.5  | 8.1          | 2.3        | 0.8        |
| $x_2y_1$                   | 3.0           | 6.0        | 5.1                               | 6.1        | 2.1                | 5.1   | 6.0          | 2.5        | 0.7        |
| <b><math>x_2y_2</math></b> | <b>6.0</b>    | <b>6.0</b> | <b>5.2</b>                        | <b>6.7</b> | <b>1.1</b>         | <b>5.3</b>  | <b>6.8</b>   | <b>3.1</b> | <b>1.7</b> |
| $x_2y_3$                   | 9.0           | 6.0        | 9.0                               | 8.0        | 2.1                | 9.1   | 8.0          | 2.3        | 0.8        |
| $x_2y_4$                   | 12.0          | 6.0        | 5.8                               | 14.8       | 10.7               | 6.0   | 14.5         | 10.7       | 2.2        |
| $x_3y_0$                   | 0.0           | 9.0        | -0.4                              | 10.4       | 1.5                | -0.4  | 10.4         | 1.5        | 0.2        |
| $x_3y_1$                   | 3.0           | 9.0        | 3.8                               | 11.2       | 2.4                | 3.4   | 11.3         | 2.7        | 0.6        |
| $x_3y_2$                   | 6.0           | 9.0        | 5.2                               | 11.1       | 2.3                | 5.0   | 10.7         | 2.9        | 1.2        |
| $x_3y_3$                   | 9.0           | 9.0        | 8.0                               | 8.8        | 1.0                | 8.1   | 8.7          | 1.3        | 0.4        |
| $x_3y_4$                   | 12.0          | 9.0        | 6.6                               | 7.0        | 5.7                | 7.5   | 6.5          | 6.3        | 1.4        |
| $x_4y_1$                   | 3.0           | 12.0       | 1.7                               | 9.4        | 2.9                | 8.8   | 7.2          | 10.5       | 11.1       |
| $x_4y_2$                   | 6.0           | 12.0       | 0.3                               | 8.2        | 6.9                | 0.3   | 8.2          | 7.2        | 1.3        |
| $x_4y_3$                   | 9.0           | 12.0       | 7.3                               | 11.6       | 1.8                | 7.6   | 11.7         | 2.7        | 1.4        |
|                            |               |            | <b>Average</b>                    |            | <b>3.6</b>         | <b>Averages</b>                                       |              | <b>4.6</b> | <b>1.8</b> |

beacon placement, there are always two equidistant farthest receivers. So, we instead perform 100 runs which randomly sample three of the available AoAs per position, and present the average estimated position. For other map configurations, discarding AoAs originating from distant beacons could be a viable heuristic, based on the RSSI. This random sampling case also provides a more realistic evaluation, since we cannot presume to have AoA estimates available for all beacons of the map at any time. Hence this case demonstrates the average accuracy if the position is estimated as soon as a sufficient number of AoAs, three, are available, regardless of origin.

Table 4 presents these estimates. The first column are the named map coordinates, while the second and third columns the true  $y$  and  $x$  position in meters. The following three columns are the respective estimated positions and absolute error. The last four columns present the position estimations using only three randomly sampled AoAs. Therefore, each row (e.g.,  $x_0y_1$ ) presents the average estimated position for the 100 sampling runs performed per position. So, while the estimation using four AoA has a single absolute positioning error, the last two columns show the average error and standard deviation of these 100 estimation runs, while the last row is the average of all rows above.

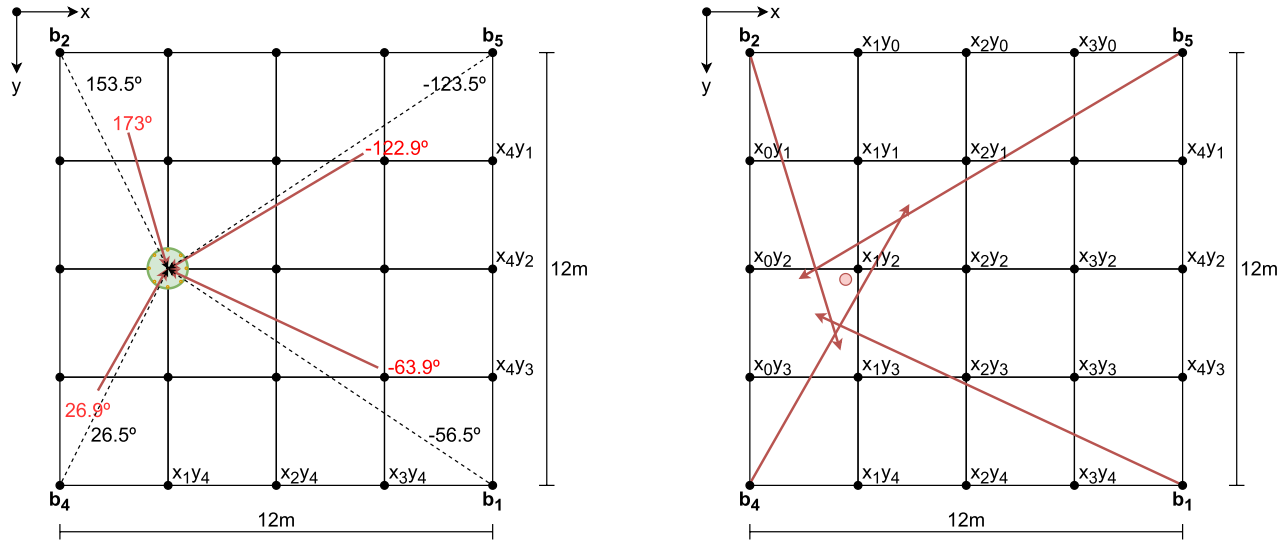
The highlighted row is the case where the smallest positioning error of 1.1 m is achieved using all four AoA, which is coincidentally the center of the map. The error is lowest despite the AoA error of  $35^\circ$  for  $b_2$ , demonstrating that, despite lower AoA accuracy, sufficient redundancy can reduce the candidate area considerably. This is observable in the global average positioning errors of 3.6 m, when using four angles, and 4.6 m, when using three angles. In the

former case, although one of the AoA may suffer from large error, it does not produce a worse result, since it could never contribute to a larger candidate area. On the other hand, randomly sampling three angles may include this more erroneous angle in detriment of a lower error instance, increasing the average positioning error.

