# Human Pose Estimation and Reconstruction using FMCW Radar


**Prepared by:**
Talon Sewnath

**Prepared for:**
Dr. Stephen Paine
EEE4022S
Department of Electrical Engineering
University of Cape Town

October 24, 2024

# Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this thesis report from the work(s) of other people has been attributed and has been cited and referenced. Any section taken from an internet source has been referenced to that source.
3. This report is my own work and is in my own words (except where I have attributed it to others).
4. I have not paid a third party to complete my work on my behalf. My use of artificial intelligence software has been limited to grammar checking and minor rephrasing on sentences written by myself.
5. I have not allowed and will not allow anyone to copy my work with the intention of passing it off as his or her own work.
6. I acknowledge that copying someone else's assignment or essay, or part of it, is wrong, and declare that this is my own work.

October 24, 2024

Talon Sewnath

Date

Word Count = 15661

# Acknowledgements

Firstly, I would like to thank my supervisor, Dr Stephen Paine, for offering his expertise and reassurance throughout the project. Your constructive feedback and support were invaluable in navigating the challenges of this project.

I would also like to extend my gratitude to the RRSG group of post graduate students for welcoming me to their lab. I want to specifically thank Mark Nyago, Kauthar Du Toit and Sebastiag Haug for their personal input and guidance.

A special thanks to Siya and Sayuri for taking the time to read and provide feedback on my report. Your recommendations are greatly appreciated.

I extend my heartfelt thanks to my mother for her endless love, sacrifices, and unwavering support in raising me. Your presence has been the foundation of my achievements. To my granny, thank you for helping raise me and for sending food during my time at university. Additionally, I am deeply thankful to my aunt Vera and uncle Sangeeth for providing eye-opening guidance and advice throughout my university experience.

I am also thankful to my friends, Joachim, Josh, Revashan, Sayuri, Sayitha, Maisha and Siya for the unforgettable memories throughout the last four years. Being united made the challenging times more manageable and the joyful moments even more enjoyable.

Lastly, I would like to thank my classmates and lecturers for enriching my university experience. The collective memories have made my time here truly memorable. Thank you all for your support and contributions to my academic journey.

# Abstract

Human Pose Estimation (HPE) finds several applications in patient monitoring, traffic control systems, autonomous vehicles and gaming to track human skeletons. Although the field has gained an increased amount of attention, it is still an under-studied topic. This study explores the viability of performing HPE for skeletal reconstruction on 13 key points using Frequency Modulated Continuous Wave (FMCW) Radio Detection and Ranging (RADAR). The use of radar makes the system robust to environmental conditions. Furthermore, radar-base poses estimation is advantageous over Computer Vision (CV) methods as it does not capture facial information thereby abiding by ethical codes. A comprehensive dataset comprising 750 samples of 2-second radar data was collected to for radar signal processing and model training. The key point targets are first detected from RDMs and then abstracted to 3D space. A novel superposition method is used to reduce sparsity which is a persisting issue in RADAR processing. A Convolution Neural Network (CNN) architecture was used for key point localization of five unique poses. The proposed method was tested in a single human scene for five poses, (i) Star Pose, (ii) Arms Raised, (iii) Left Arm Extension, (iv) Right Arm Extension and (v) Freeze Pose. The proposed method achieved a Mean Absolute Error (MAE) of 11.4cm and a Mean Per Joint Position Error (MPJPE) of 21.1cm. Future work will explore dynamic pose reconstructions and real-time localization. The detailed methodology, implementation, challenges, and validation results are presented and analysed in detail.

# Contents

# List of Figures

# Abbreviations

**ADAM** Adaptive Moment Optimizer

**AoA** Angle of Arrival

**CA-CFAR** Cell Averaging Constant False Alarm Rate

**CFAR** Constant False Alarm Rate

**CNN** Convolution Neural Network

**CSAM** Cross- and Self-Attention Module

**CUT** Cell Under Test

**CV** Computer Vision

**DTM** Doppler Time Map

**FFT** Fast Fourier Transform

**FMCW** Frequency Modulated Continuous Wave

**FN** Feature Network

**FPGA** Field Programmable Gate Array

**GCN** Graphical Convolution Network

**GRU** Gated Recurrent Unit

**HDF5** Hierarchical Data Format V5

**HPE** Human Pose Estimation

**HRNet** High-Resolution Network

**IQ** In-phase Quadrature

**LiDAR** Light Detection and Ranging

**LSTM** Long Short-Term Memory

**LVDS** Low Voltage Differential Signalling

**MAE** Mean Absolute Error

**MIMO** Multiple Input Multiple Output

**ML** Machine Learning

**MLE** Mean Localization Error

**mmWave** Millimeter Wave

**MPJPE** Mean Per Joint Position Error

**MTI** Moving Target Indicator

**MUSIC** Multiple Signal Classification

**MVDR** Minimum Variance Distortionless Response

**NMS** Non-Maximum Suppression

**OS-CFAR** Ordered Statistics Constant False Alarm Rate

**PEN** Pose Estimation Network

**PFA** Probability of False Alarm

**PJE** Per Joint Error

**RADAR** Radio Detection and Ranging

**RAZ** Range Azimuth Map

**RCS** Radar Cross Section

**RDM** Range Doppler Map

**REV** Range Elevation Map

**RF** Radio Frequency

**RGB** Red Green Blue

**RMSProp** Root Mean Square Propagation

**RNN** Recurrent Neural Network

**RPN** Regional Proposal Network

**RRSG** Radar Remote Sensing Group

**RTM** Range Time Map

**SGD** Stochastic Gradient Descent

**SMPL** Skinned Multi-Person Linear

**SNR** Signal to Noise Ratio

**ULA** Uniform Linear Array

# Chapter 1

# Introduction

## 1.1 Background

Human Pose Estimation (HPE) refers to predicting the locations of skeletal points on a human body from sensor data, typically using computer vision or radar signal processing. The importance of this technique has grown in various domains such as robotics, healthcare, and human-computer interaction, where accurate body pose information can enhance automation, diagnosis, and user experience respectively [1]. Substantial progress has been made in 2D pose estimation, however the field of 3D pose estimation remains an active research area as a result of the lack of datasets and processing systems. Traditionally, vision-based sensors such as RGB and stereoscopic cameras have

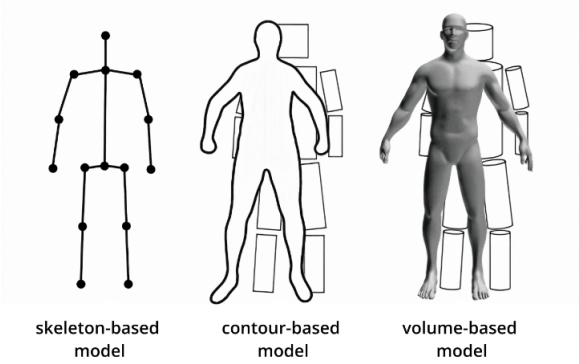



Figure 1.1: Common pose estimation models

been the primary tool for HPE however they are ineffective in adverse weather conditions, poorly lit environments or occluded scenes. In recent years, the advancements in Radio Frequency (RF) devices have made Frequency Modulated Continuous Wave (FMCW) Millimeter Wave (mmWave) more accessible as an alternative due to the reduced cost and size of the hardware. In response to vision-based limitations, mmWave offers robust performance in challenging environments.

## 1.2 Objectives

The primary aim of this project is to develop a processing pipeline to reconstruct human poses from raw RADAR data. This process can be segmented into the following key stages:

- **Data Acquisition:** Capturing raw data and establishing a dataset.
- **Signal Processing:** Creating a processing script to convert the data into a format used to train the Machine Learning (ML) model.

- **Validation:** Validating the signal processing choices through experiments.
- **Machine Learning:** Implementing an ML model to reconstruct human poses based on extracted features.

## 1.3 Scope

The scope of the project focuses on the development, implementation and validation of a pose estimation and reconstruction pipeline. The foundation of the project involves understanding FMCW radar fundamentals and applying these principles to the pipeline. Additionally, the process requires competent hardware operation and the ability to configure the system to meet the needs of the application. Lastly, optimization of computational resources and reconstruction performance is required by refining the pipeline stages.

## 1.4 Limitations

Several limitations impacted the implementation and execution of the project. Firstly, the project was limited to a 13-week timeline which restricted the amount of refinement and optimization. Secondly, a full ethics clearance was required which further delayed data capturing. The strict ethical conduct restricts the use of participants which hampers the robustness of the ML model. Although the primary radar system is available in the laboratory, a budget of ZAR2000 restricts the use of secondary hardware. Data capture was conducted in an environment with high noise and clutter potential. Additionally, the large data set resulted in the need for significant storage capacity which limited the rounds of data capturing.

## 1.5 Report Outline

The report outline is structured such that the report provides a sequential flow of the projects development. Following the Introductory Section 1, a comprehensive study of the HPE literature is presented in Section 2. The System Design section 4 then outlines the proposed processing chain whilst presenting the experiment results used to motivate the design choices. Thereafter the complete pipeline's localization performance is presented and reviewed in Section 5. Penultimately, the HPE pipeline and network results are summarized in the conclusion. Lastly, Section 7 provides a discussion on the areas for improvement and potential for future work.

# Chapter 2

# Literature Review

The study of HPE has experienced a pivotal shift, with RF based technologies facilitating new pose estimation applications. This literature review explores the use of RADAR compared to existing methods by critically evaluating its performance, advantages and limitations. By comprehensively examining current research, the review outlines the key signal processing techniques and reconstruction procedures required for HPE and thereby establishes a framework for developing radar-based pose estimation systems.

## 2.1 History of Pose Estimation and Reconstruction

HPE has undergone significant growth through the advancement of sensing-based technologies such as Computer Vision (CV), Light Detection and Ranging (LiDAR), infrared and RADAR. In 2005, Oxford pioneered pose detection by developing a CV system, 'Strike a Pose', that identified 10 body parts using RGB images [2]. Initial approaches to HPE relied heavily on CV however they were limited to 2D reconstruction [1]. A decade later, *DeepPose* by Toshev introduced a groundbreaking system that implemented a Convolution Neural Network (CNN) for key point tracking [3]. Compared to prior approaches, *DeepPose* substantially improved key point tracking whilst removing the need for handcrafted features and part-based models. This enhanced the model's ability to discern subtle pose variations. Subsequently, *OpenPose* built on Toshev's model by utilizing a multistage CNN which enabled dynamic detection of multi-person skeletal poses [4].

Although the above-mentioned developments refined spatial localization, the reliance on visual data in CV methods made them vulnerable to varying environmental factors and raised ethical concerns related to privacy. Consequently, alternative technologies were explored. LiDAR was adopted for HPE which offered precise spatial data essential for reliable pose estimation. However, its high costs and computational demands rendered LiDAR impractical for widespread use, especially in indoor environments [5]. *WiPose* was the first instance of Wi-Fi-based pose estimation which used wearable sensors to detect poses [6]. Although *WiPose* utilized easily deployable wearable sensors compared to RF hardware complexity, RF methods resulted in better skeletal spatial accuracy and robustness to complex environments. Furthermore, Wi-Fi signals cannot distinguish between different body parts and are therefore not suited for pose estimation [1].

Unlike previous non-visual systems, RF-based systems removed the need for wearable sensors whilst being able to track multi-person skeletons [7]. In 2015, *RF-Capture* developed the first RF based pipeline which identified 22 skeletal points by stitching together several x-y heatmaps [8]. Although it was ground-breaking, *RF-Capture* suffered from low spatial resolution and poor key-point localization. This limitation was addressed by RF-Pose 3D which was able to improve spatial resolution whilst

adapting to multi-person scenes and enabling live monitoring [7].

In 2021, *mmMesh* emerged as the first pose estimation system that utilized a mesh intermediate data structure. The mesh system aimed to address the issues that point cloud networks suffered from sparsity whilst improving localization accuracy[9]. Unlike the previous RADAR-based methods, *mmMesh* was able to perform a high-level shape reconstruction.

*MARS* pose estimation focused on patient monitoring systems which prioritized ethical restrictions and non-invasiveness [10]. Their system effectively performed through-wall pose estimation and accurately tracked multi-person scenes [10]. Prior systems required multiple synchronized sensors and computationally powerful machines to process the raw data into an intermediate format. Sengupta et al. (2020) implemented a point cloud-based model that reduced the computational demand whilst accurately reconstructing 25 skeletal points [11]. This innovation made RF-based systems more feasible for real-time applications by optimizing data processing and enhancing accuracy.

## 2.2 Related Works

### 2.2.1 Signal Processing Techniques

RF signals are rich in information about people and their movements [7] which helps leverage HPE whilst offering advantages such as privacy preservation and robustness to occlusions. However, due to the complexity of HPE, RF-based systems require advanced signal processing algorithms. The primary objective of radar signal processing for HPE is to extract a target's range, Doppler (velocity) and angular information. The widely adopted approach involves performing a fast time FFT to extract the range, a slow time FFT to extract speed and lastly MIMO processing to identify the angles [1, 11, 7, 9, 12, 13, 10, 5, 14].

Sengupta et al. (2020) [1] initially implemented a signal processing pipeline using three FFTs to obtain range, velocity, and angle data. This approach relied on time-division multiplexing for signal separation however it lacked mechanisms for clutter removal and noise rejection. The presence of clutter and noise resulted in the persistence of static and dynamic interference which reduced localization accuracy. Sengupta et al. (2022) [11] refined their work by incorporating Moving Target Indicator (MTI) for clutter removal and Constant False Alarm Rate (CFAR), after the two FFTs, for SNR improvement. *MARS* utilized the same approach however their system facilitated multi-person tracking and through-wall pose estimation. Similarly, *mmMesh* introduced a signal processing approach that utilized a range and Doppler FFT, followed by phase interferometry for azimuth and elevation estimation [9]. This method aimed to resolve the localization errors caused by the poor elevation resolution. While phase interferometry enhanced point cloud quality, it still struggles in environments with significant motion or clutter, further highlighting the challenges of hardware limitations.

Li et al. (2020) [13] altered the conventional approach by omitting the slow time FFT to reduce computational load. Their processing pipeline used two FFTs to generate two range angle maps as the training data input. Sizhe et al. (2022) [5] also deviated from the traditional approach. They indicated that a single radar frame struggles to capture the entire pose therefore they superimposed the FFT results from three frames together to create a complete training data sample [5]. Yuan-Hao Ho et al.

(2024) [15] introduced a hybrid method involving range and Doppler FFTs, however, they remodulated the radar signal data before performing a channel FFT [15]. The aim was to compensate for the poor elevation resolution and estimate precise angles of multiple targets. The *SUPER* system utilized the traditional method however they implemented DC compensation for clutter removal [14]. They also implemented Minimum Variance Distortionless Response (MVDR) beamforming to generate intensity spectra for each point in the target set [14]. MVDR improved localization accuracy as it was able to reject the noise interference. While these approaches improve signal quality, they add computational demand, which may not be feasible for applications requiring live processing.

Unlike the traditional FFT based approach, Yongkun Song et al. (2022) [16] applied a 2D backpropagation algorithm directly to radar data, followed by MTI and a subsequent 3D backpropagation with 3D CFAR. This multi-stage processing pipeline enhanced noise filtering and target detection accuracy but introduced significant processing power demands [16]. Zhao et al. (2018) [7] further diverted from the traditional approach by feeding pre-processed RF data directly into a neural network. This approach bypassed noise removal and coordinate extraction, relying only on the neural network's inherent ability to learn from the data. As a result, their system required extensive training and posed a challenge in non-ideal environments.

### 2.2.2 Feature Extraction and Training Data Abstraction

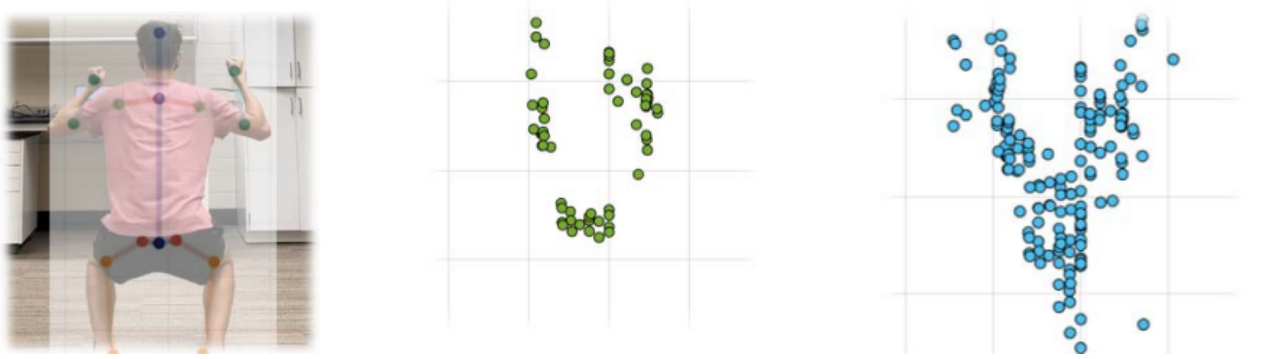
Pose reconstruction systems rely on sophisticated feature extraction techniques to transform processed RADAR data into structured spatial information. A critical evaluation of the literature revealed that different approaches were formulated to address specific limitations such as data sparsity, localization error and computational load. This section explores the processing methods and the pipeline output utilized by the relevant HPE studies.

A substantial part of assembling the training dataset involves target recognition from the processed RADAR data. Applying Cell Averaging Constant False Alarm Rate (CA-CFAR) on 2D RADAR maps was the predominant method for extracting the target positions. Jalil et al. [17] argued that for high range resolution applications such as HPE, Ordered Statistics Constant False Alarm Rate (OS-CFAR) performs better. This is due its ability to detect overlapping targets however it results in the expense of increased computational complexity.

Following target recognition, several studies such as [1, 10, 14, 5], employed a coordinate transform to convert the extracted spherical coordinates to the desired Cartesian system. Sengupta et al. (2020) grouped their coordinate set with its corresponding intensity values which were used to generate two feature maps by projecting their RGB scaled coordinates onto the XY and XZ planes [1]. By flattening the coordinates, the pipeline loses its purpose of mapping spatial relationships and instead focuses on reconstruction based of subtle map differences. Sizhe et al. (2021) and Zhang et al. (2024) both built on the coordinate transformation by incorporating the velocity information to create a 5-D coordinate set [14, 10]. The inclusion of intensity and Doppler information provides more information to the ML model for learning. *SUPER* split their dataset into a Doppler point cloud and an intensity point cloud which reduced errors across similar poses [10]. *MARS* utilized their full 5-coordinate set resulting in a successful reconstruction of dynamic poses. The addition of Doppler information enables a smooth transition between frames in dynamic pose recontruction. Compared to [1], both [10, 14] required

more complex neural architecture due to the larger sample size. Sengupta et al. (2022) [11] revised their pipeline to remove the projections and instead implemented a clustering method called DBSCAN to improve the point cloud assembly. Their improved point cloud removed the need for image-based training data thereby improved the dynamic response of the system.

*mmMesh* addressed the point cloud sparsity by integrating a Skinned Multi-Person Linear (SMPL) model which parametrizes the human mesh using a low-dimensional shape vector [9]. While the SMPL model enhances mesh reconstruction, it relies on predefined anatomical parameters which limit the system's generalization to varying body shapes. Sizhe et al. (2022) [5] focused on improving the point cloud density by fusing points from consecutive frames, creating a complete representation. Although fusing the point clouds provides a complete dataset, signal processing over multiple frames creates a significant latency in the processing. Figure 2.1 below shows the point cloud's spatial improvement by fusing three frames together.



(a) Squat pose captured by a RGB camera

(b) Extracted point cloud from a single frame

(c) Point cloud extracted from three frames fused together

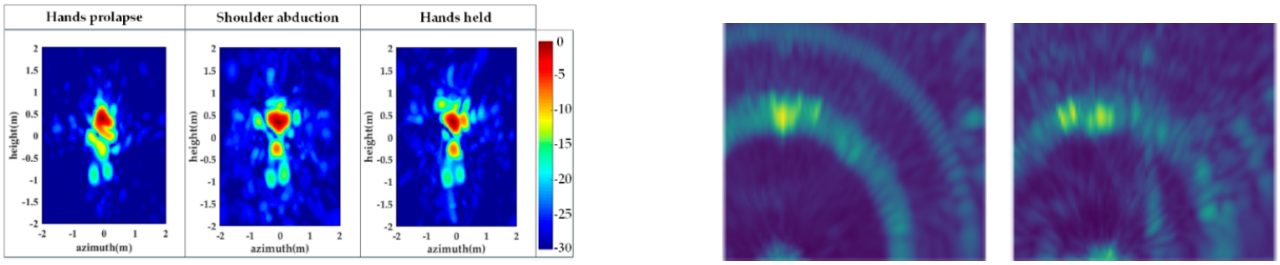
Figure 2.1: Graphical improvement resulting from the point cloud fusing process [5]

In contrast to point cloud-based methods, Song et al. (2022) [16] and Zhao et al. (2018) [7] explored intensity map-based representations to reconstruct 3D poses. To optimize the training data, Zhao et al. (2018) [7] implemented a Regional Proposal Network (RPN) which identifies the target areas in the maps. Guangzheng et al. (2020) also utilized a map-based approach however they assembled a training dataset of a range-elevation map and a range-azimuth map [13]. Although this results in a bigger input size, the full map input provides contextual information for reconstruction. As mentioned previously, this method requires extensive training and struggles to generalize in different environments. Figure 2.2 illustrates the different types of map-based training sets.

