Tampere University

MD ROBIUL ISLAM MINTO

HUMAN TRACKING AND ACTIVITIES RECOGNITION WITH MMWAVE RADAR

Faculty of Information Technology and Communication Sciences
Master of Science Thesis
November 2020

ABSTRACT

Md Robiul Islam Minto: Human Tracking and Activity Recognition with mmWave Radar.
Master of Science Thesis
Tampere University
Wireless Communication and RF Systems
Supervisors: Asst. Prof. Bo Tan and Asst. Prof. Taneli Riihonen
November 2020

Millimetre-wave (mmWave) is an extremely valuable sensing technology for the detection of objects and providing the range, velocity, and angle of these objects. A mmWave radar, having synergies with the multi-beam light detection and ranging (LiDAR) and cameras, has been considered as a must-have sensor in the connected and autonomous vehicles (CAV) in the future intelligent transportation systems (ITS). Besides the traditional target detection and ranging functions, the mmWave radar is expected to perform more intelligent tasks to improve road safety, for example recognizing the targets, especially the vulnerable road users like pedestrians and cyclists. mmWave radars are also used in indoor environments due to its high capability of working in low visibility conditions, such as smoke and debris. mmWave radar can provide the exact location of the human presence in the indoor environment with very high accuracy.

The first part of the thesis addresses the radar basics and principles, followed by a detailed discussion on FMCW radar. Also, the first part describes the micro-Doppler (μ-D) in the radar system. The second part of the thesis concentrates on the machine learning basics followed by the detailed discussion on the convolutional neural networks (CNN), recurrent neural networks (RNN).

The third part of the thesis describes a simulation study of the μ-D signatures of the pedestrian and cyclist based on mmWave vehicle radar and investigates the recognition capabilities through both the CNN, RNN and mixed convolutional and recurrent approach respectively. The result demonstrates the usability of the mmWave radar μ-D information and complementary with the video and laser data streams in the CAV auto-piloting. A paper *''Shallow Neural Networks for mmWave Radar Based Recognition of Vulnerable Road Users''* has been published in the IEEE Xplore with this simulation study.

The fourth part of the thesis concentrates on the experimental study of an object (human) detection in the indoor environment by using the Texas Instruments mmWave RADAR module. The experimental study results show the target movement in real-time by azimuth-static heatmap, range-doppler heatmap, and range-profile. The acquired data from the experiments are analyzed and demonstrated in the X-Y scatter plot which gives the analytical view of the target (human) movements.

Keywords: mmWave, RADAR, automated vehicles, micro-Doppler, object recognition, CNN, RNN, IWR6843ISK, Texas Instruments.

PREFACE

All Praise to ALLAH S.W.T the Almighty, for giving me the strength, the blessing, the chance, and patience to complete my master's degree.

This thesis works is a part of the requirement to fulfill my Master of Science degree program in Electrical Engineering at Tampere University.

I would like to express my sincere gratitude to my thesis supervisor Asst. Prof. Bo Tan who had appointed me as a research assistant for this thesis work. I have been extremely fortunate to work under his supervision. His unconditional contributions have strengthened my research work and skills significantly.

I would like to thank the Department of Wireless Communication and RF Systems at Tampere University for supporting me with all the essential facilities.

Finally, I would like to express my love to my family and my wife for their endless love and support throughout my study period.

Tampere, 27 Nov 2020

Md Robiul Islam Minto

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

RADAR	RAdio Detecting And Ranging
CNN	Convolution Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GCA	Grounding Control approach
ATC	Air Traffic Control
MTI	Moving Target Indicator
CWRADAR	Continuous Wave Radar
FM-CWRADAR	Frequency Modulated Continuous Wave RADAR
TI	Texas Instrument
mmWave	milimeter Wave
SAR	Synthetic Aperture RADAR
ISAR	Inverse Synthetic Aperture RADAR
STT	Single Target Tracker
ADT	Detection and Monitoring
TWS	Track While Search
PAT	Phased Array Tracker
RISAT	RADAR Imaging Satellite
SAR	synthetic aperture RADARs
RCS	Radar Cross-Section
RF	Radio Frequency
ADC	Analog-to-Digital Converters
MCU	Microcontrollers
DSP	Digital Signal Processors
CMOS	Metal-Oxide Semiconductor
Synth	Synthesizer
FFT	Fast Fourier transform
AoA	Angle of Arrival
LPF	Low-pass filter
μ −D	Micro-Doppler
SVM	Support Vector Machine
RBF	Radial Ba-sis Function
KNN	k-Nearest Neighbors
PCA	Principal Component Analysis
Kernel PCA	Kernel Methods – KPCA
LDA	Linear Discriminant Analysis
GDA	Generalized Discriminant Analysis
AAM	Active Appearance Methods
LBP	Local Binary Patterns
ASM	Active Shape Methods
NN	Neural Networks
LSTM	Long-Short Term Memory Networks
STFT	Short-time Fourier transform
Ped	Pedestrian
Cyc	Cyclist
2D	Two Dimensional
ReLU	Rectified Linear Unit
HU	Hidden Units
LNA	Low-Noise Amplifier
PA	Power Amplifier

MIMO	Multiple-Input And Multiple-Output
DMA	Direct Memory Access
LVDS	Low-Voltage Differential Signaling
CRC	Cyclic Redundancy Check
SPI	Serial Peripheral Interface
CAN	Controller Area Network
UART	Universal Asynchronous Receiver-Transmitter
PWM	Pulse Width Modulation
PMIC	Power Management Integrated Circuit
QSPI	Queued Serial Peripheral Interface
JTAG	Joint Test Action Group
TLV	Type-Length-Value
EVM	Evaluation Module
SDK	Software Development Kit
GPIO	General-Purpose Input/Output
SOP	Sense of Power
Fps	Frame Rate Per Second
FCL	Fully Connected Layer
MSE	Mean Equared Error
CC	Cross-Entropy
UWB	Ultra-wideband

1. INTRODUCTION

Road safety is one of the essential factors for publicizing the auto-pilot vehicles and intelligent transportation system (ITS). Protection of the vulnerable transportation participants, such as pedestrians and cyclists, draws increasing public concern with the news on 3 accidents caused by self-driving cars [1]. World Health Organization (WHO) reports over half of the 1.35 million deaths in road accidents are vulnerable road users [2]. To reduce fatalities in the accidents, sensor technologies and the intelligent signal/data inferring methods play the key role in the future connected and autonomous vehicles (CAV) [3].

The modern vehicle has integrated with variety of sensors for modelling and interpreting the static and dynamic road environmental factors. Often, we can easily find camera, radar, ultrasound and light detection and ranging (LiDAR) on commodity or trial vehicles. Regarding the environmental modelling, especially detection and recognition of the road users like pedestrians and cyclists, camera and LiDAR are two most discussed sensors. Dash board mounted cameras have been actively used for pedestrian detection and similar research topics in the last decade, due to the available dataset contributions in the community and rapid development of the neural network (NN). As high-resolution sensor, both video and Lidar data are able to capture rich details information for tracking, recognition, even identification of the targets on the road. However, both sensors suffer from the range constraint caused by light condition and dispersion effects due to the high dense particles in raining, fogy or snowing weather conditions. According to the experimental result in [4], radar sensor shows obvious advantage in such situation.

Different kinds of radar are used for human target detection. This has been widely researched over the years. Multiple-input multiple-output ultra-wideband (MIMO-UWB) radar uses short impulse as shown in [5] and has the bandwidth of 500MHz or more which is used for positional estimation for its high resolution and accuracy. The location of the human body is estimated by two or more radar terminals. The distance between the single radar terminal and the target shall be determined. The target location is determined by the distance from each radar terminal and the terminal position. The UWB signal is transmitted from the transmitter with the pulse-repetition frequency of f_{REF}. modulated impulse signal is used as UWB signal. The reflected signal is received by the receiver from the human body. After identification, the analog-to-digital converter

quantizes the signal, and each pulse is integrated. The integrated signal is used to measure the human body's distance. UWB signals are transmitted and received from more than two radar terminals and distances are measured. The position of the human body is then determined from the distances to the position of the radar terminals. Based on the cross-correlation, dual station step frequency continuous wave radar (SFCW radar) is used for target detection, shown in [6]. Cross-correlation procedure is conducted on the pre-processed pulse signals of two SFCW radars at separate positions in order to produce a correlation coefficient matrix. The constant false alarm (CFAR) detection is then used to extract the ranges between each target and the two radars from the correlation matrix, respectively. Finally, human target positions are determined using a triangulation localization algorithm.

Human motion can cause frequency shift of a radar echo signal, and then generate corresponding Doppler signatures. Thus, Doppler can also be used to detect human motion. Chen et al.[7] proposed the concept that an object or any structures on the object may have mechanical vibrations or rotations, called micro-motion dynamics. The micro-Doppler ($\mu - D$) effect is the applied frequency modulation that creates sidebands outside of the main Doppler frequency due to the motion or movement of the target called the $\mu - D$ effect. It is possible to consider a $\mu - D$ signal as a unique signature of a motion triggered by the human body. The $\mu - D$ signatures can identify the target characteristics. In the $\mu - D$, the Doppler effect is more sensitive to the higher frequency band. For millimeter-wave radar, Doppler bandwidth and Doppler precision are better, and it is easier to distinguish the $\mu - D$ signature from various targets. Using doppler frequency, the kinetic properties of humans can be determined [8].

In automotive and mobile applications, Millimeter-wave (mmWave) radar is emerging as an affordable low-power range sensor. In low visibility conditions, such as in the presence of smoke and debris, it can function well, fitting the payloads of resource-constrained robotic platforms, which is why it has been widely studied in indoor environments. MilliMap, a learning-based inductive method is demonstrated [9] for obtaining dense occupancy grid maps from lowcost mmWave radar (AWR1443) sensors, using self-supervision from partial labels from a lidar. Single-chip low-cost mmWave radar as an advanced technology offers an alternative and complementary approach for comprehensive ego-motion estimation in indoor location-based systems, rendering it feasible on resource-constrained platforms thanks to low power consumption and fast device integration. Milli-RIO, is shown in [10], an MMWave radar-based solution making use of a single-chip low-cost radar and inertial measurement unit sensor to

estimate six-degrees-of-freedom ego-motion of a moving radar in the indoor environments.

The mmWave radars working on 24 and 77 GHz are commonly used in the modern vehicles, for example, the long-range radar for active cruise control, short-/medium-range radar for cross-traffic alert, rear collision warning and spot detection. Most of the current applications for vehicles still focus on target detection, ranging and instant velocity estimation functions of radar. In fact, the intelligent functions, such as target recognition, motion classification and high-resolution imaging of modern radars, especially on the mmWave bands, can play more important roles in CAVs.

This thesis concerns the potential of the mmWave radar $\mu - D$ in road user recognition. The $\mu - D$ effect was comprehensively elaborated by Victor Chen in [7].Then, It has been widely researched for human activity recognition[11], vital sign detection[12], [13] in security[14] and healthcare[15]. The Doppler effect has also received attention by the ITS communities and has been widely investigated for collision prevention [16], vehicle classification [17], driver status monitoring [18] and pedestrian characterization [19]. Due to the fact that most current on-vehicle radars are not able to continuously output the Doppler record, the aforementioned works are based on the controlled experimental environments. Then, compared to camera and LiDAR, there is lack of $\mu - D$ data including road users of the real trails to be used for recognition modeling. Then, due to these data availability issues, it is worth using simulation strategies in present studies.

The scope of this thesis includes includes; i) Frequency-Modulated Continuous Wave (FMCW) radar followed by its basic discussion on the range measurement, velocity measurement, and angle estimation; ii) $\mu - D$ effect for target detection; iii) machine learning basics.

Structure of the following thesis are, simulation study for $\mu - D$ recognition, i) the $\mu - D$ simulation method is introduced for simulation of the pedestrian and cyclists data on the road; ii) the simulated dataset is used for recognition modeling using the convolutional and recurrent neural network strategies; iii) the pros and cons of CNN, RNN and mixed convolutional-recurrent approaches for $\mu - D$ signature-based vulnerable road users recognition are compared and discussed; iv) to identify the favorable NN layer for the $\mu - D$ data, only one or two layers shallow CNN, RNN or mixed structures are used. The identified favorable NN layers can be connected for further performance improvement in future work.

An experimental study for target (human) detection in the indoor environments, i) the TI evaluation module (EVM) is used for target detection in a controlled indoor environment;

ii) experimental study has done with TI mmWave demo visualizer; iii) the experimental dataset used to demonstrate the movement of the target; iv) an experimental study has done with the help of MATLAB; and v) these dataset gives the information about the frame number, total target, target number, and the coordinates (x, y, z), doppler (m/s), and intensity of the target which is analyzed and demonstrated.

The rest of the thesis is organized as follows. In chapter 2, background studies related to the thesis has been presented. Chapter 3, continuous with it and explored the machine learning basics followed by the feature extraction, classification, and Neural Networks (NN) is presented. In chapter 4, a simulation study for μ-D signatures recognition is presented. In chapter 5, an experimental study for object (human) detection in the indoor environment by using the Texas Instruments (TI) mmWave RADAR module is demonstrated.

2. FMCW RADAR

In this chapter, background studies related to the thesis have been presented. This chapter consists of two sections. They are (i) Radar basic; (ii) FMCW radar.

2.1 Radar Basics

An abbreviation for *Radio Detecting And Ranging* is the term RADAR. In general, modulated waveforms and directional antennas are used by radar systems to transmit electromagnetic energy to a particular location in space to scan for objects. Objects (targets) will reflect parts of this energy (radar returns or echoes) back to the radar within a search volume. The radar receiver then processes these echoes to extract target information such as velocity, range, angular position, and other target characteristics for detection. Radars can be designed to see through conditions such as fog, snow, rain, darkness, etc which are impervious to normal human vision. In addition, radar has a clear advantage of detection and range measurement and this is probably the most significant attribute it provides [20].

2.1.1 Radar Applications

The high-tech RADAR has broader implementation fields, i.e. Anti-collision aircraft systems, air and ground traffic control systems, detection systems, air defense systems, weather tracking systems, anti-missile systems, microwave astronomy systems, underwater RADARs for submarines, missile guidance systems, remote sensing, geological surveys, height and depth measurements, etc. RADAR has become a high-end surveillance device that focuses on and controls the entire globe. Nowadays, it is getting more and more popular. Radar are also used for indoor human activity recognition[11], vital sign detection[12], [13] in security[14] and healthcare[15]. Radars have been installed on the ground and identify objects on the ground, at sea, in the air, and in space for air traffic control (ATC), aircraft navigation, space, ship navigation and safety, military area, Remote sensing for monitor meteorological patterns in the atmosphere and law enforcement, etc [21].

2.1.2 Radar Operation and Basic Diagram

The fundamental theory on which RADAR works is similar to that of the reflection of sound waves [21]. RADAR uses beams of electromagnetic radiation for target identification and location. In short, its functioning may be summarized as seen below:

- RADAR transmits electromagnetic signals in all directions through the antenna.
- The function of a power amplifier (PA) is to raise the power level of the input signal.
- These radiated waves are intercepted by reflecting objects (targets) and reflected in all directions.
- The receiver in the RADAR system acquires some of the reflected waves.
- The low-noise amplifier (LNA) of the receive path is used to capture and amplify a very low-power, low-voltage signal plus associated random noise that the antenna presents to it within the bandwidth of interest.
- Via digital signal processing and amplification, the received signal is further analyzed, then a determination to determine the presence of the reflected signal from the target is taken at the reception output. If the target is present, it gets its position and other details [21].

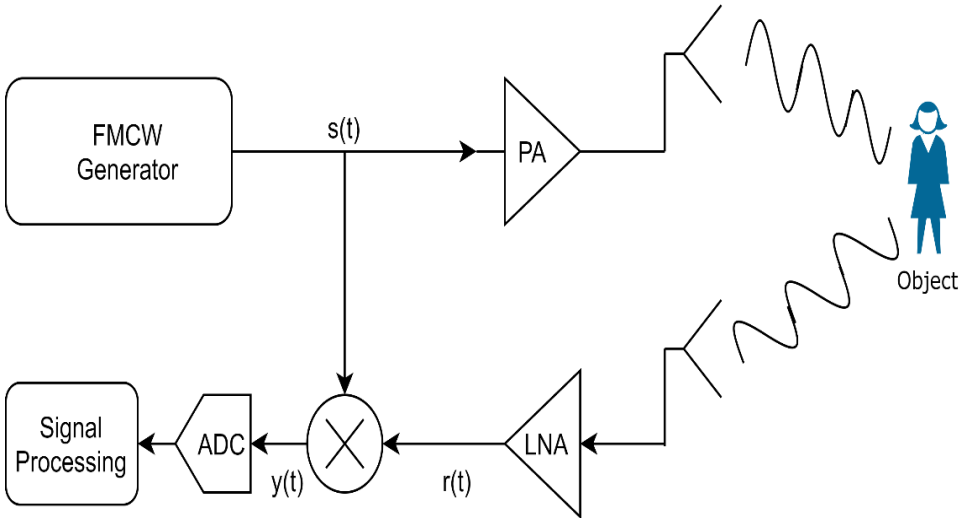


Figure 2.1 *Simplified radar block diagram.*

2.1.3 Radar Frequencies

In general, RADARs run in a frequency spectrum that ranges from 220MHz to 35GHz. Typically, the radar range is measured in the Nautical mile. 1 Nautical mile is equivalent to 1.852Km. Figure 2.2 displays the electromagnetic spectrum reflecting the range of traditional radar.

Millimeter band has extremely high frequency (EHF) designation for the band of radio frequencies in the electromagnetic spectrum from 30 to 300 gigahertz (GHz). Radio waves in this band have wavelengths from ten to one millimeter, so it is also called the millimeter band, and radiation in this band is called millimeter waves. Millimeter-wave bands brought an attractive solution for vehicular communication. These are short-range frequencies that offer high capacity and superfast response times.