Higher positioning errors appear to occur for positions closer to the edges of the map, which is consistent with prior tests via simulation [44], as well as other works in the state-of-the-art [79], [80]. We argue that this occurs since, for these positions, the receiver will be closer to two beacons, but at its maximum distance from the remaining. Therefore, only two AoAs will be of good quality, thus even use of four will not greatly reduce the candidate area down sufficiently, supporting the case for beacons along the space itself and not only the perimeters. For example, for  $x_0y_1$  the positioning error is 7.5 m, and the four AoAs suffer from errors of  $12^\circ$  and  $2.7^\circ$  for the closest beacons, but of  $62.6^\circ$  and  $37.8^\circ$  for the farthest beacons. An analogous analysis can be made for the complementary map positions.

Figure 19 exemplifies the position estimation, for the row highlighted in Table 3, using all four AoAs. In Figure 19a, the receiver is placed at position  $x_1y_2$ , and the true angles of that position relative to the beacons are shown in dotted lines. In red, the AoAs computed using the method explained in Figure 14 are shown (the angles are not exact and are illustrative only). Figure 19b shows how the receiver interprets each AoA, by knowing the absolute position of each source beacon in the map, and the red dot is the resulting estimated position, which is 1.8 m away from the real position.

This example aids in visualizing how higher precision could be achieved by placing more beacons in the map,



(a) Example of four computed AoAs for one example position, using all received packets (i.e., 600), and the explained filtering to achieve cleaner profiles (all AoAs for this case are shown in Table 3)

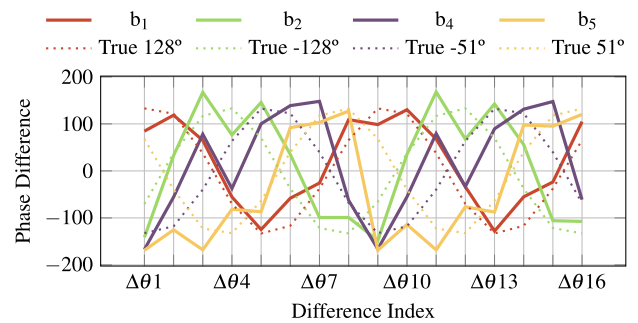
(b) Candidate area, relying on the known position of the beacons, and respective vectors at those starting points with an orientation defined by the computed AoAs

**FIGURE 19.** Example of conjugation the four AoAs for one example position,  $x_1y_2$ , to create a candidate area, and the resulting estimated position in Figure 19b.

by reducing the candidate area. As we have shown, AoA estimates with high error do not contribute to a higher estimation error, if used in conjunction with a set of accurate AoAs. We have also shown that the edges of the map suffer higher errors since fewer reliable AoAs are available, so additional beacons should be placed within the area of interest, and not only along the perimeter. For indoor scenarios, the perimeter corresponds to the walls, while the ceiling would have to be used to place anchors within the map, as the floor space is likely to be obstructed. For solutions relying on fixed anchors with cabling requirements, either placement location may imply a high installation cost. In contrast, our system topology addresses precisely this, since battery powered beacons with a single antenna can be attached to surfaces (walls or ceilings), placed on tops of shelves, or ceiling cable supports common in industrial or even office environments. Also, the beacons within the space can be easily relocated if necessary (e.g., to improve precision or if the area of interest suffers any modifications), with only an update to their position being required by the receiver software to compute its position.

**VIII. PRELIMINARY EXPERIMENTS AND FUTURE WORK**

The previous section presented the main experimental campaigns of this work, which were conducted in an outdoor area. In this section, we present brief early results for an analogous experiment conducted indoors. Also, throughout the discussions, we referenced the use of NN and hardware acceleration for potential performance improvements. We briefly present some results for these approaches, and explain future improvements that we envision can be applied to the approach.



**FIGURE 20.** Phase difference profiles after filtering, between the receiver placed at the center of the indoor test area and all beacons, and respective expected groundtruth profiles for these orientations at this location.

**A. PRELIMINARY RESULTS IN INDOOR LOCATION**

As an additional preliminary test, we conducted an experiment analogous to the previous outdoor setup, but this time in an indoor location (shown in Appendix C). The area was of 5.6 by 4.5 m, we assumed the same coordinate system with the origin placed at the top-left, and placed the four beacons at the vertices in a clockwise order of  $b_1, b_2, b_4, b_5$ , starting from the origin. The indoor location itself was an open area in an office space. Although obstructions were present, none were within the area outlined by the four beacons. However, walls were in proximity (about 1 m) of three of the sides of the outlined area. We placed the receiver in the center of the area, captured 600 packets as per the previous experiment, and applied the same processing.

Figure 20 shows the filtered phase difference profiles, along with groundtruths. Note that, since the map is rectangular, the profiles are not evenly spaced by 90° in this



case. The respective ground truths are shown, for the expected angles at this center position. The resulting computed profiles roughly follow the expected behaviour, but the profile quality is inferior relative to the analog case in the outdoor area. The likely causes are the proximity of the walls of the indoor space, obstacles, and other wireless devices operating nearby. As a result, the computed AoAs for the mentioned beacon sequence are  $121.1^\circ$ ,  $-120.974^\circ$ ,  $-93.8^\circ$ , and  $44.4^\circ$ . Only the AoA for  $b_4$  significantly deviates from the expected value. Using all four AoAs, the estimated position is  $(x, y) = (3.57, 3.00)$ , and using only the 3 best AoAs, the result is  $(x, y) = (2.37, 3.43)$ . Given the true position of  $(x, y) = (2.8, 2.25)$ , the respective errors are 1.07 m and 1.26 m. That is, although the AoA for  $b_4$  was subject to higher error, its use did result in a smaller candidate area.

### B. HARDWARE ACCELERATED IMPLEMENTATION

As preliminary work, we have also explored the implementation of phase difference calculation algorithms on an FPGA device, via High-Level-Synthesis (HLS). Specifically, we took a C/C++ implementation of a function which implemented an earlier version of the phase difference calculation algorithm (briefly presented in [45]), which was based on computing phase predictions as per [74]. We generated raw phase sample datasets (based on the theoretical model), and evaluated the execution time and power consumption of the baseline C/C++ code running on a Odroid-XU3. A modified version of the code exposing data parallelism via HLS *pragmas* was synthesized for a Xilinx UltraScale+ MPSoC ZCU102. In the former case, the Odroid's 1.2 GHz ARM processor required 22  $\mu$ s to process 361 phase samples (a single synthetic BLE packet), while the hardware accelerated implementation required only 5.4  $\mu$ s operating at 137 MHz.

This demonstrates that there is potential for real-time AoA based localization similar to our approach, if the proper algorithms are implemented as hardware within the radio systems, instead of delegating this processing to software only. The hardware implementation of the phase difference calculation portion of the signal processing pipeline is  $4\times$  faster, and when accounting for the difference in operating frequency,  $35\times$  more efficient. Further work could focus on hardware accelerated functions to directly produce the AoA, instead of only the phase differences, as we evaluated.

