Simulation-Based Neural Network Models for Improved Bluetooth Ranging and Localization

Márk Klepács

Electronic Systems 2024-06

Master's Thesis

This page is blank on purpose

Electronic systems
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Simulation-Based Neural Network Models for Improved Bluetooth Ranging and Localization

Theme:
Master Report

Project Period:
Spring 2024

Project Group:
ES-1029d

Participant(s):
Márk Klepács

Supervisor(s):
Ming Shen
Petar Popovski

Page Numbers: 45

Date of Completion:
29th of May 2024

Abstract:

This thesis investigates the application of machine learning techniques to enhance the accuracy of Bluetooth-based ranging using channel sounding. Channel responses were simulated for multiple environments using the Sionna ray tracer, generating simulated Bluetooth channel sounding measurements to construct datasets. The study evaluated multiple representations for training neural networks on this simulation dataset and explored the transferability of simulation-trained models to real measurement data. The generated data demonstrated sufficient accuracy to train models that generalize effectively to real-world scenarios. Notably, the simulation-trained models achieved ranging accuracy comparable to or exceeding that of the MUSIC algorithm.

Contents

1 Introduction

Bluetooth technology is primarily known for its device communication capabilities, enabling wireless connections between a wide range of devices. Over the years, Bluetooth has become widespread, embedded in countless devices from smartphones, wireless audio devices, computer peripherals to smart home systems. Recent advancements have extended its functionality to include location services, broadening its application scope. [1]

Bluetooth technology is now widely used as a device positioning technology to address the increasing demand for high-accuracy location services. enabling one Bluetooth device to determine the presence, distance, and direction of another device. This has made Bluetooth essential in various industries. For instance, powering the rapid growth of real-time locating systems (RTLS) used for tracking assets and people, such as locating tools and workers in warehouses or medical devices and patients in hospitals. Additionally, an increasing number of consumers are attaching Bluetooth tags to personal items like keys, wallets, and purses. Moreover, Bluetooth technology has expanded its role in access control systems, allowing smartphones to be used as secure digital keys that unlock doors and spaces as users approach their cars, homes, or offices. This technology also facilitates indoor positioning and navigation services, similar to GPS but tailored for indoor environments like airports, train stations, and shopping malls. [2] [3]

These advancements have significantly increased the demand for precise and reliable location services.

1.1 Bluetooth Ranging methods

There are several Bluetooth ranging methods, each with its own advantages and limitations.[4]

1.1.1 Recieved Signal Strength Indicator RSSI

RSSI is a well-established method for Bluetooth ranging. It relies on the principle that the amplitude of a radio signal weakens as the distance between transmitter and receiver increases. This signal attenuation allows for a rough estimation of the distance between Bluetooth devices. The benefit of RSSI is that it's a low-cost and readily available capability present in all Bluetooth devices.

However, signal strength is highly dependent on various factors such as the environment, absorption, diffraction, interference, the way the device is held, and its orientation. Consequently, this results in a low ranging accuracy, typically within the range of three to five meters. To improve the accuracy of RSSI, multiple beacons

can be used along with detailed characterization of the environment. This technique is known as fingerprinting.

1.1.2 Angle of Arrival (AoA) and Angle of Departure (AoD)

The introduction of Bluetooth core specifications v5.1 brought significant advancements in direction finding by incorporating Angle of Arrival (AoA) and Angle of Departure (AoD) measurements. These methods enable the estimation of position by using multiple anchor points and applying trigonometric calculations to determine the position and distance of a beacon. [5] [6]

AoA is measured on the receiver side, utilizing multiple antennas to capture the phase shift of the arriving signal between them. This technique allows the receiver to determine the direction from which the signal is coming. Conversely, AoD involves the transmitter sending signals from multiple antennas, and the receiver measures the angle of the outgoing signal. For AoD to work effectively, the receiver must have detailed knowledge of the transmitters antenna array design, making AoD more complex and less commonly used than AoA in practical implementations.

Despite their complexities, AoA and AoD can achieve sub-meter accuracy in localization. However, they require beacons to be placed in known locations with multiple antennas. The accuracy of these methods can be compromised by obstacles, reflective surfaces, and multipath propagation, which introduce errors in the measurements. To mitigate these challenges, measurements from multiple beacons are typically collected and processed by a positioning engine. An advanced positioning engine, equipped with a model of the specific radio environment, can account for multipath propagation and improve the accuracy of the location estimation.

1.1.3 Channel sounding

Channel sounding is the latest breakthrough in high-accuracy distance measurement. This technique measures the channel's response by sending an unmodulated constant tone signal, while the receiver captures the in-phase and quadrature components of the signal.

By analyzing these responses across multiple frequency channels, it is possible to determine the time delays and attenuation effects associated with different signal paths. This measurement method enables several methods of estimating distance, such as phase-based ranging (PBR).

The Bluetooth Special Interest Group (SIG) is actively developing channel sounding techniques to achieve 10 cm ranging accuracy. This advancement will significantly enhance applications such as secure access, digital key systems, and proximity services, providing high-accuracy positioning solutions that expand the potential applications of Bluetooth technology.

1.2 This work

This work explores machine learning techniques of utilizing Bluetooth channel sounding measurements for accurate distance estimation, exploring different options of improving ranging performance, model training, channel simulation and dataset generation.

This work considers the single antenna ranging, with static radio devices in line of sight conditions.

Acknowledgement

I would like to extend my sincerest gratitude to the supervisors, Ming Shen and Dr. Petar Popovski, for their invaluable guidance, critical insights, and enduring support throughout the process of researching and writing this thesis. Their expertise and thoughtful supervision have significantly contributed to the refinement of this work. Gratitude is also owed to Samsung Denmark Research Center, particularly to Jesper Hemming Sørensen, for their collaboration and for providing invaluable guidance, and essential data that formed the foundation of this work. Their willingness to participate and assist in our research has been instrumental in the successful completion of this thesis.

2 | Theory

This section introduces the theoretical background about the wireless channel model, Bluetooth channel sounding, and different algorithms for ranging from channel sounding measurements.

2.1 Wireless channel model

As a wireless signal $x(t)$ propagates through the channel, it experiences large scale effects such as path loss, shadowing as well as small-scale effects like constructive, destructive interference of multiple paths the signal takes. The channel's effects are typically modeled as the channel response, denoted by $h(t)$. The received signal $y(t)$ is the result of linear convolution between the transmitted signal $x(t)$ and the channel response $h(t)$.[7] This is mathematically represented by the following equation:

$$y(t) = x(t) * h(t) \tag{2.1}$$

Consider an indoor multipath environment with multiple reflectors, where the signal can travel through various paths. Each of these N paths experiences different attenuation ρ_i and delay τ_i.

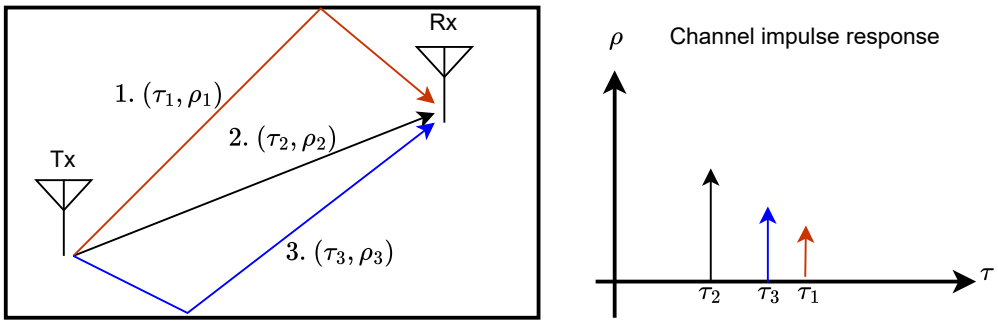

Figure 2.1: Illustration of the multipath components in an indoor environment. Each path i has an attenuation factor ρ_i and a delay τ_i. The channel impulse response is the sum of these delayed and attenuated components.

The received signal is the superposition of these delayed and attenuated versions of the original signal. Mathematically, the channel response $h(t)$ for this scenario can be expressed as[8]:

$$h(t) = \sum_{i=1}^{N} \rho_i \cdot \delta(t - \tau_i) \tag{2.2}$$

In case of Bluetooth channel sounding, the transmitted signal $x(t)$ is an unmodulated complex sinusoid transmitted at carrier frequency f^k.

$$x(t) = e^{j2\pi f^k t} \tag{2.3}$$

Substituting $x(t)$ and $h(t)$ into equation (2.1) results in:

$$y(t) = x(t) * h(t) = e^{j2\pi f^k t} * \sum_{i=1}^{N} \rho_i \cdot \delta(t - \tau_i) \tag{2.4}$$

Performing the convolution, we obtain the received signal:

$$y(t) = \sum_{i=1}^{N} \rho_i e^{j2\pi f^k (t-\tau_i)} = \sum_{i=1}^{N} \rho_i e^{-j2\pi f^k \tau_i} e^{j2\pi f^k t} \tag{2.5}$$

We can define a complex channel gain a_i for each path, which combines the effects of attenuation ρ_i experienced by the signal on that path and the phase shift introduced due to the delay τ_i:

$$a_i = \rho_i e^{-j2\pi f^k \tau_i} \tag{2.6}$$

Substituting a_i into the previous equation simplifies the expression for the received signal:

$$y(t) = \sum_{i=1}^{N} a_i e^{j2\pi f^k t} = a \cdot e^{j2\pi f^k t} \tag{2.7}$$

2.2 Bluetooth Channel sounding

In Bluetooth channel sounding, two radio transceivers participate: the Initiator and the Reflector. A channel sounding procedure consists of a series of radio operations known as channel sounding events. Each channel sounding event comprises subevents, and each subevent includes multiple channel sounding steps.

A channel sounding step involves a series of synchronized tone transmissions between devices, where the Phase Lock Loops (PLLs) remain in lock. During a channel sounding step, the initiator transmits an unmodulated, continuous tone signal S at frequency channel f^k to the reflector. This signal propagates through the channel, and the reflector measures the in-phase and quadrature components of the signal as S_R^k. (see Figure 2.2).

$$S_R^k = a \cdot e^{j2\pi f^k \tau} \tag{2.8}$$

Here, a is the channel coefficient, while τ is the delay. The starting phase of the transmitted signal depends on the initiator's internal clock phase ϕ_I, and the reflector's measurement is relative to its own clock phase ϕ_R.

$$S_R^k = a \cdot e^{j2\pi f^k \tau + \phi_I - \phi_R} \tag{2.9}$$

Next, the roles of transmitter and receiver reverse: the reflector transmits a continuous tone signal back to the initiator on the same frequency channel K. This signal propagates through the channel again and is measured at the initiator as S_I^k (see Figure 2.2).

$$S_I^k = a \cdot e^{j2\pi f^k \tau + \phi_R - \phi_I} \tag{2.10}$$

Since the roles are now reversed, the starting phase now depends on the reflector's clock phase, and it is measured relative to the initiator's phase. Taking the product of S_I^k and S_R^k results in a two-way channel response S_{TW}^k, which cancels out the clock phase offsets[9].

$$S_{TW}^k = S_I^k \cdot S_R^k = a^2 \cdot e^{-j4\pi f^k \tau} \tag{2.11}$$

Following each channel sounding subevent, the devices exchange measurement results to perform this computation. The channel sounding procedure repeats this process across all frequency channels, up to 72 channels across the entire frequency band.