### 2.2.3 Ground Truth Generation & Labelling

Reliable ground truth data is essential for developing HPE systems, however manually labelling datasets has proven impractical. Several sources have made their datasets open to the public, like *OpenPose* and *mmPose*, however the lack of raw data limits its usability [14].

*MARS* utilized Microsoft Kinect V2 to assemble their own ground truth dataset with the reliance on advanced computer vision tools for accurate data labelling [10]. *SUPER* incorporated *MARS* dataset as




(a) Radar heatmap generated by Song et al. (2021) through backpropagration [16]

(b) Vertical and horizontal range angle maps by Lee et al. (2023) using FFTs

Figure 2.2: Visualization of a map-based training data approach

well as their own data captured using a 12 Opti-Track camera system [14]. However, they encountered difficulties due to the lack of raw RF datasets This constrained the system's ability to generalize across diverse environments.

Sengupta et al. (2020) [1] utilized *OpenPose*'s dataset in conjunction with a Vicon Peak motion capture system, infrared depth sensor and Microsoft Kinect sensor. By connecting to the MATLAB API, this multi-sensor approach provided depth, azimuth, and elevation information for 25 joints. A common approach, used by [9, 5, 7], involved synchronizing an RGB and depth camera for ground truth collection and developing a neural network for dataset labelling.

Guangzheng et al. (2020) [13] adopted a CV model however they incorporated a pre-trained COCO model to produce labels. Although this method streamlined the labelling process, the ground truth quality was dependent on the quality of the pre-trained model. Similarly, *RT Pose* used the ZeDO pre-trained model on their RGB and LiDAR ground truth data. VideoPose3D is an alternative pre-trained model designed for the classification of dynamic poses [12].

Emerging technologies like YOLO v8 offer autonomous and scalable labelling by providing a comprehensive classification API [18]. YOLO v8 provides application-specific pre-trained models as well as a simple interface for developing a custom model. This mitigates the need to develop a CV neural network whilst providing independence from the quality of pre-trained models.

### 2.2.4 Machine Learning Models

As discussed in subsection 2.2.2, RADAR signal processing can extract spatial information from raw data, however it is unable to reconstruct key point poses. With the increase of computing power, ML techniques have emerged as a popular tool for determining spatial relationships between point clouds and ground truth[11]. This section will outline the purpose of the different ML models utilized in the literature. Table 2.1 below summarizes this section.

*RF-Pose* introduced a CNN-based architecture, ResNet, that decomposed high-dimensional convolutions into lower-dimensional operations [7]. Their model integrated the target recognition RPN system with the CNN for joint localization across all keypoints. The *RF-Pose* model managed to do pose reconstruction by discretizing the 3D space into voxels and treating keypoint localization as a classification task. Zhao et al. (2018) [7] indicated that by splitting their model into two parts, a Feature Network (FN) and a Pose Estimation Network (PEN), it could reconstruct multi-person

scenes. By allocating 12 layers to the FN for identifying high-level information and 6 layers to the PEN for spatial localization, Zhao et al. (2018) [7] ensured that the model was able to estimate poses in complex environments. *RF-Pose* achieved a MAE of 4.2cm for width localization (X), 4cm in height localization (Y) and 4.9cm for depth localization (Z) across all their poses [7]. However, the dual-network architecture adds significant computational load, making real-time implementation impractical in large-scale applications.

In a complementary approach, Guangzheng Li et al. (2020) [13] proposed a network that leverages 3D convolutions on range-azimuth and range-elevation heatmaps to capture spatiotemporal information. This addressed the challenge that human body reflections may not always occur in a single heatmap. While this addresses the limitation of single-heatmap human reflections, 3D convolutions demand powerful computational power and need extensive training in diverse environments. Their CNN concatenates recognized spatial features through 5 convolution layers, followed by batch normalization and the ReLU activation function. Subsequently, a fractionally strided CNN decodes the features into human poses using 4 deconvolution layers which use parametric ReLU activations and sigmoid activation in the final layer [13]. Lastly, Non-Maximum Suppression (NMS) is used to get the location of possible keypoints and remove duplicate detections.

*RT-Pose* implemented a similar network structure to Guangzheng Li et al. (2020) [13]. However to address the coarse spatial resolution, *RT-Pose* modelled a network based on High-Resolution Network (HRNet) due to its high-resolution representations throughout the network [15]. They further deviated from the structure by implementing a parallel dual convolution branch for feature processing. This system returned the confidence map of the key point locations and the relative key point positions [15]. NMS was used to remove duplication detections in multi-person scenes. Although this network improves spatial localization accuracy, the high-resolution network causes an increase in memory consumption. Ho et al. (2024) were able to achieve an MPJPE of 9.93cm across their whole dataset [15].

Conversely, Sengupta et al. (2020) [1] developed a forked CNN architecture for RADAR reflection data. The model processed an XY and an XZ projection of RGB scaled point cloud data. By incorporating reflection power levels as an additional feature, the CNN was able to differentiate between regions with varying Radar Cross Section (RCS), such as distinguishing the torso (larger RCS) from the wrists (smaller RCS). The Adam optimizer, with a variable learning rate, facilitated effective training via gradient descent due to its response to gradient changes over iterations [1]. While this approach adeptly handles spatial features, its reliance on single-frame inputs may limit its robustness to similar poses. Sengupta et al. (2020) achieved better performance compared to *RF-Pose* for depth and height [1]. They achieved a MAE of 7.5 cm in width (X), 2.7 cm in height (Y) and 3.2 cm in depth (Z) respectively.

*MARS* employs a comprehensive CNN architecture designed to process point clouds into 3D joint positions. Sizhe et al. (2022) [10] used a clever method of reshaping the 64 by 5 point clouds into 8 by 8 by 5 feature maps to take advantage of CNN's inherent ability to classify images. Their model minimized the network structure to 10 layers[10]. To mitigate overfitting, MARS incorporated dropout layers after convolution and fully connected layers. Notably, the model avoids max-pooling to preserve spatial relationships crucial for coordinate regression and implements batch normalization which facilitates faster convergence. *MARS* were not able to improve the MAE benchmark set by

[1, 7]. They achieved a MAE of 6.99cm in width (X), 6.54cm in vertical (Y) and 4.07cm in depth (Z) [10]. Although *MARS* were unable to improve the benchmark, their system set-up consited of a single RADAR module where as Sengupta et al. (2020) used two. The secondary module significantly improved elevation resolution resulting in better point clouds and hence better training data. Lastly, *MARS* used 70 minutes of data, considerably less than the previous studies.

In their second iteration, Sengupta et al. (2022) [11] utilized a different network architecture. To combat their single-frame limitation [1], they explored the use of a Recurrent Neural Network (RNN)s due to their ability to preserve temporal context [11]. Traditional RNNs struggle from the residual gradient becoming exceedingly small therefore they utilized a RNN variant called the Gated Recurrent Unit (GRU). It performs at substages thereby mitigating the vanishing gradient [11]. Despite addressing the vanishing gradient problem, this approach primarily focused on temporal limitations overlooking CNN's ability to extract spatial features, which is crucial for precise localization. Sengupta et al. (2022) [11] improved their MAE across every dimension but specifically addressed their poor width performance. They achieved a MAE of 2.68cm in width (X), 2.12cm in height (Y) and 2.37cm in depth (Z) [11]. Their width MAE significantly improved since their network performed better localization for tokenized 3D grids compared to the image pair input.

*mmMesh* also deviated from traditional CNNs by integrating a PointNet model for feature extraction from RADAR-based point cloud data [9]. Coupled with PointNet, they used the SMPL model to generate a realistic 3D human mesh. However, the reliance on the parametric SMPL model presents challenges in achieving real-time performance and adaptability to diverse body types. Furthermore, it is challenging to accurately construct the human mesh due to the presence of clutter and the error-prone nature of RF signals [9].

*SUPER* developed a hybrid network that combined the temporal information, explored by Sengupta et al. (2022) [11], and the point cloud-focused network, explored by Xue et al. (2021) [9]. They utilized a PointNet model for extracting the shape and structure of the target [14]. Thereafter they implemented an advanced PointNet++ model to identify and localize specific body parts [14]. Lastly, they utilized a RNN variant, Long Short-Term Memory (LSTM), for dynamic pose tracking. However, Sengupta et al. (2022) [1] argued that GRUs are more computationally efficient for HPE. Although the *SUPER* model provides a robust and comprehensive model, the complexity of this architecture, involving multiple network components and sophisticated parameterizations, poses challenges in terms of computational demands and training convergence. *SUPER* achieved a MPJPE 11.2cm for driving tests, 11.5cm for hand raise tests and 10.9cm for head rotation tests [14]. Although *RT-Pose* achieved a better MPJPE, their dataset consisted of simpler poses [15, 14]. Furthermore, *RT-Pose* constructed its own radar module whereas *SUPER* utilized a commercially available module resulting in inherent limitations.

*HuPR* developed a multi-structural architecture using a Graphical Convolution Network (GCN) foundation. The network begins by utilizing a fusion network, MNet, to correlate the horizontal and vertical RADAR maps [12]. Lee et al. (2023) [12] employed multiple 3D strided convolutional layers, further refined by a Cross- and Self-Attention Module (CSAM), to capture the spatial and temporal information. This structure discovers and integrates contextual information within and across radar frames, enhancing the model's learning capability. By incorporating a GCN, the model was able to competently discover relationships between the input maps and the 3D ground truth coordinates. Due

to their extensive network architecture, *HuPR* achieved a MPJPE of 6.82cm across their entire dataset [12].

| Network Architecture Summary | | | | |
|---|---|---|---|---|
| **Paper** | **Network Architecture** | **Dataset Size** | **Metrics** | **Training Data Type** |
| mm-Pose | CNN | 32 000 frames | MAE | XY, XZ Projected Point Cloud Images |
| mm-Pose NLP | RNN | 15 000 frames | MAE | Toxenized Point Cloud Cubes |
| mmMesh | PointNet, SMPL | 24 000 frames | MLE | Mesh Structure |
| RF-Pose | RPN, CNN | Not mentioned | MAE | 4D RF Tensor |
| RT-Pose | CNN, NMS | 72 000 frames | MPJPE | xyz-velocity Point Cloud |
| HuPR | MNet, CSAM, GCN | Not Mentioned | MPJPE | Range-Elevation, Range Azimuth Map |
| SUPER | PointNet, PointNet++, LSTM | 40 000 frames | MPJPE | Intensity, Doppler Point Cloud |
| MARS | CNN | 40 000 frames | MAE | xyz-Doppler-VelocityPoint Cloud |

Table 2.1: Table summarizing HPE models

### 2.2.5 System Configuration

Table 2.2 below summarizes the system configurations used in the relevant HPE studies. This section will highlight the key differences across these systems, identifying distinct approaches and methodologies.

Sengupta et al. (2020) [1] implemented two synchronized AWR1642 radar modules that house their antennas in the same horizontal plane. To estimate elevation angles, they rotated the second module by 90°. Sengupta et al. (2022) [11] diverted to two AWR1843 modules which have one more TX channel than the previous module. Although it is capable of elevation estimation Sengupta et al. [11] used a second module due to the poor elevation resolution of a single module. *HuPR* utilizes the same system configuration as Sengupta et al. (2022) [11] however they make use of the DCA1000 Field Programmable Gate Array (FPGA) for data storage whereas Sengupta et al. (2020, 2022) directly interface the radar module through the TI user interface.

Unlike the previous approaches, *mmMesh* set up a one-module system [9]. They counteracted the poor elevation resolution by developing a prediction SMPL network. Sizhe et al. (2021) captured data using a single IWR1443 module which has 4 TX and 3 RX antennas in the same plane [10]. They explained that the output of their channel FFT returned the quotient of the azimuth and the elevation angles which they used to improve the poor elevation resolution [10]. *RT Pose* implemented a simple one-module system with 12 TX and 16 RX channels. The large number of channels in both horizontal and vertical planes provided excellent spatial resolution [15].

Each approach has different implications, the first being that the two-module system degrades the practicality of the approach since most of the use cases require a compact, cost-effective and computationally efficient system. Secondly, the single-module system's poor elevation resolution requires a

large training set and a robust learning model.

| System Configurations Summary | | | | | |
|---|---|---|---|---|---|
| **Paper** | **Radar Sensor** | **Ground Truth Sensor** | **Frequency Band** | **Bandwidth** | **Reference** |
| mm-Pose | 2 TI AWR1642 | Microsoft Kinect | 77GHz | 4GHz | [1] |
| mm-Pose NLP | 2 TI AWR1843 | Microsoft Kinect | 77GHz | 4GHz | [11] |
| mmMesh | 1 TI AWR1843 | Vicon Motion Capture | 77GHz | 4GHz | [9] |
| RF-Pose | Not Stated | Vicon Motion Capture | 5.4GHz | 1.8GHz | [7] |
| RT-Pose | Self-Constructed | RGB & LiDAR | 77GHz | 4GHz | [15] |
| HuPR | 2 TI AWR1843 | RGB Camera | 77GHz | 4GHz | [12] |
| SUPER | 2 TI IWR6843 | OptiTrack system | 77GHz | 4GHz | [14] |
| MARS | 1 TI IWR1443 | Microsoft Kinect | 77GHz | 4GHz | [10] |

Table 2.2: Table comparing related work on HPE

## 2.3 Importance of Human Pose Estimation

In recent years, RADAR-based pose estimation has received significant attention in applications like autonomous driving, patient monitoring, and surveillance [10]. Unlike optical sensors,mmWave RADARs can operate in low-visibility conditions. Furthermore, RADARs can estimate poses without capturing facial features, thus addressing privacy concerns in healthcare and security [11].

A crucial autonomous vehicle application is pedestrian monitoring, where understanding pedestrian intent is critical for collision avoidance systems [1]. It is also used in healthcare where it removes the need for wearable devices, which may be uncomfortable over long periods [1]. mmWave systems also function in more complex healthcare applications such as monitoring patients with motion disorders, such as Parkinson's disease, where tracking involuntary movements is critical [7].

RADAR-based HPE extends beyond traditional applications such as autonomous driving and healthcare. 3D skeletons are useful in gaming where they can aid motion capture systems to overcome occlusions [7]. This technique has practical implications for law enforcement as well. In hostage situations, law enforcement personnel could leverage the through-wall capability offered by RADAR systems [7].

# Chapter 3

# Theoretical Development

This chapter presents the foundation required to understand the complex theory behind mmWave RADAR pose estimation and reconstruction. The discussion begins by introducing RADAR signal processing techniques. The next section presents the techniques used for coordinate mapping in generating ground truth. Lastly, the machine learning section explores some of the principles behind the algorithms used for classification. This theory section aims to provide an HPE-specific RADAR theory however a more comprehensive explanation can be found using [19, 20].

## 3.1 Fundamental Concepts of Radar

Radio Detection and Ranging (RADAR) involves the transmission of an electromagnetic wave to a target, the reception of the deflected wave at a receiver point, and signal processing to generate useful information [19]. RADAR is utilized in various applications, spanning from object detection and localization to tracking and imaging. RADAR dates back to the early 20th century, predominantly used for World War II and thereafter it evolved into a vital technology, now used in aviation, patient monitoring, and autonomous vehicles [19]. This section aims to provide a brief introduction to RADAR theory however a more comprehensive explanation can be found using [19, 20].

### 3.1.1 Components of a Radar System

A FMCW radar consists of a transmitter, a receiver, an antenna, a synthesizer and a mixer [21]. These components operate in sync to determine the range and velocity of points of interest. The figure below shows a high-level block diagram of the primary components of a radar system.

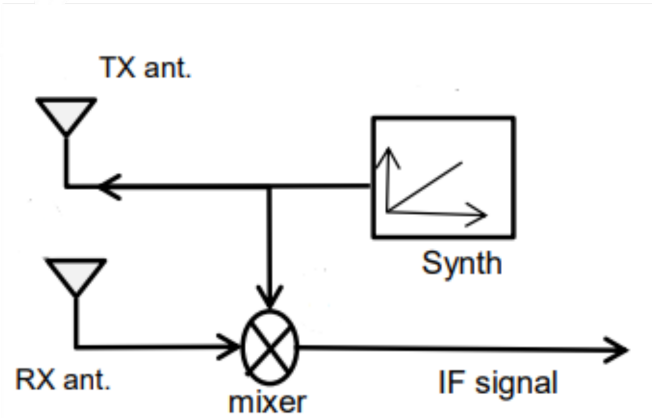



Figure 3.1: Hardware configuration of an FMCW radar system [20]

- **Transmitter**: generates electromagnetic waves to detect and locate objects.
- **Receiver**: captures the rebounded wave.
- **Antenna**: propagates transmitted energy into space and collects echoed energy on the receive [21].
- **Mixer**: combines the TX and RX signals to create a new intermediate frequency signal. This process is known as heterodyning and produces a signal that can be filtered for processing.
- **Synthesizer**: generates signals with a range of frequencies.

### 3.1.2 Frequency Modulated Continuous Wave Radar (FMCW)

FMCW RADAR uses a continuous signal whose frequency is varied over time, known as a chirp [22]. The transmitted signal is given by

$$s(t) = e^{j2\pi\left(f_c t + \frac{Bt^2}{2T_c}\right)} \tag{3.1}$$

This method is widely used over pulsed and continuous wave radar as it can detect a target's range and velocity. The figure below illustrates a chirp's time and frequency whilst indicating key features like the bandwidth, period and centre frequency which are crucial parameters for determining a target's behaviour.

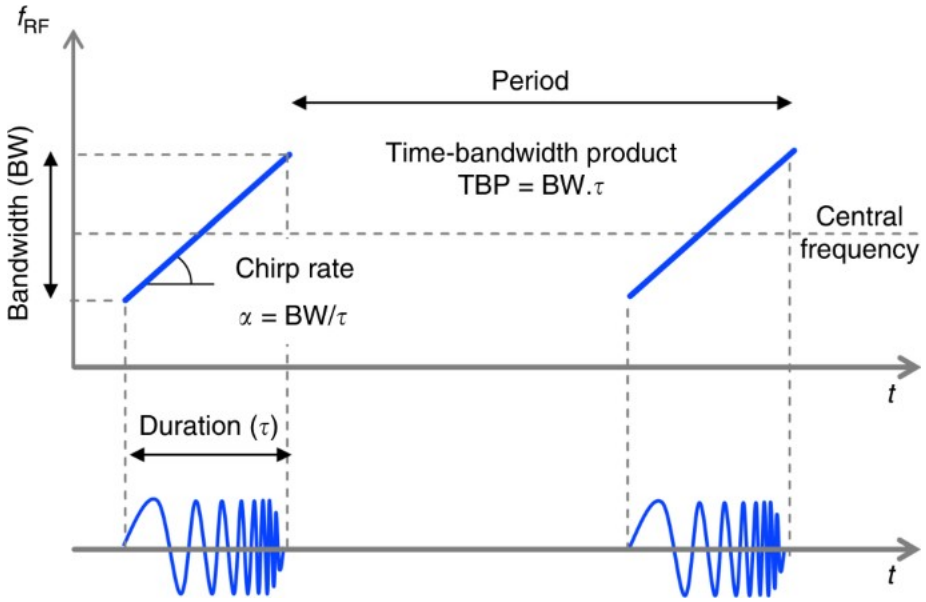



Figure 3.2: Chirp signal as a function of time and frequency [23]

## 3.2 Minimum Variance Distortionless Response

Angle of Arrival (AoA) estimation refers to the direction of the received signal. FMCW systems can determine AoAs by using a phased array of virtual channels. The difference in the distance between antennas causes an increasing phase shift along the antenna array.

Beamforming is a signal-processing technique that creates a spatial filter using a phased-antenna array

[24]. It is primarily used as a method to receive a signal whilst rejecting noise and interference however it is also used in AoA estimation by sweeping over an angle set and performing the beamforming operation at each angle.

MVDR is a beamforming technique that uses dynamically adjusted weights to minimize the overall output power (variance) while maintaining a distortionless response in the direction of the desired signal. This causes it to outperform conventional beamforming and AoA methods in noise rejection and angular resolution [24]. In a Uniform Linear Array (ULA), a steering vector is used to represent the expected phase shift across the antenna array. By aligning the steering vector with the desired signal direction and optimized weights, MVDR enhances signals arriving from that direction and suppresses interference and noise from others.

## 3.3 Data Structure

### 3.3.1 Radar Data Cube

eCAL facilitates the transfer of raw radar data which is mixed and stored in an HDF5, single array format as IQ data. Pre-processing is done on the single array which converts the data into a radar cube. The image below represents the packaged data.