### C. PRELIMINARY RESULTS USING NEURAL NETWORKS FOR AoA INFERENCE

Using the dataset generator, we also conducted a preliminary study on the use of NNs to extract the AoA via regression, using the phase differences as a feature vector. We utilized the Python TensorFlow library to define, test, and train the models [81]. We explored three NN models, and trained and tested each model by generating nine datasets where the raw phase data is affected by white noise where  $\mu = 0$ , and  $\sigma$  varies between  $0^\circ$  and  $90^\circ$ , in steps of  $10^\circ$ . Each dataset contained 72 thousand samples, i.e., 200 samples

per each possible AoA value ( $0^\circ$  to  $360^\circ$ , in a steps of  $1^\circ$ ). Also, the feature vector used contained only eight phase differences, in a range of  $[0^\circ, 360^\circ]$ , and the groundtruth AoA was normalized to a range of  $[0.0, 1.0]$  for optimization.

The first model is a simple NN with an input layer of size 8, one hidden layer with 128 neurons using the Rectified Linear Unit (ReLU) activation function, and an output layer with a single linear activation function to regress the AoA. The second model uses the same input and output layers, but contains three hidden layers with 128, 512 and 256 neurons, also using the ReLU activation function. The third model is a CNN with a total of six hidden layers: a reshape layer, a pair of 1-D convolutional and 1-D max pooling layers, and a final 1-D convolutional layer.

These preliminary experiments show that, for the second and third models, the absolute mean error in the regressed AoA remains under  $10^\circ$  when the noise applied to the raw phase samples is under  $50^\circ$ . In contrast, the error for the first model is almost  $10^\circ$  for the AoA, even for raw phase noise of also  $10^\circ$ . These preliminary evaluations did not consider reflections, as the dataset generator presently only adds white noise to each sample. As future work, we may instead subject the models to the BLE packets we retrieved during the experimental campaign presented in Section VI, including evaluating the use of all 32 phase differences as a feature set. Since inference in NN is expedient, a robust model may be promising for embedded extraction of the AoA in compute constrained devices. Given the final intended application of our topology, i.e., tracking a moving receiver, we are currently further exploring this approach.

### D. FUTURE WORK

Regarding future work, we identify three major research directions with some overlap. Namely, focus on integration with our existing simulator [44], improvements to the AoA calculation algorithm, and advancing towards real-time implementation. We subdivide this into the following items:

- The simulator currently samples AoAs retrieved from a commercial board. We will replace this with the gathered BLE data, and re-evaluate the performance by calculating the AoA as we have presented here.
- As shown in Table 4, using 3 AoAs versus 4 significantly affects the results. Future experiments should use additional beacons, including placements within the map area. The simulator can be used for early testing prior to real-world setups, by using the current phase datasets to instantiate beacons at arbitrary positions.
- Fine tuning the reference value,  $X_c$ , corresponding to the first ascending intersection for the profile for an AoA of  $0^\circ$  (see Equation (5) and Algorithm 2), since the current value of 4.5 is derived from ideal phase data. It should be attained from the true behaviour of the receiver, i.e., from the profiles resulting from packets for an AoA of  $0^\circ$ , for a better ground-truth reference.
- Improving the AoA algorithm by use of multiple intersections of the profile with the y-axis, both ascending

and descending, and comparison to respective reference values, exploiting the aforementioned data redundancy due to the four periods of the profile.

- Use all 32 phase differences to derive the AoA, effectively exploiting the data redundancy, either via curve fitting, or by using the 32 data as features to NN models.
- Estimate the AoA resorting only to 6 or 4 phase differences (i.e., discarding certain phase differences). This emulates the use of a UCA with fewer elements, allowing us to determine the trade-off between required computational time and accuracy.
- Fully re-test for datasets gathered from other experimental locations, namely indoor locations, where the reflective component of signals will have a greater influence.
- Evaluation of localization and tracking for a moving receiver. Presently, multiple packets per location were required to arrive at usable AoAs. Addressing the previous points will aid in circumventing this limitation. Use of odometry may further improve the tracking accuracy.
- Implementing the algorithms fully in the embedded software of the receiver, to evaluate the computation time on an edge device, and the resulting battery life.
- Implementing the algorithms, and future improved versions, in hardware (e.g., on FPGAs), for future integration in radios capable of directly providing the AoA based on built in knowledge of array topology.

The simulator can be used to evaluate several of the above points, especially in regards to developing further approaches to the AoA calculation, including the NN currently under development, and to implementing therein the existing state-of-the-art algorithms for a direct comparison of tracking accuracy versus computational cost.

Finally, some of the above points relate to exploiting the data redundancy available in the four periods of the profiles (i.e., 32 phase differences). This redundancy had little effect in the presented evaluation, since the filtered profiles displayed very little variation between the four periods. However, this is not likely to be the general case (e.g., in other locations, with additional interferences, and for a moving receiver, etc). However, if it is determined that there is indeed no additional benefits, the CTE may be configured to a shorter length, increasing packet reception rate, and therefore the position estimation rate, which is critical to tracking.

## IX. CONCLUSION

In this paper we have presented and experimentally evaluated the self-localization accuracy of a BLE receiver of our own design, based on Angle-of-Arrival (AoA) calculations relying on Uniform Circular Array (UCA) with eight elements. Relative to state-of-the-art algorithms for AoA calculation, we have evaluated the performance of a significantly less computationally complex algorithms, based on straightforward use of the phase difference profiles of the eight antenna

element pairs. As with recent approaches, our motivation was allowing the computation of the AoA in devices with relatively low processing power, thus contributing to IoT solutions capable of real-time response and energy efficiency.

Via two main experimental campaigns, we demonstrated that the device operates as intended in an anechoic chamber, with the expected AoAs being successfully computed with a RMSE of  $10.7^\circ$ . Regarding localization, we performed an experiment based on a proposed topology of mobile antenna arrays, and fixed omni-direction beacons. For four beacons in a an outdoor 12 m by 12 m area, 21 tested map positions, and four beacons, there are a total of 84 true AoAs. After post-processing steps to attain cleaner phase difference profiles, an average RMSE of  $24^\circ$  is achievable relative to the true angles. The respective positioning error is as low as 1.1 m in the best case, with an absolute average error of 3.6 m when four AoAs are used. We believe greater accuracy can be achieved with use of more beacons, and by the foreseen improvements to the AoA calculation algorithm.