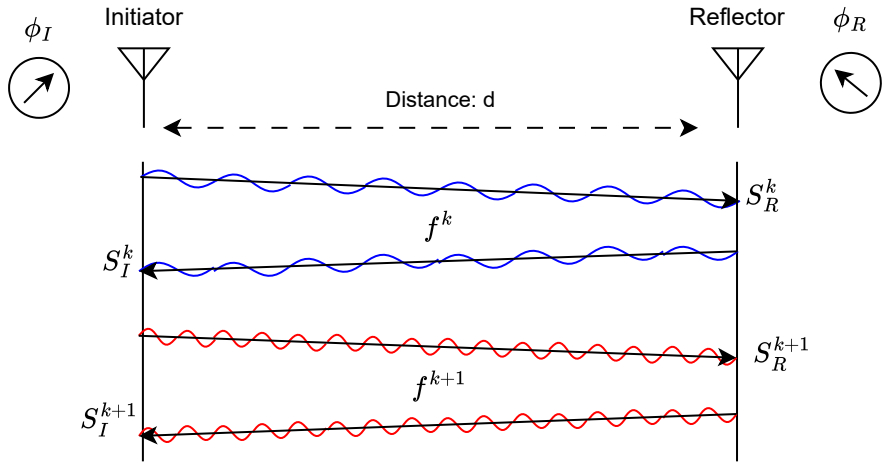

Figure 2.2: Illustration of the Bluetooth channel sounding process. The initiator transmits a continuous tone signal S at frequency channel f^k to the reflector, which measures the in-phase and quadrature components of the signal as S_R^k. The roles then reverse, with the reflector transmitting the signal back to the initiator, which measures it as S_I^k. This process is repeated across multiple frequency channels.

2.3 Ranging algorithms

Channel sounding does not specify which algorithm should be used to calculate distance estimates from the measurement data. This flexibility allows companies to tailor their solutions to various use cases, balancing computational complexity with the required estimation accuracy and the expected radio environments. Examples of ranging algorithms include simple phase difference calculations, fast Fourier transform (FFT) based methods, and advanced super-resolution algorithms designed to handle severe multipath scenarios, particularly in indoor environments.

2.3.1 Multi-carrier phase difference(MCPD)

Consider a Line of Sight (LOS) scenario, where the number of propagation paths N is 1. The received signal $y(t)$ can be described as:

$$y(t) = \rho \cdot e^{j2\pi f^k(t-\tau)} \tag{2.12}$$

The phase the received signal ϕ_{rx} is given by:

$$\phi_{rx} = \angle y(t) = 2\pi f^k \tau \tag{2.13}$$

Radio signals traveling at the speed of light c cover a distance d, resulting in a delay τ described as:

$$\tau = \frac{d}{c} \tag{2.14}$$

Substituting τ into the phase equation gives:

$$\phi_a(f^k, d) = \frac{2\pi f^k d}{c} \tag{2.15}$$

In channel sounding, a back-and-forth transmission is performed to cancel out clock offsets. Consequently, the signal travels the distance twice, resulting in a two-way phase θ^k described as:

$$\theta^k = \angle S^k_{TW} = \frac{4\pi f^k d}{c} \tag{2.16}$$

Solving for distance yields:

$$d = \frac{c}{4\pi} \cdot \frac{\theta^k}{f^k} \tag{2.17}$$

However, computing the distance from one phase measurement introduces ambiguity, as a valid distance solution exists at every wavelength. To resolve this ambiguity, two carrier frequencies can be used. The distance can be determined by:

$$d = \frac{c}{4\pi} \cdot \frac{\theta^k + \theta^{k+1}}{f^k + f^{k+1}} \tag{2.18}$$

In practice, using only two frequencies results in poor ranging accuracy. Therefore, multiple frequencies (preferably spanning the entire spectrum) are used to determine the slope, improving the accuracy of the distance measurement:

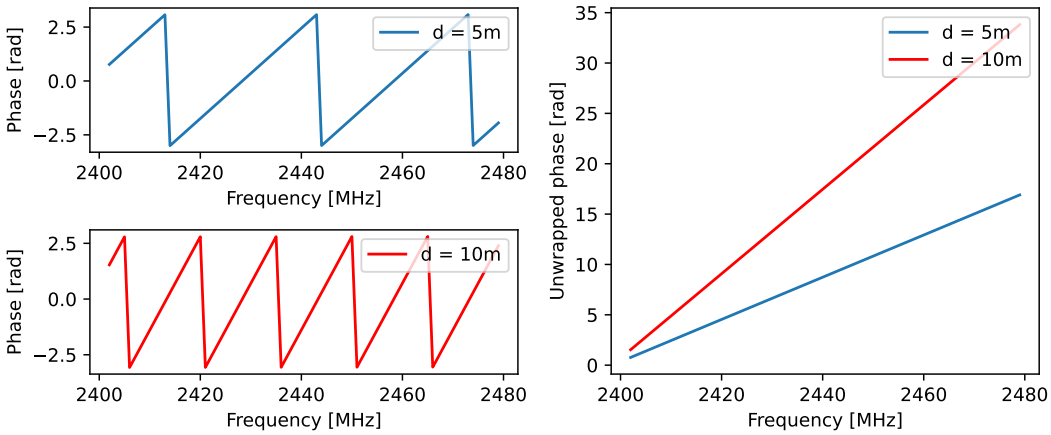

Figure 2.3: Phase diagrams of Bluetooth channel sounding at distances of 5 meters and 10 meters. The left plots show the wrapped phase as a function of frequency, while the right plot shows the corresponding unwrapped phase.

Since the measured phase values are constrained between $-\pi$ and π, a phenomenon known as phase wrapping occurs when using multiple frequencies. This wrapping creates a sawtooth-like pattern in the phase response. To address phase wrapping and obtain an accurate measurement of the phase change across the entire frequency band, a technique called phase unwrapping is employed. Phase unwrapping essentially removes these jumps and reconstructs the continuous phase behavior. Following unwrapping, the slope of the resulting unwrapped phase response can be calculated by applying linear regression. This is illustrated in Figure 2.3.

This slope is then used in the following equation to determine the distance d between the transmitter and receiver:

$$d = \frac{c}{4\pi} \cdot slope \tag{2.19}$$

However in an indoors scenario with multiple propagation paths, the measured signal is the superimposition of all propagation paths. This causes distortions on the measurements which negatively effect the slope measurement used to determine the distance.

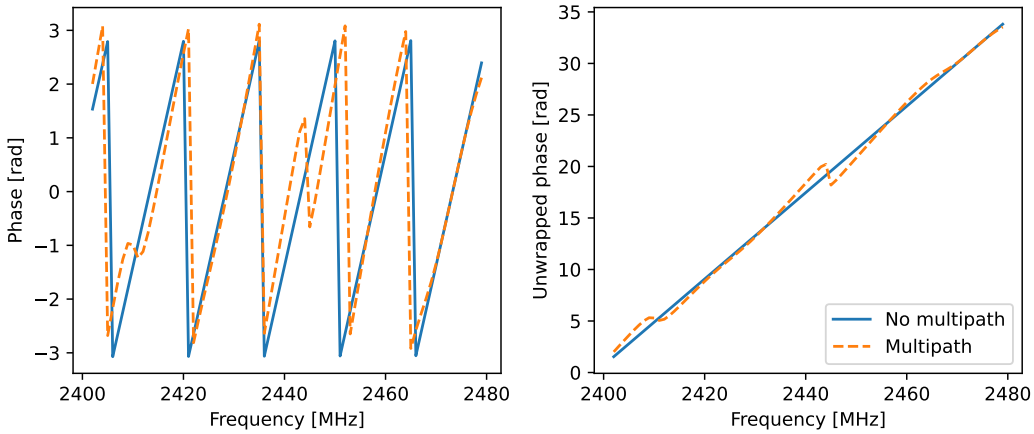

Figure 2.4: Illustration of the distortions caused by multipath to the phase angles and the unwrapped phase. The left plot shows the wrapped phase as a function of frequency for signals with and without multipath. The right plot shows the unwrapped phase for both cases. The multipath signal contains three components at distances of 10 meters, 12 meters, and 20 meters.

This ranging method is simple to implement, and provides good range estimates in non complex environments. In environments with many reflecting surfaces, this method can overestimate the distance.

2.3.2 Inverse Fourier transform (IFFT)

The channel sounding measurements are obtained across multiple frequency channels, resulting in two-way channel responses s_{TW}^{k} in the frequency domain. These frequency domain measurements can be converted back to the time domain using an inverse discrete Fourier transform (IDFT), specifically the inverse fast Fourier transform (IFFT) for computational efficiency.

By transforming the frequency domain measurements back to the time domain, it becomes possible to resolve the different arriving echoes if the paths are distinguishable with sufficient spacing. This conversion process enables the identification of individual multipath components based on their arrival times.

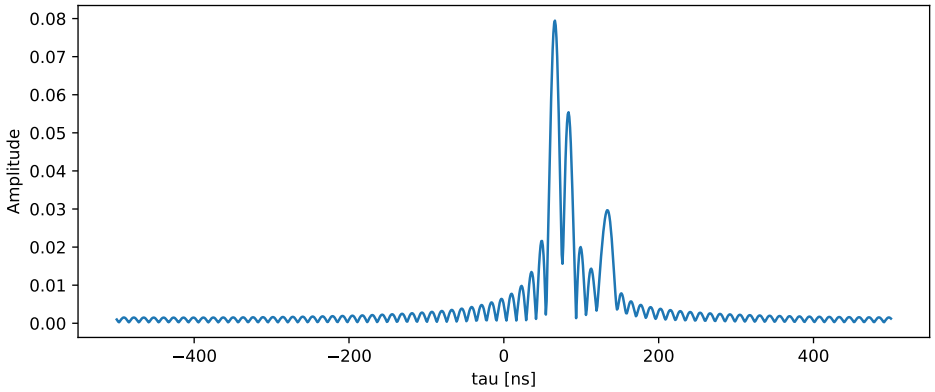

Figure 2.5: Inverse Fourier transform of a signal from three components 10,12 and 20 meters.

The peaks in the time domain representation indicate the arrival delays of the signal components. By identifying the first peak, the line of sight component's time delay τ_{LOS} can be determined. From the arrival time, distance can be determined as $d = c \cdot \tau_{LOS}$.