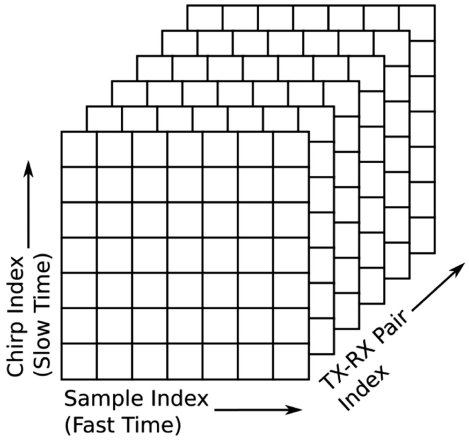



Figure 3.3: Pre-processed recorded data

The dimensions of the radar data cube are defined by the number of Rx channels per Tx channel, the number of chirps per frame, and the number of samples per chirp (nChannels x nChirps x nSamples). Fast time refers to the time between samples during a chirp whereas slow time refers to the time between chirps.

## 3.4 Data Processing

### 3.4.1 Range Doppler Maps

RDMs are two-dimensional representations displaying the detected targets' range and velocity (Doppler shift). An example of an RDM is illustrated below. Generating a RDM map involves processing the radar IQ data using a series of Fast Fourier Transform (FFT)s. Initially, an FFT is applied along

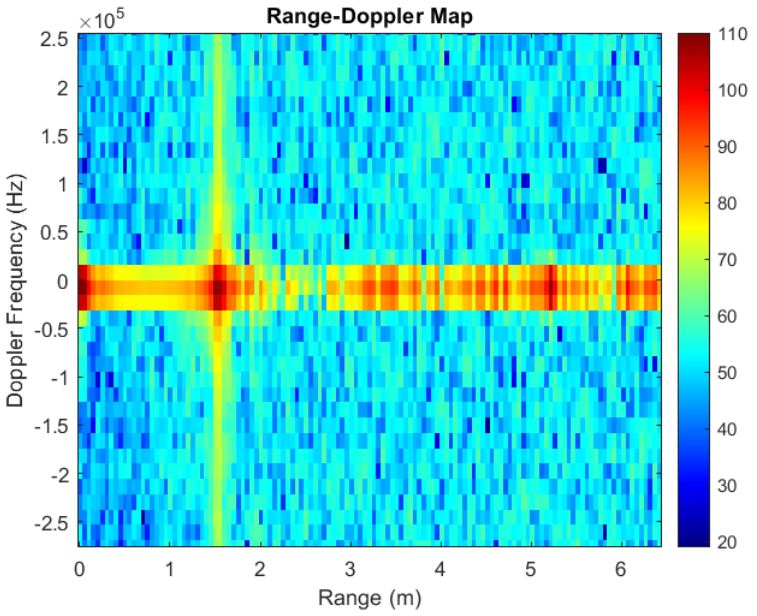



Figure 3.4: RDM of an environment with a target at 1.5m

the fast time dimension to convert the data into frequencies representing detectable target ranges. Subsequently, an FFT is performed along the slow time dimension, which extracts the Doppler shifts associated with target velocities. RDMs enable differentiation of targets based on their radial location and motion. The figure below visualizes the process.

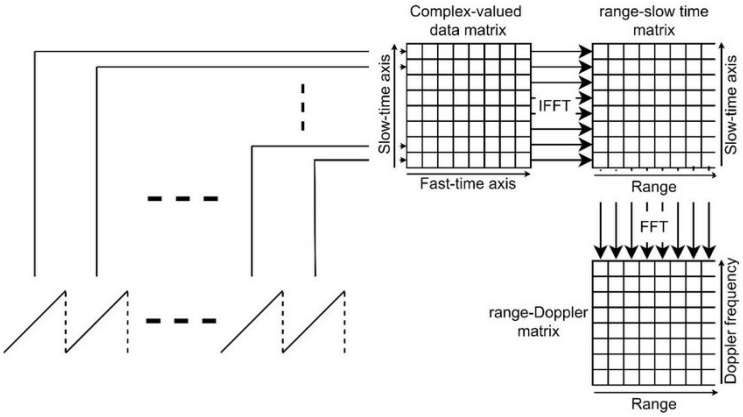



Figure 3.5: RDM Processing [25]

### 3.4.2 Range Angle Maps

RAZs integrate angular information with range data which enables spatial localization of targets. Multiple Input Multiple Output (MIMO) radar systems utilize an array of antenna elements to estimate the angle of arrival (AoA) of the reflected signals through beamforming, phase interferometry or spatial FFT techniques. After computing the range FFT, the angular information is extracted by applying and FFT across the channel dimension. Extracting azimuth and elevation requires a careful section of channels where azimuth channels require a horizontal separation whereas elevation channels require a vertical separation. The output of this process is a two-dimensional map where both range and AoA

can be visualized. The figure below depicts a range azimuth map.

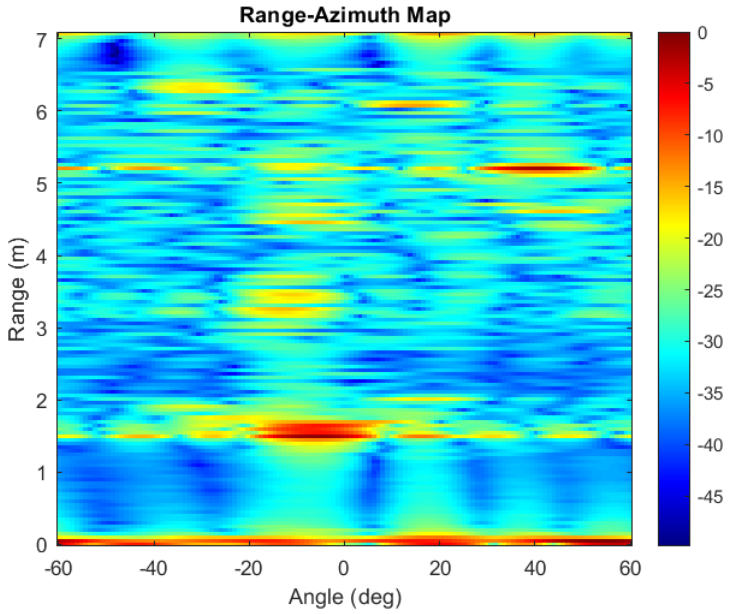



Figure 3.6: RAZ of an environment with a target at 1.5m at $0^o$

### 3.4.3 Map Scaling

RDMs and RAZs are useful for interpreting a target scene. In HPE they are used for feature extraction where the detected ranges and angles are used to map the features to 3D coordinates. As a result, associating scales with the position of detections on the maps is crucial. The following sections are vital for developing map scales.

#### RDM Equations

$$\Delta R = \frac{c}{2B} \tag{3.2}$$

where, B refers to the bandwidth and c refers to the speed of light.

Range bins refer to the individual cells in each row of the map. By splitting the range bins using the maximum range, using Equation 3.3 and the number of samples, each bin corresponds to a different range.

$$\max\_R = \frac{c \cdot sampling_rate}{2f_s} \tag{3.3}$$

where, fs refers to the frequency slop of the chirp.

A similar method can be done on the Doppler axis. Equations 3.4 and 3.5 are used respectively.

$$\text{vel\_res} = \frac{\text{wave\_length}}{4 \times T_{\text{chirp}}} \tag{3.4}$$

$$\max\_velocity = \frac{wave\_length}{4 \cdot T\_chirp} \tag{3.5}$$

Angular scaling involves the same process however, the bin sizes and maximum angles depend on the algorithm chosen.

### 3.4.4 Clutter and Noise

In radar systems, clutter and noise can significantly degrade the quality of data captured. Clutter is defined as the return signals caused by unwanted targets in the radar's field of view [26]. Noise is another disturbance that degrades the signal quality, commonly stemming from thermal noise due to electronic components and signal spillage caused by closely spaced antennas. Signal processing techniques such as MTI and CFAR reduce the impact of clutter and noise.

## 3.5 Intel RealSense Camera Projection

The Intel RealSense camera captures 3D space by combining data from multiple internal sensors [27]. Manipulating visual data is an important CV application as it uses this data to detect objects and track movements. In RADAR-based HPE, CV is a widely used technique for generating ground truth data [16]. This section outlines the theory and methods to convert camera data to 3D data points.

### 3.5.1 Pixel Projection

**Pixel Coordinates**

Each RGB frame is associated with a 2D pixel-coordinate space [27]. According to the developer documentation [27], the top left coordinate is represented by [0,0] and the bottom right coordinate by [m-1, n-1] where the frame has m columns and n rows.

**Pixel Mapping**

Real-world coordinates are referred to as point coordinates and can be linked with the pixel coordinates using the camera's intrinsic parameters [27]. Even with varying image sizes, fields of view, pixel shapes, and distortion, these intrinsic parameters ensure mapping between the 3D space and the 2D image plane.

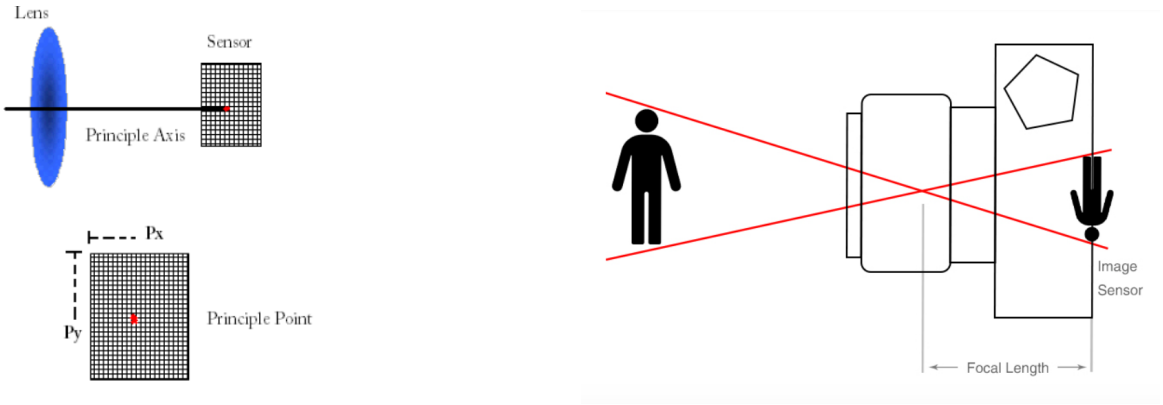
Extrinsic parameters describe the transformation between different 3D coordinate systems, particularly when multiple sensors are involved, such as depth and colour cameras [27]. Extrinsics consists of two parameters, the rotational matrix and translational vector. The first defines the relative rotation between sensors and the second describes the positional offset between the two coordinate systems.

Depth is stored as an unsigned integer associated with each pixel. The depth values have a scaling factor of 0.001 which converts the estimated depth to meters [27].

**Pixel Deprojection**

*cx, cy, fx and fy* are key intrinsic parameters that define the internal characteristics of the camera and how it maps 2D coordinates to 3D points in the real world. The principle point is defined as the point on the image plane that is center projected from the camera lens [28]. The focal lengths refer to the

distance from the point where the light meets inside the lens to the camera's sensor [29]. The intrinsic parameters are visualized in Figure 3.7.




(a) Visualization of the principle point

(b) Illustration of focal length

Figure 3.7: Intrinsic parameters of a stereo-depth camera

Equations 3.6 map the pixel coordinates from the 2D image into 3D space using the depth and the intrinsic camera parameters [27]. (u,v) are referred to as the pixel centroid coordinates.

$$\begin{aligned} X &= \frac{(u - c_x) \cdot Z}{f_x} \\ Y &= \frac{(v - c_y) \cdot Z}{f_y} \\ Z &= z \cdot scaling \end{aligned} \tag{3.6}$$

## 3.6 Convolution Neural Networks

CNNs are comprised of neurons which self-optimize through training primarily for classification and regression tasks [30]. They have become integral in HPE due to their ability to capture spatial hierarchies and complex patterns within data. When used for pose reconstruction, CNNs facilitate the transformation of point cloud data into 3D coordinate representations by handling sparsity and non-ideal spatial distribution. This theory section explores the CNN architecture design tailored for pose reconstruction. For a detailed explanation of the first principles, reference Wu et al. (2017) provides detailed insights [31].

### 3.6.1 Model Optimization

Models suffer from overfitting in the attempt to identify complex spatial relationships. As a result, regularization techniques are crucial for maintaining generalization in HPE networks.

**K Fold Cross Validation** K-fold cross-validation is a statistical method used to evaluate the generalization performance of machine learning models. It is done by partitioning the original dataset into K subsets, where training is done on K-1 folds and validated on the remaining fold [32]. This procedure is repeated such that each fold is used as validation once. The performance metric is generated by averaging the results across the K folds. This method optimizes model performance

by ensuring that the network is exposed to a wide variety of pose samples. It helps identify when the model's parameters cause overfitting. This method facilitates simple parameter variation which accelerates the model's optimization.

**L2 Regression (Ridge Regression)**

L2 regression, also known as Ridge Regression, is a regularization technique used to prevent overfitting setting constraints on large coefficients [33]. The L2 loss function is illustrated below.

$$\text{Loss} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \theta_j^2 \tag{3.7}$$

Due to the unpredictability of human movement across similar pose types, this technique mitigates the risk of overfitting to the training data [34]. Additionally, the balanced weights in the network improve its ability to handle noise and outliers in the point cloud data.

**ADAM Optimizer** Adaptive Moment Optimizer (ADAM) is an optimization technique for gradient descent. Fundamentally, it combines the qualities of two gradient descent methods: the Momentum and Root Mean Square Propagation (RMSProp) [35]. The Momentum algorithm is used to converge towards the loss minima by considering an exponentially weighted average of the gradient.

RMSProp maintains an exponential moving average of the squared gradients [35]. It modifies the standard Stochastic Gradient Descent (SGD) method by dividing the learning rate for each parameter by a running average of the previous gradient magnitudes.

Adam controls the rate of gradient descent which prevents it from settling at local minima and ensures minimal oscillation at the global minimum [35]. Equations 3.8 illustrate the ADAM equations as a combination of the Momentum and RMSProp methods.

$$\begin{aligned} m_t &= \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \\ v_t &= \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t \end{aligned} \tag{3.8}$$

ADAM is highly suitable for HPE networks due to its adaptive learning rate and efficiency [36].

### 3.6.2 Pose Specific Performance Metrics

**Mean Per Joint Position Error (MPJPE)** MPJPE calculates the average Euclidean distance between the predicted and ground truth points of each joint. It returns a single scalar value representing the error for all joints across the entire dataset. MPJPE is widely adopted for hpe due to its simplicity and interpretability [12, 10, 15].

$$\text{MPJPE} = \frac{1}{N \times J} \sum_{i=1}^{N} \sum_{j=1}^{J} \left\| \hat{\mathbf{Y}}_{i,j} - \mathbf{Y}_{i,j} \right\|_2 \tag{3.9}$$

Where
$N$ = Number of samples
$J$ = Number of joints
$\hat{\mathbf{Y}}_{i,j}$ = Predicted position of joint $j$ for sample $i$
$\mathbf{Y}_{i,j}$ = Ground truth position of joint $j$ for sample $i$
$\|\cdot\|_2$ = Euclidean (L2) norm

**Per Joint Error (PJE)** PJE computes the average Euclidean distance for each individual joint across all samples. This metric provides an error value for each key point which is important for identifying problem areas in the pipeline. Assessing individual key points reveals the systems capabilities over different joints. PJE is also highly interpretable as it is the segmented version of MPJPE.

$$\text{PJE}_j = \frac{1}{N}\sum_{i=1}^{N}\left\|\hat{\mathbf{Y}}_{i,j} - \mathbf{Y}_{i,j}\right\|_2 \tag{3.10}$$

Where
$N$ = Number of samples
$j$ = Specific joint index (ranging from 1 to $J$)
$\hat{\mathbf{Y}}_{i,j}$ = Predicted position of joint $j$ for sample $i$
$\mathbf{Y}_{i,j}$ = Ground truth position of joint $j$ for sample $i$
$\|\cdot\|_2$ = Euclidean (L2) norm

**Mean Average Error (MAE)** MAE calculates the average absolute difference between the predicted and ground truth joint positions across all joints and samples. Unlike MPJPE, which uses the Euclidean distance, MAE employs the Manhattan distance, measuring the sum of absolute errors without squaring them [37]. MAE more robust to outliers than MPJPE. While MPJPE assesses the spatial accuracy of skeletal pose reconstruction, MAE complements it by handling outliers.

$$\text{MAE} = \frac{1}{N \times J}\sum_{i=1}^{N}\sum_{j=1}^{J}\left|\hat{\mathbf{Y}}_{i,j} - \mathbf{Y}_{i,j}\right| \tag{3.11}$$

$N$ = Number of samples
$J$ = Number of joints
$\hat{\mathbf{Y}}_{i,j}$ = Predicted position of joint $j$ for sample $i$
$\mathbf{Y}_{i,j}$ = Ground truth position of joint $j$ for sample $i$
$|\cdot|$ = Manhattan (L1) norm

# Chapter 4

# System Design and Simulation Results

This section details the design considerations and the experiments used to validate the design choices. It introduces the objective of the project and the hardware available to achieve this. Thereafter, the established subsystems are explored. Using the experimental results, a critical evaluation is done on the design choices. Finally, the final pipeline is summarized at the end of this chapter.

## 4.1 Design Outline

Achieving pose estimation requires a structured approach that incorporates fundamental concepts to the practical subsystems. This section provides a system overview and by acknowledging the inherent challenges, this section aims to offer a clear understanding of how the design objectives are achieved while navigating practical considerations.

### 4.1.1 System Objective

The primary objectives of are outlined as follows:

1. **Accurate Skeletal Reconstruction:** The system must accurately reconstruct the human skeletal structure by localizing 13 key skeletal points.
2. **Sequential Processing:** Given raw data, the system should be able to output the prediction without any additional input.
3. **Practical Processing Time:** Since the system is intended for practical use, the pipeline is required to produce dynamic results.
4. **Robustness in Uncertain Environmental Conditions:** The system must be able to perform localization in cluttered environments. It must provide functionality with a robustness to lighting conditions.
5. **Practical Set Up:** The system setup can not consist of more than one RADAR module as this makes it an expensive option.

### 4.1.2 Overview of the Design Milestones

Since pose estimation is such a complex task, it was broken up into sub-systems. This ensures that expected results are produced at each point in the processing chain. By doing this, it simplifies the troubleshooting process. Figure 4.1 represents a high-level diagram of the system's sub-modules.

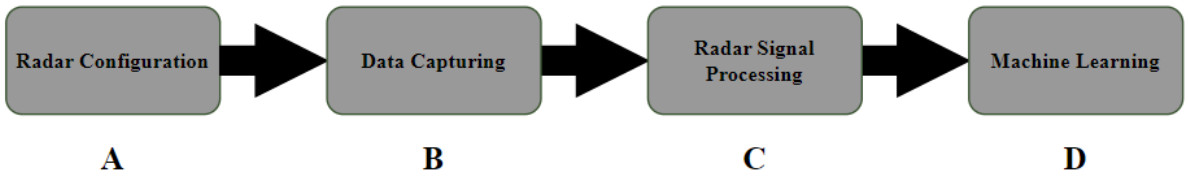



Figure 4.1: Flowchart illustrating the sequential flow between milestones

The design process began with RADAR configuration. This involved understanding the pre-existing drivers and tools available for data capturing. Thereafter, the focus shifts to the setup and calibration of the RADAR system. This phase included generating a pose estimation tailored configuration.

The next phase, data capturing, focused on assembling a pose set by gathering RADAR data for further analysis. During this stage, data was collected under controlled conditions, ensuring the data's integrity.

Following data acquisition, the RADAR signal processing phase commenced. This stage involved the conversion of raw RADAR data into a usable format, data transformation, filtering, and processing to extract meaningful information. The processed data was then visualized for system validation and further analysis.

Finally, the design concluded with machine learning integration. This stage comprised several steps, including gathering training data and ground truth labels, machine learning architecture selection and the development and training of classification models. This final stage returned the reconstructed poses.

## 4.2 Experimental Setup

In this project, the core hardware utilized was a pre-configured RADAR system belonging to the Radar Remote Sensing Group (RRSG).

### 4.2.1 Radar System

The primary setup centred around the Texas Instruments AWR1843 RADAR module, paired with their DCA1000EVM FPGA for data capture. Table 4.1 summarizes the AWR1843's key characteristics and Figure 4.2 illustrates the modules' hardware layout.

| Characteristic | Value |
|---|---|
| Frequency Range | 76–81 GHz |
| Max Bandwidth | 4 GHz |
| Number of Transmitters | 3 |
| Number of Receivers | 4 |
| Maximum ADC Sampling Rate | 25 MSPS |

Table 4.1: Key characteristics of the AWR1843 radar module

The DCA1000EVM capture card enables ADC data streaming over Ethernet. In this implementation, the DCA1000EVM capture card was used to stream the data to the host via an LVDS interface, facilitating real-time raw RADAR data acquisition.