Although our approach was tested with BLE for a UCA with 8 elements, the topology for self-localization itself, i.e., fixed beacons and mobile self-localizing receivers with antenna arrays, could be applied to other protocols capable of DF, such as Wi-Fi, and using UCAs with more elements.

We outlined several research directions for future work, but will first focus on integrating the collected BLE packet phase data into our simulator infrastructure, in order to evaluate the resulting tracking performance using this lower level real-world data, for both the presented AoA calculation algorithm, and NN based models.

## APPENDIX A ALGORITHM IMPLEMENTATIONS

The following are Octave implementations of the algorithms presented as pseudo-code in the main body of the paper.

### A. PHASE DIFFERENCE CALCULATION

The following listing is an implementation of the phase difference profile calculation. The code is generalizable for any number of samples per antenna, or number of antennas. Due to the data independence of the calculations, it is also amenable to future parallelization of iterations.

### B. ANGLE-OF-ARRIVAL ESTIMATION

The following listing is an implementation of the AoA estimation method applied, based on a simple detection of the intersection of the phase profile with the y axis at value  $0^\circ$ . The algorithm could be improved by locating multiple intersections (e.g., when receiving 32 phase differences).

### C. FILTERED PROFILE CALCULATION

The following listing illustrates the implementation which, given a dataset of packets for a given map position, selects the packets for a given beacon (*idx* parameter), and produces a clean profile, based on the histogramming method presented.

```

function [dsamplesmean] = simpleCalcDiff(rsamples,
nrAntenas = 8, samplesPerAntena = 3)

% cut down to 4 full rotations
rsamples = rsamples(:, 1:l:(4 *
samplesPerAntena * nrAntenas) + 1);

% Computes phase differences between
% consecutive antennas and stores
% the result in variable "dsamples".
dsamples = zeros(size(rsamples, 1),
size(rsamples, 2) - samplesPerAntena);
for i = 1:size(rsamples, 1)
aux1 = rsamples(i, :);
aux1(samplesPerAntena + 1:end)
= aux1(1:end - samplesPerAntena);
aux2 = rsamples(i, :) - aux1;
dsamples(i, :) = aux2(samplesPerAntena + 1:end);
end

% Apply bounding loop after difference
for i = 1:size(dsamples, 1)
for j = 1:size(dsamples, 2)
if(dsamples(i, j) < -180)
dsamples(i, j) = dsamples(i, j) + 360;
end
if(dsamples(i, j) > 180)
dsamples(i, j) = dsamples(i, j) - 360;
end
end
end

% Compute mean for each sample
dsamplesmean = zeros(size(dsamples, 1),
size(dsamples, 2) / samplesPerAntena);
for i = 1:size(dsamples, 1)
for j = 1:1:samplesPerAntena:end;
aux1 = dsamples(i, 1:samplesPerAntena:end);
for j = 2:1:samplesPerAntena
aux1 = [aux1; dsamples(i, j:samplesPerAntena:end)];
end
dsamplesmean(i, :) = mean(aux1, 1);
end
endfunction

```

Listing 1. Phase difference profile calculation.

```

function angle = estimateAngle(phaseDiffs,
nrPhaseDiffs, nrAntennas = 8)

# Reference value of zero cross
# when incident angle is 0 degrees
dp000ZeroCrossIdx = 4.5;
timePer1Deg = 8/360;

# Find zero cross
for i = 2:1:nrPhaseDiffs
if (phaseDiffs(i - 1) < 0) && (phaseDiffs(i) > 0)
break;
endif
endif

# Compute zero cross time
time = i - 1;
time += (-phaseDiffs(i - 1))
/ (phaseDiffs(i) - phaseDiffs(i - 1));
if time < dp000ZeroCrossIdx
time += 8;
endif

crossDiff = (time - dp000ZeroCrossIdx);
angle = (crossDiff / timePer1Deg);
endfunction

```

Listing 2. AoA estimation based on simple intersection method.

## APPENDIX B ADDITIONAL PHASE PROFILE EXAMPLES

Section VI illustrated the phase profiles for beacon  $b_2$  at position  $x_2y_2$ , pre- and post-filtering (Figure 16). Also shown

```

function [msamples, dsamples] = processSamples(rsamples,
idx, stacknr = 32)

% Select packets from beacon "idx" from dataset
idx1 = rsamples(:, 2) == idx;

% Normalization
% (+/-201 reported by NRF52811 to +/- 180 range)
rsamples = (180/(64*pi)) .* rsamples;
rsamples = rsamples(idx1, :);

% Use only middle sample
dsamples = simpleCalcDiff(rsamples(:, 4:3:end), 8, 1);

% Filter via histogramming
msamples = phaseDiffHistogram(dsamples, 16);

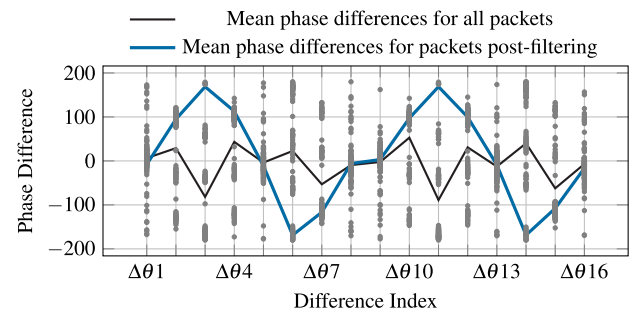
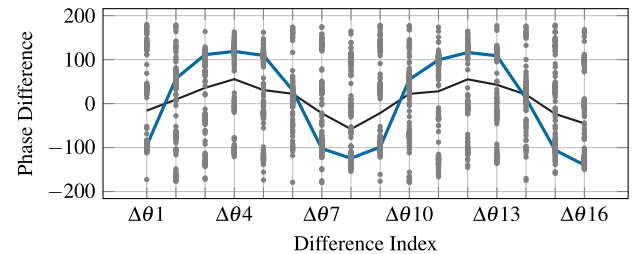
% Recover a more "sine-like" profile
msamples = sineNormalize(msamples);

% Stack profile periods and compute an average
msamples = waveStack(msamples, stacknr);

% phase diff indexes along X axis when plotting
% phase diff value in Y axis
dsamples = dsamples';
msamples = msamples';
endfunction

```

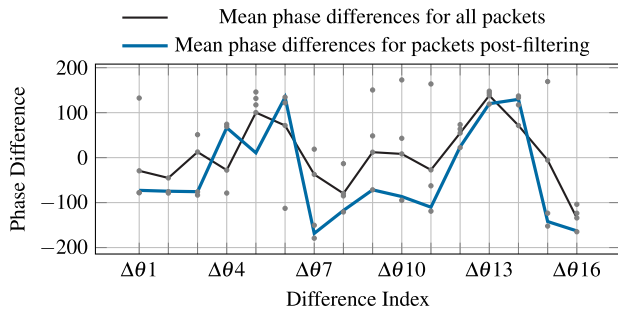
Listing 3. Phase profile filtering via dataset histogram.