Addressing the limitations of the MCPD method, the IFFT method allows for the separation of different signal paths, making it possible to isolate the line-of-sight component from multipath reflections.

However, the IFFT method's ability to distinguish between closely spaced multipath components is constrained by the resolution of the IFFT. The resolution is determined by the bandwidth of the measured frequency spectrum and the number of points used in the IFFT (nfft). Increasing the number of nfft points can improve resolution but also increases computational complexity. Moreower, IFFT assumes uniform sampling in the frequency domain. Any gaps or non-uniformities in the frequency measurements can introduce artifacts and reduce the effectiveness of the transformation.

2.3.3 Multiple SIgnal Classification (MUSIC)

Similarly to IFFT, MUSIC can decompose the arriving signal into its individual components, and detecting the first peak gives the delay of the first arrival of the signal.

The MUSIC algorithm involves several key steps. First, a Hankel matrix is constructed from the measurements. From this Hankel matrix, a covariance matrix is computed, encapsulating the statistical properties of the signals. Eigenvalue decomposition is then performed on the covariance matrix to separate the signal subspace from the noise subspace. The eigenvectors corresponding to the largest eigenvalues form the signal subspace, while the remaining eigenvectors constitute the noise

subspace.

To compute the MUSIC pseudospectrum, the array steering vectors are projected onto the noise subspace. This pseudospectrum can be evaluated at a high resolution, allowing for the distinction between closely spaced signals. Peaks in the pseudospectrum correspond to the delays of arrivals the signals. By identifying the first peak, the time delay of the line-of-sight component, τ_{LOS}, can be determined. The corresponding distance is then calculated as $d = c \cdot \tau_{LOS}$.

The MUSIC algorithm offers several advantages, including its ability to distinguish between closely spaced signal components, providing high-resolution estimates of arrival times. Additionally, by leveraging the orthogonality between the signal and noise subspaces, MUSIC can effectively separate signal components even in the presence of noise. However, the algorithm also has limitations. The eigenvalue decomposition step can be computationally intensive, potentially limiting its applicability in real-time scenarios. Furthermore, the performance of MUSIC depends on the accurate estimation of the number of signal sources, which can be challenging in practical applications.

2.3.4 Neural Network ranging

Neural networks are a type of machine learning model that can be used as a regressor to predict continuous variables, such as distance. These networks process input data through layers of artificial neurons, or "nodes." Each neuron applies a transformation to its inputs and passes the results to the next layer, with the final layer producing an output prediction.

Machine learning methods, such as neural networks, utilize data to learn optimal transformations that minimize prediction errors. The prediction error is calculated by a loss function, typically mean squared error in the case of regression. During training, the model adjusts the transformations using a method called backpropagation and an optimization algorithm like gradient descent. This process iteratively refines the model to improve its accuracy.

In a previous semester project, the effectiveness of a neural network for providing distance estimates from the phase angles of channel sounding measurements was demonstrated [10].

3 | Simulation

Recent advancements in simulation technology have made it possible to create highly accurate channel simulations. These simulations make the generation of training datasets possible that would otherwise be very expensive and time-consuming to produce in real life. By leveraging these simulation datasets, it becomes feasible to create generalized datasets that encompass a wide range of environments and configurations, thus enhancing the robustness and applicability of the models developed.

A comprehensive system has been developed to generate datasets, train and benchmark different machine learning models, and compare these models to other ranging methods.

Sionna ray tracer generates channel impulse responses for various environments, then these channel responses are used for generating Bluetooth channel sounding scans to form simulation datasets. These simulation datasets are used to conduct machine learning experiments, and evaluate their performance.

3.1 Sionna ray tracer

Sionna is an open-source software library for physical layer research developed by NVIDIA. It enables the prototyping of complex communication system architectures with a native support for machine learning since its implementation is based on python using TensorFlow.

Sionna implements a number of features such as Forward error correction (FEC), channel models, MIMO processing, Orthogonal frequency-division multiplexing (OFDM), and ray tracing. In this work, the ray tracing module is used to create channel impulse responses which is later used to generate the Bluetooth channel sounding scan specific to an environment[11].

The ray tracing module's first component is a scene, which contains objects and its geometry and material. This can be a three dimensional environment exported from Blender with the Mitsuba Blender plugin installed, and with the help of the Blender-OSM plugin, the geometries of buildings and real cities can also be imported from Open Street Map.

The software includes several built-in example scenes, such as the area around the Frauenkirche in Munich (Figure 3.1), a metallic box, and a simple reflector plane. Additionally, these scenes can feature transmitters and receivers, each with their own antenna arrays. The antenna arrays are customizable, with parameters including the number of rows and columns of the antenna elements, their spacing, position, orientation, radiation pattern, and polarization.

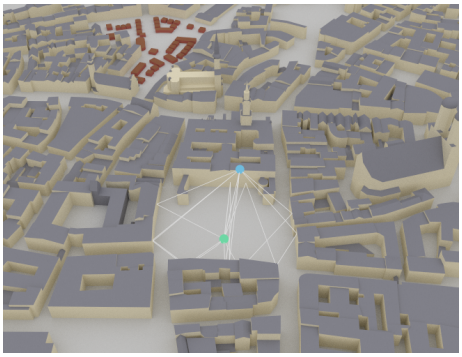

Figure 3.1: Visualized signal paths between a transmitter and a receiver in the Munich example scene built-in to Sionna.

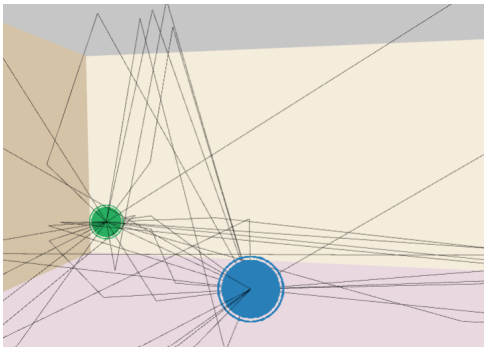

Figure 3.2: Visualized signal paths between a transmitter and a receiver in a box with concrete walls, simulating an indoor environment. The rays are computed up to two bounces of depth.

After the transmitters and receivers are set up in the scene, the paths between them can be computed (Figure 3.2). In order to optimize the trade-off between accuracy and computational complexity, parameters such as the maximum number of bounces a ray can make between the antennas, or whether an exhaustive ray tracing method is used, or a Fibonacci approach. After the paths are determined, the radio materials and their properties are taken into account to compute the channel coefficient a_i and the delay τ_i for each of the i paths (Illustrated in Figure 3.3).

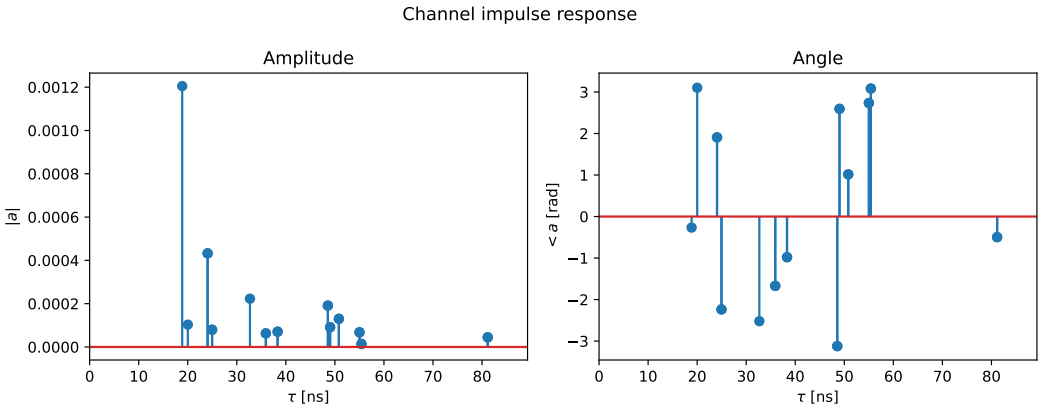

Figure 3.3: Simulated channel impulse response in a concrete room environment, showing the multi-path components with their respective delays and attenuations.

3.1.1 Generated datasets

Different datasets are generated in three different environments using this simulation method. These environments include an open field, a concrete box, and a street

canyon area. Each environment is chosen to represent a variety of real-world conditions that might affect Bluetooth signal propagation differently.

For each environment, a training and testing dataset are created separately. To generate the training datasets, the antenna positions are randomly generated within a specified coordinate range to ensure diversity in the training samples. The generated coordinates are checked to ensure they are not closer than 10 centimeters to avoid overlapping scenarios that might skew the data. This random placement helps simulate a wide range of possible real-world conditions and interactions.

For the testing dataset, the setup is slightly different to mimic real-world measurement scenarios more closely. One of the transmitters is fixed at a specific position, while the other transmitter is placed at predefined distances, such as 1-meter increments. This configuration aims to replicate the typical process of collecting measurement data in the field, where one device remains stationary, and the other is moved manually to different positions.

For all the generated datasets, the τ delays and the corresponding a channel coefficients are saved in a file. These parameters capture the essential characteristics of the signal propagation in the simulated environments. In the subsequent Bluetooth scan generation step, these τ delays and a coefficients are used to produce simulated Bluetooth channel sounding scans. These scans are created based on the specific channel parameters of each environment, ensuring that the generated data accurately reflects the conditions under which the measurements were taken. This process enables the creation of realistic and diverse datasets for training and evaluating the performance of various machine learning models and other ranging methods.

Open Field Dataset

The Open Field dataset is designed to simulate an open outdoor environment with minimal obstacles. This dataset is useful for understanding signal propagation in a clear line-of-sight scenario. A 30x30 meter flat ground plane is created in Blender, with the material set to itu_medium_dry_ground for Sionna to interpret its radio characteristics from the material name. The scene is then exported to a Mitsuba XML format and imported into Sionna, where transmitter and receiver antennas are placed.

Open Field Dataset Properties

Radio Material:		Medium dry ground
Size	X	-30 : 30m
	Y	-30 : 30m
	Z	0m
Realizations:		3000
Test dataset		1 - 30m (.5m)

Table 3.1: Open Field Dataset Properties

Random coordinate range	
X	-30 : 30m
Y	-30 : 30m
Z	1 : 3m

Table 3.2: Range of random antenna coordinates used in the Open field training dataset.

Concrete Box Dataset

The Concrete Box dataset simulates an indoor environment with significant multipath reflections due to the concrete walls. This setting is useful for studying signal behavior in enclosed spaces where reflections and obstructions are prevalent. A 10x10x5 meter concrete box is created in Blender, and the material is set to itu_concrete to accurately represent the radio characteristics. The scene is exported to Mitsuba XML format and imported into Sionna, where the antenna configurations are set up.