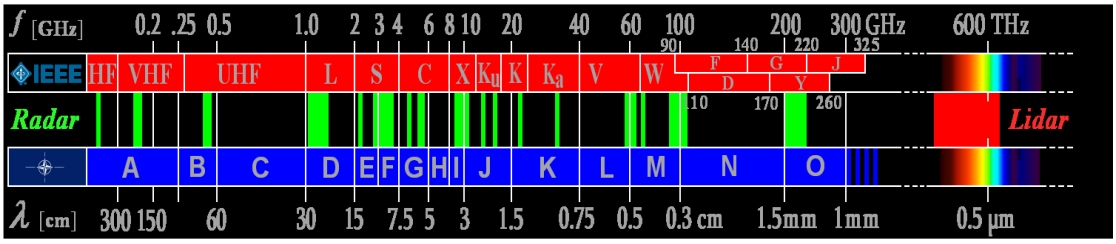


Figure 2.2 *Radar frequencies and electromagnetic spectrum.*

2.1.4 Radar Equation

The radar equation can be written as a,

$$R_{max} = \sqrt[4]{\frac{p_t G_{Rx} G_{Tx} c^2 \sigma N T_r}{f_c^2 \times (4\pi)^3 \times kT \times NF \times SNR_{det}}} \quad (2.1)$$

Where,

$p_t \rightarrow T_x$ output power G_{Rx}

$G_{Tx} \rightarrow R_x$ and T_x antenna gain

$\sigma \rightarrow$ RCS of the object

$N \rightarrow$ Number of chirps

$T_r \rightarrow$ Chirp time

$NF \rightarrow$ Noise figure of the receiver

$SNR_{det} \rightarrow$ Minimum SNR required by the algorithm to detect an object

$k \rightarrow$ Boltzman constant

2.1.5 Radar Cross Section

Radar cross-section (RCS) is a calculation of how a radar target is observable. A greater RCS means that an object is identified more quickly. An object reflects to the source a restricted amount of radar energy. The factors that influence this include:

- The material of which the target is made.
- The size of the target compared to the illuminating radar beam wavelength.
- Target absolute size.
- The angle of the event (angle where the radar beam reaches a certain part of the target, which depends on the form of the target and its direction to the source of the radar).
- The angle of reflection.
- Transmitted and the received radiation polarization with respect to the orientation of the object.

A target's radar cross-section is the (fictional) region the intercepts the amount of electricity. When spread uniformly in both directions, it generates an echo equal to that of the target on the radar.

In other terms,

$$\sigma = \frac{power\ reflected\ toward\ source/unit\ solid\ angle}{incident\ power\ density/4\pi} \tag{2.2}$$

$$\sigma = \lim_{R\to\infty} 4\pi R^2 \left|\frac{E_R}{E_i}\right|^2 \tag{2.3}$$

Where R→Distance between radar and target.
E_R→ Reflected field strength of radar.
E_i →Incident field strength of a target.

2.1.6 Radar Signal Processing

The signal processing is that part of the system which separates targets from the clutter on the basis of Doppler content and amplitude characteristics. Radar returns from each pulse repetition interval (PRI) are stored in memory for further processing.

- **Fast time processing:** Fast time refers to the different time slots composing a PRI, sampling rate dependent. This comparison of a single pulse and its echo to capture time delay when the peak power is detected. Range resolution inversely proportional to pulse (sweep) duration, proportional to bandwidth.

  Match filter (impulse response):

  $h(t) = ax^*(-t)$ $[\cdot]^*$ → Conjugation (in case $x(t)$ is a complex signal) (2.4)

  Output for range bins by filtering received signal:

  $y(t) = \int x_r\, h(t-\tau)d\tau$ (Cross-correlation of T_x and R_x signal) (2.5)

- **Slow time processing:** Slow time updates every PRI. Slow time process multiple pulses and their echoes at one operation. It has benefits of (i) SNR gain of Integration of pulse/sweeps; (ii) Doppler processing; (iii) Radar imaging.

- **Information cube:** Information Cube is an extension to Data Matrix including spatial sampling. In cases that the radar uses multiple receiving channels, the data matrices from each receiver are stacked to form a data cube.

2.1.7 Limiting Factors of the Radar

Some of the limiting factors of a radar are described in [22],

Beam path and range: The line of sight (LOS), the maximum non-ambiguous range, Radar sensitivity, and the power of the return signal as computed in the radar equation can affect the radar performance.

Noise: Signal noise, created by all electronic components, is an intrinsic cause of spontaneous variations in the signal.

Interference: The unintentional man-made electromagnetic interference also affects radar performance.

Clutter: Clutter is an electromagnetic wave that is reflected from unwanted objects from the environments which affect the radar performances.

Jamming: Intentional jamming, in terms of noise or incorrect targets, by an electronic counter measuring device can drop the radar performance.

2.2 FMCW Radar

A special kind of radar sensor that radiates continuous transmitting power like a basic continuous-wave radar (CW-Radar) is the FMCW radar (Frequency-Modulated Continuous Wave Radar). In comparison to this CW radar, the FMCW radar may adjust its operating frequency during the calculation, i.e. the frequency (or phase) modulation of the transmitting signal. Radar measurement possibilities via runtime measurements are only theoretically feasible with this frequency (in-phase) shifts. The drawback of basic CW radar systems without frequency modulation is that it does not determine the target range because it lacks the timing mark required to allow the sensor to precisely time the interval of transmission and reception and translate it into range. Such a time reference may be created by the frequency modulation of the transmitted signal to calculate the distance of stationary artifacts. In this technique, a signal is transmitted which periodically increases or decreases in frequency. When an echo of a signal is received, this frequency transition becomes a delay like the pulse radar technique (by runtime shift). However, in pulse radar, the runtime must be explicitly calculated. In FMCW radar, the variations in phase or frequency between the signal currently sent and the signal received are measured instead [23].

Basic features of frequency modulated continuous wave are,

- Capacity to measure very narrow target ranges (the minimum measured range is equivalent to the wavelength transmitted).
- Ability to calculate the target range and its relative velocity simultaneously.
- Strong high degree of range estimation accuracy.
- Signal processing after blending is carried out at a low-frequency level, making the realization of the processing circuits considerably simpler.
- Protection in the absence of high peak intensity pulse radiation.

2.2.1 FMCW Radar Operation

The FMCW radar is describe from Figure 2.1,

- A chirp is generated by a synthesizer (synth).
- A transmit antenna (Tx ant) transmits the chirp.
- A reflected chirp captured by the receive antenna (Rx ant) is produced by the reflection of the chirp by an object.
- To generate an intermediate frequency (IF) signal, a "mixer" combines the Rx and Tx signals. A frequency mixer is an electronic component that combines two signals to create a new signal with a new frequency.

2.2.2 Chirp

In radar systems, the basic principle is the propagation of an electromagnetic signal reflected in its direction by objects. A signal frequency increases linearly with time on FMCW radars is known as a chirp signal [24] shown in Figure 2.3.

$$x_t = sin(w_t t + \phi_t) \tag{2.6}$$

Where w is the angular frequency and ϕ is the phase of the signal.

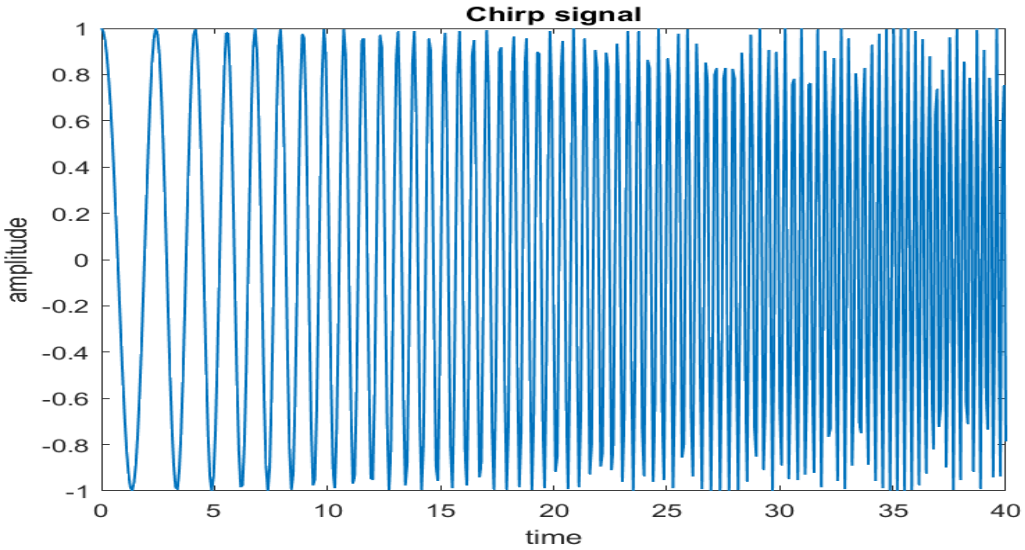


Figure 2.3 *A chirp signal with amplitude as a function of time.*

2.2.3 Range Measurement

To measure range, An FMCW radar system transmits a chirp signal and captures the signals reflected by objects in its path. A signal's frequency increases linearly over time with a slope of S, with frequency f and with total chirp time of T_c. Assuming there is an object in front of the radar at a distance of d, a chirp will reach the object in the time $t = \frac{d}{c}$, since it propagates at the speed of light, c.

The signal is then reflected and arrives back at the radar after the same amount of time t, making the total round-trip time.

$$\tau = \frac{2d}{c} \tag{2.7}$$

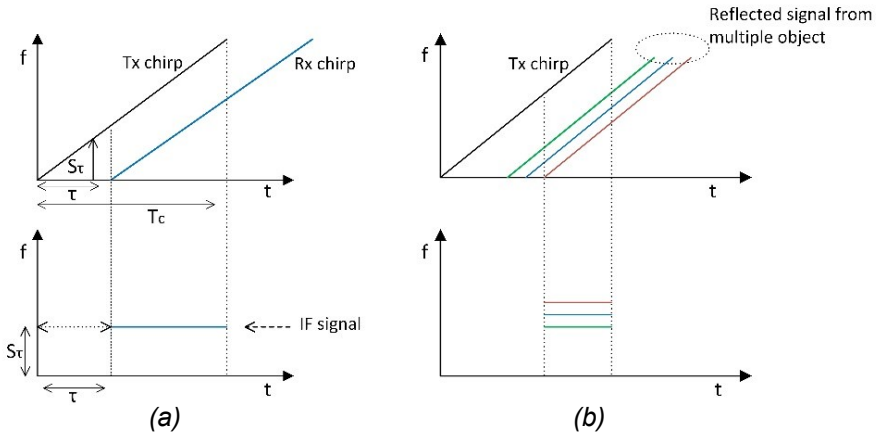


Figure 2.4 *IF frequency is constant (a), Multiple IF tones for multiple-object detection (b).*

At the radar, the reflected signal is received and sent to a mixer along with the originally transmitted signal. A mixer takes two sinusoids and gives the output. Let's say, $x_1 = sin(w_1 t + \phi_1)\ and\ x_2 = sin(w_2 t + \phi_2)$ are these signals. Hence in the mixer,

$$x_{out} = sin[(w_1 - w_2)t\ +\ (\phi_1 - \phi_2)] \tag{2.8}$$

Now, Figure 2.4 (a), for one object in front of the radar, Rx chirp is the delayed version of Tx chirp with round trip delay (τ) and slope (S). The mixer output is the difference of instantaneous frequency of the Tx chirp and Rx chirp as shown in Figure 2.4 (a), this is a straight line. Therefore, frequency of this tone is,

$$S_\tau\ = \frac{S2d}{c} \tag{2.9}$$

Now considering more than one object, radar transmitting a single chirp, and multiple reflected chirps are generated from different objects. Each delayed by a different amount depending on the distance to that object. So, the IF signal will have tones corresponding to each of these reflections. And the frequency of these tones is directly proportional to the range. The mixer output will consist of several different frequencies shown in figure 2.4 (b). Therefore, a range-FFT is applied to the mixer output to identify the different distances of objects but resolution also needs to be considered to identify the amount of space between two objects and still mixer shows the tones of those frequencies. The resolution of the range-FFT is given by

$$d_{res}\ = \frac{c}{2B} \tag{2.10}$$

where B is the bandwidth swept by the chirp *($B\ = ST_c$),* and the maximum unambiguous range is given by,

$$d_{max} = \frac{F_S c}{2s} \tag{2.11}$$

where F_S is the analog to digital converters (ADCs) sampling rate of the radar [24].

2.2.4 Velocity Measurement

To measure the velocity of a target the phase is needed. The phase of the mixer output is determined by the difference in the initial phase of the two signals. An FMCW radar transmits two chirps, separated by T_c, in-order to determine velocity. To detect the object's range (range-FFT), each reflected chirp is processed by FFT. In the same location, but with a different phase, the range-FFT corresponding to each chirp will have peaks. The calculated difference in phase corresponds to a motion inside the vT_c object.

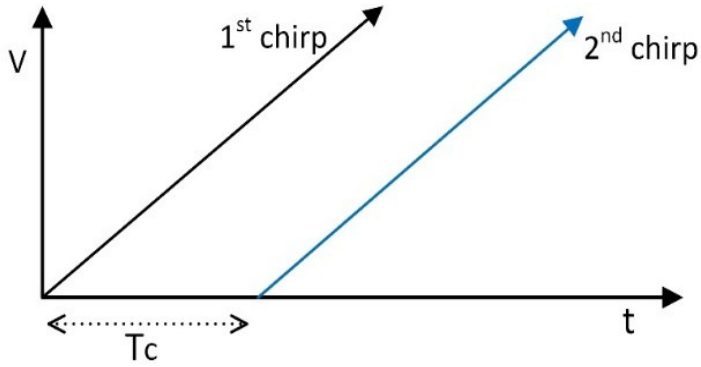


Figure 2.5 *Velocity measurement with two chirps.*

The phase difference is defined by the equation

$$\Delta\phi = 4\pi v T_c/\lambda \tag{2.12}$$

Can be written as,

$$V = \lambda\Delta\phi/4\pi T_c \tag{2.13}$$

There will be ambiguity as velocity based on the phase difference. The measurement is unambiguous only if $[\Delta\phi] < \pi$. Maximum relative speed (V_{max}) measured by two chirps spaced T_c apart, Higher V_{max} required shorter transmission times between two chirps. If several moving objects with differing velocities are at the time of measurement, all at the same distance from the radar, the two-chirp velocity measurement system does not operate. As these objects are at the same distance, with identical IF frequencies, they can produce reflective chirps. As a result, the range-FFT would result in a single peak, which from both of these equi-range objects reflects the cumulative signal. It would not operate for a simple phase contrast method. In this situation, more than two chirps must be emitted by the radar system to determine the speed. It transmits a series of N equally spaced chirps. It's called a chirp frame for this group of chirps [24].

3. MACHINE LEARNING BASICS

In this chapter of machine learning basics, the feature extraction and classification, and Nural Networks (NN) have been presented.

Machine learning is a data analysis method that automates the analytical models. It is a part of artificial intelligence focused on the premise that systems, with minimal human interaction, can learn from data, recognize patterns, and develop decisions. Like statistical models, machine learning is aimed at understanding the structure of the data, fitting theoretical distributions to well-understood data. There is a hypothesis behind the model that is mathematically proven with statistical simulations, but this includes data to satisfy some strong assumptions as well. The machine learning has evolved based on the ability to use algorithms to evaluate data for structure. The test for a machine learning model is not a theoretical test that confirms a null hypothesis, but a validation error on new data. Since machine learning also uses an iterative technique to learn from data, it is simple to automate learning. Up until a robust pattern is found, passes are run through the data. For feature extraction and classification, machine learning approaches can be effectively used and are also applicable to biometric systems. Biometrics deals with people's identification based on physiological and behavioral attributes. Biometric recognition utilizes automated recognition approaches and this is why machine learning is closely linked to it [26] [27].

3.1 Feature Extraction and Classification

Nowadays it is becoming more common to work with a bigger dataset which has hundred of features or even more. If the number of features in a dataset becomes similar or bigger than the number of observations stored in a dataset, then this will most likely lead to overfitting of a Machine Learning algorithm. To avoid this type of problem, dimensionality reduction techniques (Feature Extraction) or regularization can be applied. The dimensionality of a dataset is equal to the number of variables in the machine learning used to represent it. This technique certainly reduces the risk of overfitting but there are also many other advantages, for instance, improved data visualization, accuracy improvements, overfitting risk reduction, speed up in training [28].

For pattern recognition, feature extraction is very important which deals with the transformation of data from original data space to a feature space. Dimensionality is the same for both feature space and data space but the dimensionality of the space of the

feature can be minimized (reduction of dimensionality) when the transformation is selected correctly, and the data can be decorrelated. The methods of data transformation can be linear or non-linear. Linear methods can be of the 2nd order(e.g. Factor Analysis, Principal Component Analysis (PCA), Linear Multilayer Perceptron (MLP), Linear PCA networks,) or of higher-order (e.g. Independent Component Analysis (ICA), Projection Pursuit). Examples of nonlinear methods are Nonlinear PCA applied by nonlinear MLP [27].

3.1.1 Feature Extraction Methods

There are several approaches that the term machine learning encompasses. Examples of them are PCA, Kernel Methods – KPCA (Kernel PCA), Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), Generalized Discriminant Analysis (GDA), Active Appearance Methods (AAM), Local Binary Patterns (LBP), Active Shape Methods (ASM), etc [27].

3.2 Classification

Support Vector Machine (SVM), Random forest, and k-Nearest Neighbors (K-NN) are briefly explained in this section. However, These classifications are not used directly in this thesis, Henceforth, the classifications are described shortly in the section.

3.2.1 Support Vector Machine (SVM)

A linear model for classification and regression problems is SVM or Support Vector Machine. For several practical issues, it can overcome linear and non-linear issues and perform well. The SVM principle is simple: the algorithm produces a line or hyper-plane that divides the data into groups.

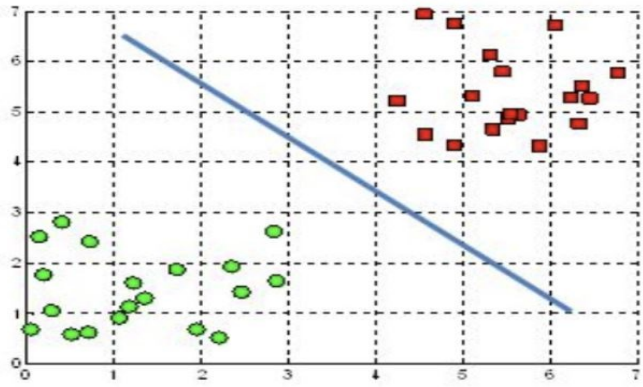


Figure 3.1 *SVM Hyperplane* [30].

At first, SVMs find a separating line (or hyperplane) between data of two classes. SVM is an algorithm that takes the data as an input and outputs a line that separates those classes if possible. The data points between the hyperplane are classified to a certain class (eg. green and red as in Figure 3.1). The difference between the hyperplane and the first point (for all the various classes) on each side of the hyperplane is a measure of certain the algorithm is regarding its classification decision. The bigger the distance and the more certain that SVM is making an accurate decision.

Support Vectors are considered the data points nearest to the hyperplane. Support Vectors evaluate the hyperplane's orientation and position in-order-to optimize the margin of the classifier (and thus the classification score). Depending on the applications, the number of Support Vectors the SVM algorithm can use can be randomly selected. Two main types of SVM classification algorithms are used one is hard margin another is soft margin [29].

3.2.2 Random Forest

A Random Forest is just a bunch of bundled Decision Trees. The basic concept behind a random forest is to combine multiple decision-making trees into a single model. Predictions made by decision trees (or humans) individually may not be correct, but taken together, the predictions would be on average closer to the mark. They can also manage categorical characteristics very well. This algorithm can handle high dimensional spaces as well as a large number of training examples [31] [32].