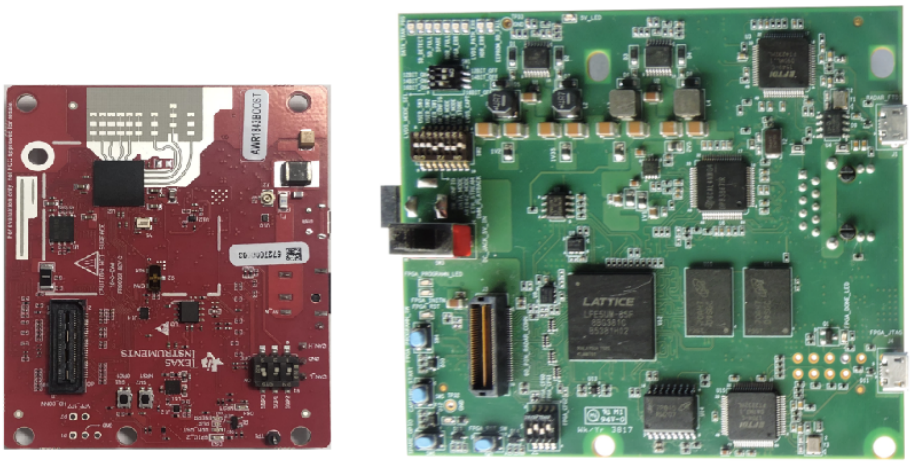


Figure 4.2: Top view showing the AWR1843 and DCA1000EVM hardware respectively

In this system configuration, the NVIDIA Jetson performs an important function providing the interface for the radar module. The Jetson is a compact computing platform designed for efficient real-time data collection. It handles the initial data collection and the setup involving loading configuration files, executing pre-processing scripts and simplifying file transfer.

### 4.2.2 Image Capture System

The image capture system used in this setup consists of the Intel RealSense Stereo Depth Camera D400, which captures both depth and RGB information. As the secondary system, it complements the RADAR data by providing comprehensive visual and depth data. The D400's ability to generate real-time depth images from stereo vision makes it ideal for 3D applications.

### 4.2.3 Data Processing Hardware

Although it is designed for ML, the NVIDIA Jetson lacks the sufficient computational power required for RADAR signal processing in conjunction with ML [38]. This is due to its limited GPU and CPU resources compared to a high-performance PC. RADAR signal processing requires a large amount of memory and floating-point performance, which the Jetson's architecture is not well-suited for [38]. As a result, an Intel i5 PC with 8GB of RAM was selected for developing the processing chain, offering sufficient performance for prototyping and debugging. To execute the entire processing pipeline across the full dataset, a powerful CPU was used with the following specifications: Intel i7, 6-core processor, 24GB RAM, and with an independent GPU.

## 4.3 Development Environment

Initially, MATLAB was used to interface with HDF5 files as it provides an extensive list of utility functions. Moreover, MATLAB's workspace panel allows for easy interaction with variables. Additionally,

MATLAB offers a wide range of signal processing toolboxes essential for RADAR signal processing. Lastly, its plotting capabilities coupled with its well-established community make it an ideal platform for data analysis, development and debugging. As a result, MATLAB was used for developing the processing pipeline.

Python was chosen for the ML segment due to its simplicity and extensive libraries. One of the key advantages of using Python is access to cloud-based platforms like Google Colab, which provides additional virtual computing power to reduce training time. TensorFlow was selected as the framework due to its available documentation and industry-wide adoption. Its integration with Python ensures compatibility with other Python libraries.

## 4.4 Data Collection

### 4.4.1 Pose Selection

The AWR1843 has an elevation resolution of $57^o$. This poor resolution is caused by the module having two virtual antennas in the vertical plane compared to the eight virtual antennas in the horizontal plane. The antenna placement can be found in Appendix D. Due to the RADAR's physical limitations, careful consideration was taken to select poses that optimize the ML model's output. The poses that had several vertically close-spaced key points were ignored due to the AWR1843's poor elevation resolution. By moving the target closer to the sensor, the elevation angular distance between key points increases however there is a tradeoff with the field of view. As the target moves closer, extreme key points move out of the frame. As the target moves away from the sensor, the azimuth angular distance reduces which weakens the azimuth estimation capabilities. An experiment was done to determine the optimal range position to record poses. The table below summarizes the performance at different ranges.

| **Range (m)** | **Shoulder-Shoulder Angle (°)** | **Knee-Shoulder Angle (°)** | **Key Points missing** |
|---|---|---|---|
| 1 | 28.2 | 45 | 5 |
| 1.5 | 18.9 | 33.7 | 2 |
| 2 | 14.3 | 26.56 | 0 |
| 3 | 9.52 | 18.43 | 0 |
| 4 | 7.15 | 14.1 | 0 |

Table 4.2: Range optimization experimental results

As illustrated by Table 4.2, poses recorded between 1.5m and 3m optimize the three metrics.

Furthermore, large varying poses were prioritized since this would minimize the overlap between the between the dataset and the ground truth. Using these criteria, the pose set consisted of five poses: Star Pose, Right Arm Extension, Left Arm Extension, Hands Raised and Freeze Pose. The minimal overlap of the key joints across these five poses aims to reduce generalization ambiguity.

### 4.4.2 Scene Preparation

To ensure data integrity, several precautionary measures were implemented. First, all moveable clutter was removed from the scene, reducing the risk of unwanted reflections. Second, data was captured in a

controlled environment with no movement, ensuring that only the intended subject was recorded.

### 4.4.3 Radar Configuration

Before capturing data, the RADAR requires a configuration. The configuration parameters of a RADAR system play a crucial role in determining its performance. This affects key aspects such as range resolution, maximum range, and velocity detection. Pose estimation requires comprehensive key point recognition therefore it demands a high range resolution. Since this pose estimation system handles static poses, preference is placed on the range resolution.

| **Parameter** | **Result** |
|---|---|
| Range Resolution | 0.05m |
| Maximum Detectable Range | 7.12m |
| Velocity Resolution | 15.1 m/s |
| Maximum Detectable Velocity | 0.81 m/s |

Table 4.3: Ideal System Parameters

Bandwidth directly influences the RADAR's range resolution; a wider bandwidth improves resolution, allowing the RADAR to detect closely spaced targets. The chirp duration influences the maximum detectable range, where longer chirps enable the RADAR to detect objects further away. The velocity resolution depends on the amount of chirps and the period between them. Due to computational restraints and hardware limitations, a balance must be found for each parameter.

The Texas Instruments mmWave Sensing Estimator provided an automated tool for chirp design and parameter configuration [39]. Table 4.4 summarizes the final RADAR configurations utilized for data capturing.

| **Radar Parameter Configuration** | |
|---|---|
| **Parameter** | **Details** |
| Start Frequency | 77MHz |
| Frequency Slope | 63.2 MHz/$\mu$s |
| Sampling Rate | 3000 ksps |
| Bandwidth | 2.99GHz |
| Idle Time | 7$\mu$s |
| Ramp Time | 56.3$\mu$s |
| Frame Period | 33.33ms |
| Antennas | 3 TX, 4 RX |
| Samples | 142 |
| Chirps | 37 |

Table 4.4: Table summarizing key configuration parameters

Several configurations were generated and tested. Maximizing the bandwidth resulted in an improved range resolution however the number of samples doubled thereby doubling the file size. Reducing the number of chirps, reduced the velocity resolution howver this degraded the ability to recognize micro-Doppler shifts in the RDM. The parameters in Table 4.4 yielded optimal radar performance whilst achieving the desired resolutions.

### 4.4.4 Ground Truth Raw Data

To acquire the raw ground truth data, the resolution of the RealSense D400 camera was set to 1280x720, ensuring that high-quality data was captured. High-resolution data capture ensures robust performance of the labelling model. Following this, the RADAR data and the RGB-depth data were synchronized using time stamps which allowed the two data streams to be collected in parallel. By doing this, the samples could later be accurately correlated for ML training.

## 4.5 Signal Processing Pipeline

### 4.5.1 Data Pre-Processing

During data capture, raw data is streamed from the AWR1843 module via a two-lane LVDS stream [40]. Figure 4.3 illustrates the raw ADC data streaming structure.

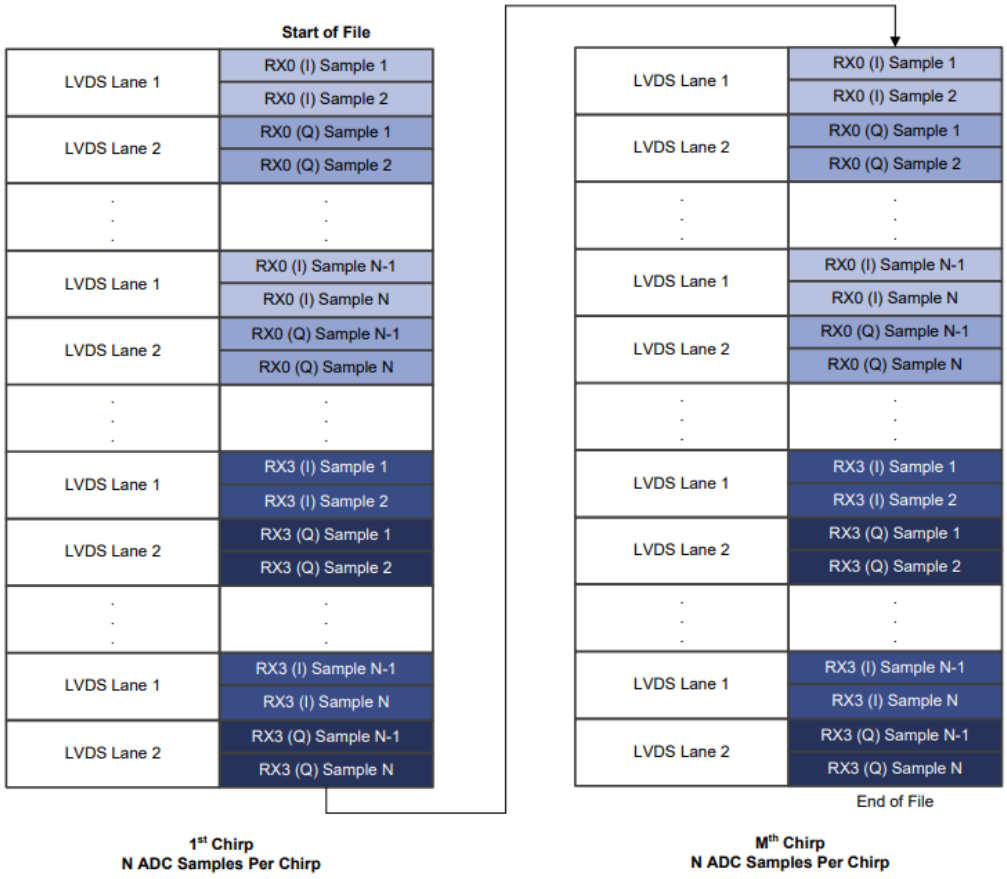



Figure 4.3: AWR1843 Complex Data Format Using DCA1000 [41]

Following data capture, the ADC data was stored using a 1-dimensional complex array in a frame-by-frame configuration. Thereafter, the 1-D array was transformed into a RADAR data cube using existing Python scripts written by Nicholas Bowden. Using the configuration file, the RADAR cube was instantiated with dimensions of channels by chirps by samples. The Python script was executed over the entire dataset resulting in each frame having an associated RADAR data cube stored in the HDF5 format. After reading the data into MATLAB, the imaginary and real data cubes were

concatenated ensuring IQ data representation.

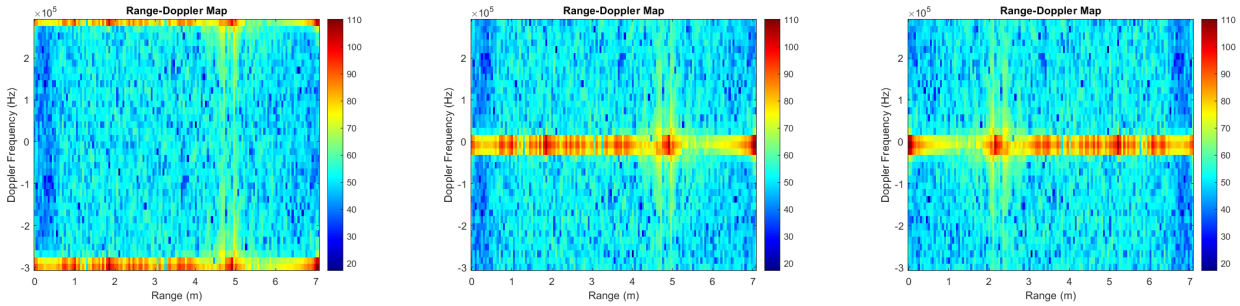
### 4.5.2 Range Doppler Map Generation

In RADAR-based pose estimation, Range Doppler Map (RDM)s play a critical role in validating the system's functionality. Furthermore, it is essential to validate a sample's data integrity before capturing a large dataset. These maps are generated by processing the RADAR's raw data, outlined in Section 3.4.1, to represent the data's range and Doppler (velocity) information.

Using the method described in Section 3.4.1, The RDM returned unexpected results. The output of the FFT orders the array such that the DC point is at the start of the array. Due to this, the positive frequencies appear in the first half and the negative frequencies appear after. Figure 4.4a below shows the map generated after computing two FFTs.

As a result, MATLAB offers a fftshift operation which reorders the frequencies into an intuitive order. After plotting the updated RDM, as seen in Figure 4.4b, the map appears to be plausible however the target was placed at 2.3m.

During pre-processing, the data was transposed. In RADAR systems with close-spaced antennas, there is crosstalk causing noise at the 0 range bin. From the map, it can be seen that the 0 range bin noise occurs at the maximum range further proving that the data was transposed. By implementing the flip function, the RDM represents the true measurement shown in Figure 4.4c.




(a) Range Doppler Map of the recording before shifting and flipping operations

(b) Range Doppler Map of the recording before the flipping operation

(c) The final RDM returning expected results of a stationary target at 2.2m

Figure 4.4: Visual changes from clutter removal

Three other experiments were conducted to test the robustness of RADAR graphical data. Firstly, recordings were taken in two extreme lighting conditions. Both RDMs displayed the target at the same range indicating that the system is immune to lighting conditions. Secondly, multiple corner reflectors were spaced out at different ranges. In this test, the RDM was able to detect all three targets however the most central target had the dominant reflection. Lastly, objects with different RCS were tested. The reflection of a corner reflector returned a higher intensity than that of a ball at the same range. These results can be found in the Appendix.

RDMs are better-suited for pose estimation compared to spectrograms, RTMs, or DTMs due to their ability to simultaneously capture range and velocity information. In the static pose application, RDMs

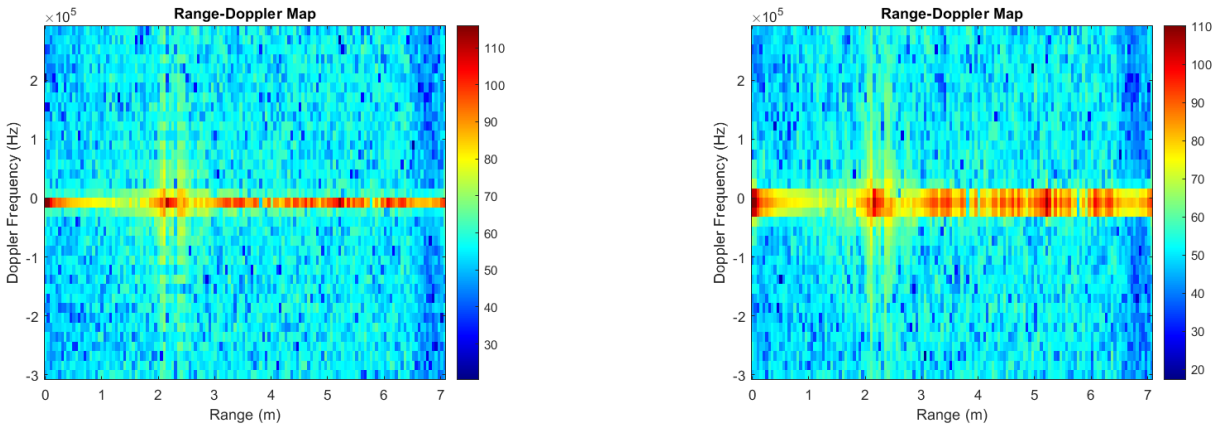
are preferred over range-angle maps as they ensure that ranges are isolated from the angles for target recognition. By extracting range-angle information, further processing has to be done to extract the secondary angle and pair the coordinates. Conversely, after range extraction, the two angles can be estimated and paired in parallel.

### 4.5.3 Windowing

Windowing is a signal processing technique used to prevent the spectral leakage that occurs when performing a FFT on finite-length RADAR data [42]. This is caused when energy from one frequency bin 'leaks' into adjacent bins.

In radar systems, the signal is collected over a finite time window, meaning there are abrupt cuts at the beginning and end of each frame. When this data is transformed into the frequency domain via an FFT, the abrupt cut-offs introduce unwanted high-frequency components, which distort the signal's frequency content [43].

Different windowing functions (such as Rectangular, Hanning, or Blackman) offer different trade-offs. The Hanning window prevents leakage whilst maintaining frequency accuracy however the cost is a small amplitude error [42]. By reducing the side lobes, windowing helps to refine the Range-Doppler Maps (RDMs), making them more accurate and reliable for tracking dynamic human poses.




(a) Range Doppler Map without using windowing

(b) Range Doppler Map using windowing

Figure 4.5: Visual effect of applying a Hanning window function

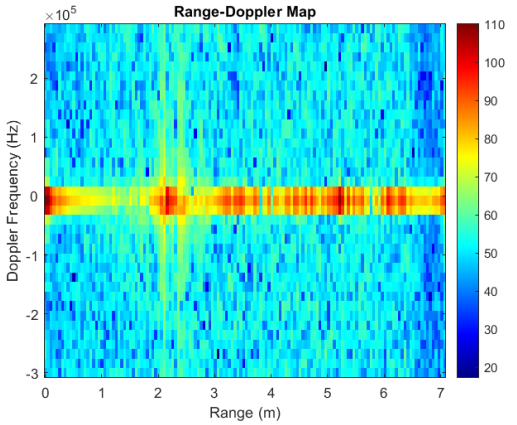
Without windowing, spectral leakage can cause strong clutter to mask weak moving targets even if the clutter and target are separated in the Doppler [44]. As seen in Figure 4.5, a side effect of windowing causes the spectrum to spread over neighbouring spectral lines. If detecting weak moving targets is important, Koks et al (2014) advise that this can be done by removing clutter [44].

### 4.5.4 Clutter and Noise Removal

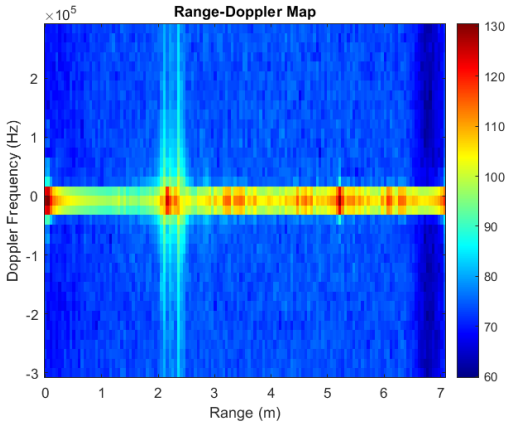
Noise and clutter are two factors that can significantly degrade system performance. Noise can arise from various sources such as thermal noise and electronic interference. Clutter consists of unwanted echoes from surrounding objects. These clutter signals can mask weaker signals from important targets. Effective noise and clutter removal techniques are essential to ensure expected performance.

In each of the previous RDMs, a high-intensity band was consistently observed at the zero range bin. This is due to close-spaced antennas that caused crosstalk. To address this, a widely used method, mean subtraction which removes the DC offset, was tested to minimize the noise. However, a simpler method involves cropping out the zero range bin. In this application, this method is a suitable simplification since targets should not be in the zero range bin.

Additionally, since noise can be considered as Gaussian white noise therefore summing the RDMs across all the channels amplifies the signal while the varying noise is averaged out. This method significantly improves SNR. Figure 4.6 below illustrates the difference between a single channel RDM and a summed channel RDM.