Concrete Box Dataset Properties

Radio Material:		itu_concrete
Size	X	-5 : 5m
	Y	-5 : 5m
	Z	0 : 5m
Realizations:		10,000
Test dataset		2.5 - 6.5m (.2m)

Table 3.3: Concrete Box Dataset Properties

Random coordinate range	
X	-4.9 : 4.9m
Y	-4.9 : 4.9m
Z	0.1 : 4.9m

Table 3.4: Range of random antenna coordinates used in the Concrete box training dataset.

Street Canyon Dataset

The Street Canyon dataset represents an urban environment with multiple reflective surfaces and obstructions typical of a city street. This dataset helps in understanding signal propagation in an outdoor urban settings. This environment model is offered by Sionna as an example scene, featuring a ground plane and multiple rectangles with different materials, resembling buildings.

Street Canyon Dataset Properties

Radio Material:	Multiple
Realizations:	3,000
Test dataset	5 - 45m (2m)

Table 3.5: Street Canyon Dataset Properties

Random coordinate range	
X	-17 : 31m
Y	-36 : 36m
Z	0.1 : 3m

Table 3.6: Range of random antenna coordinates used in the Street canyon training dataset.

3.2 Bluetooth scan generator

This component in the software loads the file saved by the Sionna ray tracer to generate Bluetooth channel sounding scans from the a and tau values describing a scene in a specific radio environment.

The received signal is a superposition of all individual signal paths. To cancel out the individual clock offsets, a two-way measurement is performed. Consequently, the channel sounding measurement is simulated as follows:

$$s_{TW}^{k} = \sum_{i=0}^{N} a_i \cdot e^{-j4\pi c f^{k} \tau_i} \tag{3.1}$$

This is computed across all 79 frequency channels that Bluetooth operates on from 2402 to 2480 MHz. The resulting generated scan is illustrated in Figure 3.4.

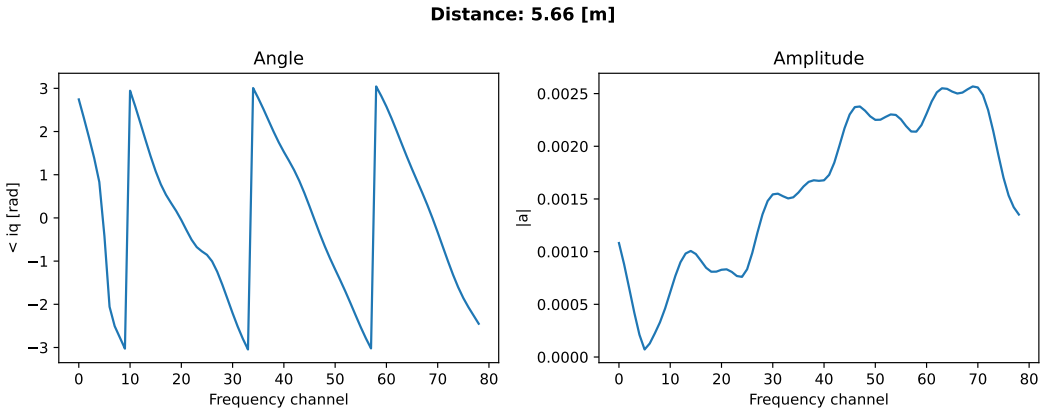

Figure 3.4: The phase angles and amplitudes of a simulated channel sounding scan, generated from the values in Figure 3.3

While processing real measurement datasets with the linear regression method, a constant bias of 8 meters was observed. This can be explained by a delay in the devices radio frontend causing a delay of $8/c \approx 27ns$. To align the generated dataset with the measurements, this hardware delay is added to the τ values.

In every position, a random phase offset is added in order to encourage the model to generalize and not overfit to the specific values. This step also functions as a data augmentation technique, as every impulse response can be used to generate multiple simulated scans.

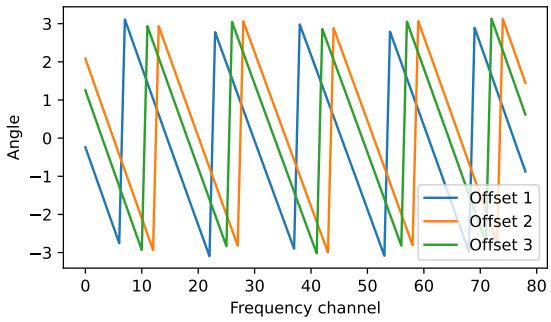

Figure 3.5: Random phase offsets of simulated channel sounding measurements generated from the same channel response.

This process is applied to all channel response datasets generated by Sionna across all environments, including both the training and testing sets. The resulting generated channel sounding scans, along with their corresponding distance labels, are saved in a file to be used in subsequent steps for training and evaluation.

3.3 Representation experiment for machine learning

In this section the generated datasets are loaded from the files, and used for performing different machine learning experiments.

Different representations of the same data can be more suitable for different tasks. The simulated measurements used in this project consist of 79 complex numbers. These complex numbers can be represented in various ways. One representation involves extracting the angle and magnitude of the complex number, another uses the imaginary and real parts, and a third discards the magnitudes to calculate an angle-only representation.

Conventional ranging methods, such as linear regression, only consider phase angles, while decomposition-based ranging methods like IFFT and MUSIC can utilize the complex magnitudes to better distinguish between different signal arrivals. Alternative representations might include an unwrapped angle or an IFFT-transformed signal. These representations can be used as inputs for a neural network.

In this experiment, four representations are compared on the simulation dataset to assess how well they help the model fit the training data and generalize to the testing dataset. The four tested representations are angle-abs, real-imag, angle-only, and IFFT-transformed.

The input data for the different representations are normalized to ensure most values fall within a standard range. For amplitudes, which can only be positive, values are scaled to fall between 0 and 1. For angles and the real-imaginary representation,

where negative numbers can occur, values are scaled to fall between -1 and 1. This normalization is achieved by scaling all values by a predetermined constant.

For all the representations in this experiment, the same neural network architecture is used, implemented using scikit-learn in python. The input is one representation flattened to a vector, while the output is the distance prediction. The model's parameters can be seen at Table 3.7.

Parameter	value
Hidden layer sizes	(200,100,50,20,10)
Activation	ReLU
Learning rate	adaptive
Tolerance	1e-4
N iter no change	1000

Table 3.7: Parameters of the neural network architecture used in the representation experiment.

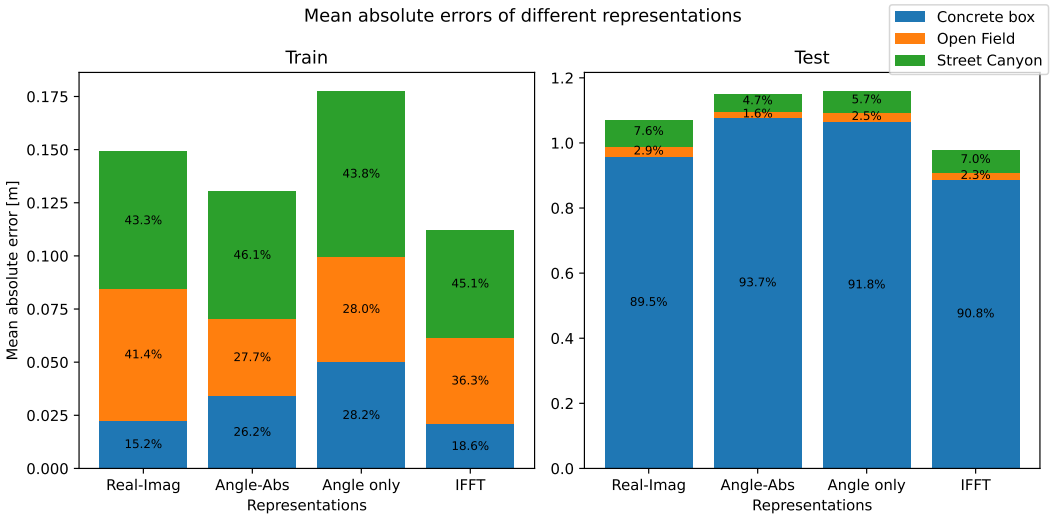

Figure 3.6: This figure compares the mean absolute errors (MAE) in meters achieved by neural networks trained on different representations. The left chart shows the MAE on the training dataset, while the right chart shows the MAE on the testing dataset. Each column represents a specific representation (Real-Imag, Angle-Abs, Angle only, IFFT), and the colors within the column represent the contribution of different datasets (Concrete Box, Open Field, Street Canyon) to the overall MAE for that representation.

After training, all the representations converged to fit the training dataset to a 14cm error on average, with the IFFT representation reaching the lowest error of 11cm, followed by the angle-abs representation at 13cm, the real-imag representation with 15cm, and the angle-only representation with 17.7cm.

For the testing dataset, the average ranging error across the different representations

is around 1 meter. The IFFT representation again performs the best with the lowest overall error of 0.97 meters, followed by the Real-Imag representation at 1.06 meters, the Angle-Abs representation at 1.15 meters, and the Angle-Only representation at 1.16 meters. Analysis of the error contributions reveals that the majority of the ranging error on the testing dataset comes from the Concrete Box dataset.

From these observations its evident that the IFFT representation achieves the lowers ranging error on both the training and testing datasets. Performing an inverse Fourier transform reduces the ranging problem to picking the peak of the first arrival on the spectrum.

3.4 Comparison to conventional ranging methods

In this section, the performance of the trained neural network is compared to Linear regression and IFFT ranging. The chosen representation for the neural network in this analisys is the real-imag representation, since it performed well with simple preprocessing. Each of these three methods are tested on all of the three simulation testing datasets. The predictions of these different methods can be seen in Figure 3.7.

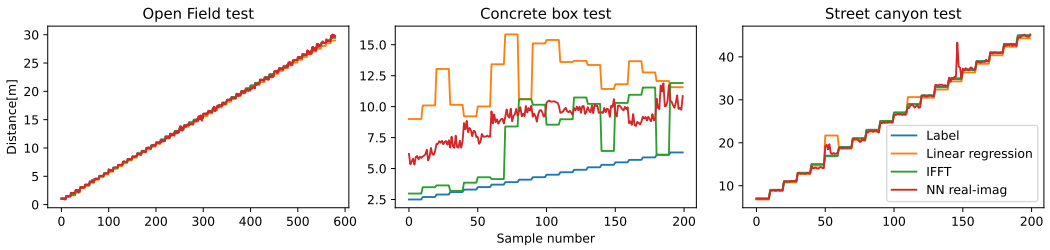

Figure 3.7: Comparison of ranging performance across different methods: Linear Regression (LR), Inverse Fast Fourier Transform (IFFT), and a Neural Network (NN) with real-imag representation. The graphs display the performance on three distinct simulation environments: Open Field, Concrete Box, and Street Canyon. Each graph shows the predicted distance versus the sample number

In the Open Field test, the distance predictions align closely with the actual distances, indicating high accuracy for all methods in a simple environment with minimal obstacles.