(a) phase difference profiles for beacon  $b_5$ , with receiver at position  $x_1y_3$  (farthest), from which 129 packets are received(b) phase difference profiles for beacon  $b_5$ , with receiver at position  $x_3y_1$  (closest), from which 190 packets are received

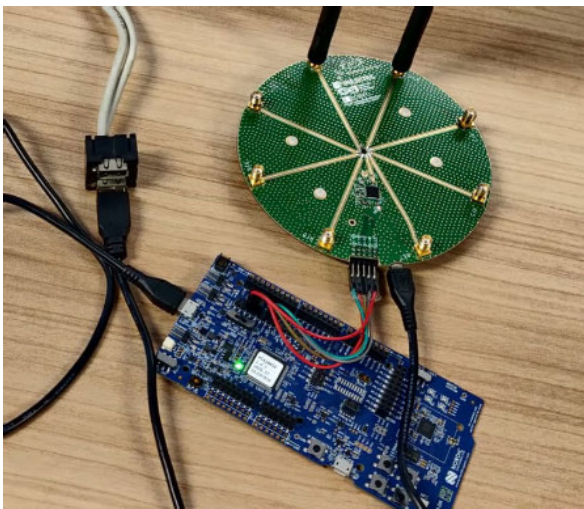
**FIGURE 21. Additional phase difference profile examples, pre- and post-filtering, for beacon  $b_5$ , for the farthest ( $x_1y_3$ ) and closest distance ( $x_3y_1$ ).**

are the post-filtered profiles for all beacons at the same center position. Analogous profiles had to be computed per map position, per beacon, to attain the best achievable AoA. For completeness we now show three additional examples, presenting only the first 16 phase differences.

Figure 21 shows two phase profile difference examples for beacon  $b_5$ , produced from the packets received by the antenna array at the nearest (Figure 21b), and farthest positions (Figure 21a). These correspond to distances of 4.2 m and



**FIGURE 22.** Additional phase difference profile examples, pre- and post-filtering, for beacon  $b_1$ , with the receiver at the farthest distance  $(x_x y_1)$ , from which only 6 packets are received.



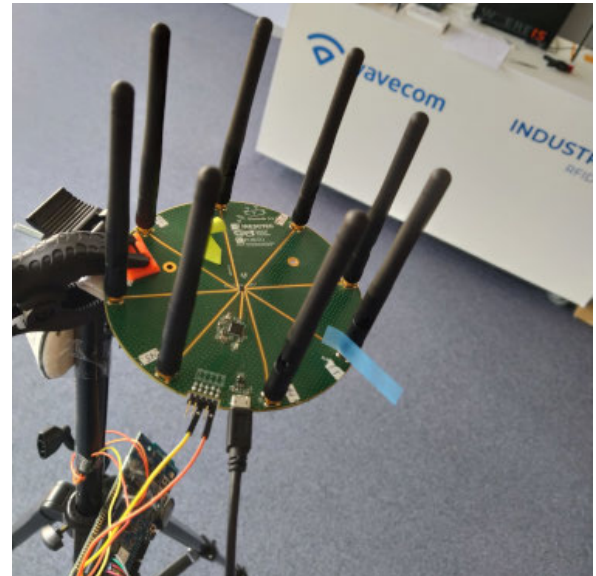
**FIGURE 23.** Connections between development kit, antenna array design, and laptop.

12.7 m, respectively. For the nearest distance (approximate to the device distance in the anechoic chamber), a very clean filtered profile is achievable. Both pre- and post-filter profiles present the same intersections with the  $y$  axis, meaning the same AoA would be computed using Algorithm 2.

However, as highlighted before, this does not mean the filtering stage is unnecessary or does not improve results, as this case cannot be extrapolated for all others (i.e., positions and beacons). Figure 21a demonstrates this, since the intersection of the simple mean occurs near index  $\Delta\theta_3$ , and therefore the low complexity, but naive, AoA estimation algorithm would miscalculate the AoA. Table 3 had also shown this by comparison of the mean absolute errors and RMSE of the AoA of using the pre- and post-filter profiles.

Future versions of the algorithm must rely on the magnitude of each phase difference, combined with multiple intersections with the  $y$  axis for a more robust calculation (i.e., methods akin to curve fitting), especially since the reflective or interferent components will likely not be as constant as those observed in the example histograms (Figure 15) for any other experimental setups or for a moving receiver.

Figure 22 shows the phase difference profile produced from the packets received at the farthest distance



**FIGURE 24.** Photographs of experimental setup for preliminary experiments in indoor area of 5.6 by 4.5 meter.

(i.e., position  $x_1 y_1$  for the receiver). The last 16 phase differences are omitted for brevity, but there is no periodic behaviour relative to the first 16 shown. As mentioned in Section IV-C, fewer packets were captured from this beacon for all positions, with only six packets received for this particular case.

Even for the closest placement of the receiver relative to  $b_1$  (i.e.,  $x_3 y_3$ ), the number of packets is only 102, with more packets having been received from the other three beacons (see Table 2). This is likely due to a lower transmission power or hardware issue. Even so, a sinusoidal and periodic profile is obtainable after filtering for this position.

### APPENDIX C ADDITIONAL SETUP PHOTOGRAPHS

For completeness, Figures 23 to 25 show additional photographs of the device configurations, experimental locations and respective setups. Figure 24 the indoor location and a closeup of the receiver mounted on the tripod, and Figure 25



**FIGURE 25.** Photographs of experimental setup for data collection of 600 packets per each 21 test locations of the 12 by 12 meter test area.

shows the setup in the outdoor location including tripod setup using the four beacons, one receiver, and the laptop for data collection.

## REFERENCES

[1] C. Han, Y. Wang, Y. Li, Y. Chen, N. A. Abbasi, T. Kurner, and A. F. Molisch, "Terahertz wireless channels: A holistic survey on measurement, modeling, and analysis," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 3, pp. 1670–1707, 3rd Quart., 2022.

[2] H.-J. Song and N. Lee, "Terahertz communications: Challenges in the next decade," *IEEE Trans. Terahertz Sci. Technol.*, vol. 12, no. 2, pp. 105–117, Mar. 2022.