(a) Range Doppler Map generated from a single channel




(b) Range Doppler Map generated from summing over all channels

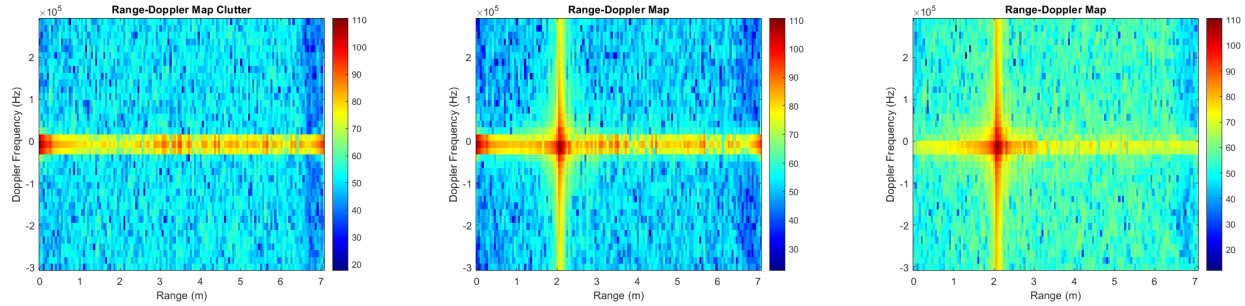
Figure 4.6: Comparison of Range Doppler Map with and without channel summation

In the literature [11, 16], MTI is a popular method for clutter removal. Although widely used, it is not suitable for this application since the poses being studied are primarily stationary. One method that is suitable, involved capturing a sample of the static clutter and subtracting it from each subsequent recording. Since the clutter is present in both samples, this subtraction effectively eliminates the clutter, resulting in cleaner data. This is not practical for wide use since clutter samples would be required for each environment in which it is deployed. Furthermore, removing all clutter prevents the ML network from engaging with non-ideal data. This may cause it to be vulnerable when tested on data from different sources.

In Figure 4.7c, the clutter that was present on the zero velocity band has been removed. Furthermore, the zero-range bin noise has also been removed. This clutter-removal process reduces the likelihood of false target detections from the target recognition algorithm.

### 4.5.5 Target Recognition

After clutter removal, target recognition remains critical for isolating key points within the RADAR frame. HPE relies on an accurate coordinate transformation which requires computationally intensive processes. To reduce the processing load, the dataset must be minimized. CFAR plays a crucial role by dynamically adjusting detection thresholds based on surrounding noise and clutter levels, ensuring




(a) Range Doppler Map of static clutter

(b) Range Doppler Map of a pose surrounded by clutter

(c) Range Doppler Map of a pose with clutter removed

Figure 4.7: Visual changes from clutter removal

that only true targets are identified and reducing false positives.

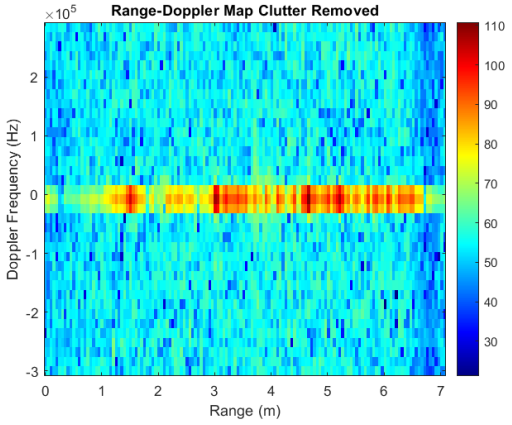
Before implementing CFAR, it was important to find optimal parameters such as guard cell dimensions, training cell dimensions and Probability of False Alarm (PFA). Guard cells are placed around the Cell Under Test (CUT) to protect it from interference caused by nearby targets. Nearby targets influence the noise estimate which causes missing detections amongst weaker targets. Given that the human body is assumed to be less than 2 meters in length and the range resolution is 5 cm, a pose can occupy a maximum of 20 distinct range bins (given that the RADAR is 1m high). Furthermore, the strongest Doppler intensity spans 3 bins therefore the guard cell matrix is determined to have dimensions of 20 by 3. Training cells are used to gather an average noise estimation for the CUT. Using the guard cell matrix, the training cell matrix size was set to 30 by 5. The PFA was determined through an iterative process whilst keeping the threshold constant.

| **PFA** | **Threshold (dB)** | **Number of Detections** |
|---|---|---|
| $10^{-3}$ | 100 000 | 86 |
| $10^{-4}$ | 100 000 | 64 |
| $10^{-5}$ | 100 000 | 58 |
| $10^{-5}$ | 200 000 | 54 |
| $10^{-5}$ | 400 000 | 51 |

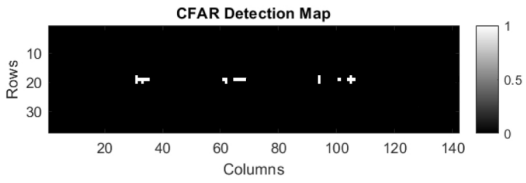
Table 4.5: Iterative CFAR tests

A controlled experiment was done to assess CFAR's performance. Three targets were placed at different ranges and around 60 detections were expected. As seen in Table 4.5, expected detections were missed beyond $10^{-4}$. As a result, the configuration used in the second experiment was used as the final CFAR set. The last two experiments were computed to show that both PFA and threshold could be varied to limit the detections.

Figure 4.8b illustrates the final CFAR set's ability to detect targets. As seen in the figure, the three targets are detected including a small number of false detections which are removed in the pose-processing.




(a) Range Doppler Map of a 3 target scene




(b) Detection locations from the Range Doppler Map

Figure 4.8: Target extraction using CFAR

### 4.5.6 Angle Estimation

The extracted ranges required an associated elevation and azimuth angle before computing a point cloud transformation. Since the horizontal plane contained eight virtual antennas, it provided a workable system for estimation. Extracting accurate angles was imperative as the quality of the coordinate transform depends on both azimuth and elevation. This meant that although the range and azimuth estimations were sufficient, the entire point cloud would not be a true representation of the target due to the elevation error. Minimizing the the elevation error was the most important task due to the two vertically separated channels. Several approaches were tested in an attempt to optimize the angular resolution and processing time.

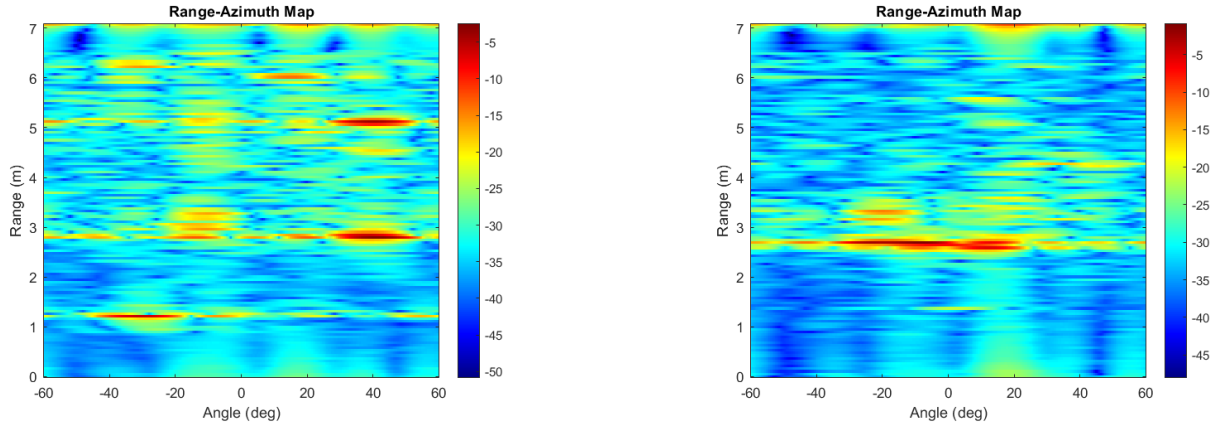
#### Channel FFT Approach

The FFT based approach was implemented as the initial approach due to its simplicity and computational efficiency. It used a simple process for angle estimation similar to the range and Doppler FFT transformation approach, allowing for straightforward analysis of the radar data.

Channels 1,2,3,4,9,10,11,12 were used for azimuth estimation. This configuration was chosen since channels 1 to 4 each have a separation of half a wave length. Channels 9 - 12 also have a half wavelength separation however channel 9 is separated from channel 4 by 2 two wavelengths. This means that the separation between all eight channels is half a wavelength. Although there are four pairs of elevation angles, only two were capable over returning distinct angles. As a result, channels 8,12 were chosen.

To validate this angle estimation approach, controlled experiments were conducted where Range Azimuth Map (RAZ)s and Range Elevation Map (REV)s were generated. The first experiment involved testing the system's angular accuracy and its ability to capture multiple targets. Figure 4.9a shows that the system was successful in capturing two targets, the first at 1.2m offset by $-30°$ and the second at 2.7m offset by $45°$. The second experiment tested the system's resolution by moving the targets closer until ambiguity occurred. Overlapping occurred when the angular distance between the two targets was $10°$. In MIMO processing, angle estimation has a beamwidth, which determines resolution, at different power levels [45]. Using the 3dB standard, the RADAR has a $15°$ beamwidth in the horizontal plane. The beamwidth plot can be seen in Appendix D.2. This indicates that spatial

separation is not possible using this technique if the angular difference is less than 15°.



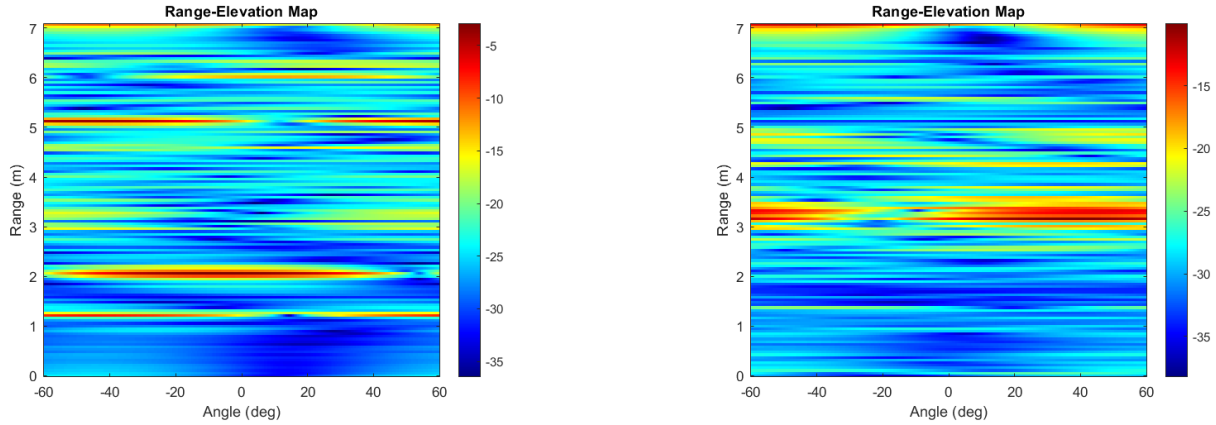

(a) Range Azimuth Map of a two-target scene with corner reflectors at −30°, 1.2m and at 45°, 2.7m

(b) Range Azimuth Map of a two-target scene with corner reflectors at −5°, 1m and at 5°, 2m

Figure 4.9: Azimuth Estimation Validation

### Phase Interferometry Approach

The second validation experiment involved measuring the elevation estimation performance. As discussed earlier, the beamwidth is a determining factor in the system's estimation capability. In the vertical plane, the RADAR has a 3dB beamwidth of 28° [45]. As a result, the FFT-based approach is not suitable for elevation estimation. Figure 4.10a and Figure 4.9b further prove this as there is a large spectral band at the correct range with no definite angle.




(a) Range Elevation Map of a target at 0°, 2m

(b) Range Elevation Map of a target at −30°, 3m

Figure 4.10: Elevation Estimation Validation

As shown in Equation **??**, the angular resolution depends on the physical hardware. Therefore signal processing is unable to improve it. The subsequent methods do not rely on the number of channels to improve angular resolution.

Phase interferometry was the second approach explored for elevation estimation. Table **??** presents the elevation estimation results. The results show that although the approach can work, as seen in tests 1,3 and 7, the data was not robust since there were only two samples. Noise from the environment and poor

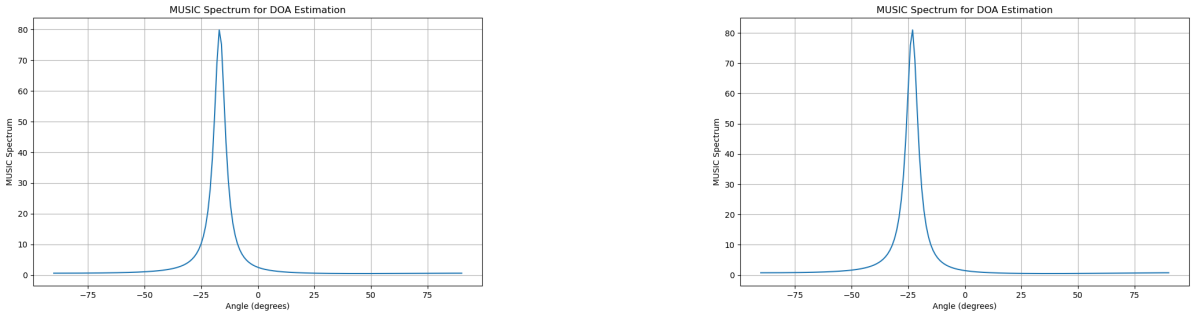
mixing could alter the received signal significantly resulting in poor estimation. Multi-path reflections are also a major challenge for phase interferometry, degrading the estimation results. Furthermore, this approach is unsuitable for HPE since it can only determine the dominant angle where pose estimation requires an elevation angle corresponding to each azimuth angle estimated.

| **Target** | **Expected Angle (°)** | **Estimated Angle (°)** |
|---|---|---|
| Corner Reflector | 10 | 12.4 |
| Corner Reflector | 10 | -32.3 |
| Corner Reflector | 10 | 15.2 |
| Corner Reflector | 15 | 23.6 |
| Corner Reflector | 15 | 1.8 |
| Star Pose | 0 | -2.3 |
| Star Pose | 5 | 46.1 |

Table 4.6: Phase Interferometry Test

### MUSIC Algorithm Approach

This section presents the experimental outcomes of the MUSIC algorithm, focusing on its angular resolution and ability to estimate multiple angles. The first experiment tested angular resolution by shifting the corner reflector to vary the elevation. Figure 4.11 shows the reduced beamwidth which provides an improved angular resolution. Moreover, the algorithm was able to determine a target at $-17°$ and $-23°$.




(a) MUSIC Spectrum showing a target at $-17°$

(b) MUSIC Spectrum showing a target at $-23°$

Figure 4.11: MUSIC Spectrum resolution test

The second test was not conducted since the MUSIC algorithm was unable to estimate more angles than the number of antennae on the RADAR. Since MUSIC splits the subs-space into noise and signal, it is unable to estimate more than n-1 angles (where n is the number of antennae) as the noise subspace has to be non-zero.

In the horizontal plane, MUSIC was able to detect between 3 and 5 key point angles. Although MUSIC provided improved accuracy compared to the FFT approach, its extended processing time for the same number of estimations makes it impractical for this investigation.

### MVDR Beamforming Approach

Unlike the prior approaches, MVDR uses a beamforming technique that minimizes the output power in

all directions except the direction of the desired signal. In this approach, tests focused on maximizing the number of estimated angles whilst maintaining accuracy. MVDR was able to identify two elevation peaks in a single frame and resulted in wider beamwidths than MUSIC. Despite returning the same amount of distinct angles, other high-power angles which neared peaks, were also extracted. Although MVDR could not estimate all the key point angles, gathering the high-power angles reduced the point cloud's sparsity.

MVDR was deployed for both azimuth and elevation estimation. The algorithm improved spatial resolution and reduced interference resulting in better spherical coordinate extraction. Although MVDR outperformed the previous techniques, the amount of matrix operations led to a significant reduction in processing speed.

### 4.5.7 Point Cloud Extraction and Dataset Generation

Point cloud generation is the final RADAR signal processing stage in the pose estimation pipeline. In this stage, extracted features (range, azimuth, elevation) are transformed from spherical to Cartesian coordinates. These coordinates describe the target's position relative to the RADAR's reference frame. The point cloud serves as the initial reconstruction from the RADAR system, facilitating classification and pose estimation.

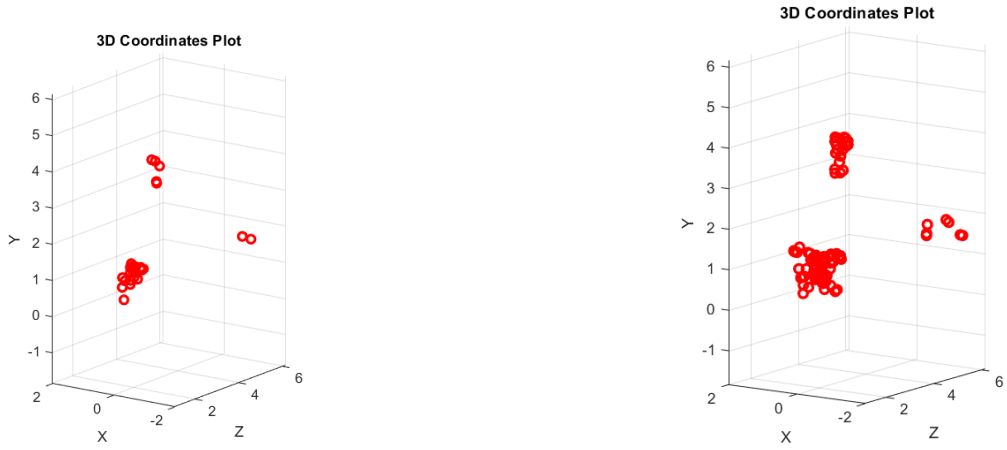
Equations 4.1 were used to perform the coordinate transformation.

$$\begin{aligned} X &= \rho \cdot \sin\phi \cdot \cos\theta \\ Y &= \rho \cdot \sin\phi \cdot \sin\theta \\ Z &= \rho \cdot \cos\phi \end{aligned} \tag{4.1}$$

Due to the limited number of angle estimates, the point cloud suffers from sparsity and is difficult to visualize the initial pose. This can be seen in Figure 4.12a. Apart from poor angle estimates, frames do not contain reflections from all the key points. Considering this, an approach was used where point clouds were generated and superimposed to maximize the key point completeness. This was a valid solution since the pose set consisted of static recordings. The point density difference can be seen in Figure 4.12b.

Although the point cloud structure improved, false detections from noise and persistent clutter remained. To ensure that only key point detections were used for classification and regression, a simple approach of zooming into a subspace was used. The subspace consisted of $x \in [-1, 2]$, $y \in [-1, 2]$, and $z \in [0.5, 4]$. In a complex task where the basis is determining spatial relationships, it was critical to reduce sources of confusion (a small amount was needed for robustness).

After generating a single point cloud, it was observed that the process took a significant computation time of 12 minutes. This was primarily due to the intensive amount of matrix and linear algebra computation required for angle estimation. The processing time grew linearly as the number of angles for estimation and frames superimposed increased. This made the current method difficult to execute on a processing power budget. To reduce the processing time, parallelization was used. This technique involved dividing a large computational task into sub-tasks that were executed in parallel on multiple




(a) 3D abstraction of a single frame point cloud

(b) 3D abstraction of four frames superimposed

Figure 4.12: Comparison of a sparse point cloud to a multi-frame point cloud

processors which reduced overall computation time [46]. MATLAB offers a comprehensive set of tools as part of the Parallel Computing toolbox. By implementing a parallel for-loop structure with pre-allocated storage, the processing time was reduced to 1 minute per sample.

The detected points were then sorted to gather the highest intensity points. Afterwards, the data was sorted by x (ascending) with y and z values breaking up equal-magnitude points. The data was then stored in an 8 by 8 by 4 where each dimension represented one of x, y, z or intensity. This reshaping process created point cloud image structures mimicking RGB image structures. Sorting was imperative as image-based ML requires consistency between the data to realize spatial patterns.

### 4.5.8 Final Processing Pipeline

This section outlines the rationale behind each component of the final pipeline design, providing a comprehensive analysis of the selected processes in achieving pose reconstruction. By examining the suitability of each sub-process, the pipeline is optimized for reliable, computational efficient results. Additionally, the summarizing graphic 5.4 presents a visualization of the pipeline flow.

Each recording sample consisted of around 60 frames which were further divided into 6 batches of 9 samples. One batch represented a single pose sample which dictated by the sample sizes required to optimize the MVDR algorithm's performance.

RDMs were generated from each batch in preparation for clutter removal and target extraction. Although the static clutter subtraction method returned promising results, it was not implemented in the final design. It is not practical to capture static scenes before each use of a system and therefore the ML model would struggle to generalize to data containing clutter. The alternative method of performing a channel sum was used to increase the reflection power whilst levelling the contribution from noise. In addition to this, the pipeline partitioned into the subspace of interest which maintained enough clutter for generalization whilst removing outlier detections. CFAR was used on the summed RDM map and its detection points were used on each of the twelve maps. The RDM sum reduces the amount of noise which is beneficial for CFAR detection.

Using the channel FFT for azimuth in conjuction with MVDR for elevation was the most computationally efficient angle estimation system. However both azimuth and elevation employed MVDR AoA due to the availability of a high-performance PC, as detailed in Section 4.2.3. Although this process was computationally expensive, it generated point clouds that were dense enough to create input feature maps of adequate size. MVDR was preferred over MUSIC as MVDR performed spatial sweeping whereas MUSIC split the signal into subsets. Similar to MUSIC, phase interferometry was fundamentally unable to estimate multiple angles in the elevation direction.