The Concrete Box test introduces more complexity due to reflections from the walls. Here, the IFFT method outperforms the others by initially successfully distinguishing the line-of-sight component from the other reflections. The neural network is observed to have a larger bias, while performing better than linear regression.

In the street canyon environment, IFFT ranging performs reliably, with linear regression, and neural network having some deviation in specific distances.

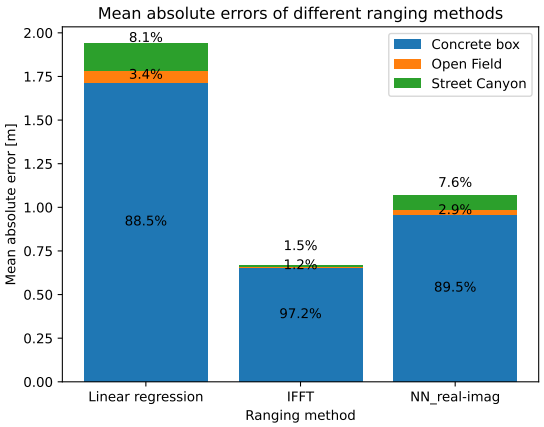

Figure 3.8: Mean absolute errors of different ranging methods across various environments. The bar graph shows the contribution of each environment (Concrete Box, Open Field, Street Canyon) to the overall error for each method.

Figure 3.8 provides a comparison of the mean absolute errors (MAE) for the different methods across the three environments. The linear regression method shows the highest overall error of 1.93 m, followed by the neural network with 1.06 m, and the IFFT ranging achieving the lowest error of 0.67 m. Most of the error originated from the concrete box dataset, being the most challenging environment with tightly spaced reflections.

4 | Transfering to real measurements

Simulated data provides a controlled environment for initial testing and training, but to prove real-world applicability, its crucial to transfer these models to real measurements.

In this chapter, the measurements taken are introduced, compared to the simulations, followed by different machine learning experiments.

4.1 Measurement datasets

Measurements were conducted using two developer phones, with one phone serving as the initiator and the other as the reflector. During the channel sounding process, a test operator placed one device at various distances while keeping the other fixed.

Each channel sounding subevent measured the entire spectrum, adhering to Bluetooth specifications. Therefore, the channels reserved for advertising were excluded from the measurements. 2402, 2403MHz from the beginning of the spectrum, 2425, 2426, 2427MHz in the first part of the spectrum, causing a gap in the measurements, and the last channel 2479 MHz.

These frequency gaps necessitate additional considerations when implementing various ranging algorithms. For instance, MCPD relies on phase unwrapping to determine the slope of the phase-frequency diagram. The presence of frequency gaps can introduce errors in this unwrapping process. Similarly, decomposition methods such as IFFT and MUSIC assume uniform frequency sampling, which is not met due to these gaps. This non-uniform sampling can affect the accuracy and reliability of these algorithms. On the other hand, when neural networks are used for ranging, it does not suffer a significant loss in ranging accuracy[12].

Measurements were performed in three distinct environments, each with different number of reflecting surfaces, posing a different level of difficulty for ranging. Each measurement is performed with the devices in line of sight.

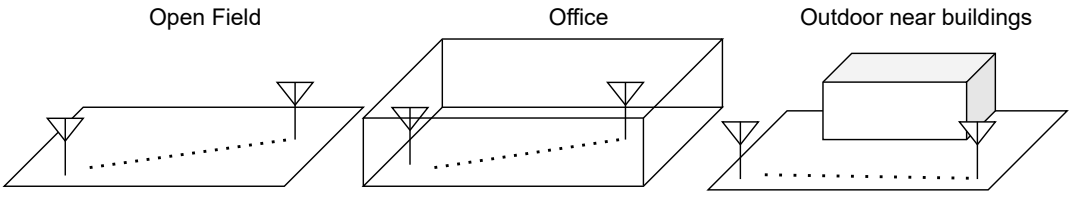

Figure 4.1: Measurement environments used for creating channel sounding datasets.

4.1.1 Open Field

This dataset was collected outdoors in a field with no nearby buildings, so the main reflecting surface was the ground below the devices, causing minimal interference. Under these conditions, ranging performance is expected to be close to ideal for all methods.

Measurements were taken at distances ranging from 1 meter to 30 meters, in 1 meter increments, totaling 30 different distances. At each distance, measurements were repeated between 100 and 129 times.

4.1.2 Office

This dataset was collected indoors in an office with reflecting surfaces such as walls and furniture. These conditions pose a significant challenge to ranging algorithms due to multiple reflections.

Measurements were taken at distances ranging from 0.2 meters to 6.2 meters, in 0.2 meter increments, totaling 31 different distances. At each distance, measurements were repeated between 3400 and 11000 times.

During the measurement, the test operator when labelling the data had the option of clicking 'moving' on the screen to label the measurements as moving, and when the device has been placed at a new position, the appropriate distance label was selected manually. This allowed to save time by performing measurements continuously when making measurements.

4.1.3 Outdoor near buildings

This dataset was collected outdoors near buildings, where both the ground and nearby building surfaces influenced the measurements, though from larger distances compared to the indoor scenario. This dataset serves as an intermediate difficulty level between the ideal open field environment and the challenging indoor office scenario.

Measurements were taken at distances ranging from 0.2 meters to 6 meters, in 0.2 meter increments, with additional measurements at 8 and 10 meters, totaling 32 different distances. At each distance, measurements were repeated between 1400 and 3800 times.

These measurements were also taken continuously, with the operator labeling the data as 'moving' when necessary.

4.2 Comparing the generated dataset to the measurements

The generated channel sounding dataset and the measurement from an Open field environment are compared by visual inspection.

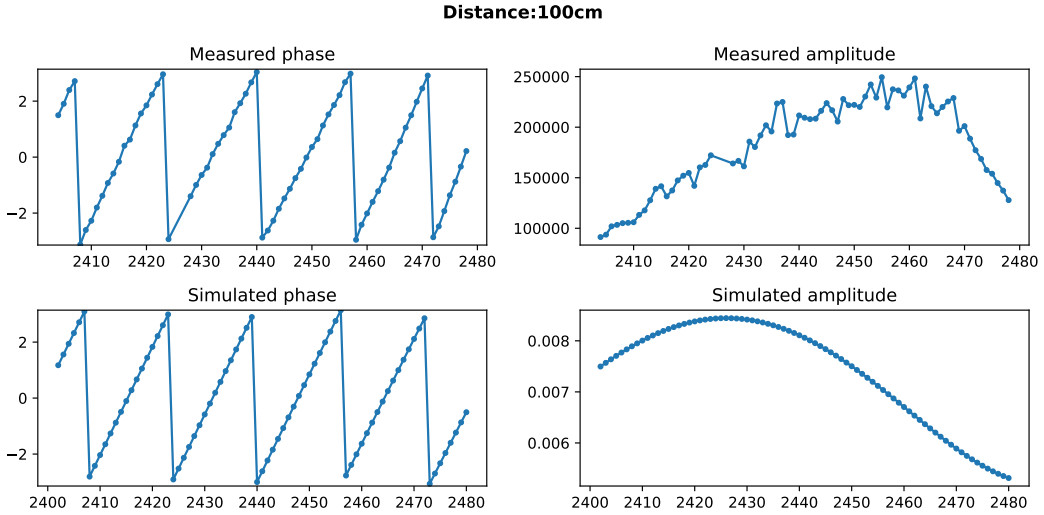

Figure 4.2: Comparison of the generated dataset to the measurements for a distance of 100 cm. The top row shows the measured phase and amplitude, while the bottom row shows the simulated phase and amplitude. The phase diagrams align well with the measurements, but the amplitude values differ in both range and general shape.

On Figure 4.2, the phase angles and the amplitudes are plotted for the resulting scan at a distance of 100 centimeters. The phase diagram aligns well to the measurements across all distances of the dataset. The amplitude values however differ both in the range of values, and also in the general shape. The amplitude in this context is a dimensionless quantity, proportional to the received power, which decreases as the distance is increased between the two devices. This amplitude decrease is observable in the simulation dataset, however on the measurement dataset the amplitude is observed to fluctuate unrelated to the distance between the two devices as illustrated on Figure 4.3.

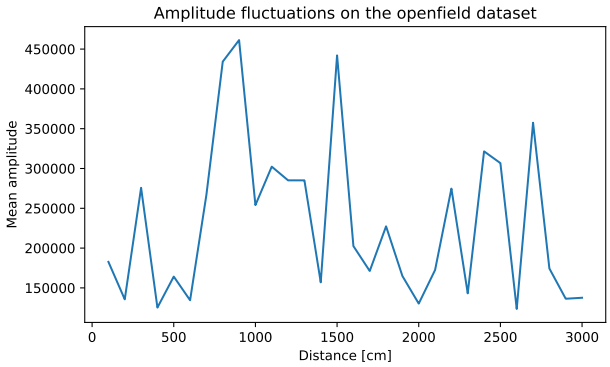

Figure 4.3: Amplitude fluctuations in the open field dataset. This figure shows the mean amplitude values at different distances.

These fluctuations are due to the device's Automatic Gain Controller (AGC) adjusting the gains to ensure correct reception. However, the measurements in the dataset are not normalized to cancel out this effect.

4.3 Training on measurement datasets

In this experiment, a neural network model with the same architecture as used in the representation tests was trained on real measurement datasets. The goal was to evaluate the model's performance and its ability to generalize across different environments.

The three datasets (Open Field, Office, and Outdoor Near Buildings) allow for a combination of two datasets for training while reserving one for testing. This setup enables the creation of a training dataset with a mixture of easier and more challenging examples, which is crucial for robust model training.

Given that the Open Field dataset is considered the least challenging and covers the widest range of distances, it was included in the training dataset. The Office dataset, which includes the most challenging examples for ranging among the three datasets, was also included in the training experiment. This leaves the Outdoor dataset set aside for testing purposes, serving as a medium-difficulty dataset to evaluate the model's generalization capabilities.

Preprocessing

The complex measurements with corresponding distance labels were loaded from a file. Each channel sounding scan was preprocessed by converting the measurements to phase angles, then normalizing these angles by dividing by π, resulting in an input range of -1 to 1 for the model. The distance labels in the measurement dataset,

originally in centimeters, were divided by 3000 to normalize the distances between 0 and 1, considering the largest measured distance was 30 meters.

Pretraining

Since the amplitudes are not well correlated with the distance, an angle-only representation approach is used in this experiment. The model was first pretrained on the combined dataset of the Office and Open Field measurements. This pretraining resulted in a mean absolute error (MAE) of 3.39 centimeters, indicating a good fit on the training dataset. This pretrained model was then tested on the reserved Outdoor dataset, resulting in a mean absolute error of 174.74 centimeters.