3.2.3 k-Nearest Neighbors (K-NN)

A supervised learning algorithm used for both regression and classification is k-Nearest Neighbors (KNN). The KNN algorithm does not calculate a predictive model from a training dataset, as in logistic or linear regression, to make a prediction. Indeed, a predictive model need not be developed by KNN. To be able to make a prediction, without any training process, a KNN uses the dataset to generate a result.

The mean (or median) of the y variables of the K nearest observations would be used for predictions if KNN is used for a regression problem. If KNN is used for a classification problem, the mode of the variables y of the K closest observations that will be used for predictions is the mode (the value that occurs most often). It is robust to noisy training data and is effective in the case of a large number of training examples [32] [33].

3.3 Neural Networks

Neural networks are a clear example of methods of deep learning with attractive features. Since the majority of real signals are nonlinear, non-stationary, and non-Gaussian, conventional methods are sub-optimal. With such signals, neural networks can simply function and can produce optimum performance. Neural networks and deep learning currently provide the best solutions to many problems in image recognition, speech recognition, and natural language processing.

3.3.1 Convolutional Neural Network

Convolution Neural Network (CNN) also called feedforward NN is one of the most common deep learning algorithms, a form of commonly used deep learning method in which a model learns to conduct classification tasks directly from images, video, text, or audio. For finding patterns in images to recognize objects, faces, and scenes, CNNs are particularly useful. They learn from image data directly, use patterns to classify images, and eliminate the need for manual extraction of features. This network is usually used for object recognition, object detection, and computer vision such as self-driving vehicles, etc.

CNN Architecture:

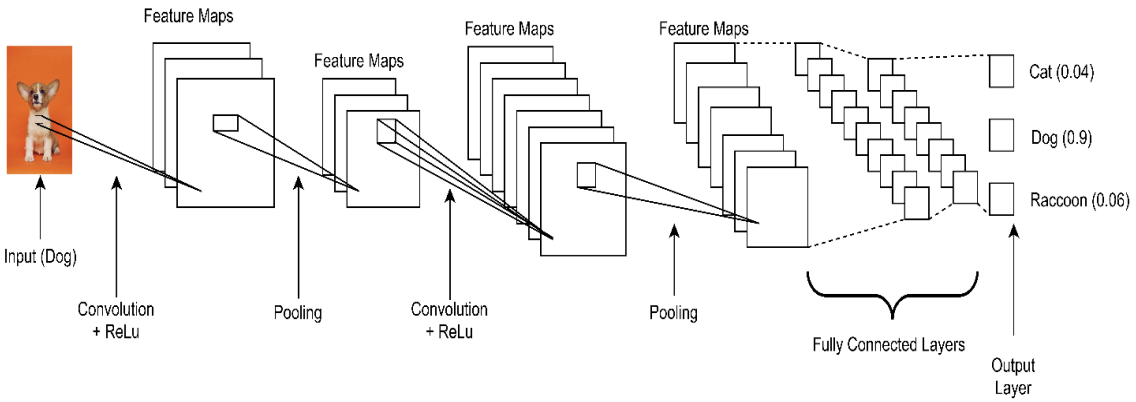


***Figure 3.2** The CNN architecture.*

A convolutional neural network can have lots of layers according to the task and each of the layers learns to detect different features of an image. Filters are applied at different resolutions to each training image, and each convoluted image's output is used as the input to the next layer. The most common layers are convolution, ReLU or activation, and pooling in the CNN.

- **Convolution layers:** Convolution uses a set of convolutional filters to place the input images, each of which activates certain features of the images.

- **Rectified linear unit (ReLU) or activation:** By mapping negative values to zero and maintaining positive values, the rectified linear unit (ReLU) enables faster and more efficient training. This is sometimes referred to as activation because the next layer is carried forward by only the activated features.
- The pooling layer is responsible for the convolved feature's spatial size reduction. This is to reduce the computational power required through dimensionality reduction to process the data. Also, it is useful for extracting rotational and positional invariant dominant characteristics, thus maintaining the model's process of effective training. There are two types of pooling average pooling and max pooling.

These operations are repeated over and over within the layers and with each layer's network learn to identify different features. After learning the features in these layers CNN shifts to classification layers. The next-to-last layer is a fully connected layer that produces a K-dimensional vector, where K is the number of classes that can be predicted by the network. For each class of any image being classified, this vector contains the probabilities. To provide the classification output, the final layer of the CNN architecture utilizes a classification layer such as softmax [34].

Mathematical Operation of CNN:

Considering a CNN with L convolutional layers, a mapping relation $f(X_0;\theta) = \mathbb{R}^{N_0} \rightarrow \mathbb{R}^{N_L}$. $\mathbb{R}^{N_L}$ exist between the input layer and the last convolution layer where $N_0 > N_L$ due to the shared weights in the convolution step and pooling that makes a significant reduction in the number of weights. The l_{th} layer r_l depends on both the output of $(l-1)_{th}$ layer and a set of weights θ_l belonging to $\theta = \{\theta_1, \ldots . \theta_l\}$, denoting the set of all parameters of the L layers of the network. The mathematical operation in a local receptive field of the l_{th} convolution and activation layers can be described as:

$$f_l(X_{l-1};\theta_l) = \delta(Z_l X_{l-1}; b_l) \tag{3.1}$$

where $Z_l \in \mathbb{R}^{K_l K_l}$, with K_l the size of the kernel window, that is applied as a sliding window on the local areas of $X_{l-1} \cdot b_l \in \mathbb{R}^{N_l}$, is the bias in the l_{th} layer and $\delta(\cdot)$ is called the activation function that induces non-linearity into the features. After the L layers of deep extracted features, there are the decision layers, where the similarity of the feature into the classes is quantified. For example, as probabilities for assigning the features into one of the classes.

Usually, the last layers include a fully connected layer (FCL). In order to train the network, to find the optimum weights, a loss function for minimization or performance for

maximization is used. Considering the vector of image labels as Y and the predicted labels as $\hat{Y}$, then the loss function for CNN can be defined as:

$$Loss\ (\theta) = \frac{1}{S} \sum_{i=1}^{S} loss(\hat{y}_i, y_i) \tag{3.2}$$

Where S is the size of the training dataset. The goal of the training process is to find out the values of the parameter set θ that minimizes the $Loss(\theta)$. The metric of the loss function can be mean equared error (MSE) or categorical cross-entropy (CC). The most popular training process method is stochastic gradient descent (SGD) [35]. In the work, the input data sample X_0 for training is the 2D $\mu - D$ signature D from STFT [36].

3.3.2 Recurrent Neural Network

Recurrent Neural Network (RNN) is another deep learning architecture that processes the sequences of the data by using feedback connections. For example, it is used for the analysis of time-series such as speech and video data. The instant Doppler detection in the $\mu - D$ signatures can be considered as a one-time instance of a sequential event. RNN keeps track of temporal dependencies in the input sequences. RNNs may use their internal state (memory) to handle sequences of inputs, unlike feed-forward neural networks. All the inputs are independent of each other in other neural networks, but all the inputs are related to one another in RNN. The problem with RNN is they might have good short-term memory but handling the long or big data this performed badly. Long-Short Term Memory Networks or LSTMs is an RNN variant that addresses the former's long-term memory problem. In this thesis, the LSTM [37] is used for the analysis of the $\mu - D$ signatures as time-series data.

Long Short-Term Memory Networks (LSTM): The LSTM has a more complex cell structure than the regular recurrent neuron which allows better regulate the function of learn and forget. The function in the LSTM is controlled by cell state (cell memory). Basically, the cell state encodes the data of the inputs (relevant info) observed (at every step) up to that point. Another state in the LSTM is the hidden state (cell output) which is an output of the cell denoted by ‘h’. The regular RNN has no cell state often denoted by ‘c’ (‘g’ for cell candidate), therefore RNN suffers for accessing the information a long time ago. The hidden state is used in the LSTM for prediction which contains the information of previous input along with current input [38].

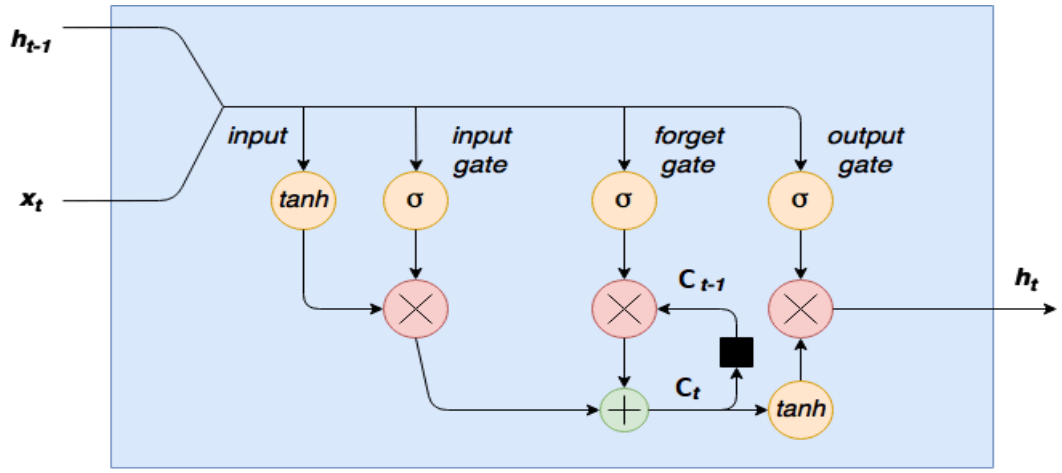


Figure 3.3 *LSTM cell view* [38].

An LSTM has three of these gates, to protect and control the cell state. The input gate, the cell, will determine whether or not to change the cell state. The cell can delete its memory with the forgotten gate, and with the output gate, the cell can determine whether or not to make the output information available.

1. **Input Gate:** Also called the save vector. These gates decide which information should enter in the cell state or long-term memory and which information should save in the cell state or should be forgotten. From the Figure 3.3, the *tanh* activation function squashed the input between -1 to 1. Then this squashed input (from tanh) is multiplied by the output of the input gate elementwise. The input gate is effectively a hidden layer of sigmoid enabled nodes, with weighted x_t and input values h_{t-1}, which outputs values between 0 and 1, and when multiplied by an input decides the inputs are turned on and off.

2. **Forget gate:** The forget gate is sometimes called the remember vector. By multiplying 0 to a position in the matrix, the output of the forget gate informs the cell state what data to forget. If the output of the forget gate is 1, it will hold the information in the cell state.

3. **Output Gate:** The output gate is often referred to as the focus vector. It ultimately highlights, out of all possible values from the matrix (long memory), which information can pass on to the next hidden state [39].

The trainable weights of LSTM layers are Input Weights $W = [W_i, W_f, W_g, W_o]^T$, the Recurrent Weights $R = [R_i, R_f, R_g, R_o]^T$, and the Bias $b = [b_i, b_f, b_g, b_o]^T$,. Assume the input time sequence is X, h_t and c_t are the output (or hidden) state and cell state of the time step t.

The update of cell and output states can be described as follows:

$$c_t = f_t \circ c_{t-1} + i_t \circ g_t \tag{3.3}$$

$$h_t = o_t \circ \delta_c(c_t) \tag{3.4}$$

where o denotes the element-wise product of vectors, δ_c denotes the state activation function. h_o and c_o are initialized to 0. The update of the gates and cell candidate can be described as follows: where $\sum = [\delta_g, \delta_g, \delta_c, \delta_g]^T$ is the activation vector, and δ_g is the gate activation function, and $[.]^T$ is the transposition of a vector. In the RNN context, the input time sequence X for training $\mu - D$ is signature D obtained from STFT. The column vector d_t in D is the time sequence status at time t which corresponds to x_t in [39][40].

3.3.3 CNN-LSTM Network

The convolutional model is good at representing the spatial features and the recurrent model is good at revealing the temporal feature of the input data. Thus, a CNN-LSTM model is developed by connecting the features of the early layers of CNN to the LSTM network. Therefore, the model benefits from both sequential deep spatial features and LSTM temporal features for analysis of the mmWave radar $\mu - D$ signatures of vulnerable road user's movements.

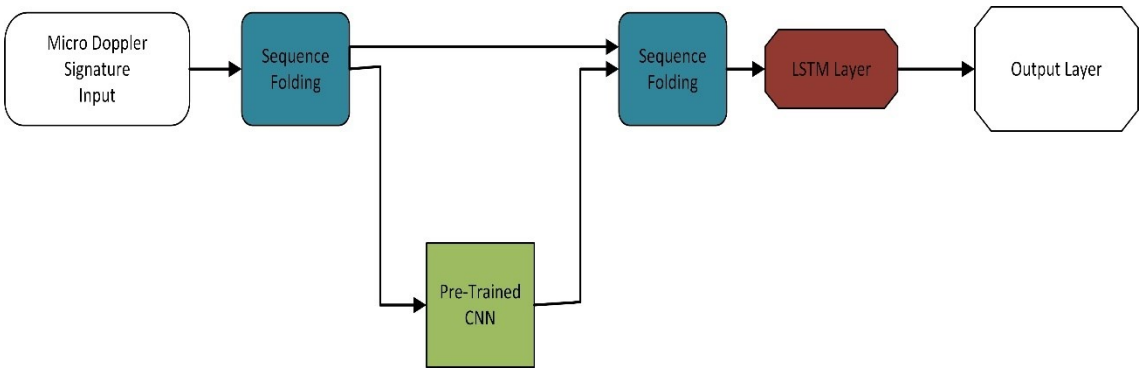


Figure 3.4 *CNN-LSTM model.*

3.3.4 Model Implementation

In this thesis, The CNN, LSTM, and CNN-LSTM models are implemented in Matlab. The Matlab functions are used to train and test the data set. Matlab® pedestrian backscatter signal function and Matlab® backscatter radar signal function are used to generate pedestrian and the cyclist model, respectively, details are described in the next chapter.

4. SIMULATIONS STUDY OF MICRO DOPPLER RECOGNITION

In this chapter, a simulation study for micro doppler recognition has been presented. This chapter consists of five sections (i) micro-doppler signature, (ii) simulation setting and scenario description, (iii) datasets description, (iv) target recognition performance, and (v) identify the favorable NN layer for the $\mu - D$ data.

4.1 Micro Doppler Signature

The $\mu - D$ effect, induced by micromotions (vibration or rotation) of a target or structures on the target, can be derived from the theory of the electromagnetic back-scattering field. In this paper, we used a simplified model to interpret the $\mu - D$ phenomenon. Assume an object is moving with radial velocity v with respect to the radar. The Doppler shift f_D induced by this object can be written as

$$f_D = 2f_c\frac{v}{c} \tag{4.1}$$

where f_c is the carrier frequency, c is the free-space propagation speed. When handling a complex target, which is composed of N moving parts and each moving part has distinguishable time-varying motion status $v_i(t)$. Assume the reflected signal from the i_{th} part can be denoted by $r_i(t) = \alpha_i s(t-\tau_i)e^{j2\pi f_{Di}t}$. f_{Di} is the Doppler shift caused by the i_{th} moving part, α_i and τ_i are the amplitude coefficient the signal delay from the i_{th} part. The reflected signal of the whole target on the receiver side can be considered as the sum of reflections from all parts and the additive white Gaussian noise (AWGN) which is denoted by n in equation (4.2)

$$r(t) = \sum_i r_i(t) + n \tag{4.2}$$

This assumption is more realistic for road users such as pedestrians or cyclists. As shown in Figure 4.1, the Doppler observation of the walking human is caused by the combined effect of the bulk torso rocking and partial rotations of legs and arms. The Doppler shifts of a cyclist can be considered as the result of the bulk human body, bike frame, quick spinning wheels, and rotating legs. The receiving signal $r(t)$ contains multiple Doppler (frequency) components because of the complex target motion status. However, the Doppler shifts can not be directly observed from equation (4.2).

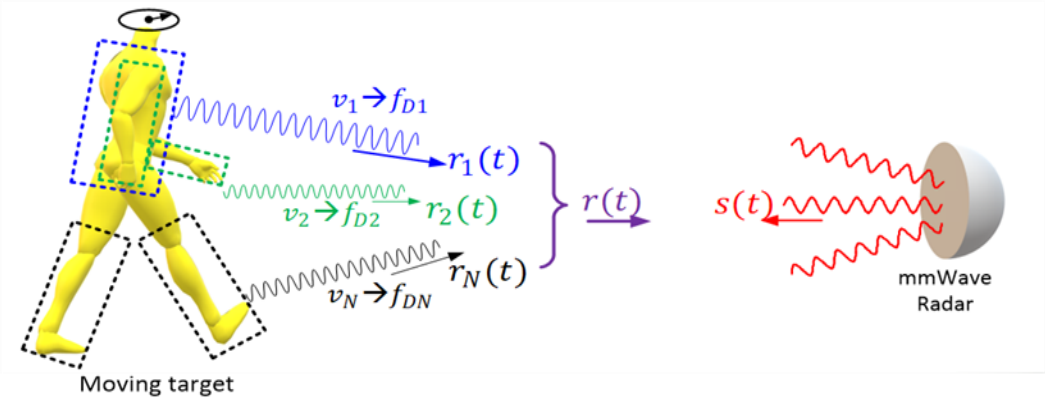


Figure 4.1 *Walking padestrian observed by the radar*

Usually, short-time Fourier transform (STFT) is applied on the radar receiving signal $r(t)$ to visualize the $\mu - D$. The STFT of the receiving signal $r(t)$ is shown as below:

$$STFT\{r(t)\}(\tau, f) = \int_{\propto}^{-\propto} r(t)\omega(t-\tau)e^{j2\pi ft}\,dt \tag{4.3}$$

where $w(\tau)$ is the window function, commonly a window centered around zero (e.g., Hann, Gaussian, or Kaiser window). The STFT of the signal $r(t)$ shows the distribution of energy in the two-dimension (2D) time-frequency space. Depending on the target motion status, this energy distribution varies so that different motions result in different energy distribution patterns. Thus, STFT can be used to recognize different types of road users based on the on-vehicle mmWave radar data. Also, the real STFT pattern is influenced by the orientation, distance, and direction of the target with respect to the radar, as well as the operating frequency of the radar system. The result of the STFT operation is a 2D matrix with complex entries.

In practice, the phase information of the entries are usually ignored, and the normalized logarithmic magnitudes of the entries in the 2D matrix are considered. They are called $\mu - D$ signature and denoted by $\mathcal{D}$. There are two ways to interpret the $\mathcal{D}$. First, it can be treated as an image and convolutional models can be employed to recognize different $\mu - D$ signatures. Second, the $\mathcal{D}$ can be treated as a sequence. Each column of $\mathcal{D}$, d_{ti} , corresponds to a Doppler detection at the time t_i . Then, the $\mathcal{D}$ can be recognized by using recurrent models.

4.2 Simulation Parameter Settings

In this thesis, to generate the radar returns from objects, the parameters were set up as follows;

- **Radar operating parameters:** In this section, the waveform bandwidth, sampling frequency, carrier frequency, and waveform repetition time has been set up.