The final point cloud was sorted by intensity to retain 64 points. The amount of points retained, was decided based of the amount of detections that theoretically belong to a pose and was constrained to a square value due to CNNs ability with square feature maps. Since CNNs require structured data, sorting the data in preferential order of x, y and then z created a spatial ordered in the map. Prioritizing x ensure that the feature map retained its spatial relationships with other surrounding points.

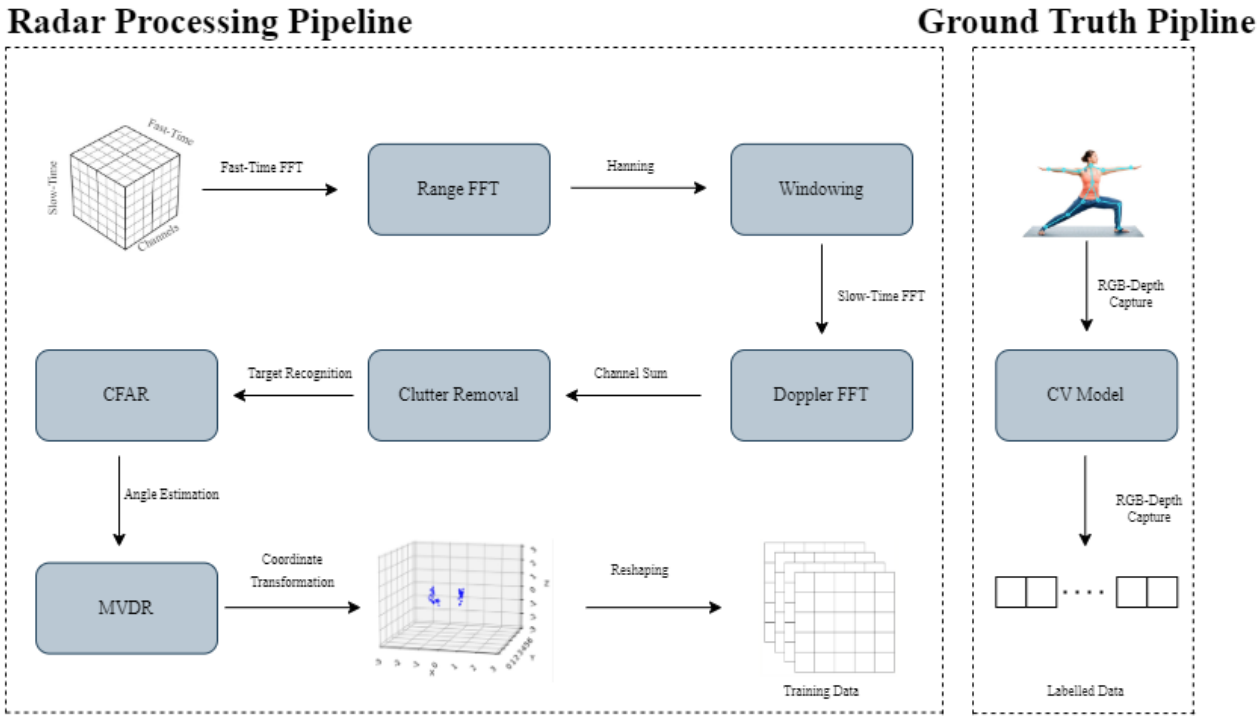



Figure 4.13: Block Diagram of the radar processing pipeline in parallel with the ground truth pipeline

## 4.6 Ground Truth Pipeline

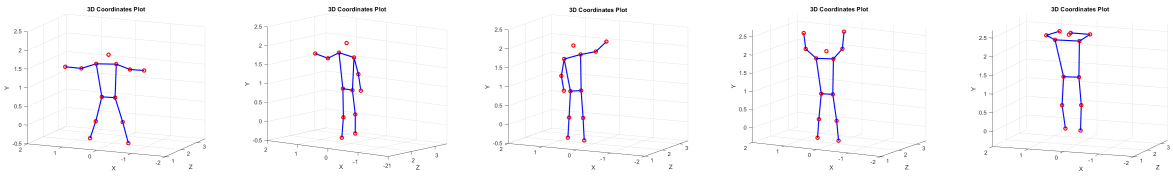
### 4.6.1 Ground Truth Model

RADAR point clouds lack detailed information about the original pose and therefore ML methods are required to find spatial relationships between the data points and the true key points. Ground truth data provides a benchmark to tune hyper-parameters and improve the learning process.

Repositories such as *OpenMMLab* and *MARS*, provide access to several ground truth datasets [47, 48]. However, these sets are not usable since the spatial relationship between the ground truth and the

pipeline's raw data is not the same as it is to the third party's raw data used to generate the labels. As a result, the Intel RealSense Stereo-Depth camera was used to capture raw ground truth data. Before capturing data, the RADAR, depth and colour data were synchronized using time-stamps which enabled simple correlation in the post-processing. Manually labelling the dataset requires 13 labels per sample and would require 13,000 labels in total which is impractical. YOLO v8 is a CV framework that provides pre-trained models and a simple interface to train new models for automatic labelling [49]. Since YOLO v8 developed a model for key point detection, it was used in the ground truth pipeline to identify 13 points.

From the existing models, the yolov8n-pose (nano) model was used as it required the least amount of CPU power with performance errors that were repairable in post-processing. Although deploying the model was simple, its broad capabilities necessitated data extraction from its Python object output. The model's output object consisted of six fields where the second field contained a key-point location sub-object and the fourth field contained the names of the type of objects detected whilst the rest of the object were parameters required for live-stream CV detection and classification. After extracting the single-person key points, the data was plotted to validate the labelling process. Ground truth generation was successful for each pose as seen in Figure 4.14.



(a) 5 Limb Star Pose (b) Right Arm Extended Pose (c) Left Arm Extended Pose (d) Arms Raised Pose (e) Knees Bent Freeze Pose

Figure 4.14: The different poses in the dataset

### 4.6.2 Ground Truth Data Integrity

Upon closer inspection of the ground truth dataset, three errors were found: 16% of the dataset had missing feet, 18% were non-sensical reconstructions and 8% had the left wrist missing in the freeze pose. By removing all the non-ideal reconstructions, there would not be enough data for model training. The 18% of non-sensical data was removed however correction algorithms were used on the rest of the data. Firstly, to correct the missing feet, the algorithm assumed the midpoint between the foot and the hip to be the knee. As a result, it solved for the feet since the hips and knees were detected. Although this constrains the spatial relationship for this portion of the data, the correction resulted in visually sensical poses with all the key points present. Secondly, the left wrist placement involved finding a dissection plane and then mirroring the right wrist over to create a left wrist. The dissection plane was found by finding a plane that passed through the midpoint of the feet, hips and shoulders. As a result, the remaining reconstruction samples contained 13 sensical key points.

## 4.7 Machine Learning Implementation

### 4.7.1 Model Architecture

A CNN model designed for pose estimation from RADAR point clouds requires a sophisticated architecture to efficiently extract features and perform regression on the input data. The architecture consists of multiple convolutional blocks for hierarchical feature extraction, followed by fully connected layers that map these features to the ground truth coordinates. To accommodate sparsity and inconsistent point distribution, the model incorporates several optimization techniques. TensorFlow was utilized to develop the network due to its extensive range of tools and the support of a large community, which facilitated efficient development and enhanced model performance.

The model begins with an input layer accepting data of shape (8, 8, 3). The architecture was tested using different numbers of convolution layers. When using one layer, the network was unable to capture enough spatial information to reconstruct poses. After adding another layer, the model reconstructed poses however its predicted differed to the ground truth. Finally, by incorporating three convolution layers, the model was able obtain all the details ranging from broad global patterns to local spatial relationships. Large filters were used as they were crucial in low resolution feature maps. Furthermore, they are capable of maximizing the amount of relationships determined in the early stages of the network. Batch normalization ensured stable learning and an accelerated convergence since the poses were taken at different ranges with subtle variations. The network was tested using drop out layers in the initial stages. Although drop outs prevent overfitting, reducing neurons in small neuron applications resulted in a significant prediction error.

Max Pooling layers were set to a pool size of (2×2) to reduce the spatial dimensions of the feature maps. These layers prevented overfitting to the ground truth which is important as there were subtle variations between samples in the pose set. This ensures that the model reconstructed based on the input point cloud rather than overfitting to the general ground truth structure. Max pooling prevents the network from being vulnerable to pose variations reduces the chances of poor MPJPE at a high classification accuracy.

The Flatten layer integrated the key point features extracted by the convolutional layers which were used in the regression task. The model includes two fully connected layers with 512 and 256 units respectively. The substantial number of units allows the network to model intricate relationships between the training data and ground truth. L2 regularization was applied to these layers with a coefficient of 0.001 to set the network to learn more generalizable patterns. Dropout layers with rates of 0.5 were incorporated after each fully connected layer which are used a an additive generalization technique.

This model used the ReLU activation function after each convolutional and dense layer. It allowed the network to introduce non-linearity, essential for pose estimation tasks. ReLU is computationally efficient due to its simple mathematical operation. ReLU also addresses the vanishing gradient problem often encountered with other activation functions which prevents the network from learning. The output uses a linear activation function as the model attempts to predict continuous key point values that should be unrestricted.

The final output layer returns 39 outputs which correlates to 3 values per key point creating a flattened coordinate system. The model is compiled using the Adam optimizer with a learning rate of 0.0005. The loss function used is MAE, suitable for regression tasks and specifically for pose estimation. Additional metrics like MPJPE and PJE were used for a deeper evaluation allowing for a nuanced assessment. Figure 5.4 summarizes the networks architecture.

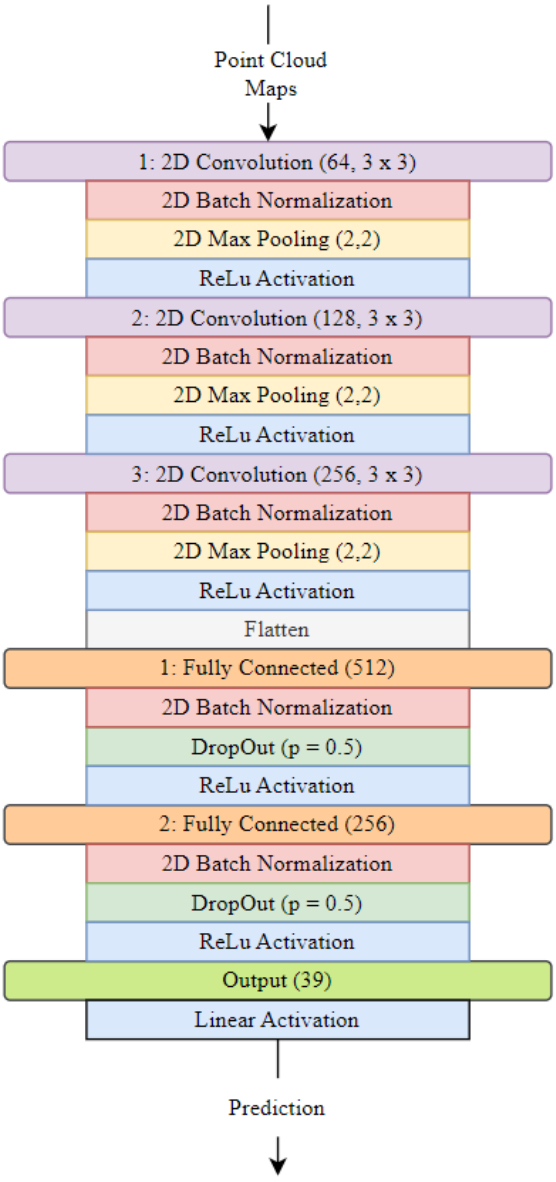



Figure 4.15: Diagram of the CNN architecture

# Chapter 5

# Results and Analysis

## 5.1 Signal Processing Results

The final pose reconstruction accuracy primarily depends on the CNN's ability to find spatial relationships. However, quality training data reduces the dependency on the networks architecture. As a result, this section analyses the processing chain's point cloud output.

### 5.1.1 Star Pose Cloud

This point cloud contained a high density of points in the torso region. This is due to the torso having the highest RCS. Arms and legs have lower RCS due to their smaller surface areas. As a result, there are fewer detected points. As seen in Figure 5.1 the arms visibly have a higher point density than the legs. This stems from the poor elevation resolution. Although the MVDR aglorithm estimates ten angles in the vertical plane, only two angles were received from distinct directions (due to the hardware's limitation). The arms were unaffected sharing elevation angles with the torso as they are in the same vertical vicinity. Only the upper leg was reconstructed as it shares similar angles with the torso.

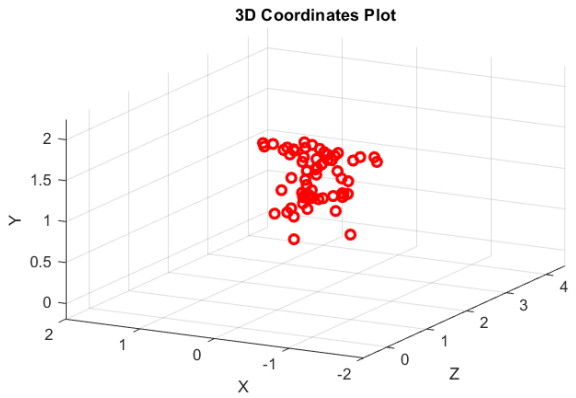



Figure 5.1: Point cloud from a Star Pose recording

### 5.1.2 Arms Raised Cloud

In Figure 5.2, the arm reconstruction becomes distorted. This is different to the star pose as the arms raised enter a different elevation region to the torso. The torso density is greater than the star pose as a result of the ambiguous arm points drifting into that region.

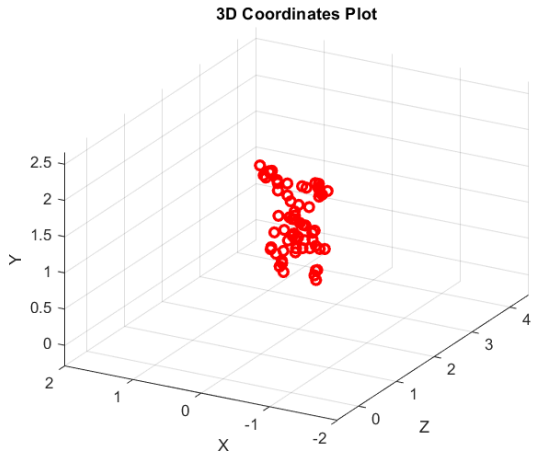
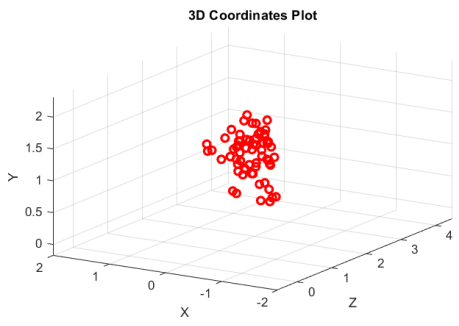


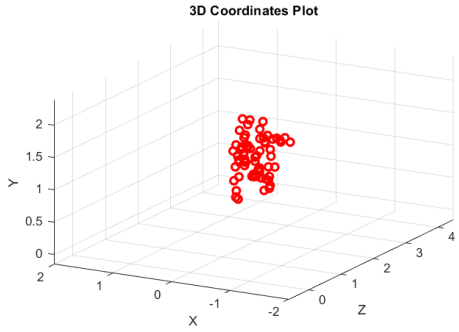

Figure 5.2: Point cloud from an Arms Raised Pose recording

### 5.1.3 Left/Right Arm Extended Cloud

In Figure 5.3, the extended arm in both point clouds can be seen. This is alike the star pose where the arm shares an elevation angle. The resting arm is undetected as it blends into the point cloud. The missing limb should not result in complications since the important feature was captured. Noise in the rest arm detection can cause ambiguity as the noise can be mistaken for a poorly detected raised arm. This crosses over into the star pose's detection set.




(a) Point cloud from a Left Arm Extended Pose recording




(b) Point cloud from a Right Arm Extended Pose recording

Figure 5.3: Left/Right Arm Point Cloud Similarity

### 5.1.4 Freeze Pose Cloud

The freeze pose resembles the star pose however its distinction is a higher density in the arms. The pipeline was not able to capture a high level representation of the crossed arms due to the small vertical seperation between the hand and arm. In theory, the freeze pose should be further distinguishable from the star pose by the seperation distance of the legs. This did not translate to the output point clouds since the legs were poorly reconstructed.

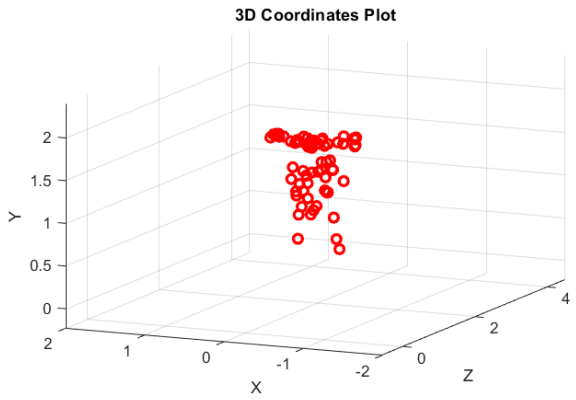



Figure 5.4: Point cloud from a Freeze Pose recording

## 5.2 Training, Validation and Test Performance

This section details the outcomes of the training phase analysing the effect of varying the model's parameters. Thereafter, the validation results, due to training changes, are discussed. Properly splitting data is essential to ensure the model generalizes well and performs effectively on unseen data.

### 5.2.1 Data Split

The input data was tested using three splits which can be seen in Table 5.1. In the first split, there was not enough training data. This resulted in the model being unable to understand the relationships the point cloud and the key point coordinates. The last split minimized training loss due to the large amount of data accessed. However, the model overfit to the training data which caused heavy reliance on the learnt information. The second split ensured that the model was trained sufficiently to minimize validation loss. As a result, the second split was used for optimizing the model.

| Split (Train:Val:Test) | Train Loss (m) | Val Loss (m) | Train MAE (m) | Val MAE(m) |
|---|---|---|---|---|
| 40:30:30 | 0.723 | 0.849 | 0.376 | 0.452 |
| 60:20:20 | 0.469 | 0.514 | 0.330 | 0.377 |
| 80:10:10 | 0.314 | 0.646 | 0.307 | 0.410 |

Table 5.1: Table showing the models results using different data splits (MSE used as the loss function)

K-Fold Cross Validation was also to test the robustness of the model. The dataset was split into five folds for validation. Performing K-Fold Cross Validation showed that the MPJPE and MAE fell within one standard deviation of the other folds. The consistent performance across different folds indicates that the model was able to generalize well to data variations. This also shows that the CNN does not overfit to a specific dataset.

### 5.2.2 Overall Performance

Figure 5.10 illustrates the CNN's performance. The error function reduces to 60% of its initial value after 20 epochs which indicates that the model was able to extract the broad and abstract spatial relationships. This is attributed to the the filters across three convolution layers. In conjunction

ADAM optimizer accelerated the convergence using its adaptive gradient adjustment which ensured quick arrival at the minimum point.

The figure also shows the steady error improvement between 60 and 120 epochs. By setting a learning rate reduction in the linear region, the model was able to make gradual weight adjustments until the minimum was reached. At around 120 epochs, the network reached its plateu point. The early convergence to this plateau demonstrates that the model's architecture is well-optimized for the task without unnecessary complexity.

With regards to the plateau, it can be see that the validation curve is in a close range to the trainings loss. This indicates the the network was not overfitting to the training data. The model prevented this by setting an early stopping call back which ends the learning when the training loss plateaus and the validation loss is at risk of increasing. The constant dropouts after the convolution layers prevented neuron dependancies which cause overfitting.

The curve is smooth compared to the model without batch normalization and drop outs. By ensuring a consistent statistical distribution throughout the layers, the network was able to update the parameters smoothly. The batch size of 128 further smoothed the plot as it captured a large enough sample for analysis.

The last key observation involves the low MPJPE validation loss. This behaviour could be attributed to the use of batch normalization. Since batch normalization is exclusively used in training, it introduces differences between training and validation. The difference arises from the computation base of varying statistics [50]. Conversely, validation uses a moving average which results in lower losses.