The prediction plot indicates that the model was able to generalize to an unseen dataset to some extent. However, higher errors were observed at both shorter and longer distances within the dataset.

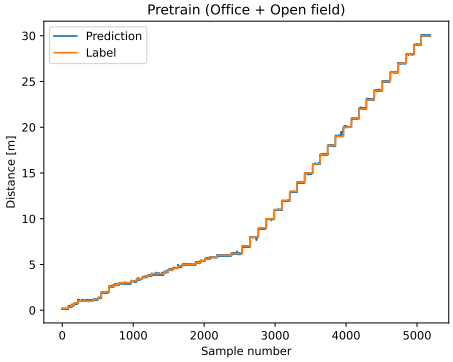

Figure 4.4: Pretraining on Office and Open Field datasets.

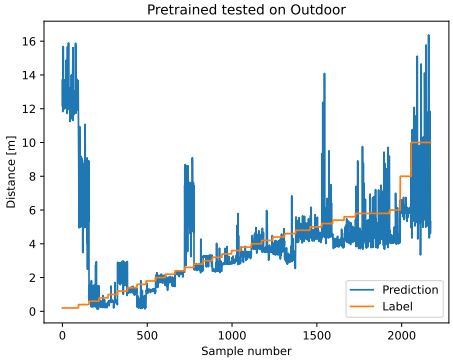

Figure 4.5: Pretrained model tested on Outdoor dataset.

Finetuning

To further improve the model's performance, finetuning was performed by taking every other distance from the Outdoor dataset and appending these new training samples to the previous combination of Open Field and Office datasets. This served as a new finetuning training dataset.

After finetuning, the model was tested again on the still unseen distances from the Outdoor dataset. The results showed a reduction in the overall prediction error from 174.74 centimeters to 87.23 centimeters.

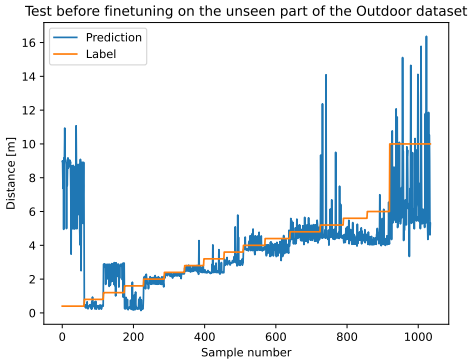

Figure 4.6: Test before finetuning on the unseen part of the Outdoor dataset.

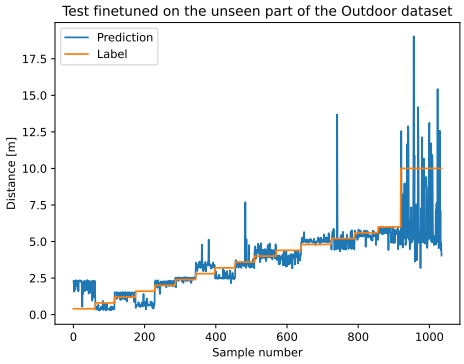

Figure 4.7: Test finetuned on the unseen part of the Outdoor dataset.

The finetuning on the dataset, expanded with new distances from the Outdoor dataset, effectively reduced the overall prediction error on the unseen distances from 143.2 centimeters to 87.23 centimeters.

This improvement demonstrates the model's ability to generalize better with additional training data from varied environments. These results suggest that incorporating diverse training samples can enhance the robustness and accuracy of the ranging models in real-world applications.

4.4 Transferring from simulation

Training on real datasets demonstrated generalizability between datasets. Since the measurements are taken in a static environment, repeating the measurement in the same condition resuls in consistent measurements with little variance. Therefore it is more valuable to make measurements in smaller distance increments, or alternatively, measuring in a changing environment with moving reflecting surfaces while keeping the same distance.

Collecting these measurements is a time-consuming process. However, leveraging simulation data for pretraining a model can significantly reduce this effort. Depending on how closely the simulations match the real measurements, the model can be pretrained on simulated data, then finetuned on real data, or even trained entirely on simulated data.

When comparing the open field simulation datasets to real measurements, it was observed that while the phase angles closely matched, the amplitudes in the measurement dataset were inconsistent due to the influence of each device's gain controller. This lack of normalization makes the amplitude data less useful for distance estimation and model training.

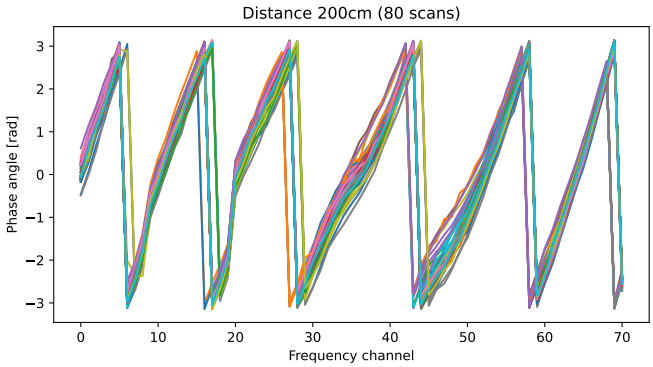

Figure 4.8: Phase diagram of 80 measurements in the office dataset with devices placed 200 cm apart. This figure demonstrates that repeated measurements in a static environment yield consistent results.

4.4.1 Transfer angle-only representation

This experiment tests a model's performance trained on the simulation datasets in the angle-only representation.

Sine the phase angles were possible to simulate to a suitable accuracy, training a model on angle-only representation can result in a good transfer from simulation to measurement data.

The simulation training datasets are combined to one dataset for training to provide the model examples from different scenarios.

Preprocessing

The simulated complex measurements with corresponding distance labels were loaded from a file. Each channel sounding scan was preprocessed by converting the measurements to phase angles, then normalizing these angles by dividing by π, resulting in an input range of -1 to 1 for the model. The distance labels in the measurement dataset, stored in meters, were divided by 30 to normalize the distances in line with the measurement datasets.

Additionally, the simulations generated measurements for all Bluetooth channels. However, due to Bluetooth specifications, some channels cannot be used for channel sounding. These unusable channels were discarded from the simulation dataset before training.

Training

This data is preprocessed as detailed above. The model is the same architecture as in the representation experiment.

Since this training was performed on a simulation dataset, all the real measurement datasets are unseen to the model, therefore the model's performance can be assessed in all measurement datasets.

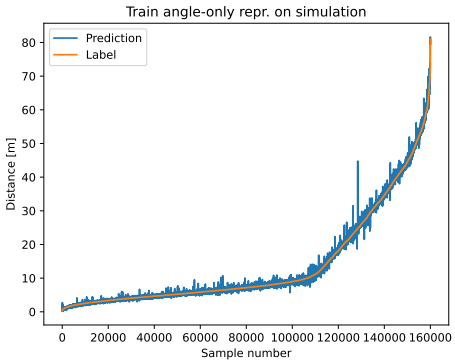

Figure 4.9: The trained angle-only model's fit on the its training dataset.

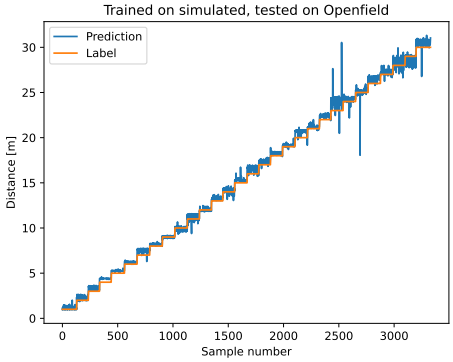

Figure 4.10: Simulation trained angle-only representation model tested on openfield measurements.

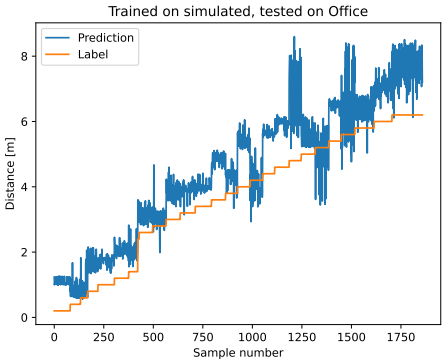

Figure 4.11: Simulation trained angle-only representation model tested on office measurements.

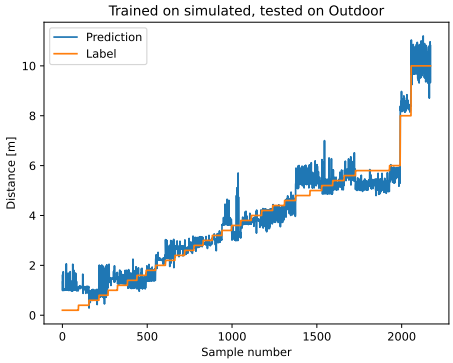

Figure 4.12: Simulation trained angle-only representation model tested on outdoor measurements.

This training run can be directly compared to the model trained purely on real measurements since both use the angle-only representation and the outdoor dataset remains unseen in both cases. Training on limited real measurements resulted in a mean absolute error of 174.74 cm on the outdoor dataset. In contrast, training on simulation data resulted in a significantly lower MAE of 37.08 cm on the same dataset. This comparison demonstrates the effectiveness of simulation training in improving the model's generalization to unseen real measurements.

Angle-only transfer	
Dataset	MAE (cm)
Train set	20.27
Openfield	36.82
Office	90.52
Outdoor	37.08

Table 4.1: Mean Absolute Error (MAE) in centimeters for the angle-only model on different datasets.

Finetuning on measurements

A model initially trained on simulation data can be further refined using real measurements to better align with real-world data. Similar to Section 4.3, a combination of the Open Field and Office measurement datasets can be used as a training dataset, while the Outdoor dataset is reserved for testing.

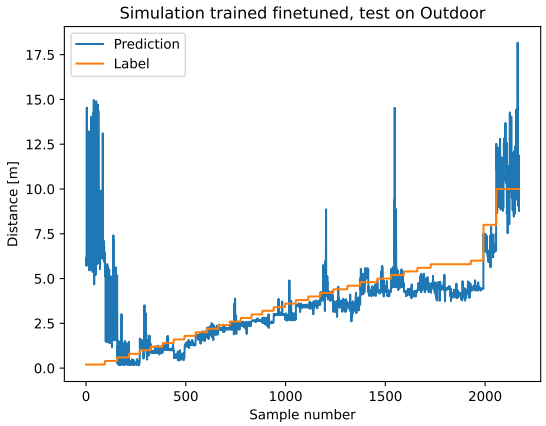

Figure 4.13: Simulation-trained angle-only model further finetuned on the combined Office and Open Field dataset, tested on the Outdoor dataset.

After only a few iterations of finetuning, the model achieves a very low mean absolute error (MAE) of 4 cm on the combined training data. However, when tested on the Outdoor dataset the predictions start to deviate quickly, resulting in an error exceeding 1 meter similar to the results observed in the measurement-only training case.