[3] A. Zandamela, A. Chiumento, N. Marchetti, and A. Narbudowicz, "Angle of arrival estimation via small IoT devices: Miniaturized arrays vs. MIMO antennas," *IEEE Internet Things Mag.*, vol. 5, no. 2, pp. 146–152, Jun. 2022.

[4] M. Capra, R. Peloso, G. Masera, M. R. Roch, and M. Martina, "Edge computing: A survey on the hardware requirements in the Internet of Things world," *Future Internet*, vol. 11, no. 4, p. 100, Apr. 2019.

[5] N. A. Sulieman, L. Ricciardi Celsi, W. Li, A. Zomaya, and M. Villari, "Edge-oriented computing: A survey on research and use cases," *Energies*, vol. 15, no. 2, p. 452, Jan. 2022.

[6] C. Xu, S. Jiang, G. Luo, G. Sun, N. An, G. Huang, and X. Liu, "The case for FPGA-based edge computing," *IEEE Trans. Mobile Comput.*, vol. 21, no. 7, pp. 2610–2619, Jul. 2022.

[7] S. Biookaghazadeh, M. Zhao, and F. Ren, "Are FPGAs suitable for edge computing?" in *Proc. USENIX Workshop Hot Topics Edge Comput.*, 2018, pp. 1–6.

[8] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2568–2599, 3rd Quart., 2017.

[9] D. Vecchia, P. Corbalan, T. Istomin, and G. P. Picco, "TALLA: Large-scale TDoA localization with ultra-wideband radios," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2019, pp. 1–8.

[10] S. Yiu, M. Dashti, H. Claussen, and F. Perez-Cruz, "Wireless RSSI fingerprinting localization," *Signal Process.*, vol. 131, pp. 235–244, Feb. 2017.

[11] N. Singh, S. Choe, and R. Punmiya, "Machine learning based indoor localization using Wi-Fi RSSI fingerprints: An overview," *IEEE Access*, vol. 9, pp. 127150–127174, 2021.

[12] A. A. A. Alkhatib, M. W. Elbes, and E. M. A. Maria, "Improving accuracy of wireless sensor networks localisation based on communication ranging," *IET Commun.*, vol. 14, no. 18, pp. 3184–3193, Nov. 2020.

[13] M. Woolley. (2021). *Bluetooth Direction Finding—A Technical Overview*. Bluetooth SIG. [Online]. Available: <https://www.bluetooth.com/bluetooth-resources/bluetooth-direction-finding/>

[14] H. Krim and M. Viberg, "Two decades of array signal processing research: The parametric approach," *IEEE Signal Process. Mag.*, vol. 13, no. 4, pp. 67–94, Jul. 1996.

[15] Z. Tian and H. L. Van Trees, "DOA estimation with hexagonal arrays," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, vol. 4, May 1998, pp. 2053–2056.

[16] E. Gentilho, P. R. Scalassara, and T. Abrão, "Direction-of-arrival estimation methods: A performance-complexity tradeoff perspective," *J. Signal Process. Syst.*, vol. 92, no. 2, pp. 239–256, Feb. 2020.

[17] P. K. Eranti and B. D. Barkana, "An overview of direction-of-arrival estimation methods using adaptive directional time-frequency distributions," *Electronics*, vol. 11, no. 9, p. 1321, Apr. 2022.

[18] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. AP-34, no. 3, pp. 276–280, Mar. 1986.

[19] A. Barabell, "Improving the resolution performance of eigenstructure-based direction-finding algorithms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 8, Sep. 1983, pp. 336–339.

[20] R. Roy and T. Kailath, "Esprit-estimation of signal parameters via rotational invariance techniques," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 7, pp. 984–995, Jul. 1989.

[21] B. H. Fleury, D. Dahlhaus, R. Heddergott, and M. Tschudin, "Wideband angle of arrival estimation using the SAGE algorithm," in *Proc. IEEE Int. Symp. Spread Spectr. Techn. Appl. (ISSSTA)*, vol. 1, Sep. 1996, pp. 79–85.

[22] L. C. Godara, "Application of antenna arrays to mobile communications. II. Beam-forming and direction-of-arrival considerations," *Proc. IEEE*, vol. 85, no. 8, pp. 1195–1245, Aug. 1997.

[23] M. Wolfel and J. McDonough, "Minimum variance distortionless response spectral estimation," *IEEE Signal Process. Mag.*, vol. 22, no. 5, pp. 117–126, Sep. 2005.

[24] P. Gupta, K. Aditya, and A. Datta, "Comparison of conventional and subspace based algorithms to estimate direction of arrival (DOA)," in *Proc. Int. Conf. Commun. Signal Process. (ICCSPP)*, Apr. 2016, pp. 251–255.

[25] A. Paulraj, B. Ottersten, R. Roy, A. Swindlehurst, G. Xu, and T. Kailath, "16 subspace methods for directions-of-arrival estimation," *Handbook Statist.*, vol. 10, pp. 693–739, Jan. 1993.