- **Radar parameter:** Radar position, velocity is defined here. The Matlab® phased platform function is used to define these parameters.
- **Radar waveform setting:** The sample rate, sweep time, sweep bandwidth has been set up.
- **Simulation setup:** The maximum pedestrian and cyclist speed were set up. The oversampling factor and simulation time duration were set up.
- **Signal initialization:** Finally, the signal was initialized and lock the area of interest.

4.3 Simulation Settings and Scenario Description

In this thesis, for generating different $\mu - D$ samples of pedestrians and cyclists a four steps procedure was developed.

- **S1-target scattering modeling:** For the pedestrian object, the stick man model in [40] is used, which is also used by Matlab® pedestrian backscatter signal function [41]. The backscatter signal function produces an entity that simulates signals from a walking pedestrian. The pedestrian walking model coordinates the motion of 16 body parts in order to simulate natural motion. The model also simulates the reflectivity of each portion of the body. From this model, the direction and velocity of each segment and the overall backscattered radiation can be obtained as the body travels. After creating the pedestrian model the Matlab® move function is used to move the pedestrian and Matlab® reflect function is used to get reflection from the pedestrian. There are more moving parts in the cyclist object. The Matlab® backscatter radar signal function [42] is used to generate the cyclist model, The model simulates backscattered radar signals reflected from a moving bicyclist. Both the bicycle and its rider make up the bicyclist. Henceforth, The model also takes the number of spokes, gear transmission rates, and pedaling status into account.

- **S2-trajectory generation:** For both pedestrian and cyclist objects, the initial heading measured in the xy-plane from the x-axis towards y-axis in units of degrees, walking speed is set to 1.4 times of the pedestrian height set in the Height property [Initial position, Heading, Speed] is defined to describe the trajectory. The detailed distribution of these parameters are in Table 4.1.

- **S3-propagation modeling:** Free space propagation models are used in both pedestrian and cyclist simulation. Signal propagation speed, signal carrier frequency, and sample rate are the main property of the free space propagation

is used. The two-way propagation was set to perform round-trip propagation between the source and destination.

- **S4-$\mu - D$ signature generation:** The backscattered signal from either pedestrian or cyclist are fed to STFT to obtain the $\mu - D$ signature for recognition. The STFT parameters are listed in Table 4.1. The setting of the simulation in this work are shown in Table 4.1.

Table 4.1 $\mu - D$ signature generation settings of the pedestrian and cyclist objects.

	Pedestrian		**Cyclist**	
Object Properties	Height	$U(1.5,\ 2)$ in m	Spokes	36
			Gear rate	$U(1,\ 10)$ in m/s
			Pedaling	$Bern(0.5)$
Motion Properties	Initial Position	$U([-10,\ 10][5,\ 45])$ in m	Initial Position	$U([-10,\ 10][5,\ 45])$ in m
	Heading	$U(-180,\ 180)$ in degree	Heading	$U(-180,\ 180)$ in degree
	Speed	$U(0,\ 1.4\ \times\ height)$ in m/s	Speed	$U(0.5,\ 6\ )$ in m/s
Radio Propagation	Propagation Speed	3×10^8 m/s	Propagation Speed	3×10^8 m/s
	Carrier Frequency	$24\ GHz$	Carrier Frequency	$24\ GHz$
STFT Parameters	Window Type	Kaiser	Window Type	Kaiser
	FFT size	200	FFT size	200
	Overlap rate	6	Overlap rate	6

where the overlap rate in Table 4.1 is defined as the ratio of window size and shift step size. Based on the parameters in Table 4.1, the $\mu - D$ dataset can be generated that covers random factors such as, the size of the object (impact radar RCS), moving orientation, initial position, moving speed, etc. Thus, the trained neural network models will be more robust against the target variation.

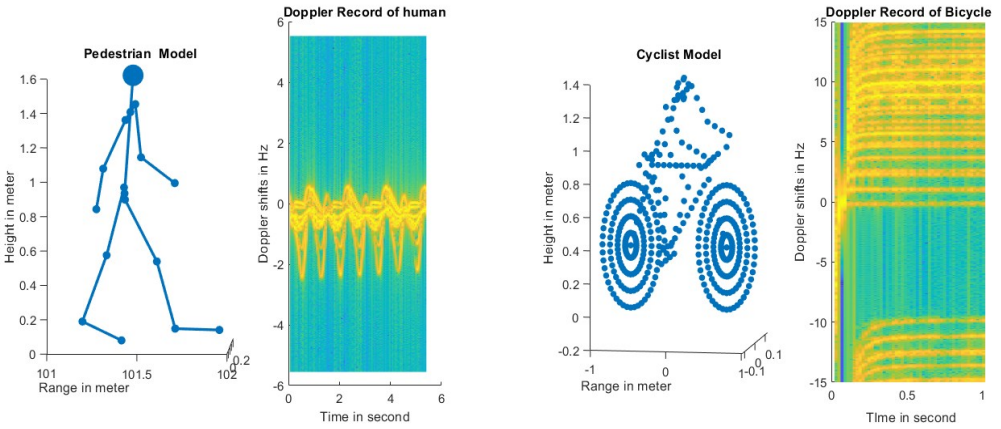


Figure 4.2 *Illustration of the simulated* $\mu - D$ *samples of the pedestrian and cyclist.*

4.4 Datasets Description

Five datasets are created to train the CNN, LSTM, and CNN-LSTM models for simulation. The main difference between the datasets is the duration of the $\mu - D$ signature samples as shown in Table 4.2. The duration of the $\mu - D$ is indicated by the number of the time bins of D. Each dataset contains 25000 labeled samples that cover five road scenarios: single pedestrian (1 ped), single cyclist (1 cyc), two pedestrians (2 ped), two cyclists (2 cyc), one pedestrian & one cyclist (ped+cyc) and 80% of randomly sampled data are used as a training set and the rest 20% samples are used as a testing set. Example samples of the five cases are shown in Figure. 4.3.

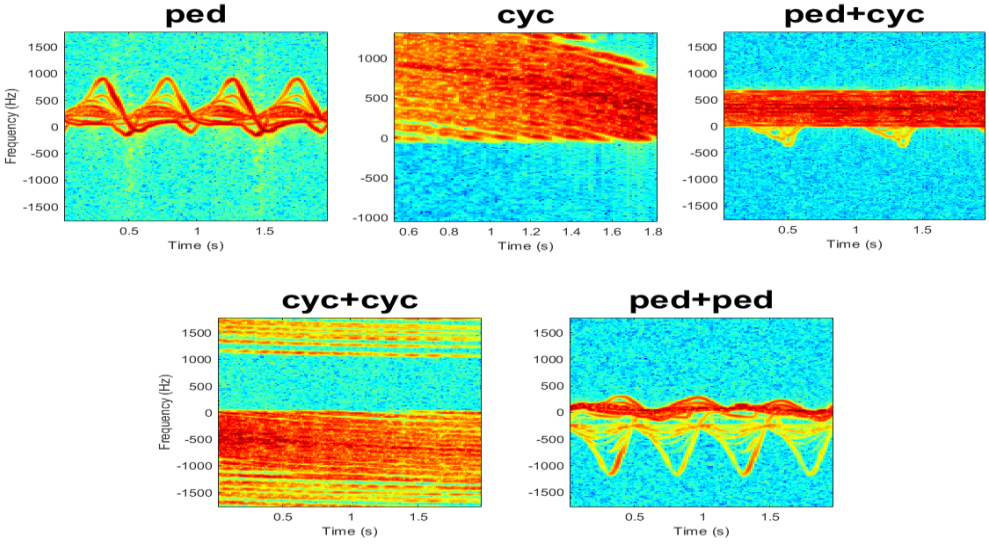


Figure 4.3 *Example of the 2-second duration of Doppler signature sample* $(400 \times 144$ *pixel image) of the five road user cases in Set-4.*

Each $\mu - D$ signature example in Figure 4.3 contains 400 Doppler (frequency) bins and 144-time bins.

Table 4.2 Five datasets for modeling and test experiments

Dataset	Set-1	Set-2	Set-3	Set-4	Set-5
Duration	0.5 sec	1.0 sec	1.5 sec	2.0 sec	
Time bins	36	72	108	144	$\mathcal{N}(72,14)$

4.5 Target Recognition Performance

Different numbers of time bins are considered to test the impact of Doppler recording duration on the recognition performance.

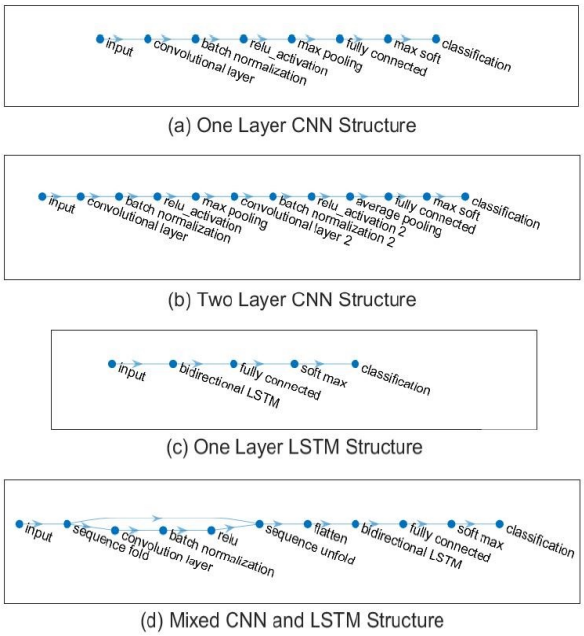
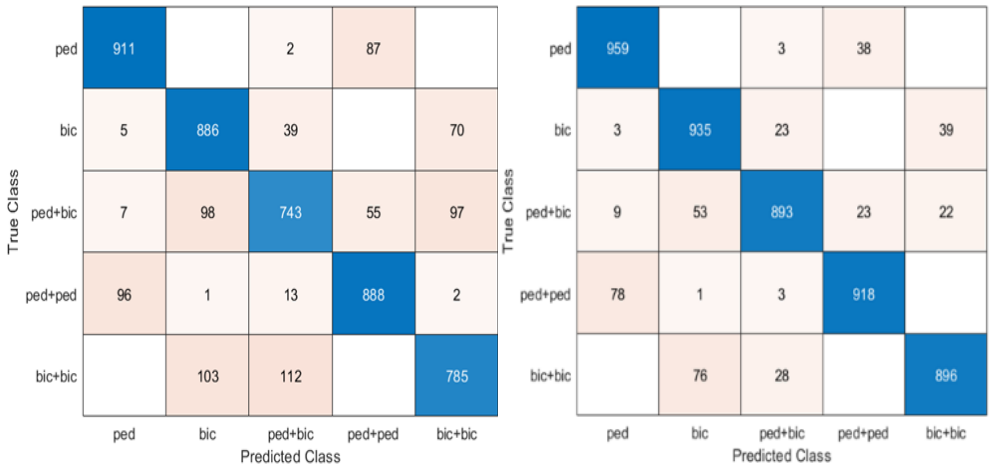


***Figure 4.4** One-layer CNN (a), two-layers CNN (b) CNN, (c) One layer LSTM, and (d) CNN-LSTM structures used for mmWave $\mu - D$ signature recognition.*

Only two simple CNNs are used to test the availability of $\mu - D$ recognition in this simulation: one-layer and two-layer CNN in Figure 4.4 (a) and (b). The dense layer type, activation function, and loss function used in the two CNNs are used and a simple shallow one-layer LSTM NN in Figure 4.4 (c) as is used to demonstrate the capability of recurrent models. For the benefit from both sequential deep spatial features and LSTM temporal features for analysis of the mmWave radar $\mu - D$ signatures of vulnerable road users movements, in the CNN-LSTM in Figure 4.4 (d), the input of the LSTM network is connected to the CNN and re-train.

4.5.1 Feature and Classification Based Performance

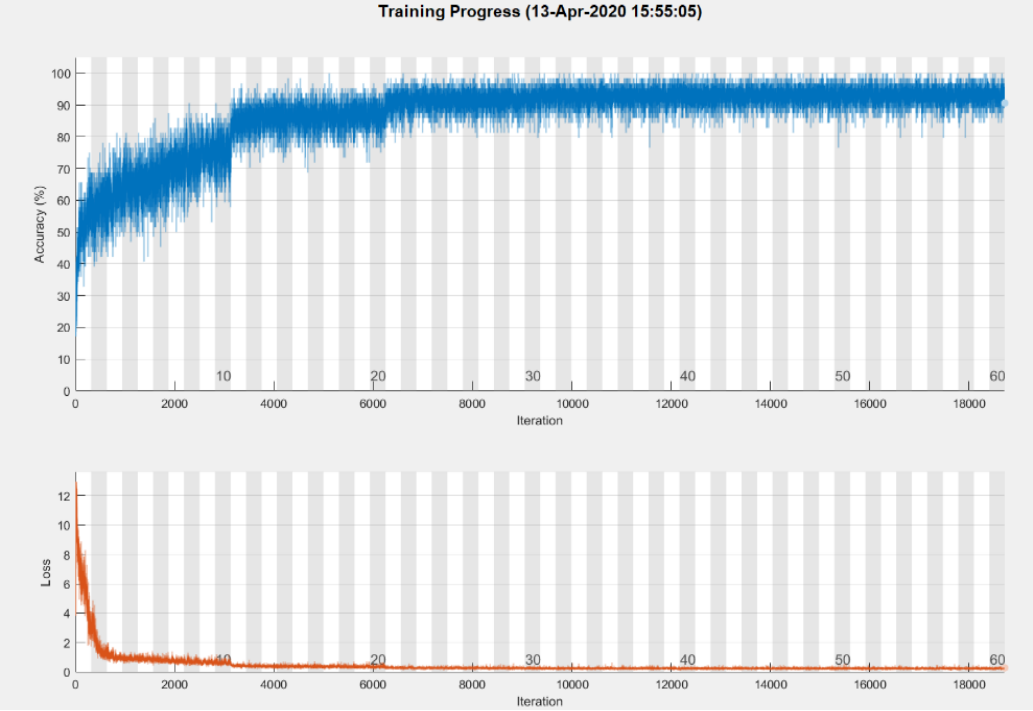
- **One- and two-layer CNN Performance using Set-4:** In the convolutional layer of the one-layer CNN, 16 2D feature maps of size 10×10 is used. This is followed by applying a rectified linear unit (ReLU) activation and 10×10 max-pooling layers. In the case of the two layer CBNN, the first convolution layer uses the same number and type of filters as the one-layer CNN, while the second convolutional layer uses 32 filters of size 5×5 followed by ReLU activation and 5×5 average pooling layers.



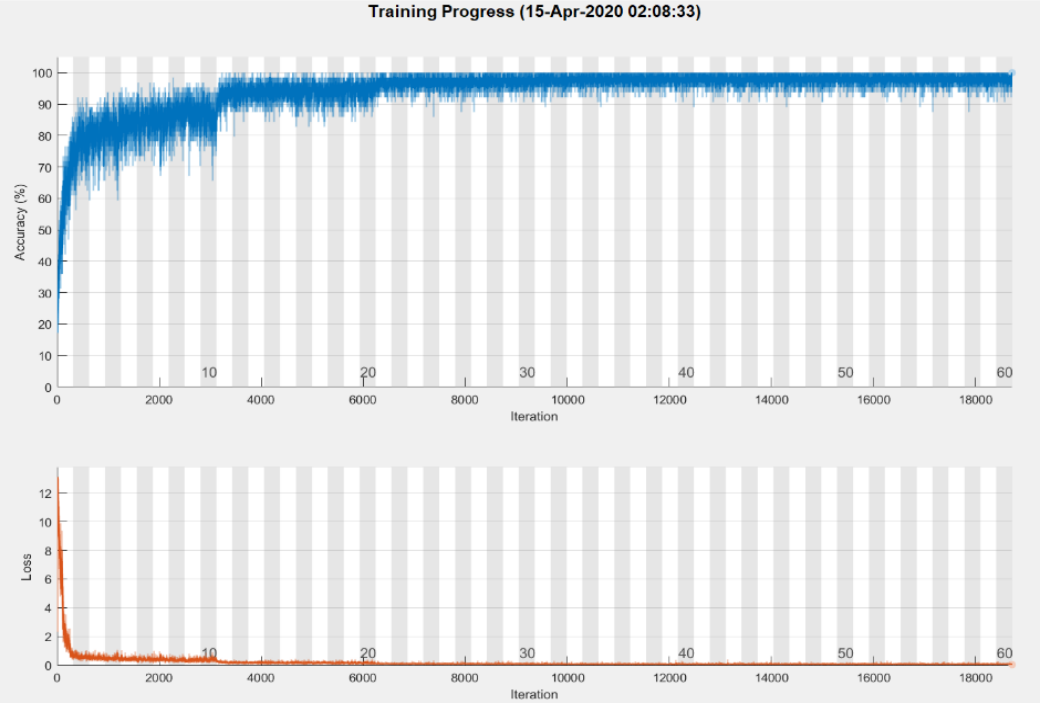
(a) One-layer CNN (b) Two-layer CNN

Figure 4.5 *Confusion matrixes of (a) one-layer CNN, (b) two-layer CNN.*

Regarding the output, both the one- and two-layer CNN use the combination of fully connected, softmax, and classification layers. The training time varies depending on the various factors such as MiniBatchSize, execution environment (gpu, cpu) and MaxEpochs, etc. For this experiment, the MiniBatchSize and MaxEpochs are 64 and 60 has been taken respectively. The overall recognition accuracy of the One- and two-layer CNNs are 0.8426 and 0.9202 and the confusion matrices are shown in Figure. 4.5 (a) and (b) revealing that the increased convolutional layer boosts the performance significantly but increasing the layers in the network required more time to perform shown in Table 4.3.



(a) Single layer CNN



(b) Two-layer CNN

Figure 4.6 *(a) Single-layer CNN and (b) Two-layer CNN training progress.*

It can be clearly seen from Figure 4.6 (a) and (b) performance graph didn't show many changes after the 30 iterations for both two-layer CNN and single layer CNN but certainly, 60 apoches provided a little bit better performance for both NN structures.