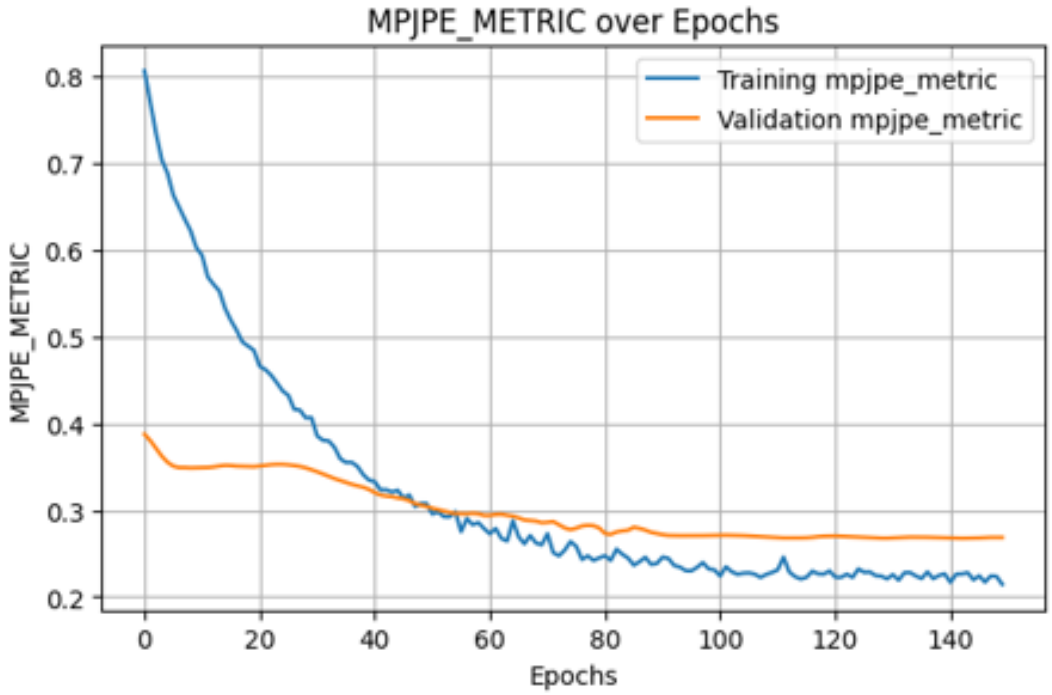



Figure 5.5: Figure showing the decay of the Mean Per Joint Position Error over epochs

## 5.3 Pose Performance

This section details the network's generalization ability across each pose. By reviewing the key point specific errors, shortcomings were found in the CNN. Tables 5.2 and 5.3 provide the Per Joint Error (PJE) required to assess each pose performance. The key to the table is as follows: 1: Head 2: Right Shoulder 3: Left Shoulder 4: Right Elbow 5: Left Elbow 6: Right Wrist 7: Left Wrist 8: Right Hip 9: Leftt Hip 10: Right Knee 11: Left Knee 12: Right Ankle 13: Left Ankle

| **Pose** | **1 (cm)** | **2 (cm)** | **3 (cm)** | **4 (cm)** | **5 (cm)** | **6 (cm)** | **7 (cm)** |
|---|---|---|---|---|---|---|---|
| Star | 11.8 | 13.6 | 12.4 | 15.6 | 20.0 | 22.4 | 24.4 |
| Left Arm Extension | 14.5 | 13.0 | 16.2 | 19.2 | 25.4 | 20.2 | 26.8 |
| Right Arm Extension | 19.8 | 32.6 | 37.1 | 51.1 | 32.9 | 108.2 | 48.4 |
| Arms Raised | 12.9 | 12.6 | 14.1 | 16.0 | 23.9 | 12.4 | 14.5 |
| Freeze | 37.6 | 42.4 | 45.9 | 45.1 | 54.4 | 91.1 | 64.1 |
| Average | 19.32 | 22.8 | 25.1 | 29.4 | 31.3 | 30.56 | 35.6 |

Table 5.2: Table presenting the Per Joint Error for the first seven key points

| **Pose** | **8 (cm)** | **9 (cm)** | **10 (cm)** | **11 (cm)** | **12 (cm)** | **13 (cm)** |
|---|---|---|---|---|---|---|
| Star | 12.6 | 12.5 | 14.1 | 12.9 | 23.93 | 21.3 |
| Left Arm Extension | 12.6 | 13.5 | 12.1 | 12.9 | 15.9 | 16.4 |
| Right Arm Extension | 41.3 | 43.5 | 39.1 | 51.0 | 41.8 | 55.7 |
| Arms Raised | 13.3 | 14.1 | 14.9 | 15.6 | 21.6 | 22.6 |
| Freeze Pose | 37.6 | 34.08 | 39.42 | 41.6 | 49.5 | 51.2 |
| Average | 23.5 | 23.5 | 23.94 | 26.8 | 30.54 | 33.4 |

Table 5.3: Table presenting the Per Joint Error for key points eight to thirteen

### 5.3.1 Star Pose

The network was able to generalize best to the star pose. This can be seen in Tables 5.2 and 5.3 as it has the lowest PJE across all joints. Due to spread of each limb, the star pose suffered the least from poor angular resolution. As a result, there were several points outside the torso region which the network used to perform a full localization. Although the star pose in Figure 5.6 presents a sufficient reconstruction, samples that contained limbs which deviated from an ideal star pose were not mapped correctly. This indicates that the model overfits to the shape of the predicted pose and mapped to the general spatial area which the raw data was contained in.

### 5.3.2 Left Arm Extension Pose

Left Arm extensions were also sufficiently reconstructed as it returned a low MPJPE. However, a small number of samples were ambiguous with the star pose since torso points overlapped into the right arm

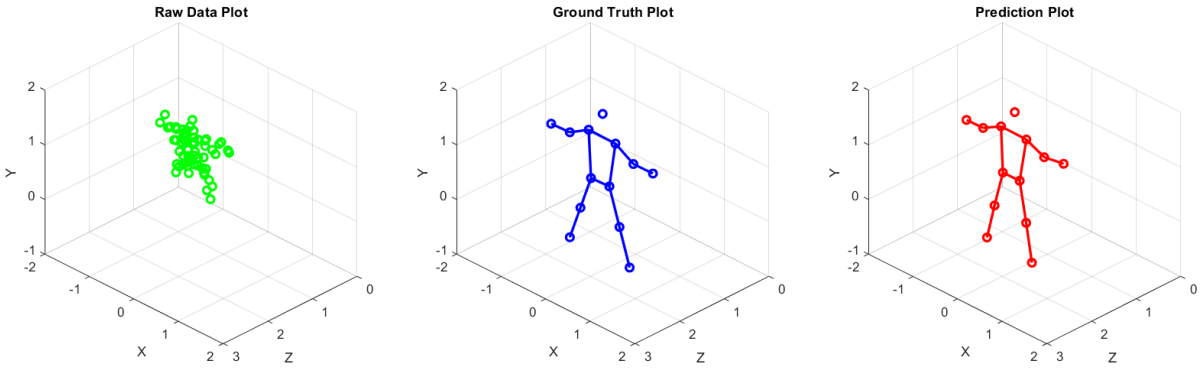



Figure 5.6: Comparison of the network's reconstruction of a Star Pose

position. This is due to sparsity which is vulnerable to noise and randomness in the RADAR's signal capturing. The ambiguous reconstruction can be seen in Appendix E.1.

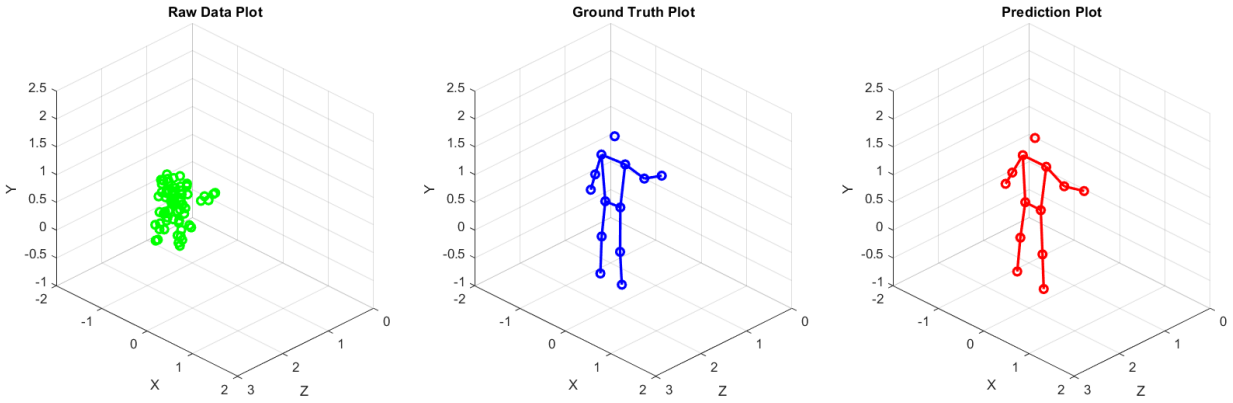



Figure 5.7: Comparison of the network's reconstruction of a Left Arm Extension Pose

### 5.3.3 Right Arm Extension Pose

The model performed poorly for the right arm extension reconstruction. The PJE results show large errors across all limbs where the PJE increased as the joint deviated from the pose's center. Out of the total 175 pose samples, the CNN was only able to accruately reconstruct 38 outputs. The incorrect reconstructions overlapped with the star pose's and the left arm extension's basic structure. This is due the RADAR point clouds' inability to capture enough points corresponding to limbs. As a result, the model relied on a small number of points to determine the output's structure. The ambiguous reconstruction can be seen in Appendix E.2.

The model performed competently for the left arm extension which suggests that it should have performed in accordance for the right arm version. Since the left arm samples were trained first, the right arm's PJE suggests that the model overfit to the previous training set and was unable to

determine the key features of the right arm's point cloud.

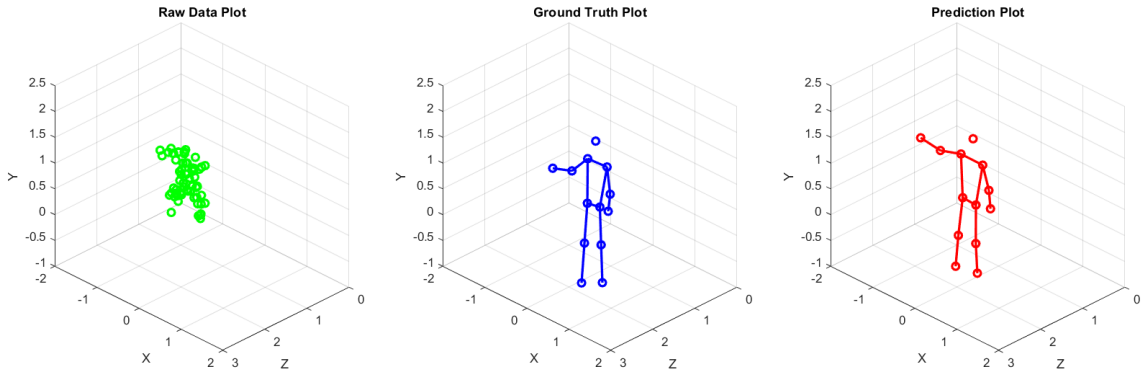



Figure 5.8: Comparison of the network's reconstruction of a Right Arm Extension Pose

### 5.3.4 Arms Raised Pose

The vertical variation in this pose allowed the network to distinguish it from the other poses. This resulted it achieving a small PJE. The wrist position served a consistent problem area across the other poses due to its low RCS and the point clouds sparsity. However, the network was able to accurately localized the wrist position due the this pose's spatial layout. Although, its spatial uniqueness allowed successful reconstruction, the CNN was not able to reconstruct precise arm positions. It consistently reconstructed the average arm position across the entire pose set with a small shift influenced by the cloud data.

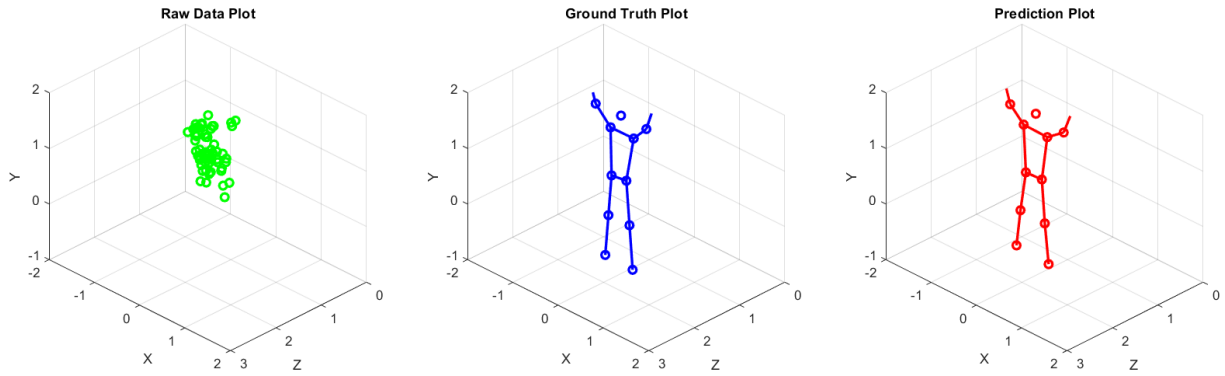



Figure 5.9: Comparison of the network's reconstruction of a Raised Arm Pose

### 5.3.5 Freeze Pose

The ML model was incapable of reconstructing the freeze position due to its overlap of key features with the star and arms raised pose. The high density of points relating to the upper limbs were not sufficiently displaced in the vertical axis. In addition to the overlap, the freeze pose set was reduced to 90 samples as a result of large number of missing key-points from the ground truth generation. The

reduced pose set prevented the model from engaging with this set enough to perform localization. The poor arm localization can be seen Appendix E.3.

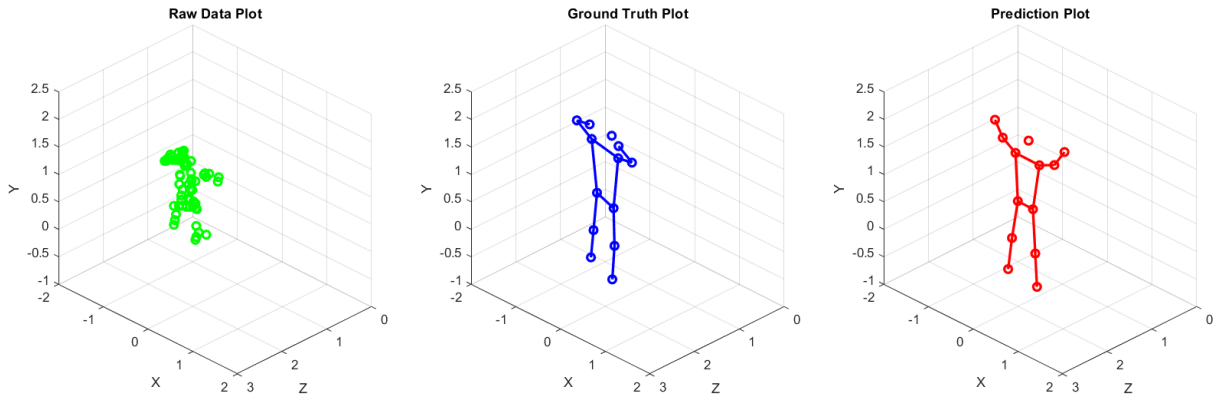



Figure 5.10: Comparison of the network's reconstruction of a Freeze Pose

## 5.4 Performance on Unseen Data

### 5.4.1 External Data Validation

The *MARS* dataset, made publicly available by Sizhe An et al. was used to test the networks integrity to data captured by external systems. It can be found on the GitHub repositoryhere. This dataset consists of ten times more samples than used in this study. The data is in the same basic format with additional intensity and Doppler dimensions associated with the coordinates. Furthermore, the *MARS* dataset contains high-quality input data which ensures that the model's true performance can be observed. This provides a framework to tune the model without non-ideal data features skewing the results.

The model, without any adjustments, was able to achieve a MAE of 5.5cm and a MPJPE of 11.1cm. The model returned errors that were half the size that were produced on the data captured in this study. In addition, the model reduced the error by 90% after 5 epochs and converged at 30 epochs. This indicates that the architecture is capable of extracting features quickly. As a result, major improvement in the loss functions requires better input data. The network's performance on the *MARS* dataset can be seen in Appendix E.4.

### 5.4.2 Unique Pose Performance

Three unseen poses were used to test the network's performance. One of which was a variation of the star pose with one leg raised. The remaining two poses contained no overlap with the existing poses. The model was unable to accurately reconstruct these poses. Instead the model reconstructed three star poses as the star set consisted of the most samples. This shows that the architecture requires additional structures to adapt to unseen poses.

## 5.5 Related Work Comparison

This section presents a comparison of the proposed system's results with the related works. By benchmarking against established models, the potential of the system can be evaluated for further development. The evaluation involves comparing the network architecture against the state-of-the-art solutions using MAE and MPJPE.

As illustrated by Table 5.4, the proposed model was unable to improve the benchmarks set by the state of the art solution. Considering the dataset sizes, all of the related works training their respective architectures using datasets substantially bigger than used in this study. By increasing the amount of samples per pose, the network is able to localize better. This was proven by training the model using *MARS* open dataset discussed in Section 5.4.1.

In addition to larger datasets, *mars* dataset included intensity and Doppler information compared to this study using only x, y, z coordinates. Increasing the amount of information provides the network with more resources to learn from to distinguish between poses. Furthermore, *SUPER* and *mm-Pose* used two radar modules which improved their elevation resolution. As a result, their point clouds returned more information about limbs which simplifies localization.

Lastly, *SUPER* and *mm-Pose* used a sequence of networks tasked with individual components of localization. The increased complexity ensured that at each stage, the data's quality was optimized for learning. Although the CNN used provides adequate localization, the sequences of LSTMs and RNNs ensure better adaption to unseen data.

| **Research** | **Num. Frames** | **MAE (cm)** | **MPJPE (cm)** |
|---|---|---|---|
| *MARS* [10] | 40 000 | 6.23 | N/A |
| *SUPER* [14] | 40 000 | N/A | 12.2 |
| *mm-Pose* [1] | 32 000 | 4.47 | N/A |
| Proposed System | 6 750 | 11.4 | 21.1 |

Table 5.4: Table showing the models results using different data splits (MSE used as the loss function)

## 5.6 Results Summary

The RADAR processing pipeline was able to achieve the objectives set in Section 4.1.2. By achieving a respectable MAE and MPJPE, the pipeline-network combination proved that it was able to successfully reconstruct poses. Although the system is split between Python and MATLAB, the pipeline developed in MATLAB is easily transferrable to Python. This study performed a drastic reduction from fifteen minutes per sample to one minute per sample. This objective requires improvement before it can be adapted to practical applications.

# Chapter 6

# Conclusion

The objective of this study was to determine the viability of performing HPE using mmWave FMCW RADAR. By implementing the necessary theory and literature, a successful HPE processing pipeline was establish.

Prior to the system's development, a comprehensive literature review and theoretical background was completed. The literature review examined the history of HPE developments followed by a critical evaluation of the approaches taken by each of the relevant studies. The work completed in this study was heavily influenced by Sengupta et al. (2020) [1] and Sizhe An et al (2021) [48]. Sengupta et al. (2020) [1] provided the basis for RADAR signal processing whereas Sizhe An. et al introduced a novel CNN grid-based network for localization.

Chapter 3 provided the foundational understanding into required to develop the system. It commenced by introducing fundamental RADAR concepts followed by RADAR processing techniques. Thereafter, the theory behind camera coordinate mapping was detailed for ground truth generation. The chapter concluded by discussing the theory behind CNN architecture and its optimization.

The key to this projects success was improving the quality of RADAR point clouds. In the first stage of the study, radar configurations were chosen to maximize the number of target detections relating to human body parts. This was done by using a configuration that set the number of samples withing a chirp to 142. This reduced the range resolution to 5cm. Although the RADAR was capable of a more samples, the tradeoff between range resolution and data size was not worth the change.

Prior to feature extraction, RDMs were used to validate the systems performance using controlled experiments. After system validation, the dataset was established consisting of 750 samples of five poses. CA-CFAR was employed as the target recognition algorithm due to its integration simplicity and computational efficiency. A significant point-cloud quality factor was the processing chain's ability to estimate angles. As a result, an extensive process was conducted to select the optimal angle estimation algorithm. Due to the limited amount of elevation virtual antennas, the system suffered from poor elevation resolution. As a result, angle estimation techniques that relied on subspace estimation such as channel FFT, MUSIC and phase inteferometry were not suited for this investigation. MVDR proved to be the most effective approach as it performed spatial sweeping. Although computationally expensive, the availability of a high performance PC made it possible to use MVDR in both elevation and azimuth.

Thereafter, a coordinate transformation was computed which generated a 3D abstraction of the target information. Although the point clouds for different poses were distinguishable, they were unable to provide a visual representation of the ground truth. As a result, ML was chosen as the framework to transform the point cloud into a skeleton reconstruction. Ground truth raw data was captured using the Intel RealSense camera which provided depth and image information. A pre-trained YOLOv8

model was employed for automatic labeling of the thirteen key points. The pipeline was concluded by the implementation of a CNN architecture.

Finally, Chapter 5 provided an in-depth presentation and analysis of the pipeline's output. Each point cloud generated was validation before it was converted to training data for the network. In the end, the model was able to successfully reconstruct four out of the five poses with an overall MPJPE of 21.1cm. Although it was not able to improve the bench marks set by the state-of-the-arts, this study provided a good introduction into HPE using RADAR given the project's constraints; however RADAR-based HPE requires substantial development in its processing time, system setup and unseen pose performance before it can be practically used.