- [26] R. L. Haupt and Y. Rahmat-Samii, "Antenna array developments: A perspective on the past, present and future," *IEEE Antennas Propag. Mag.*, vol. 57, no. 1, pp. 86–96, Feb. 2015.
- [27] S.-H. Jeong, B.-K. Son, and J.-H. Lee, "Asymptotic performance analysis of the MUSIC algorithm for direction-of-arrival estimation," *Appl. Sci.*, vol. 10, no. 6, p. 2063, Mar. 2020.
- [28] Y. Jung, H. Jeon, S. Lee, and Y. Jung, "Scalable ESPRIT processor for direction-of-arrival estimation of frequency modulated continuous wave radar," *Electronics*, vol. 10, no. 6, p. 695, Mar. 2021.
- [29] Z. Dai, Y. He, V. Tran, N. Trigoni, and A. Markham, "DeepAoANet: Learning angle of arrival from software defined radios with deep neural networks," *IEEE Access*, vol. 10, pp. 3164–3176, 2022.
- [30] G. Avitabile, A. Florio, and G. Coviello, "Angle of arrival estimation through a full-hardware approach for adaptive beamforming," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 12, pp. 3033–3037, Dec. 2020.
- [31] J. S. Tavares, H. H. Avelar, H. M. Salgado, and L. M. Pessoa, "A Gaussian window for interference mitigation in Ka-band digital beamforming systems," in *Proc. 13th Int. Symp. Commun. Syst., Netw. Digit. Signal Process. (CSNDSP)*, Jul. 2022, pp. 197–201.
- [32] A. Florio, G. Avitabile, and G. Coviello, "Multiple source angle of arrival estimation through phase interferometry," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 3, pp. 674–678, Mar. 2022.
- [33] A. A. Hussain, N. Tayem, M. O. Butt, A.-H. Soliman, A. Alhamed, and S. Alshebeili, "FPGA hardware implementation of DOA estimation algorithm employing LU decomposition," *IEEE Access*, vol. 6, pp. 17666–17680, 2018.
- [34] Bluetooth SIG. (2019). *Bluetooth Specification Version 5.1*. [Online]. Available: <https://www.bluetooth.com/bluetooth-resources/bluetooth-core-specification-v5-1-feature-overview/>
- [35] C. Li, J. Zhen, K. Chang, A. Xu, H. Zhu, and J. Wu, "An indoor positioning and tracking algorithm based on angle-of-arrival using a dual-channel array antenna," *Remote Sens.*, vol. 13, no. 21, p. 4301, Oct. 2021.
- [36] J. Baek, Y. Choi, C. Lee, J. Suh, and S. Lee, "BBUNS: Bluetooth beacon-based underground navigation system to support mine haulage operations," *Minerals*, vol. 7, no. 11, p. 228, Nov. 2017.
- [37] M. N. K. Boulos and G. Berry, "Real-time locating systems (RTLs) in healthcare: A condensed primer," *Int. J. Health Geographics*, vol. 11, no. 1, p. 25, 2012.
- [38] J. Konecny, M. Prauzek, R. Martinek, L. Michalek, and M. Tomis, "Real-time patient localization in urgent care: System design and hardware perspective," in *Proc. IEEE 20th Int. Conf. E-Health Netw., Appl. Services (Healthcom)*, Sep. 2018, pp. 1–5.
- [39] R. Giuliano, F. Mazzenga, M. Petracca, and M. Vari, "Indoor localization system for first responders in emergency scenario," in *Proc. 9th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jul. 2013, pp. 1821–1826.
- [40] A. Rácz-Szabó, T. Ruppert, L. Bántay, A. Löcklin, L. Jakab, and J. Abonyi, "Real-time locating system in production management," *Sensors*, vol. 20, no. 23, p. 6766, Nov. 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/20/23/6766>
- [41] G. Pau, F. Arena, Y. E. Gebremariam, and I. You, "Bluetooth 5.1: An analysis of direction finding capability for high-precision location services," *Sensors*, vol. 21, no. 11, p. 3589, May 2021. [Online]. Available: <https://www.mdpi.com/1424-8220/21/11/3589>
- [42] G. Iannizzotto, M. Milici, A. Nucita, and L. Lo Bello, "A perspective on passive human sensing with bluetooth," *Sensors*, vol. 22, no. 9, p. 3523, 2022. [Online]. Available: <https://www.mdpi.com/1424-8220/22/9/3523>
- [43] I. The MathWorks. (2022). *Bluetooth Toolbox—Simulate, Analyze, and Test Bluetooth Communications Systems, Natick, Massachusetts, United State*. [Online]. Available: <https://www.mathworks.com/help/bluetooth/>
- [44] N. Paulino, L. M. Pessoa, A. Branquinho, and E. Goncalves, "Evaluating a novel Bluetooth 5.1 AoA approach for low-cost indoor vehicle tracking via simulation," in *Proc. Joint Eur. Conf. Netw. Commun. 6G Summit (EuCNC/6G Summit)*, Jun. 2021, pp. 259–264.
- [45] N. Paulino, L. M. Pessoa, A. Branquinho, and E. Goncalves, "Design and experimental evaluation of a Bluetooth 5.1 antenna array for angle-of-arrival estimation," in *Proc. 13th Int. Symp. Commun. Syst., Netw. Digit. Signal Process. (CSNDSP)*, Jul. 2022, pp. 625–630.
- [46] S. Zekavat, R. M. Buehrer, G. D. Durgin, L. Lovisolo, Z. Wang, S. T. Goh, and A. Ghasemi, "An overview on position location: Past, present, future," *Int. J. Wireless Inf. Netw.*, vol. 28, no. 1, pp. 45–76, Mar. 2021.
- [47] S. He, H. Long, and W. Zhang, "Multi-antenna array-based AoA estimation using Bluetooth low energy for indoor positioning," in *Proc. 7th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2021, pp. 2160–2164.
- [48] Z. Hajiakhondi-Meybodi, M. Salimibeni, K. N. Plataniotis, and A. Mohammadi, "Bluetooth low energy-based angle of arrival estimation via switch antenna array for indoor localization," in *Proc. IEEE 23rd Int. Conf. Inf. Fusion (FUSION)*, Jul. 2020, pp. 1–6.
- [49] F. A. Toasa, L. Tello-Oquendo, C. R. Peñafiel-Ojeda, and G. Cuzco, "Experimental demonstration for indoor localization based on AoA of Bluetooth 5.1 using software defined radio," in *Proc. IEEE 18th Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2021, pp. 1–4.
- [50] Y. Fan, C. E. Ardıc, M. Trinh-Hoang, and M. Pesavento, "Decentralized online direction-of-arrival estimation and tracking," in *Proc. IEEE 12th Sensor Array Multichannel Signal Process. Workshop (SAM)*, Jun. 2022, pp. 6–10.
- [51] N. Bnilam, E. Tanghe, J. Steckel, W. Joseph, and M. Weyn, "ANGLE: Angular location estimation algorithms," *IEEE Access*, vol. 8, pp. 14620–14629, 2020.
- [52] A. Alteneiji, U. Ahmad, K. Poon, N. Ali, and N. Almoosa, "Angle of arrival estimation in indoor environment using machine learning," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, Sep. 2021, pp. 1–6.
- [53] M. Chen, Y. Gong, and X. Mao, "Deep neural network for estimation of direction of arrival with antenna array," *IEEE Access*, vol. 8, pp. 140688–140698, 2020.
- [54] W. Zhu, M. Zhang, P. Li, and C. Wu, "Two-dimensional DOA estimation via deep ensemble learning," *IEEE Access*, vol. 8, pp. 124544–124552, 2020.
- [55] W. Suleiman, M. Pesavento, and A. Zoubir, "Decentralized direction finding using partly calibrated arrays," in *Proc. 21st Eur. Signal Process. Conf.*, 2013, pp. 1–5.
- [56] A. Florio and G. Avitabile, "Characterization of a multisource angle of arrival estimation technique based on phase interferometry," in *Proc. 18th Int. Conf. Synth., Modeling, Anal. Simulation Methods Appl. Circuit Design (SMACD)*, Jun. 2022, pp. 1–4.
- [57] A. A. Hussain, N. Tayem, A.-H. Soliman, and R. M. Radaydeh, "FPGA-based hardware implementation of computationally efficient multi-source DOA estimation algorithms," *IEEE Access*, vol. 7, pp. 88845–88858, 2019.
- [58] N. BniLam, D. Joosens, J. Steckel, and M. Weyn, "Low cost AoA unit for IoT applications," in *Proc. 13th Eur. Conf. Antennas Propag.*, 2019, pp. 1–5.
- [59] A. Herzog and E. A. P. Habetts, "Eigenbeam-ESPRIT for DOA-vector estimation," *IEEE Signal Process. Lett.*, vol. 26, no. 4, pp. 572–576, Apr. 2019.
- [60] Z. Li, W. Wang, R. Jiang, S. Ren, X. Wang, and C. Xue, "Hardware acceleration of MUSIC algorithm for sparse arrays and uniform linear arrays," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 69, no. 7, pp. 2941–2954, Jul. 2022.
- [61] A. M. Ahmed, O. Eissa, and A. Sezgin, "Deep autoencoders for DOA estimation of coherent sources using imperfect antenna array," in *Proc. 3rd Int. Workshop Mobile Terahertz Syst. (IWMTS)*, Jul. 2020, pp. 1–5.
- [62] A. Khan, S. Wang, and Z. Zhu, "Angle-of-arrival estimation using an adaptive machine learning framework," *IEEE Commun. Lett.*, vol. 23, no. 2, pp. 294–297, Feb. 2019.
- [63] Y. Kase, T. Nishimura, T. Ohgane, Y. Ogawa, D. Kitayama, and Y. Kishiyama, "DOA estimation of two targets with deep learning," in *Proc. 15th Workshop Positioning, Navigat. Commun. (WPNC)*, Oct. 2018, pp. 1–5.
- [64] O. Bialer, N. Garnett, and T. Tirer, "Performance advantages of deep neural networks for angle of arrival estimation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 3907–3911.
- [65] H. Huang, J. Yang, H. Huang, Y. Song, and G. Gui, "Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system," *IEEE Trans. Veh. Tech.*, vol. 67, no. 9, pp. 8549–8560, Jun. 2018.
- [66] S. Navabi, C. Wang, O. Y. Bursalioglu, and H. Papadopoulos, "Predicting wireless channel features using neural networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.
- [67] S. A. Shaikh and A. M. Tonello, "Radio source localization in multipath channels using EM lens assisted massive antennas arrays," *IEEE Access*, vol. 7, pp. 9001–9012, 2019.
- [68] C. Usha Padmini and P. S. Naidu, "Circular array and estimation of direction of arrival of a broadband source," *Signal Process.*, vol. 37, no. 2, pp. 243–254, May 1994.
- [69] A. Manikas and C. Proukakis, "Modeling and estimation of ambiguities in linear arrays," *IEEE Trans. Signal Process.*, vol. 46, no. 8, pp. 2166–2179, Aug. 1998.