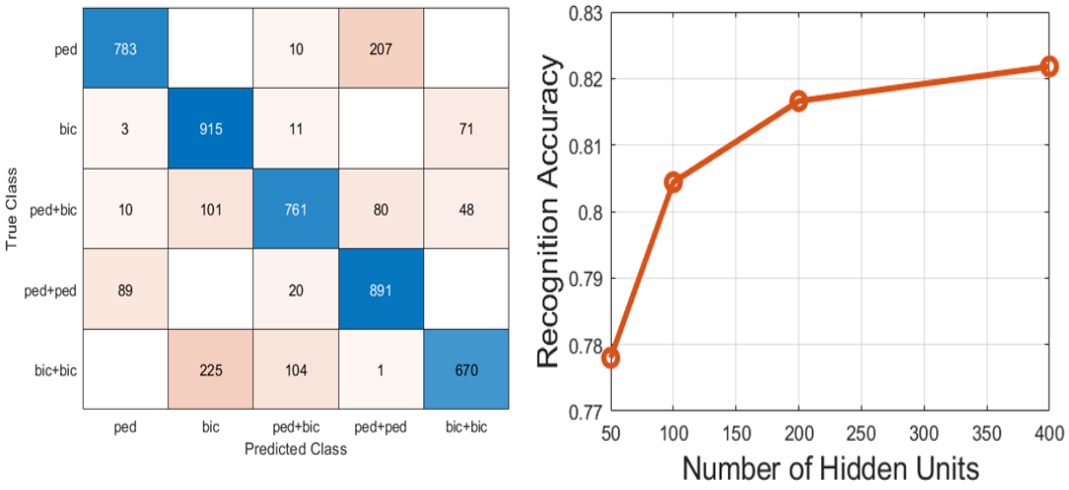
Table 4.3 Requirement time for two layers CNN and single-layer CNN.

NN structure	Time	Epoch
Two-layer CNN	543 minutes	60
Single-layer CNN	473 minutes	60

- **One-layer LSTM Performance with Set-4:** In this thesis, a bidirectional LSTM layer is used. The output layers are the same as the CNN models setup. The MaxEpochs and MiniBatchSize are 60 and 27 taken respectively. The recognition accuracy of the single-layer LSTM model with 400 hidden units is 0.8218, shown as the confusion matrix in Figure 4.7 (a).

Table 4.4 single layer LSTM performance with different number of hidden units for 2-second duration set-4.

Number of HUs	50	100	200	400
One-layer LSTM accuracy	0.7780	0.8044	0.8166	0.8218

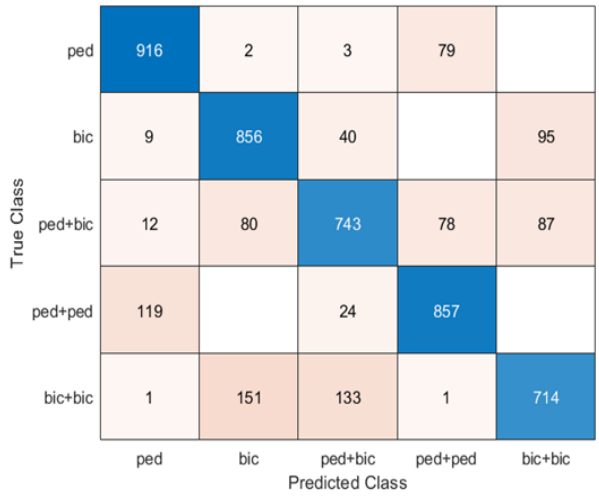


(a) (b)

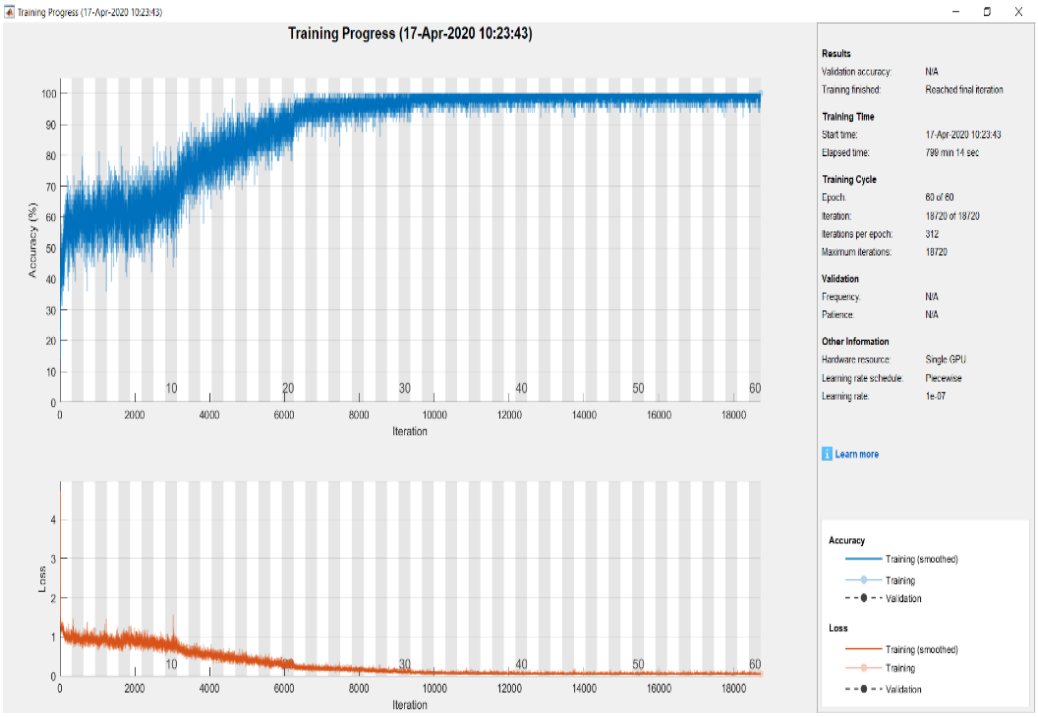
***Figure 4.7** (a) Confusion matrix of single-layer LSTM with 400 HU, (b) LSTM recognition accuracy with different number of HU.*

For the LSTM NNs, the number of HUs impacts the recognition performance. In Table 4.4, we show how the increase in the HUs improves the performance of the LSTM.

- **CNN-LSTM Model Performance on Set-4:** In this CNN-LSTM model, the $\mu - D$ sample is first fed to a convolutional layer and the number of filters and size are defined. The MiniBatchSize and Maxepochs are used same as the one-layer CNN. Then, the output of the previous layer are fed to the LSTM layer which has 400 HUs.



(a)



(b)

Figure 4.8 *(a) CNN-LSTM Confusion matrix, (b) Training progress.*

The overall recognition accuracy of the CNN-LSTM model reaches to 0.8686. The confusion matrix is shown in Figure 4.8 (a) and training progress is shown in Figure 4.8 (b)

- **Recognition with different $\mu - D$ durations:** Using the 2-second $\mu - D$ samples in Set-4, all the convolutional, recurrent, and CNN-LSTM models achieve over 80% recognition accuracy. However, short reaction time is already a favorable feature in the road traffic scenarios. Thus, the performance of the models is tested when using different $\mu - D$ sample durations for Set-1 to 4. The

same network layer structures and training setups as the previous sections are used.

Table 4.5 Classification performances of one- and two-layer CNN, one-layer LSTM, and CNN-LSTM models using different $\mu - D$ signature durations.

	Set-1 0.5 Sec	**Set-2 1.0 Sec**	**Set -3 1.5 Sec**	**Set-4 2.0 Sec**
One-layer CNN	0.8288	0.8544	0.8726	0.8426
One-layer LSTM	0.7608	0.7810	0.8302	0.8040
Two-layer CNN	0.8710	0.9042	0.9158	0.9202
CNN-LSTM	0.8106	0.8172	0.8480	0.8686

From Table 4.5, it can be observed that the longer $\mu - D$ signature duration resulted in higher recognition accuracy in general.

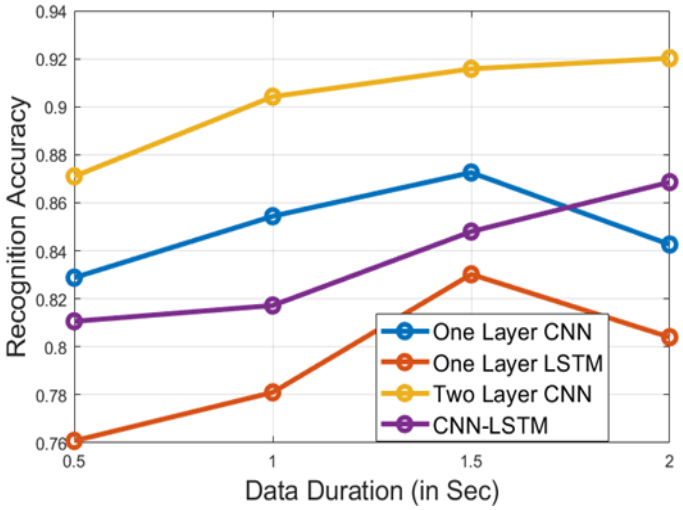


***Figure 4.9** Recognition accuracy with different $\mu - D$ duration.*

However, the conclusion is not always stand, due to the capability of the shallow NNs. For example, the performance of 2-second duration Set-4 drop in both one-layer CNN and one-layer LSTM cases. That is because the single NN layer may not be able to cope with both spatial and/or sequential features of the long $\mu - D$ data. Due to their deeper structure, the two-layer CNN and CNN-LSTM models can handle the long duration data well, as shown in Table 4.5.

- **Recognition accuracy in random $\mu - D$ sample duration:** In the previous sections, the recognition performances were evaluated by using the Set-1 to 4 that contain uniformed $\mu - D$ sample durations, which are not common in practice. This section will evaluate the models using Set-5 that contains $\mu - D$ samples with random durations.

The recognition accuracies are shown in Table 4.6 using the same network structures and training setups in the previous experiments.

Table 4.6 classification performances of one- and two-layer CNN, one-layer LSTM, and CNN-LSTM models on set-5.

	1L CNN	**1L LSTM**	**2L CNN**	**CNN-LSTM**
Accuracy	0.8166	0.6224	0.8528	0.8212

From the result in Table 4.6, we can clearly see that the performance of the one-, two-layer CNN models and one-layer LSTM model drop by $4{\sim}6\%$, $5{\sim}6\%$, and $14{\sim}20\%$ respectively when comparing the fix duration datasets (e.g. Set-2, Sect-3). While the CNN-LSTM model shows improvement when comparing with the Set-2 and only around 1% drop when comparing with Set-3. Also, with further optimization of the convolutions layers and length of the HU, the CNN-LSTM is expected to outperform other models meanwhile demonstrates robustness to the randomness of the $\mu - D$ duration.

4.5.2 NN Based Performance

- **Convolution Model:** From the recognition accuracy results in Figure 4.5 (a), (b), and Table 4.5, is concluded that despite of the additional complexity to the model, increasing the number of layers positively improves the performance. In addition, the performance drop of one-layer CNN model in the case of the 2-second duration dataset shown in Table 4.5, is evidence that the number of convolutions layers is more important for longer periods signals and deeper features are required in these cases.
- **Recurrent Model:** The results in Figure. 4.7 (a) and Table 4.5, show that the recognition accuracy of the one-layer LSTM model is not as high as one-layer CNN due to the lack of spatial feature representation. However, it shows some level of robustness for longer durations of the $\mu - D$ samples. Furthermore, As shown in Table 4.4, the number of hidden units is a key factor for the recognition capability of the longer $\mu - D$ sequence.
- **CNN-LSTM Model:** The CNN-LSTM model takes the advantage of the spatio representation of 2D $\mu - D$ samples from the convolutional layer and temporal feature from LSTM layer. Thus, this model shows robustness for $\mu - D$ datasets

with the dataset of random $\mu - D$ sample durations, which is a promising sign for practical applications.

- **Potential problems in practice:** The current work is conducted based on the simulation $\mu - D$ datasets. Though the simulation datasets provide diverse variables for testing the strengthen and weakness of different models, there are other factors which are difficult to be include in the simulation data. For example, the noise and interferences of the reflections from others.

4.6 Conclusion

In this chapter, a simulation study of $\mu - D$ signature of vulnerable road users (pedestrian and cyclist) was carried out. In the absence of $\mu - D$ data from road trials, the simulation data still valuable for developing and testing the object recognition methods. CNN, LSTM, and CNN-LSTM models are tested for different road scenarios and varying Doppler record duration. The preliminary results show that the pedestrian and cyclist $\mu - D$ signatures originated from the mmWave is distinguishable with even shallow one or two-layers neural networks. In general, the two-layer convolutional models outperform the recurrent model (LSTM in this paper). However, it is sensitive to $\mu - D$ signature duration. This fact is against the requirement of an agile response in an urgent situation. The LSTM recurrent model presents stability over different $\mu - D$ durations. The CNN-LSTM model combines merits from both convolutional and recurrent models, especially in the case of random duration $\mu - D$ samples in practical situations.

5. EXPERIMENTAL STUDY FOR TARGET (HUMAN) DETECTION

In this chapter, an experimental study has been presented by the Texas Instruments (TI) mmWave Evaluation Module (EVM). This chapter consists of the sections (i) a detailed description of the IWR6843ISK module and mmWAVEICBOOST; (ii) experiments with TI mmWave demo visualizer followed by result and discussion, (iii) experimental study with MATLAB program followed by result and discussions. The Experiments has been conducted in the indoor environment. Millimeter-wave (mmWave) radar hardware is used in this thesis is an affordable low-power range sensor and can identify the target within 100 meters.

5.1 Hardware

The IWR6843ISK and MMWAVEICBOOST are specific hardware introduced by Texas Instruments used in this thesis for the experimental study. The IWR6843ISK device consists of whole mmWave blocks, customer-programmable DSP, customer-programmable MCU and has three transmitters and four receivers with the analog baseband signal chain. This system is used as a radar-on-a-chip in application cases with modest memory, processing power, and application code size specifications. This may be cost-sensitive implementations of industrial radar sensing. Examples are

- Sensing at the industrial level
- Sensor fusion for factory automation with radar
- Monitoring of traffic intersections with radar
- Industrial surveillance
- Gesturing
- People counting
- Object detections

For signal processing, radar signals for FFT, amplitude, tracking, and other uses, the IWR6843ISK has an integrated DSP. This board act as a radar front-end board because this contains a 60 Gigahertz transceiver in which on the PCB board or on the packager the antennas are attached. The MMWAVEICBOOST is an add-on board used for the mmWave sensor from TI, used with all starter kits to provide the mmWave sensors with more interfaces and PC compatibility. The MMWAVEICBOOST Using a capture board such as the DCA1000 evaluation module, the MMWAVEICBOOST board provides an

interface for the mmWave Studio tool to configure the Radar system and capture raw analog-to-digital converter (ADC) data. The EVM antenna module IWR6843ISK and MMWAVEICBOOST has everything needed to start developing the interface for on-chip C67x DSP core and low-power ARM R4F controllers. The EVM provides an interface through 40-pin LaunchPad™/BoosterPack™ connectors for the MSP43xx boards [43].

5.2 Functional Block Diagram of IWR6843ISK

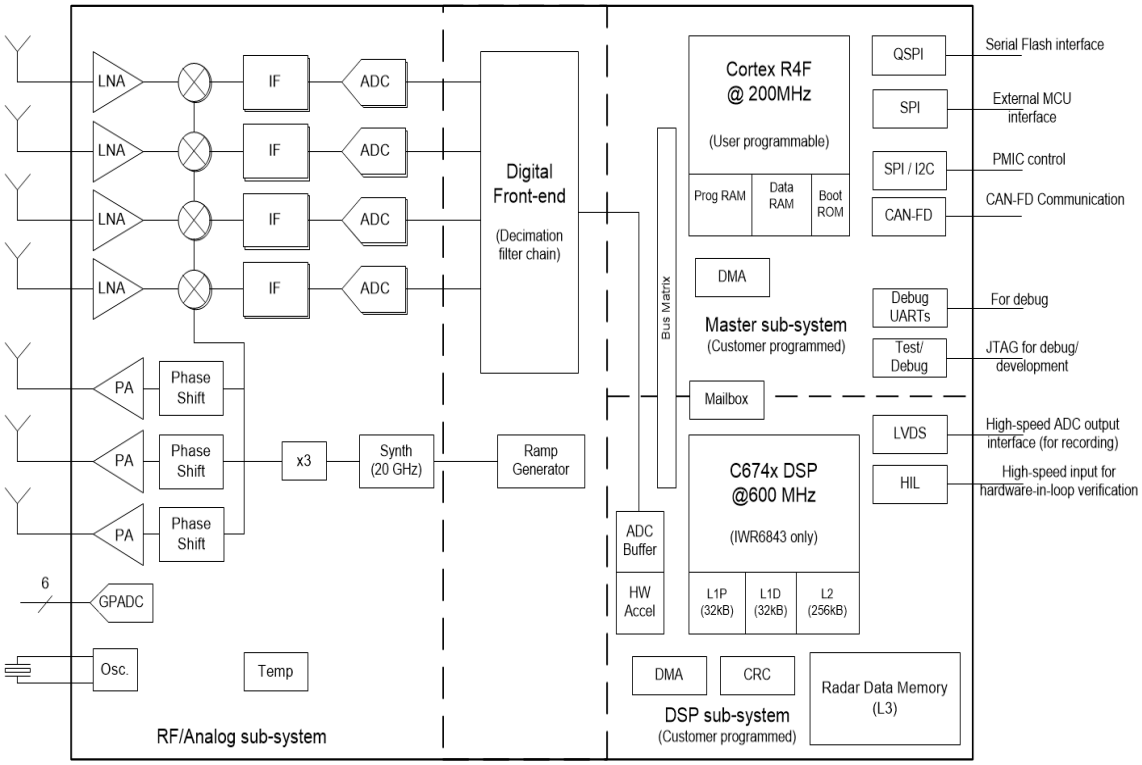


Figure 5.1 *Functional diagram of IWR6843ISK* [43].

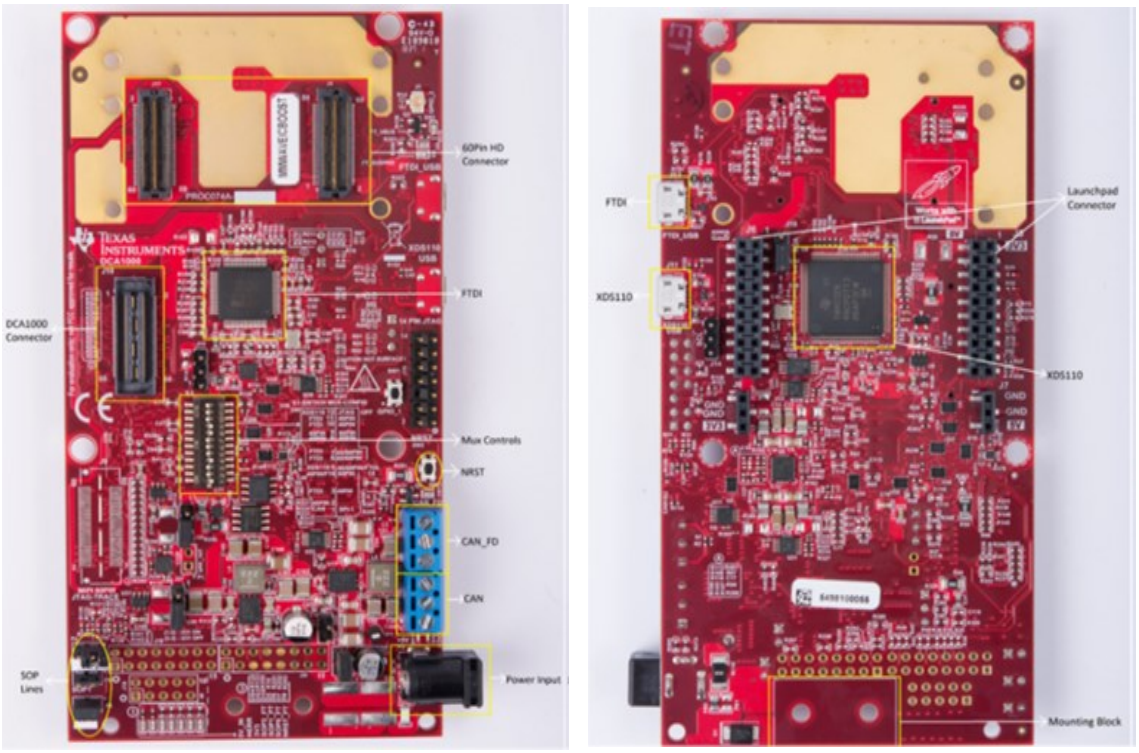
The IWR6843ISK functional block diagram consists of RF and analog sub-system, which including the synthesizer, PA, LNA, amplifier, IF, and ADC, are part of the RF and analog subsystem. The crystal oscillator and temperature sensors are also included in that subsystem. The three transmitting channels can be operated in 1.3-V mode, up to a maximum of two at a time (simultaneously). The simultaneous operation of the three transmission channels is only enabled with 1-V LDO bypass and PA LDO disabled mode for beamforming purposes, as needed.