This outcome suggests that while finetuning on a limited set of real measurements can initially improve performance, it is not sufficient for achieving robust generalization across different environments. A more extensive and diverse measurement dataset would be required to train a model exclusively on real measurements or to use for further finetuning.

It is also important to note that the angle-only representation proved to be the least effective in the representation experiment on simulation data. Therefore, it is worth exploring other representations to improve performance. Experimenting with different representations, such as angle-abs or real-imag, could potentially lead to better generalization and accuracy in the predictions.

4.4.2 Transfer normalized real-imag representation

The representation experiments demonstrated on simulation data that including amplitude information improves performance. The real-imag representation showed promising performance, with the lowest error after the compute intensive IFFT representation.

However, the amplitude measurements in the real dataset being not normalized, is not a good feature for distance estimation. This experiment tests if transfering the promisingly performing real-imag representation model to real measurements, by normalizing the minimum and maximum magnitudes in each scan to fall between -1 and 1. This removes the information about the absolute signal magnitude, but preserves the shape of the signal. In frequency selective deep fading scenarios it could still retain information about lower amplitudes relative to the minimum and maximum which are normalized to -1 and 1.

Preprocessing

The simulated complex measurements with corresponding distance labels were loaded from a file. Each channel sounding scan was preprocessed by taking their real and imaginary components, as visualized in Figure 4.15. The real and imaginary components are appended to construct a one dimensional vector, then the resulting vector is normalized, so all of its values fall between -1 and 1. This served as the input for the model as depicted in Figure 4.17. The distance labels in the measurement dataset, stored in meters, were divided by 30 to normalize the distances in line with the measurement datasets.

Additionally, the simulations generated measurements for all Bluetooth channels. However, due to Bluetooth specifications, some channels cannot be used for channel sounding. These unusable channels were discarded from the simulation dataset before training.

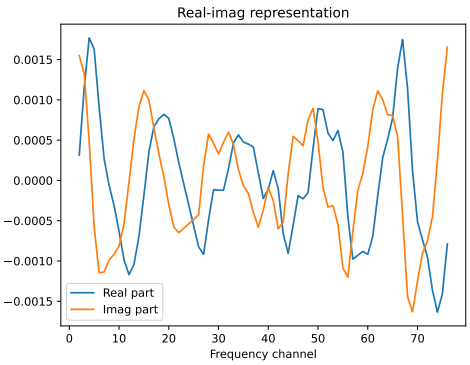

Figure 4.14: Real-imag representation of the channel sounding scan. The graph shows the real and imaginary parts of the measurements across the 79 Bluetooth frequency channels.

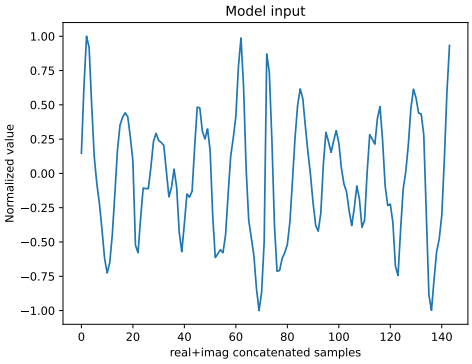

Figure 4.15: Model input after preprocessing. The graph shows the normalized values of the concatenated real and imaginary components from the channel sounding scan, scaled between -1 and 1.

Training

The model is the same architecture as in the representation experiment, with the normalized real-imag representation described above. The training resulted in a better fit on the simulation training dataset as compared to the same architecture trained on the angle-only representation (Table 3.7).

However, when loading and combining all measurement datasets, and applying the same preprocessing, the model does not provide valid predictions as depicted in Figure 4.17.

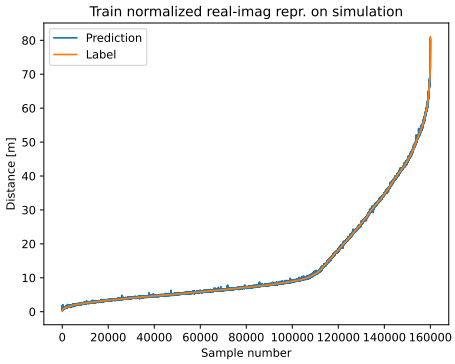

Figure 4.16: Training predictions of the model with normalized real-imag representation on the simulation dataset.

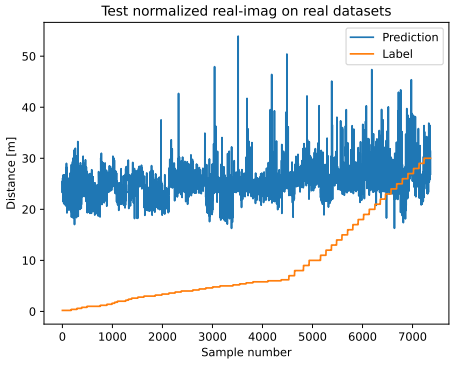

Figure 4.17: Testing predictions of the model with normalized real-imag representation on the real measurement datasets, highlighting the model's failure to generalize to the real datasets

This experiment demonstrated a failure to make the model generalize to a real measurement dataset in a real-imag representation, even when the absolute amplitude values are cancelled out by the normalization technique.

4.4.3 Combined model

This experiment is meant to leverage a good performing simulation pretrained angle model, and combine its predictions with an amplitude model trained on measurements for finetuning.

To facilitate the construction of this architecture, this implementation is based on PyTorch. This also allowed GPU accelerated training, and larger batch sizes, as well as on-line logging.

Preprocessing

Each channel sounding scan was preprocessed by converting the simulated measurements to phase angles, then normalized these angled by dividing by π, resulting in an input range of -1 to 1 for the model. The distance labels in the measurement dataset, stored in meters, were divided by 30 to normalize the distances in line with the measurement datasets.

Additionally, the simulations generated measurements for all Bluetooth channels. However, due to Bluetooth specifications, some channels cannot be used for channel sounding. These unusable channels were discarded from the simulation dataset before training.

Pretraining

The model is trained on the combination of all simulation training datasets to provide the model examples from different scenarios. This data is preprocessed as detailed above.

Parameter	Value
Hidden layer sizes	(200,100,50,20,10)
Activation	ReLU
Learning rate	0.001
Optimizer	Adam
Loss function	MSE

Figure 4.18: Parameters of the PyTorch implementation of the angle model

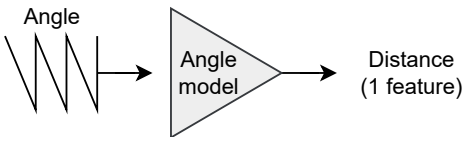

Figure 4.19: Depiction of the angle model

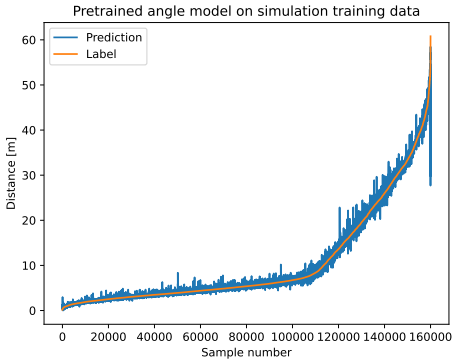

Figure 4.20: The angle model's fit on the simulation training data.

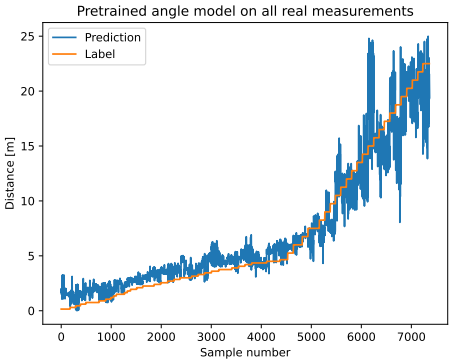

Figure 4.21: The angle model's fit on real measurements.

The model achieved a mean absolute error of 25 centimeters on the simulation training dataset. Note that Scikit-learn implementation achieved a 20.27 cm error on the same dataset, indicating a better fit than the Pytorch implementation. This is possibly due to differences like a fixed learning rate, different batch sizes.

On the combined real measurement dataset, The model achieves a 161.2 cm error. Observable higher deviance is observable on the higher distances, which belong to the openfield dataset.

Finetuning

The simulation pretrained model is reused. it provides a distance estimate. The weights of the angle model were not frozen when its put into the combined model.

An amplitude model is constructed to extract useful information for ranging, which is compressed down to a representation of 10 features. This is done since the amplitude values themselves are not sufficient to provide a distance estimate, and intermediate 10 feature representation is kept instead.

This 10 features, and the 1 feature being the distance estimate of the angle model is appended to a 11 feature vector, which serves as an input to a combiner network to produce a refined distance estimate.

The model could be expanded with additional features, which can be useful to further refine the distance estimate.

The finetuning is performed on real measurement data, to use the real measurement's amplitude information to refine the pretrained angle model's measurement. Since these measurements are performed in a static evironment, The dataset was separated by distance. Half of the available distances are used for the finetuning training, while the other half of the distances are used for evaluating.

Parameter	Value
Amplitude model layer sizes	(200,100,50,20,10)
Combiner layer sizes	(50,20,10)
Activation	ReLU
Learning rate	0.001
Optimizer	Adam
Loss function	MSE

Table 4.2: Parameters of the combined model

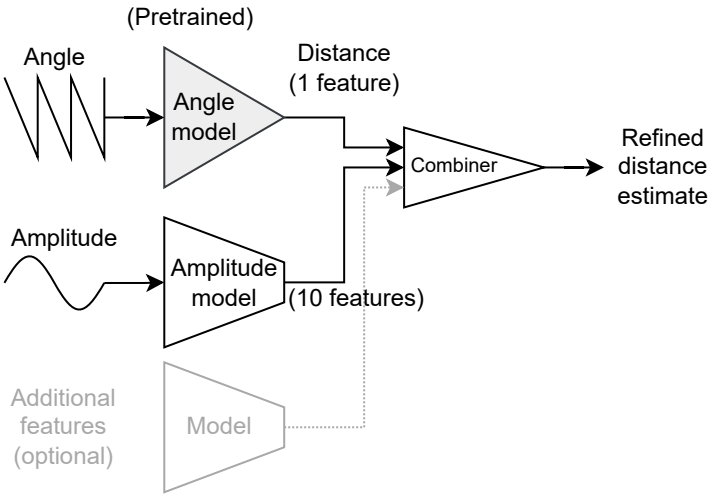

Figure 4.22: Architecture of the combined model.