- [70] P. Ioannides and C. A. Balanis, "Uniform circular arrays for smart antennas," *IEEE Antennas Propag. Mag.*, vol. 47, no. 4, pp. 192–206, Aug. 2005.
- [71] S. V. Ballandovich, Y. G. Antonov, G. A. Kostikov, L. M. Liubina, and M. I. Sugak, "Development of the ultra-wideband circular antenna array," in *Proc. Syst. Signal Synchronization, Generating Process. Telecommun. (SYNCHROINFO)*, 2020, pp. 1–4.
- [72] A. V. S. Swathi, V. V. S. S. S. Chakravarthy, and M. V. Krishna, "Circular antenna array optimization using modified social group optimization algorithm," *Soft Comput.*, vol. 25, no. 15, pp. 10467–10475, Aug. 2021.
- [73] N. Semiconductor. (2021). *nRF52811 Product Specification v1.1*. [Online]. Available: [https://infocenter.nordicsemi.com/pdf/nRF52811\\_PS\\_v1.1.pdf](https://infocenter.nordicsemi.com/pdf/nRF52811_PS_v1.1.pdf)
- [74] M. Cominelli, P. Patras, and F. Gringoli, "Dead on arrival: An empirical study of the Bluetooth 5.1 positioning system," in *Proc. 13th Int. Workshop Wireless Netw. Testbeds, Experim. Eval. Characterization*, 2019, pp. 13–20.
- [75] J. Traa, "Least-squares intersection of lines," Univ. Illinois Urbana-Champaign, Champaign, IL, USA, 2013.
- [76] Skyworks. (2019). *SKY13418-485LF 0.1-6.0 GHz SP8T Antenna Switch*. [Online]. Available: <https://www.skyworksinc.com/Products/Switches/SKY13418-485LF>
- [77] T. Instruments. (2001). *TPS76925*. [Online]. Available: <https://www.ti.com/lit/ds/symlink/tps769.pdf>
- [78] N. Semiconductor. (2018). *nRF52840 Development Kit*. [Online]. Available: [https://infocenter.nordicsemi.com/pdf/nRF52832\\_PS\\_v1.4.pdf](https://infocenter.nordicsemi.com/pdf/nRF52832_PS_v1.4.pdf)
- [79] X. Zhao, Z. Xiao, A. Markham, N. Trigoni, and Y. Ren, "Does BTLE measure up against WiFi? A comparison of indoor location performance," in *Proc. Eur. Wireless 20th Eur. Wireless Conf.*, 2014, pp. 1–6.
- [80] M. G. Jadidi, M. Patel, J. V. Miro, G. Dissanayake, J. Biehl, and A. Girgensohn, "A radio-inertial localization and tracking system with BLE beacons prior maps," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat.*, 2018, pp. 206–212.
- [81] M. Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: <https://www.tensorflow.org/>



**NUNO PAULINO** received the M.Sc. degree in electrical and computer engineering from the Faculty of Engineering, University of Porto, in 2011, and the Ph.D. degree in electrical and computer engineering from the University of Porto, in 2015. He is currently an Assistant Professor at the University of Porto and a Researcher with INESC TEC. His research interests include reconfigurable and embedded systems in FPGAs, computing architectures, and tools for hardware/software co-design.



**LUÍS M. PESSOA** (Senior Member, IEEE) received the Licenciatura and Ph.D. degrees in electrical and computer engineering from the Faculty of Engineering, University of Porto, in 2006 and 2011, respectively. He is currently a Senior Researcher at the Centre of Telecommunications and Multimedia, INESC TEC, where he leads the Optical and Electronic Technologies Group. His research interests include coherent optical systems, radio-over-fibre, RF/microwave devices and antennas, and underwater wireless power/communications.

• • •