The Clock Subsystem of IWR6843 generates 60 to 64 GHz from a 40-MHz crystal input reference. Three parallel transmission chains consist of the IWR6843 transmission subsystem, each with independent phase and amplitude control. For MIMO radar, the system supports 6-bit linear phase modulation, Tx Beam shaping Applications, and minimize intervention. Programmable backoff for device optimization is also supported by the transmission chains. Four parallel channels compose of the IWR6843 receiving

subsystem. An LNA, IF filtering, A2D conversion, mixer, and decimation constitute a single receive channel. For device optimization, all four receive channels may be active at the same time as an individual power-down option is also available.

DSP subsystem that comprises TI's high-performance C674x DSP (IWR6843 only), high-performance high-bandwidth interconnect (128-bit, 200MHz), and related peripherals-four data transmission DMAs. LVDS for measurement output, L3 data cube memory for radar, ADC buffers, CRC generator, and data handshake memory. The master subsystem manages all device peripherals and device housekeeping operations. The Master subsystem includes the processor Cortex-R4F (Master R4F) and related peripherals and housekeeping components such as DMAs, CRC and Peripherals (I2C, SPIs, CAN, UART, PWM, PMIC clocking module and others) connected via Peripheral Central Resource (PCR interconnect) to the Master Interconnect [43].

5.3 MMWAVEICBOOST Hardware and Block Diagram



(a) Front view (b) Rear view

Figure 5.2 *MMWAVEICBOOST (a*) Front *view* (b) Rear *view [43].*

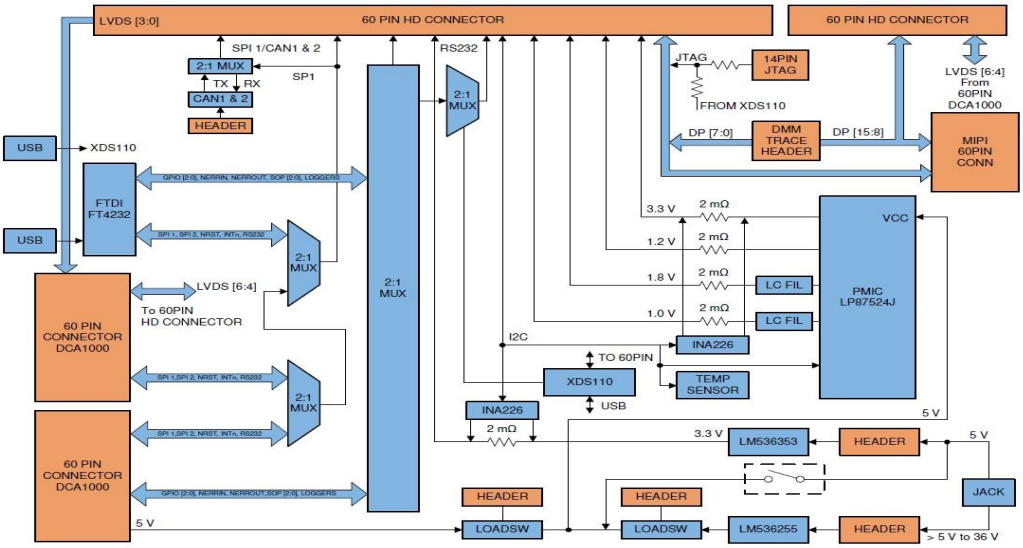


Figure 5.3 *Block Diagram of MMWAVEICBOOST*[43].

5.4 Experiment with the TI mmWave Demo Visualizer

Experiments have been done by using TI mmWave demo visualizer. This is Texas Instruments gallery APP is used for configuring the mmWave sensor and identifying the object in a certain range which is generated by the mmWave SDK demo. This app is used in conjunction of the evaluation module and mmWave SDK demo.

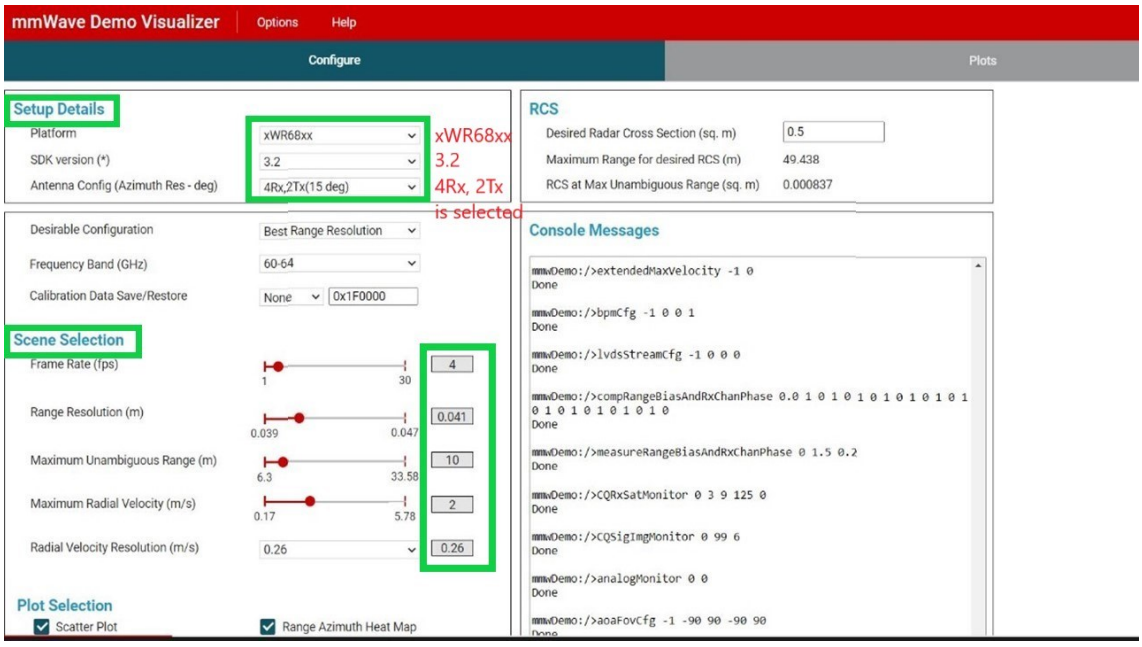


Figure 5.4 *GUI of the mmWave demo visualizer.*[44]

For the experiment, SOP is selected in functional mode. The mmWave sensor is connected to the PC by XDS110 UART interface and power up the sensor by 5V power supply. The browser, mmWave device, and serial port are sets up as shown in Figure 5.4 [44].

5.4.1 Configuration

- The platform is selected to the xWR6843.
- SDK version is selected to the 3.2
- Antenna config (Azimuth res-deg) is selected to the 4Rx, 2Tx (15 degrees).
- Selected the best range resolution in the desirable configuration.
- Frequency band wide band (60 to 64 GHz) is selected[44].

5.4.2 Scene Selection

- **Frame rate:** Frame rate (fps) is selected to 4fs for the range azimuth heat map and range doppler heat map.
- **Range resolution:** Amount of separation between objects or points in the cloud defined by the range resolution(m), 0.041 meters of range resolution is selected for the experiment.
- **Maximum Unambiguous Range(m):** Maximum unambiguous range is selected to the 10 meters.
- **Maximum Radial Velocity (m/s):** Maximum radial velocity is selected to 2.0 meters per second.
- **Radial Velocity Resolution:** Radial velocity resolution is selected 0.24 meters/second.

5.4.3 Plot Selection

1. **Scatter plot:** Scatter plot provides the detected object numbers sent out by the targets to the EVM and displays it on the doppler range plot and scatter plot.
2. **Range Profile:** Range profile enables the log-magnitude range profile information at zero doppler to be sent out by the target and show it on the range profile plot.
3. **Range Azimuth Heat Map:** Range azimuth heat map to send all range bins and all antennas to the zero Doppler radar cube matrix for out and display it on the range azimuth heatmap plot.
4. **Range Doppler Heat Map:** Range doppler heat map enables to send the whole detector matrix to the target (mmWave sensor) unit [45].

5.5 Experiment Scenario Description

Four experiments have been conducted with the mmWave demo visualizer. The first experiment was done with an object at a distance of 1.0 meter from the mmWave sensor device. The 2nd and 3rd experiments were done with a distance of 1.5 meters and 2 meters respectively. For the 4th experiment, a metal object with a height and width of 1m was moved from 2 meters to 1 meter towards the mmWave sensor device shown in Figure 5.5.

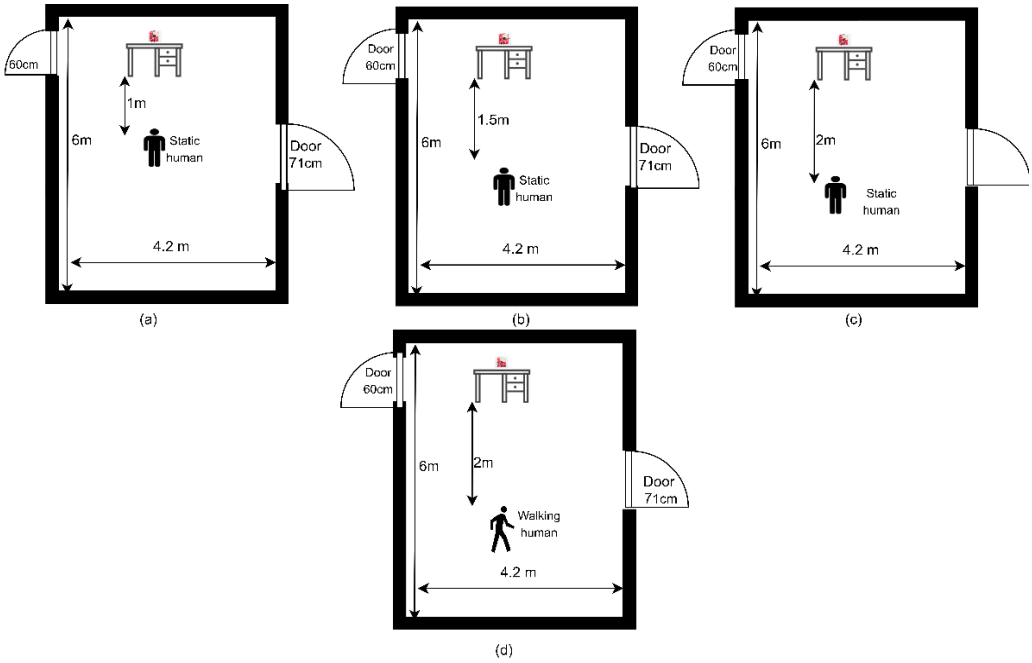


Figure 5.5 *Experiment illustration for the static object (human) (a), (b), (c) at a distance 1m, 1.5m, 2m respectively and human walking (d) towards the mmWave sensor from 2m distance.*

5.6 Experiment Procedure

The experiment procedure is described below step by step,

- The EVM was flashed and the image was loaded by the UniFlash tool.
- The SOP mode was changed from the flashing mode to FUNCTIONAL mode.
- The MMWAVEICMODE was set to the STANDALONE mode.
- The evaluation module was connected with the PC by the XDS110 Emulator/UART interface and the sensor device powered up by the 5V power supply.
- From the mmWave demo visualizer, serial ports were configured in the ‘configure’ tab.
- The configuration, scene selection, and plot selection have been done according to the 5.5.1 and 5.5.2 sections.

- Finally, the configuration was sent to the mmWave device and the data was logged for 5 seconds.

5.7 Result and Discussion

5.7.1 Experiment with Static Target and Moving Target

The data is logged when a static object (human) was at a distance of 1.0 meters, 1.5 meters, and 2 meters from the mmWave sensor for 5 seconds, and the total data was plotted in an X-Y scatter plot. This logged data gives the results with the information about the Frame Number, Total Target, Target Number, X (m), Y (m), Z (m), Doppler (m/s), and Intensity of the target.

For the first experiment, 5 frame numbers were generated with 47 of total targets. For the second experiment, 8 frame numbers and 64 total targets were generated and in the third experiment, 7 frame numbers and 64 total targets were generated. In these experiments, the maximum intensity of 111.2(log), 105.5(log), and 101.9(log) was recorded, respectively. From Figure 5.6 (a), (b), and (c) the circle size of the dots indicates the doppler of the object. The color of the dots indicates the intensity of the power of the target and blue to yellow color indicates the minimum to the maximum intensity of the target. It can be clearly seen most of the circle sizes are identical because of zero movements of the target (object) during the experiments. It also can be seen that the maximum intensity of the target is recorded at around 1m,1.5m, and 2m distance by the yellow circle shown in the figures. The intensity was high for the 1m distance object due to the shorter distance of the object than the 1.5m and 2m distance. Similarly, Intensity was lowest for the 2m distance object due to the higher distance than the other experiments. There is some scattered circle can be seen in the scatter plot due to the presence of other objects in the experimental environment, but the intensity of these circles is mostly below 90 (log) and has zero doppler.

In the 4th experiment, a metal object with a height and width of 1m was moved towards the mmWave sensor device from 2 meters distance to 1-meter distance and the data was logged for 5 seconds. The maximum intensity of 115.5 (log) was recorded during this experiment.

From Figure 5.6(d), it can be seen that the highest intensities are between the 2m and 1m distance indicated by the yellow circle. The Doppler can also be identified for the change in the circle size between the 2m and 1m distance.

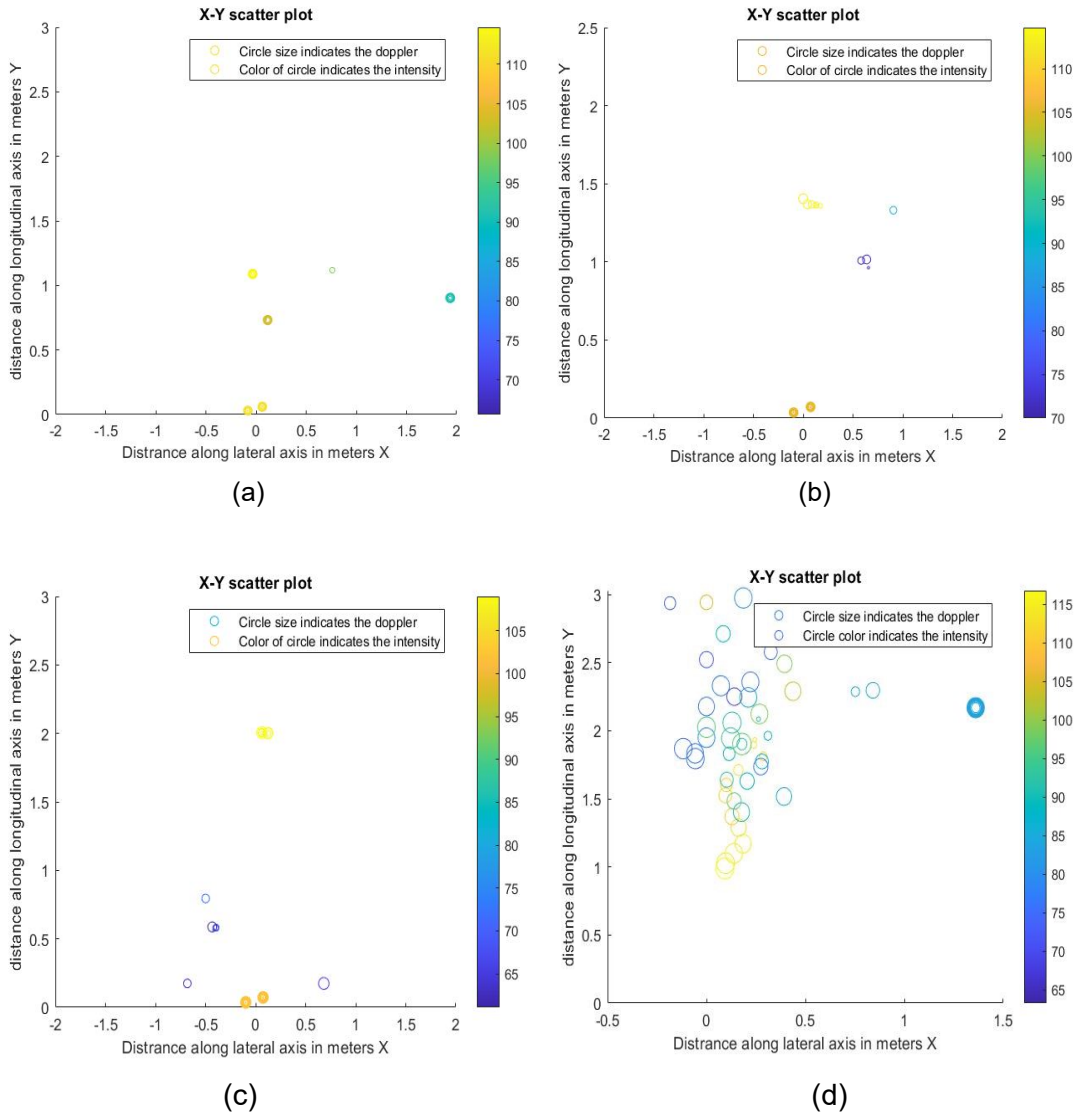


Figure 5.6 *X-Y scatter plot for the static object (human) at a distance 1m(a), 1.5m(b), 2m(c), and the object moving towards mmWave sensor from 2m(d).*

5.8 Experiment with Matlab

Real-time data is captured by the mmWave radar board using the Matlab based on the mmWave demo from the Texas instrument. For the experiments with Matlab, the SOP configuration and mux configuration were the same as the mmWave demo visualizer experiment. EVM was powered ‘on’ by the 5V power supply and the sensor device is connected with the laptop. At the very first, the initial properties are defined for the radar set up, the azimuth resolution, range resolution, maximum unambiguous range, maximum radial velocity, frame duration, range detection threshold, Doppler detection thresholds are predefined in the cfg file is uploaded to the script. The frame rate 3fps, range resolution 0.044 (m), maximum unambiguous rage 3.95 (m), maximum radial velocity 0.17 (m), maximum radial resolution 0.33 (m/s) has taken for the experiment.