# Chapter 7

# Recommendations

### 7.0.1 Areas of Improvement

Angle estimation is crucial for quality coordinate transformations. This study extracted angles that returned the highest power. Although this is a reasonable metric for extraction, certain body parts have lower power therefore the selection would neglect low power angles relating to key points. Instead, one resolution for this involves a method that determines the peaks in the angle spectrum and dynamically determines the amount of angles to be extracted at each peak.

After angle estimation, point clouds require sorting and filtering before they can be used as training data. In this study, the points were sorted based on intensity and then filtered in the x, y and z dimensions. This ensured that body parts were selected over cutter. In many cases, points outside the torso region were neglected as these limbs have lower RCS. Instead of sorting on intensity, a method that is able to gather points from each segment of the subspace would ensure that points are widely distributed. Ensuring this would provide the ML network with data throughout the body which improves the localization accuracy.

Lastly, instead of using 3 coordinates in the training data, incorporating Doppler and intensity provides the model with more information for learning.

### 7.0.2 Future Work

This iteration of HPE using RADAR provided a good basis for future improvements. In this project, the network was able to successfully reconstruct static poses. Although this is useful, it is imperative to improve the processing chain's computational load in order for it to be deployed for real-time sensing. On the topic of static poses, there is potential for RADAR-based pose estimation from videos. By optimizing the pipeline to perform a frame by frame reconstruction, video reconstruction can be achieved. Lastly, developing meta learning networks for RADAR-based HPE which are able to localize unseen poses remains largely undiscovered.

# Bibliography


[1] A. Sengupta, F. Jin, R. Zhang, and S. Cao, "mm-pose: Real-time human skeletal posture estimation using mmwave radars and cnns," *IEEE Sensors Journal*, vol. 20, no. 17, pp. 10 032–10 044, 2020.

[2] D. Ramanan, D. A. Forsyth, and A. Zisserman, "Strike a pose: Tracking people by finding stylized poses," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 271–278.

[3] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1653–1660.

[4] G. H. Martınez, "Openpose: Whole-body pose estimation," Ph.D. dissertation, Carnegie Mellon University Pittsburgh, PA, USA, 2019.

[5] S. An and U. Y. Ogras, "Fast and scalable human pose estimation using mmwave point cloud," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 889–894.

[6] W. Jiang, H. Xue, C. Miao, S. Wang, S. Lin, C. Tian, S. Murali, H. Hu, Z. Sun, and L. Su, "Towards 3d human pose construction using wifi," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020, pp. 1–14.

[7] M. Zhao, Y. Tian, H. Zhao, M. A. Alsheikh, T. Li, R. Hristov, Z. Kabelac, D. Katabi, and A. Torralba, "Rf-based 3d skeletons," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, 2018, pp. 267–281.

[8] F. Adib, C.-Y. Hsu, H. Mao, D. Katabi, and F. Durand, "Capturing the human figure through a wall," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, pp. 1–13, 2015.

[9] H. Xue, Y. Ju, C. Miao, Y. Wang, S. Wang, A. Zhang, and L. Su, "mmmesh: Towards 3d real-time dynamic human mesh construction using millimeter-wave," in *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, 2021, pp. 269–282.

[10] S. An and U. Y. Ogras, "Mars: mmwave-based assistive rehabilitation system for smart healthcare," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 5s, pp. 1–22, 2021.

[11] A. Sengupta and S. Cao, "mmpose-nlp: A natural language processing approach to precise skeletal pose estimation using mmwave radars," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 11, pp. 8418–8429, 2022.

[12] S.-P. Lee, N. P. Kini, W.-H. Peng, C.-W. Ma, and J.-N. Hwang, "Hupr: A benchmark for human pose estimation using millimeter wave radar," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5715–5724.

[13] G. Li, Z. Zhang, H. Yang, J. Pan, D. Chen, and J. Zhang, "Capturing human pose using mmwave radar," in *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2020, pp. 1–6.

[14] B. Zhang, Z. Zhou, B. Jiang, and R. Zheng, "Super: Seated upper body pose estimation using mmwave radars," in *2024 IEEE/ACM Ninth International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 2024, pp. 181–191.

[15] Y.-H. Ho, J.-H. Cheng, S. Y. Kuan, Z. Jiang, W. Chai, H.-W. Huang, C.-L. Lin, and J.-N. Hwang, "Rt-pose: A 4d radar tensor-based 3d human pose estimation and localization benchmark," *arXiv preprint arXiv:2407.13930*, 2024.

[16] Y. Song, T. Jin, Y. Dai, and X. Zhou, "Efficient through-wall human pose reconstruction using uwb mimo radar," *IEEE Antennas and Wireless Propagation Letters*, vol. 21, no. 3, pp. 571–575, 2021.

[17] A. Jalil, H. Yousaf, and M. I. Baig, "Analysis of cfar techniques," in *2016 13th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, 2016, pp. 654–659.

[18] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments," *Procedia computer science*, vol. 199, pp. 1066–1073, 2022.

[19] M. A. Richards, J. Scheer, W. A. Holm, and W. L. Melvin, "Principles of modern radar," 2010.

[20] S. Rao, "Introduction to mmwave sensing: Fmcw radars," *Texas Instruments (TI) mmWave Training Series*, pp. 1–11, 2017.

[21] M. I. Skolnik, "Radar handbook," 1970.

[22] W. L. Melvin, J. A. Scheer *et al.*, "Principles of modern radar: Volume 3: Radar applications," SciTech Publishing Inc, Tech. Rep., 2013.

[23] H. Guillet de Chatellus, L. Romero Cortés, C. Schnébelin, M. Burla, and J. Azaña, "Reconfigurable photonic generation of broadband chirped waveforms using a single cw laser and low-frequency electronics," *Nature communications*, vol. 9, no. 1, p. 2438, 2018.

[24] M. Litchman, "A guide to sdr and dsp using python." [Online]. Available: https://pysdr.org/content/doa.html

[25] Y. E. Acar, İ. SARITAŞ, and E. Yaldiz, "Comparison of ml algorithms to distinguish between human or human-like targets using the hog features of range-time and range-doppler images in through-the-wall applications," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 30, no. 6, pp. 2086–2096, 2022.

[26] D. A. Shnidman, "Radar detection in clutter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 3, pp. 1056–1067, 2005.

[27] Intel RealSense, "Projection in intel realsense sdk 2.0," https://dev.intelrealsense.com/docs/projection-in-intel-realsense-sdk-20, 2024, accessed: 2024-10-16.

[28] Catalyst Earth, "Principal point in camera calibration," https://catalyst.earth/catalyst-system-files/help/concepts/orthoengine_c/Chapter_45.html, 2024, accessed: 2024-10-16.

[29] Adobe, "Focal length explained: What it is and how it works," https://www.adobe.com/creativecloud/photography/discover/focal-length.html, 2024, accessed: 2024-10-16.

[30] K. O'Shea, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[31] J. Wu, "Introduction to convolutional neural networks," *National Key Lab for Novel Software Technology. Nanjing University. China*, vol. 5, no. 23, p. 495, 2017.

[32] D. Berrar *et al.*, "Cross-validation." 2019.

[33] P. Jacob Murel and E. Kavlakoglu. (2023) Ridge regression. [Online]. Available: https://www.ibm.com/topics/ridge-regression

[34] A. I. Aramendia. (2024) L1 and l2 regularization part 1: A complete guide. [Online]. Available: https://medium.com/@alejandro.itoaramendia/l1-and-l2-regularization-part-1-a-complete-guide-51cf45bb4ade

[35] GeeksforGeeks. (2024) Adam optimizer. [Online]. Available: https://www.geeksforgeeks.org/adam-optimizer/

[36] D. Wei. (2024) Demystifying the adam optimizer in machine learning. [Online]. Available: https://medium.com/@weidagang/demystifying-the-adam-optimizer-in-machine-learning-4401d162cb9e

[37] S. I. Garcia. (2018) L0 norm, l1 norm, l2 norm, l-infinity norm. [Online]. Available: https://montjoile.medium.com/l0-norm-l1-norm-l2-norm-l-infinity-norm-7a7d18a4f40c

[38] S. Mittal, "A survey on optimized implementation of deep learning models on the nvidia jetson platform," *Journal of Systems Architecture*, vol. 97, pp. 428–442, 2019.

[39] "Texas instruments mmwave sensing estimator," https://dev.ti.com/gallery/view/mmwave/mmWaveSensingEstimator/ver/2.4.0/, accessed: 2024-10-15.

[40] T. Instruments, "Dca1000evm + awr1843: Raw data format when using different chirp profiles and mmwave studio," https://e2e.ti.com/support/sensors-group/sensors/f/sensors-forum/1388662/dca1000evm-awr1843-raw-data-format-when-using-different-chirp-profiles-and-mmwave-studio, 2018, accessed: 2024-10-15.

[41] *mmWave Radar Device ADC Raw Data Capture*, 2018, accessed: 2024-10-15. [Online]. Available: https://www.ti.com/lit/an/swra581b/swra581b.pdf

[42] P. Wickramarachi, "Effects of windowing on the spectral content of a signal," *Sound and vibration*, vol. 37, no. 1, pp. 10–13, 2003.

[43] F. D. Enggar, A. M. Muthiah, O. D. Winarko, O. N. Samijayani, and S. Rahmatia, "Performance comparison of various windowing on fmcw radar signal processing," in *2016 International Symposium on Electronics and Smart Devices (ISESD)*. IEEE, 2016, pp. 326–330.

[44] D. Koks, "How to create and manipulate radar range-doppler plots," *Australian Government, Department of Defence, Defence Science and Technology Organisation*, 2014.

[45] Texas Instruments, *xWR1843 Evaluation Module (xWR1843BOOST) Single-Chip mmWave Sensing Solution*, revised ed., May 2020, user's Guide, SPRUIM4B, December 2018 – Revised May 2020.

[46] Lenovo, "Parallelization," 2018, accessed: October 19, 2024. [Online]. Available: https://www.lenovo.com/us/en/glossary/parallelization/#:~:text=Parallelization%20is%20the%20technique%20of,of%20reducing%20overall%20computation%20time

[47] OpenMMLab, "Openmmlab github repository," https://github.com/open-mmlab, 2023, accessed: 2024-10-19.

[48] S. An, "Mars github repository," https://github.com/SizheAn/MARS, 2021, accessed: 2024-10-19.

[49] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[50] S. Abusaleh, "Why my training loss is higher than validation loss: Is the reported loss even accurate?" 2024, accessed: 23-Oct-2024. [Online]. Available: https://siddiqueabusaleh.medium.com/why-my-training-loss-is-higher-than-validation-loss-is-the-reported-loss-even-accurate-8843e14a0756

# Appendix A

# AI Use

## A.1 Commands used in ChatGPT

1. **Grammar:** Check the grammar correctness in this sentence [insert sentece].
2. **Sentence Flow:** Check the flow in this sentence [insert sentece].

# Appendix B

# GA informaion

## B.1 GA table

Table B.1: Table Showing the GAs and How They Were Achieved

| GA | Requirement | Justification |
|---|---|---|
| 1 | Problem Solving | The project focuses on developing a mmWave FMCW RADAR-based processing pipeline for human pose estimation. It is in demand in various industries such as healthcare, gaming, and surveillance. I utilized radar fundamentals along with mathematical and physics first principles to overcome this complex problem. I was able to meet the objectives outlined in the project description. |
| 4 | Investigations, Experiments, and Data Analysis | Experiments such as range and angle detection, RCS tests, field of view validation, and more outlined in the report were conducted to validate the radar's configuration. Furthermore, several investigations were performed in software to compare different angle estimation results. Subsequently, the data was analyzed to validate the design choices. |
| 5 | Use of Engineering Tools | I have used engineering software such as MATLAB and Python. I built a machine learning model using the TensorFlow framework in Python. The radar signal processing was done in MATLAB. Furthermore, I used the AWR1843 Texas Instruments radar module to record data and the DCA1000 FPGA for processing the raw data. |
| 6 | Professional and Technical Communication | The report demonstrates proficient technical communication. I have also showcased professional communication by engaging with my supervisor and master's students through Teams and in the lab. |
| 8 | Individual Work | In my project, I worked independently on tasks like developing signal processing algorithms and analyzing radar data, while validating data outputs with other students. All the experiments were conducted by myself, and the report was written entirely by myself. |
| 9 | Independent Learning Ability | I have self-taught myself the radar theory fundamentals required to complete this project. I also took personal responsibility for my learning, adapting to uncertain challenges by seeking feedback, reflecting on my progress, and recognizing areas for improvement. |

# Appendix C

# Processing Files

## C.1 GitHub

The processing scripts used in the implementation of this project can be found in the following GitHub repository:
Talon Sewnath EEE4022S FMCW Human Pose Estimation Repo

# Appendix D

# TI AWR1843

## D.1 AWR1843 Virtual Antenna Layout

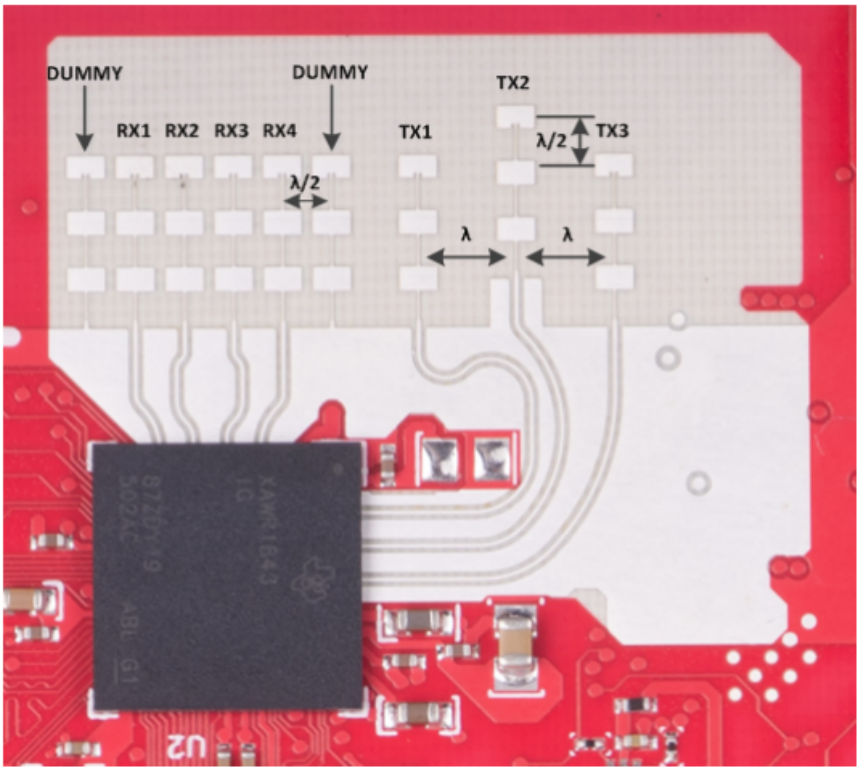
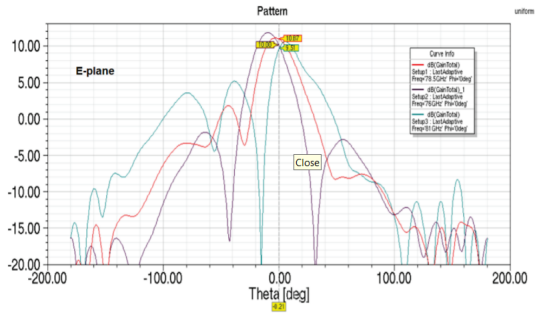
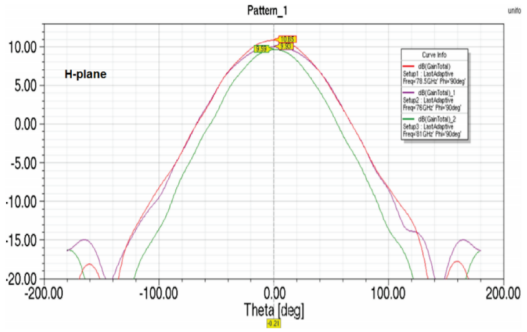



Figure D.1: Image showing the layout of the AWR1843's virtual antennas

## D.2 AWR1843 Beamwidth




(a) Plot showing the azimuth beamwidth across different intensity levels




(b) Plot showing the elevation beamwidth across different intensity levels

Figure D.2: AWR1843 beamwidth for azimuth and elevation

# Appendix E

# Machine Learning Plots

## E.1 Left Arm Ambiguity

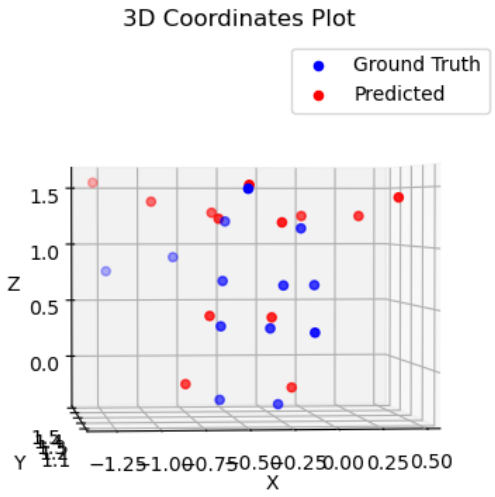



Figure E.1: Image showing the incorrect left arm extension reconstruction

## E.2 Right Arm Ambiguity

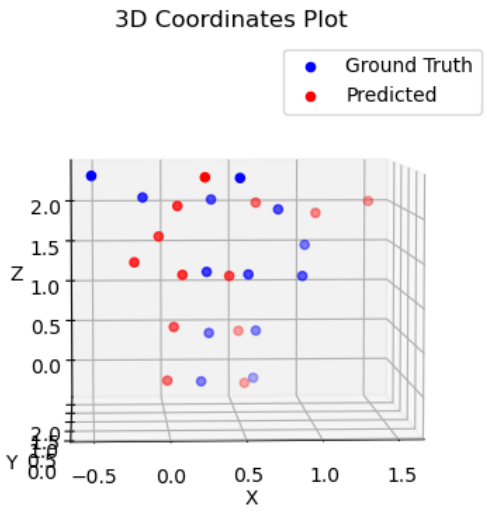



Figure E.2: Image showing the incorrect right arm extension reconstruction

## E.3 Freeze Pose Ambiguity

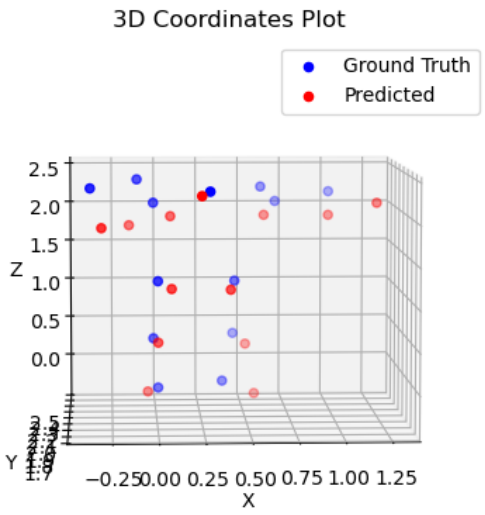



Figure E.3: Image showing the incorrect freeze pose reconstruction

## E.4 MARS dataset performance

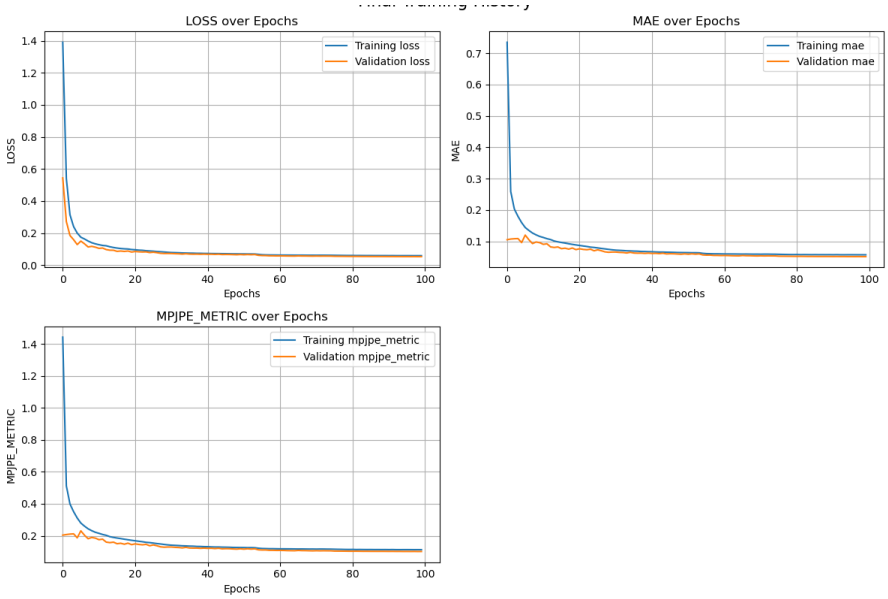



Figure E.4: Plot showing the network's performance on the MARS dataset