The finetuning resulted in a close fit of 6 cm MAE on the training distances as illustrated on Figure 4.23, but a 4.9 meter mean absolute error on the validation distances as illustrated on Figure 4.26

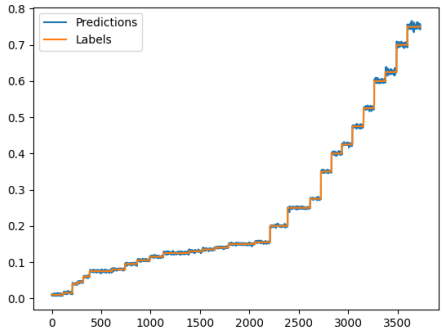

Figure 4.23: The combined model's fit on the training distances.

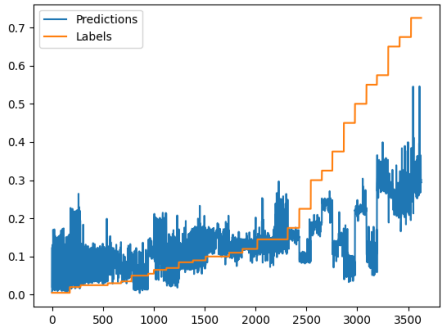

Figure 4.24: The combined model's fit on the validation distances.

This experiment resulted in the inability of the model to generalize better than the angle-only model. This can be explained by the limited number of training data available, and the amplitude's unpredictability.

4.5 Comparison to MUSIC ranging

By using a simulation-trained model, it can be fairly evaluated and compared against other ranging methods without contaminating the results with the model having seen the samples during training.

The best performing model is the angle-only model implemented with Scikit-learn. This model is compared against predictions from the MUSIC algorithm applied to the current datasets.

Predictions are made for all measurements, including the scans with 'moving' labels. This uncertainty in the labels makes it difficult to evaluate ranging errors accurately. Additionally, there are NaN values in the Office dataset where predictions cannot be made for certain scans. As a result, visual inspection of the predictions is used as the primary method of evaluation.

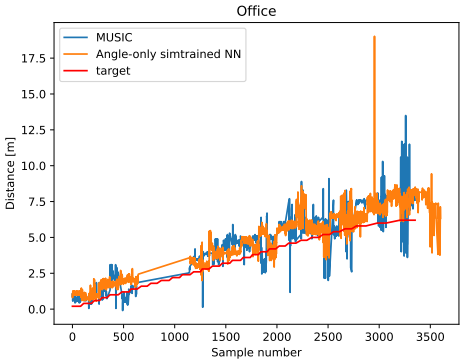

Figure 4.25: Comparison of MUSIC and angle-only simulation-trained neural network on the Office dataset.

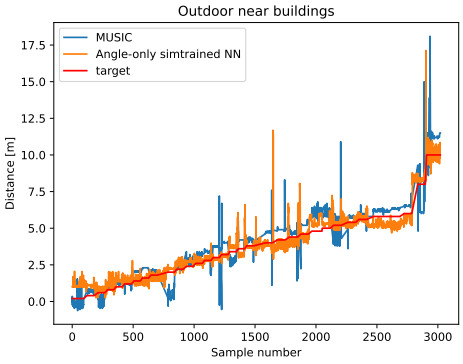

Figure 4.26: Comparison of MUSIC and angle-only simulation-trained neural network on the Outdoor dataset.

In Figures 4.22 and 4.23, the predictions of the angle-only simulation-trained neural network are compared to those of the MUSIC algorithm for the Office and Outdoor Near Buildings datasets, respectively. The target values represent the ground truth distances. Both methods show a general trend of increasing as the distance increases, but differences in prediction accuracy and consistency can be observed. The MUSIC algorithm tends to have more fluctuation in its predictions, particularly in the Office environment, which contains more reflecting surfaces and multipath effects.

5 | Discussion

In this work, machine learning techniques were explored to improve range estimation from Bluetooth channel sounding measurements. using the amplitudes, and explore machine learning, and data generation options to improve ranging performance

Datasets were generated with the help of Sionna ray-tracer in three different environments. An open field environment, a concrete box environment imitating an indoor scenario, and a street canyon environment. For each environment, a training and a testing set was created separately. The training set is generated with random antenna positions, while the test sets were created at discrete distances imitating a real measurement. These channel response simulations were used to simulate Bluetooth channel sounding scans.

This simulation dataset was used for a machine learning experiment, where different representations of the complex numbers of the simulated scan. The performance of real-imaginary, angle-abs, angle-only, and IFFT representations using the neural network models with the same architecture were trained and tested. Each representation's fit on the training set, and performance on the testing set was compared. This resulted in similar fit for all representations of 14 centimeters on average, and one meter on the testing set in terms of mean absolute error. Most of the error originating from the challenging concrete box dataset. Overall, IFFT representation showed the lowest ranging errors, followed by the real-imag, the angle-abs and lastly the angle-only representation. The IFFT representation was not studied further in favor of other representations with less demanding preprocessing.

The performance of the real-imag representation was compared to Linear regression and IFFT on the simulation testing datasets.

Real measurements were carried out in three different environments. an open field, inside an office, and outdoors near buildings. The amplitudes of the measurements was observed to fluctuate as the automatic gain controller adjusted the gains. This made the amplitudes not a good feature for ranging purposes. This left the phase angles as a reliable feature for ranging.

Training an angle-only representation model on real measurements from the mixed openfield and the office dataset, was able to generalize to the third, outdoors dataset to a mean absolute error of 174.74 cm. Further finetuning with half of the available distances of the outdoor dataset was tried, and was able to reduce errors from 143.2 cm to 87.23 cm on the unseen testing distances.

The next experiment attempted to perform training the angle-only representation model on the simulation data, and test it on real datasets. This resulted in mean absolute errors of 36.82 cm on openfield, 90.52cm on office, and 37.08cm on outdoor.

Further finetuning of this model on the combined dataset of openfield and office resulted in the same performance as purely measurement training.

Since the angle-only model was the worst performing in the representation experiment, it was tested to train a real-imag representation model with the magnitudes of the values normalized between -1 and 1 within a full scan. This was meant to cancel out the effects of the gain controller. The model pretrained on simulation data was not able to transfer to the real dataset using this representation.

To leverage on a trained angle model, it was tried to combine its prediction with an amplitude model to extract important information from the amplitude data and refine the distance estimate. This combined model was attempted to train on half of all the measurement distances, and validated on the unseen half, but it was unable to fit to the the unseen distances.

Lastly, the best performing angle-only model was tested against the MUSIC algorithm, resulting in a similar performance.

6 | Conclusion

The study demonstrates that machine learning is applicable in Bluetooth channel sounding ranging.

Demonstrated using channel simulations for generating training data which could be too time consuming and costly to perform in real life.

A representation experiment revealed that most majority of the useful information for ranging lies in the phase angles, however improved performance was observed on the angle-abs and real-imag representation. The IFFT representation achieved the best performance, since it transforms the ranging problem to selecting the position of the signal's first arrival's peak.

The real measurements proved insufficient to train a model on to be able to generalize with good accuracy.

The measurements were performed in a static environment, therefore repeated measurements of the same position resulted in very similar measurements. Data collections for training purposes therefore benefits the most from measuring in smaller distance increments, diverse environments, or moving reflecting surfaces.

The ability to train a model, and transfer to real measurements validates that the accuracy of the simulation dataset.

The current iteration of the dataset measurement method does not normalize the amplitudes according to the gain controller, which negatively impacts performance of machine learning methods, and conventional methods alike. Implementing amplitude normalization is expected to improve ranging accuracy, as it was also shown on the representation experiment.

Comparison with the MUSIC algorithm's and the best performing simulation trained angle-only model's prediction revealed similar ranging errors. The angle-only model generally follows the trend of the target line more closely, with fewer deviations from the target than the MUSIC model. Both methods are expected to improve with more reliable amplitude data.

Bibliography

[1] Silicon Labs. *Silicon Labs Announces New Bluetooth Location Services with Advanced Hardware and Software*. https://news.silabs.com/2022-06-21-Silicon-Labs-Announces-New-Bluetooth-R-Location-Services-with-Advanced-Hardware-and-Software. Accessed: 2024-05-29. 2022.

[2] Bluetooth SIG. *Using Bluetooth Technology as a High Accuracy Positioning Solution*. Accessed: 2024-05-30. 2023. URL: https://www.bluetooth.com/blog/using-bluetooth-technology-as-a-high-accuracy-positioning-solution/.

[3] Bluetooth SIG. *Location Services*. Accessed: 2024-05-30. 2024. URL: https://www.bluetooth.com/learn-about-bluetooth/solutions/location-services/.

[4] Bluetooth SIG. *Bluetooth Channel Sounding: A Step Towards 10 cm Ranging Accuracy for Secure Access, Digital Key, and Proximity Services*. Accessed: 2024-06-01. 2024. URL: https://www.bluetooth.com/blog/bluetooth-channel-sounding-a-step-towards-10-cm-ranging-accuracy-for-secure-access-digital-key-and-proximity-services/.

[5] Silicon Labs. *UG103.18: Bluetooth Direction Finding Fundamentals*. Accessed: 2024-06-01. 2022. URL: https://www.silabs.com/documents/public/user-guides/ug103-18-bluetooth-direction-finding-fundamentals.pdf?source=Partner&detail=Bluetooth%20SIG&cid=prt-sig-blu-021022.

[6] Bluetooth Special Interest Group. *Bluetooth Direction Finding: Technical Overview*. Accessed: 2024-06-01. 2021. URL: https://www.bluetooth.com/wp-content/uploads/Files/developer/RDF_Technical_Overview.pdf.

[7] David Tse and Pramod Viswanath. "Fundamentals of Wireless Communication". In: Accessed: 2024-05-29. Cambridge, UK: Cambridge University Press, 2005. Chap. 2. URL: https://web.stanford.edu/~dntse/Chapters_PDF/Fundamentals_Wireless_Communication_chapter2.pdf.

[8] Qasim Chaudhari. *Carrier Phase-Based Ranging in Indoor Multipath Channels*. https://wirelesspi.com/carrier-phase-based-ranging-in-indoor-multipath-channels/. Accessed: 2024-06-01. 2023.

[9] Pepijn Boer et al. "Performance of High-Accuracy Phase-Based Ranging in Multipath Environments". In: *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. 2020, pp. 1–5. DOI: 10.1109/VTC2020-Spring48590.2020.9128721.

[10] Mark Klepacs and Pantelis Stefanakis. *Bluetooth Distance Estimation Using Phase-Based Ranging*. Accessed: 2024-05-29. 2023. URL: https://kbdk-aub.primo.exlibrisgroup.com/permalink/45KBDK_AUB/a7me0f/alma9921650692405762.

[11] Jakob Hoydis et al. *Sionna: An Open-Source Library for Next-Generation Physical Layer Research*. 2023. arXiv: 2203.11854 [cs.IT].

[12] Alireza Sheikh et al. *Phase-based Ranging in Narrowband Systems with Missing/Interfered Tones*. 2023. arXiv: 2211.00502 [cs.IT].