Figure 5.7 *mmWave sensor setup with the laptop.*

5.8.1 Radar Set up

For serial port configuration, The UART COM port was configured and the Baudrate was set to 115200. Similarly, DATA COM port was configured and Baudrate was set up to 921600. The function was set up to read the configuration file and parse the configuration file. The Channel, profile, and frame parameters were configured. In the channel configuration, the antennas were configured. In the profile configuration, starting frequency, Idle time, ramp end time, number of ADC samples were configured. Chirp start index, chirp end index, number of loops, number of frames, frame periodicity are configured in the frame configuration. The number of chirps per frame, number of Doppler bins, Number of range bins, range resolutions, range index to meters, maximum range, maximum velocity etc are also configured for setting up the RADAR.

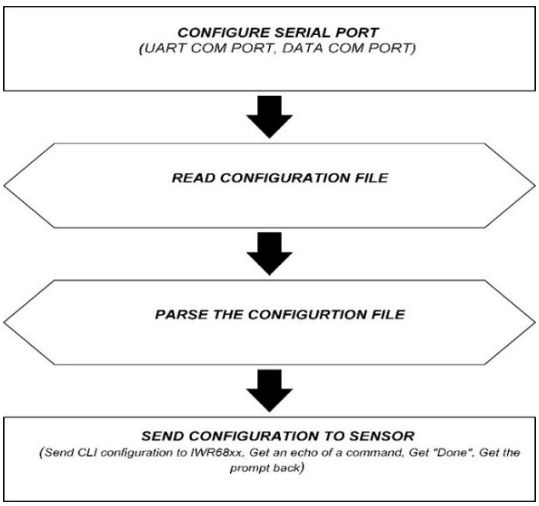


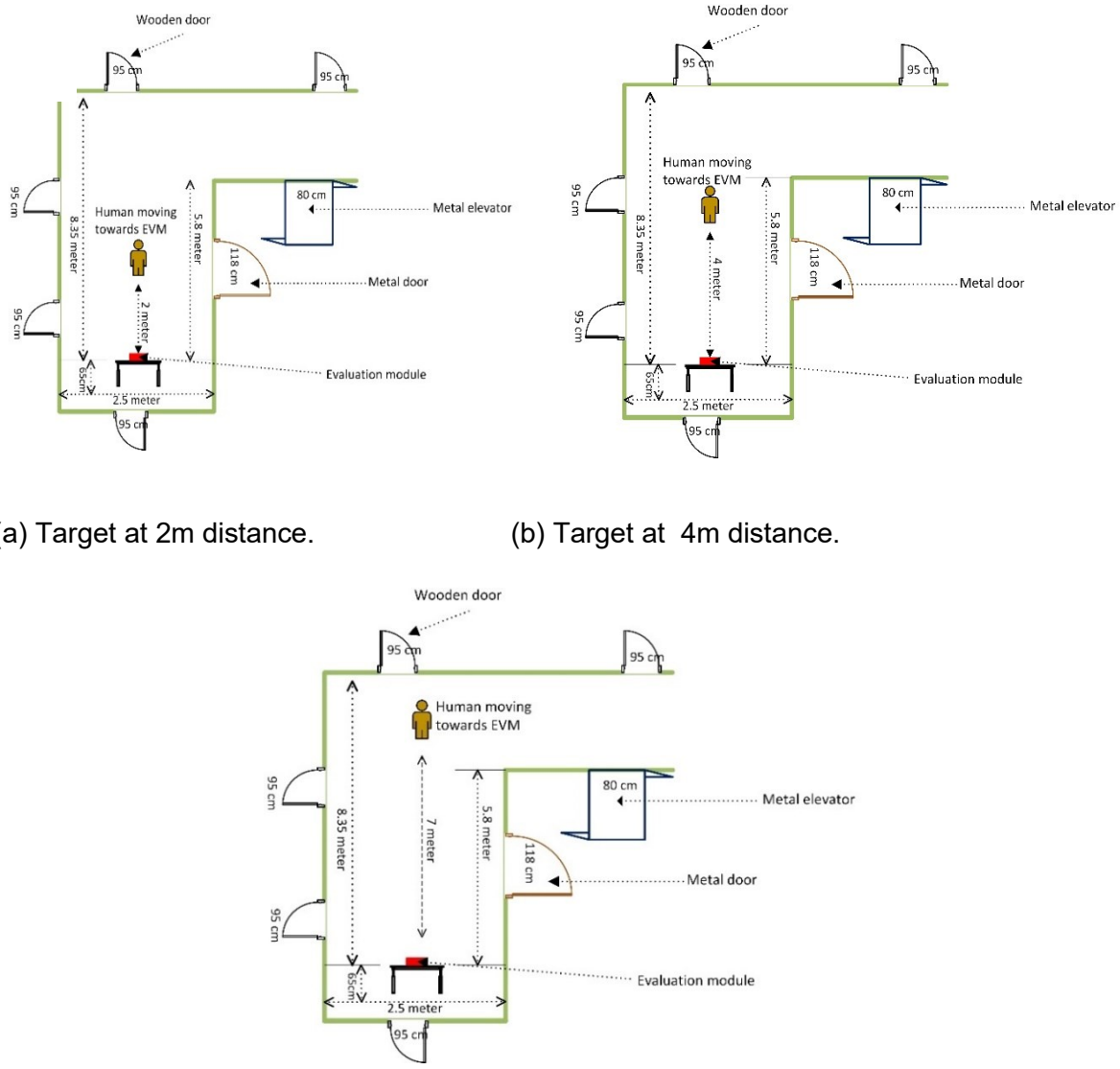
Figure 5.8 *Radar configuration procedure*

The serial port configuration sends the CLI command to the radar which provides the outputs of the serial targets for the data and CLI ports. In parse configuration, the

configuration file was parsed to extract the configuration parameters. This returns the parameter dictionary with the extracted values. Finally, the configuration was sent to the sensor device. The radar set up processing flow is shown in Figure 5.8.

5.8.2 Experiment Scenario Description

For the experiments with the mmWave sensor, a corridor of 9 meters long and 2.5 meters in width was chosen.



(a) Target at 2m distance.

(b) Target at 4m distance.

(c) Target at 7m distance.

Figure 5.9 *Experiment illustration for the object (human) walking towards the mmWave sensor from the distance of 2m(a), 4m(b), and 7m(c).*

The experimental environment had a metal door at a distance of 3.25 meters from the mmWave sensor with a width of 95 cm on the right side of the corridor. There were three other doors in the experiment environment, two were on the left side of the corridor and one was in front of the corridor. The width of all the doors was 95 cm. For the real-time

experiment, three experiments were conducted, one by one. The three experiments were done for the moving object (human) walking towards the mmWave sensor from the 2m, 4m, and 7m distance respectively shown in Figure 5.9 (a), (b), and (c).

5.8.3 Data Processing

The data processing consists of taking ADC samples as input and producing detected objects (point-cloud and other information) to be shipped out of UART port to the PC. The algorithm processing is realized using the Data Path Manager (DPM) registered object detection Data Path Chain (DPC) shown in Figure 5.10. The output packet with the detection information is sent out every frame through the UART. Each packet consist of the header and the number of TLV items containing data information and each TLV item consists of type, length, and payload information. Since output packet length depends on the number of detected objects and it can vary from frame to frame. The end of the output packets is padded so that the total packet length is always multiple of 32 bits. The frame numbers, detected points, 3D positions, the velocity of the target given by the function which returns a Boolean variable that stores data if data was correct. The detected objects contain the range, angle, and velocity information of the target from the sensor device, and the coordinates (x, y, z) are in Q format in the descriptor field.

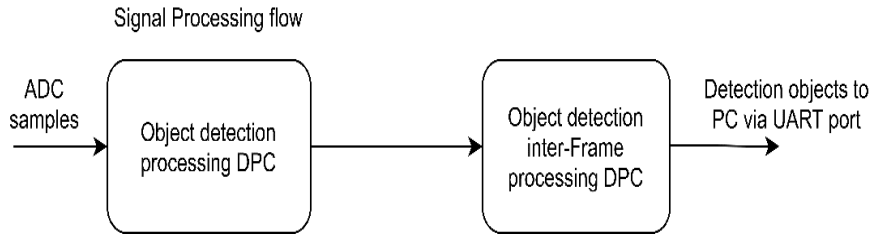


Figure 5.10 *Data Processing chain*

5.8.4 Data Structure

The data structure of the header, detected objects, range profile, azimuth static heatmap, and doppler range heatmap [45].

- **Header and Detected Objects Data Structure:**

Table 5.1 Header data structure and detected objects data structure

Header	Detected Objects
MagicWord (8 bytes)	Structure Tag (4 bytes)
Version (4 bytes)	Length of Structure (4 bytes)
Total Packet Length (4 bytes)	Descriptor (4 bytes)
Total Packet Length (4 bytes)	Object Struct (while (i++ != #DetObj))
Frame Number (4 bytes)	Range Index (2 bytes)
Time (CPU Cycles) (4 bytes)	Doppler Index (2 bytes)
Number of Detected Objects (4 bytes)	Peak Value (2 bytes)
Number of Data Structures in package (4 bytes)	X coordinate (2 bytes)
	Y coordinate (2 bytes)
	Z coordinate (2 bytes)

- **Range Profile:**

The range profile is extracted from the payload of the output packet from UART. The range profile contains a structure tag and the length of the structure. They also contain 1D array of log magnitude Range FFTs. In the Matlab function, the object is detected by range profile with

$$Size = RangeBins \times \ 4\ bytes \tag{5.1}$$

where range bins are the discrete boxes in which the range-FFT divides the signals.

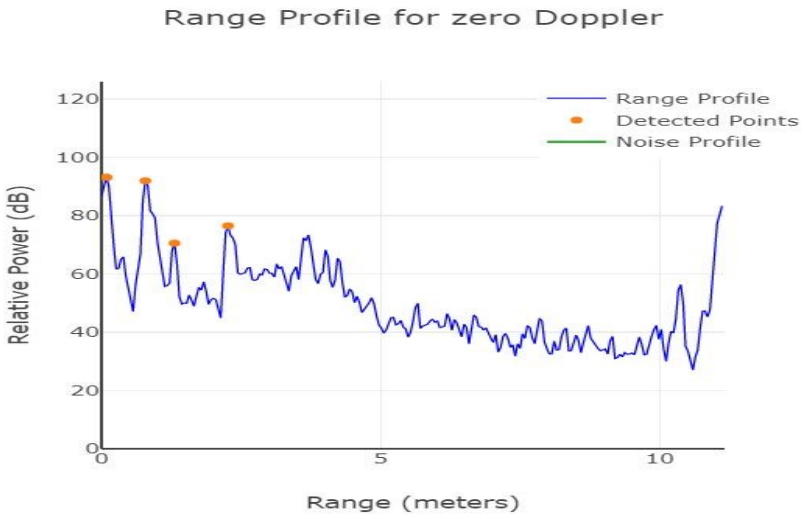


Figure 5.11 *Range Profile.*

To generate the range profile, ADC sample is configured in the cfg file which is 256. As mentioned in section 2.2.3, The IF signal will have tones corresponding to each of the reflected signals and the frequency of these tones is directly proportional to the range. In Figure 5.11 it can be clearly seen that the object detected points in a certain range. The range profile at zero doppler for a static object indicated by the blue color and the orange dots show the detected objects at the 0^{th} doppler range bin.

- **Azimuth Static Heatmap:**

  The Azimuth static heatmap is extracted from the payload of the output packet from UART and set up the parameters for plotting in the GUI monitor with 256 range bins and 64 angle bins. In the Matlab function, the size of the heatmap is given by,

$$Size = Range\ bins \times Virtual\ antennas \times 4\ bytes \quad (5.2)$$

  the column-based number of angle bin point FFT is taken with padded zeros (appendix: *Case mmwave demo UART azimuth static heatmap*). The heat map is a matrix which then provides the elements consist of intensities and its indexes are arbitrary quantities. From Figure 5.12, it can be seen that the object is detected in the range of 2 meters and gives the estimation angle of the object. Heatmap shows the result at the 0^{th} doppler of the radar cube matrix and all antennas and all range bins.

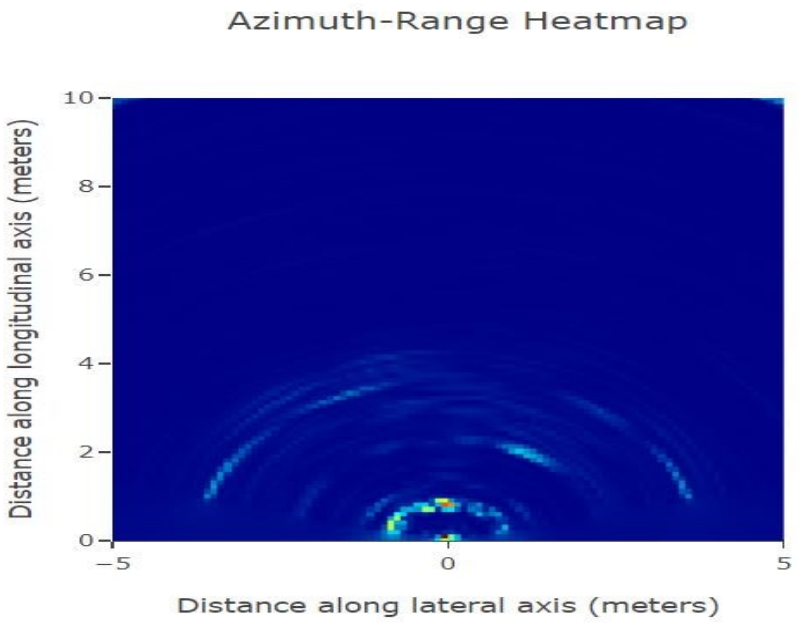


Figure 5.12 *Azimuth-Range Heatmap.*

- **Range-Doppler Heatmap:**

  Similarly, the range doppler heatmap is extracted from the payload of the output packet from UART. For plotting the range doppler heatmap, The length of the heatmap is defined in the Matlab function (*Appendix: Case mmw demo range doppler heatmap*). Then The size of the heat map is given by,

  $$Range\ bins \times Doppler\ bins \times 4\ bytes \quad (5.3)$$

  Where, $$Doppler\ bins = Chirps\ per\ frame / Transmission\ antennas \quad (5.4)$$

  The result of the heatmap taken doppler values as its row indices and range value as its column indices. Figure 5.13 shows the result of the maximum doppler and minimum doppler at different ranges indicated by the red and blue color respectively. The red color indicates the maximum doppler captured in the given frame which is 1-2 meters in the experiment. In range and doppler coordinates, this plot shows the entire radar cube matrix using the heatmap plot.

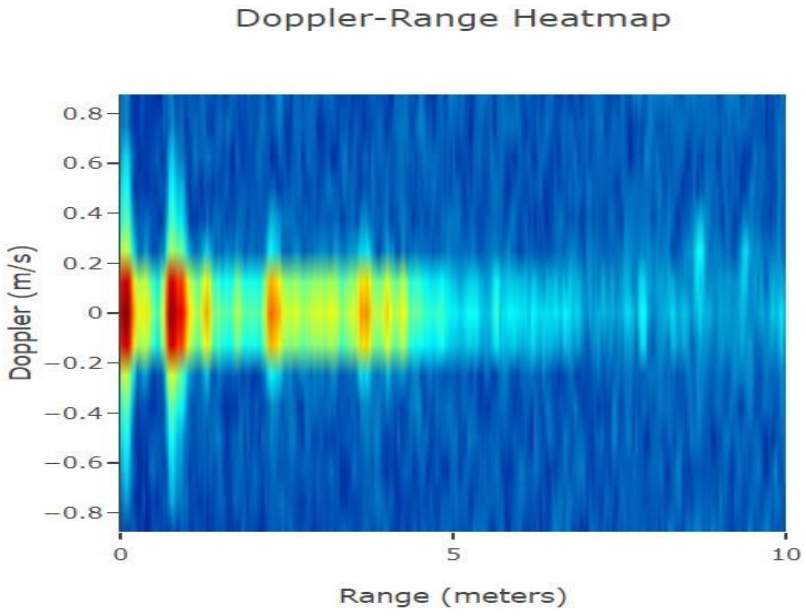


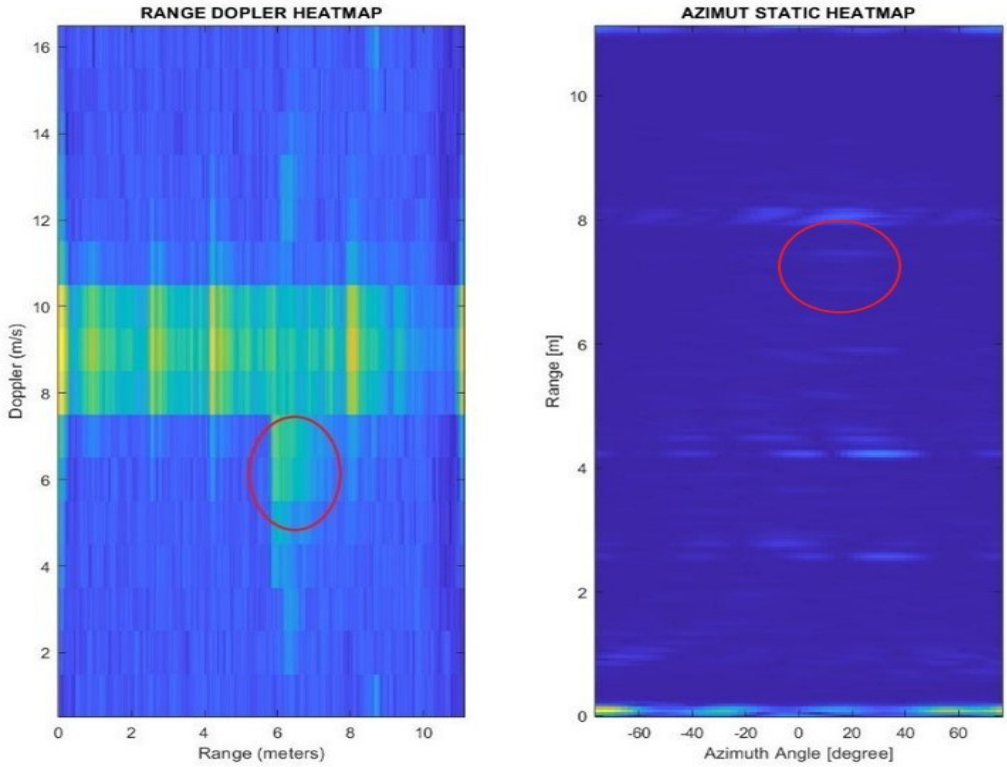
Figure 5.13 *Doppler-Range Heatmap*

5.9 Result and Discussion

5.9.1 Target at 7m Distance

The following results were captured by performing a test in the corridor where a target (human) was walking from 7 meters distance to the mmWave sensor while the results were being logged and plotted as shown in Figure 5.15. From Figure 5.14(a), the range-doppler heat map contained 256 range bins and 16 doppler bins performed a range FFT corresponding to each chirp and after all individual chirps in frame, it performed Doppler

FFT which shown the target movement (red circle) at around 7.0 meters distance in the heatmap.



(a) (b)

Figure 5.14 *Range-doppler Heatmap(a), Azimuth static heatmap(b) for a target (human) walking towards EVM from around 7 meters distance.*

The azimuth-static heatmap in Figure 5.14(b), The result is illustrated in the heatmap for $\mp 60^0$ angular field of view (FoV) with 64 angle bins and $15^0(4R_x, 2T_x)$ azimuth resolution provide the angle of target by using multiple R_x antennas. This plot displays the radar cube matrix for zero doppler only but across all range bins and all antennas. This can be seen from the heat map, The angle of the target within (5-10) degrees at around the 7-meter range.

The target movement can be easily shown in the x-y scatter plot in Figure 5.15 where circle size indicates the doppler and color of the circle indicate the intensity of the target. when target moved towards the EVM, the intensity (color of the dots) becomes higher.

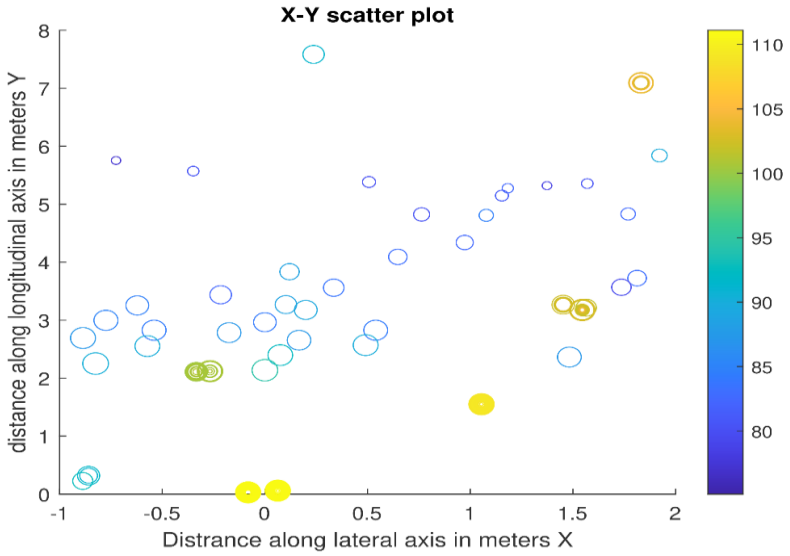
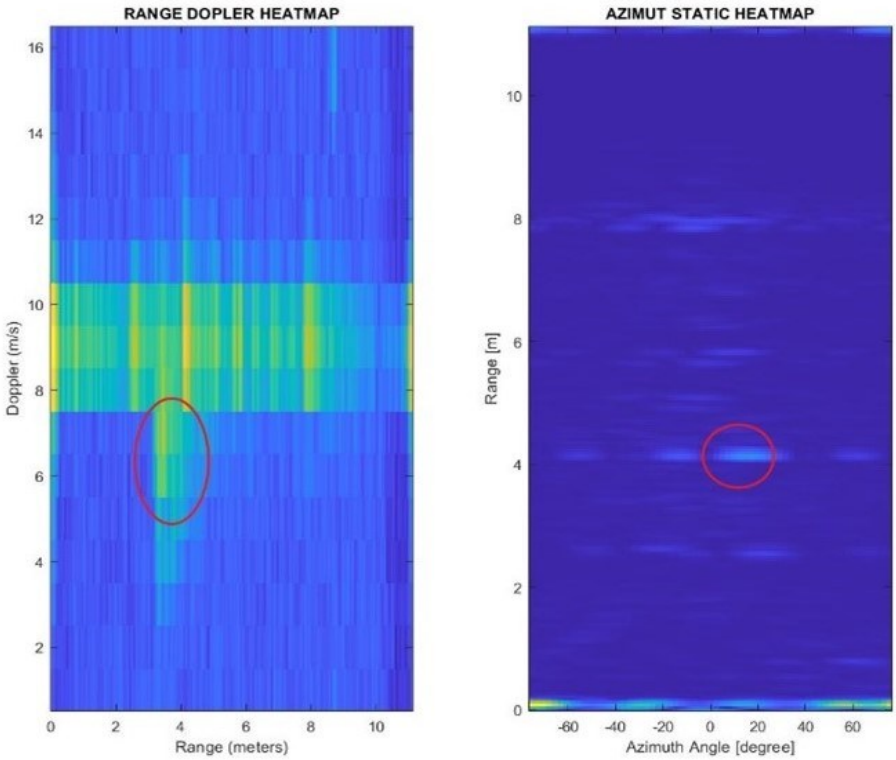


Figure 5.15 *X-Y scatter plot for a moving target towards EVM from 7m.*

5.9.2 Target at 4m Distance

The following results were captured by performing a test in the corridor where a target (human) was walking from 4 meters distance to the mmWave sensor while the results were being logged and plotted as shown in Figure 5.17.



(a) (b)

Figure 5.16 *Range-doppler Heatmap(a), Azimuth static heatmap(b) for a target (human) walking towards EVM from around 4 meters distance.*

From Figure 5.16 (a), the range-doppler heat map contained 256 range bins and 16 doppler bins performed a range FFT corresponding to each chirp and after all individual chirps in frame, it performed Doppler FFT which shown the target movement(red circle) at around 4.0 meters distance in the heatmap.

The azimuth-static heatmap in Figure 5.16 (b), The result is illustrated in the heatmap for $\mp 60^0$ angular field of view (FoV) with 64 angle bins and $15^0(4R_x, 2T_x)$ azimuth resolution provide the angle of target by using multiple R_x antennas. This plot displays the radar cube matrix for zero doppler only but across all range bins and all antennas. This can be seen from the heat map angle of the target within (5-10) degrees at 4-meter.

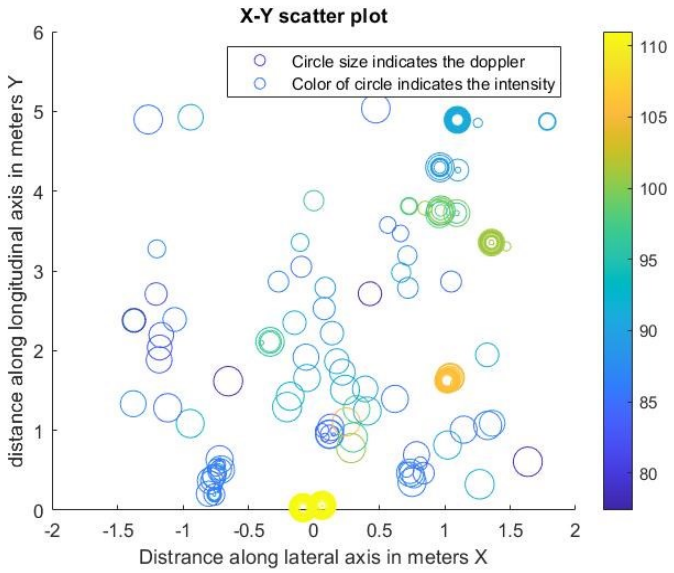


Figure 5.17 *X-Y scatter plot for a moving target towards EVM from 4m.*

Target moves from 4 meters shown in Figure 5.17 with around 100 (log) intensity at the 1-meter lateral axis. Then the Doppler shift was observed at the 1.4-meter lateral axis of the scatter plot and kept moving towards the mmWave sensor. As the target moved towards the EVM, the intensity (color of dots) become higher.

5.9.3 Target at 2m Distance

The following result was captured by performing a test in the corridor where a target (human) was walking from 2.0 meters distance to the mmWave. From figure 5.18 (a), the range-doppler heat map contained 256 range bins and 16 doppler bins performed a range FFT corresponding to each chirp and after all individual chirps in frame, it performed Doppler FFT which shown the target movement(red circle) at around 2.0 meters distance in the heatmap. The azimuth-static heatmap in figure 5.18(b), The result is shown in the heatmap for $\mp 60^0$ angular field of view (FoV) with 64 angle bins and

$15^0(4R_x, 2T_x)$ azimuth resolution provide the angle of target by using multiple R_x antennas. This plot displays the radar cube matrix for zero doppler only but across all range bins and all antennas. This can be seen from the heat map angle of the target within (5-10) degrees at the 2-meter range.

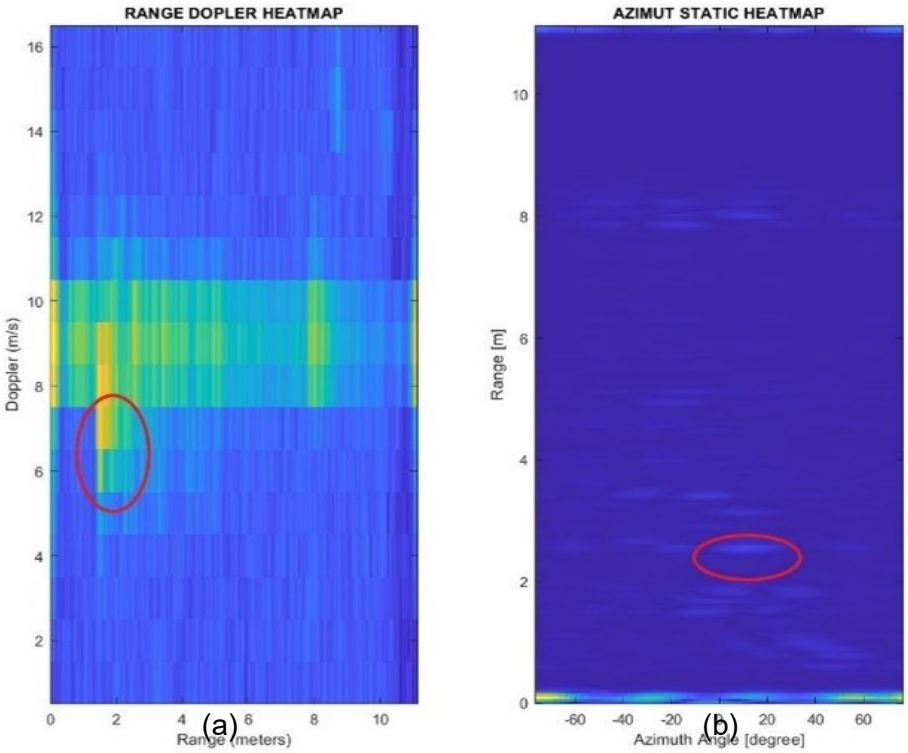


Figure 5.18 *Range-doppler Heatmap(a), Azimuth static heatmap(b) for a target (human) walking towards EVM from 2 meters distance.*

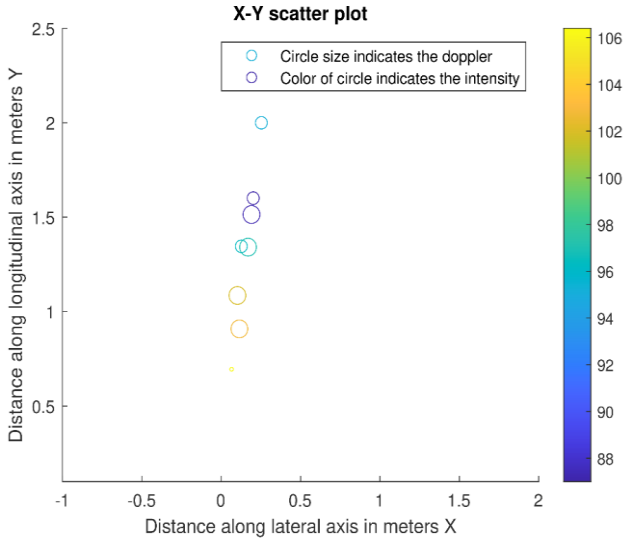


Figure 5.19 *X-Y scatter plot for a moving target towards EVM from 2m.*

The target moved from the 2-meter distance shown in Figure 5.19. As the target moved towards the EVM, the intensity (color of dots) becomes higher.

5.10 Conclusion

The experiments were conducted with mmWave demo visualizer and Matlab by using TI IWR6843ISK evaluation module. Four experiments have been conducted with mmWave demo visualizer while first three experiments were carried out for static target at a distance 1m, 1.5, and 2m respectively and fourth experiment was carried out for walking target towards the EVM from 2m distance and logged data for 5 seconds. The acquired data shows the target detection and movements in the X-Y scatter plot. Similarly, Three experiments were also carried out with Matlab for a target moving from different ranges (2m, 4m, 7m) to the EVM. The real-time human movements are shown in the Azimuth-static heatmap, range-doppler heatmap. The data acquired from the experiments were plotted in the X-Y scatter plot which shows the target path, doppler, and intensity of the target.

6. CONCLUSION

Road safety is an important part of the ITS. The number of accidents especially in the foggy, snowy, raining condition becomes much higher and higher in the vulnerable roads. Through the camera, radar, ultrasound, and light detection and ranging (LiDAR) on commodity or trial vehicles are the most used technology but these technologies is unable to provide sufficient support to the drivers in bad weather conditions. The thesis simulation and experimental result shows the potential of the mmWave radar Micro Doppler ($\mu - D$) in road user recognition.

In the simulation, different kinds of road scenarios are tested by using CNN, LSTM, and CNN-LSTM models. The results show that the two-layer convolutional models outperform the recurrent model (LSTM in this paper). The performance is drop for one-layer CNN model in case of two seconds duration. However, the number of convolutions layers is more important for longer periods signals, and deeper features are required in these cases. The LSTM recurrent model presents stability over different $\mu - D$ durations. However, The number of hidden units have great impact on the performance. The CNN-LSTM model combines merits from both convolutional and recurrent models, especially in the case of random duration $\mu - D$ samples in practical situations.

In the experimental study, the experiments were carried out by using TI IWR6843ISK evaluation module. Experiments have been conducted with the target (human) moving from different ranges (2m, 4m, 7m) to the EVM. The results are shown in the Azimuth-static heatmap, range-doppler heatmap, and X-Y scatter plot. Range-doppler heatmap provides target movement (doppler) in the range and the X-Y scatter plot shows the target path, doppler, and intensity after analysis of the doppler data. These results give a better understanding of the target in real-time in terms of target speed, the angle which could be very useful in the extreme weather condition in the vulnerable roads for vehicle users.

Shallow neural network is used in this thesis for $\mu - D$ recognition. So, Deep neural networks can be used in the future. The experimental study has been conducted for object detection. Henceforth, the experiments can be conducted for $\mu - D$ recognition. Also, these experiments can be extended to monitor the vital signs of a human like heart rate and breathing rate.

REFERENCES

[1] Wakabayashi Daisuke. Self-Driving Uber Car Kills Pedestrian in Arizona, Where Robots Roam. https://www.nytimes.com/2018/03/19/technology/uber-driverless-fatality.html.

[2] Road traffic injuries, https://www.who.int/health-topics/road-safety#tab=tab_1 (accessed November 2, 2020).

[3] T. S. Combs, L. S. Sandt, M. P. Clamann, and N. C. McDonald, “Automated vehicles and pedestrian safety: Exploring the promise and limits of pedestrian detection,” American Journal of Preventive Medicine, vol. 56, no. 1, pp. 1–7, 2019.

[4] L. Daniel, D. Phippen, E. Hoare, A. Stove, M. Cherniakov, and M. Gashinova, “Low-THz radar, lidar and optical imaging through artificially generated fog,” in International Conference on Radar Systems (Radar 2017), Oct 2017, pp. 1–4.

[5] Shingu G, Takizawa K, Ikegami T. Human body detection using MIMO-UWB radar sensor network in an indoor environment. In: Parallel and Distributed Computing, Applications and Technologies, PDCAT Proceedings. 2008, pp. 437–442.

[6] Jia Y, Guo Y, Yan C, et al. Detection and localization for multiple stationary human targets based on cross-correlation of dual-station SFCW radars. Remote Sensing; 11. Epub ahead of print June 1, 2019. DOI: 10.3390/rs11121428.

[7] Chen VC, Li F, Ho S-S, et al. Micro-Doppler Effect in Radar: Phenomenon, Model, and Simulation Study.

[8] Singh AK, Kim YH. Analysis of Human Kinetics Using Millimeter-Wave Micro-Doppler Radar. In: Procedia Computer Science. Elsevier B.V., 2016, pp. 36–40.

[9] Lu CX, Rosa S, Zhao P, et al. See Through Smoke: Robust Indoor Mapping with Low-cost mmWave Radar, http://arxiv.org/abs/1911.00398 (2019).

[10] Almalioglu Y, Turan M, Lu CX, et al. Milli-RIO: Ego-Motion Estimation with Low-Cost Millimetre-Wave Radar. IEEE Sensors Journal 2020; 1–1.

[11] Q. Chen, B. Tan, K. Chetty, and K. Woodbridge, “Activity recognition based on micro-Doppler signature with in-home Wi-Fi,” in 2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom), Sep. 2016, pp. 1–6.

[12] Q. Chen, K. Chetty, K. Woodbridge, and B. Tan, “Signs of life detection using wireless passive radar,” in 2016 IEEE Radar Conference (RadarConf), May 2016, pp. 1–5.

[13] W. Li, B. Tan, and R. J. Piechocki, “Non-contact breathing detection using passive radar,” in 2016 IEEE International Conference on Communications (ICC), May 2016, pp. 1–6.

[14] Q. Chen, Y. Liu, F. Fioranelli, M. Ritchie, B. Tan, and K. Chetty, “DopNet: A deep convolutional neural network to recognize armed and unarmed human targets,” IEEE Sensors Journal, vol. 19, no. 11, pp. 4160–4172, June 2019.

[15] B. Tan, A. Burrows, R. Piechocki, I. Craddock, Q. Chen, K. Woodbridge, and K. Chetty, “Wi-Fi based passive human motion sensing for in-home healthcare applications,” in 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), Dec 2015, pp. 609–614.

[16] J. Huang, Y.-H. Lee, T.-W. Lin, Y.-L. Chen, H.-W. Tseng, and Y.-S. Ho, “The use of Doppler effect in early warning system for vehicle collision at crossroad,” Microsystem Technologies, July 2019.

[17] J. Fang, H. Meng, H. Zhang, and X. Wang, “A low-cost vehicle detection and classification system based on unmodulated continuous-wave radar,” in 2007 IEEE Intelligent Transportation Systems Conference, Sep. 2007, pp. 715–720.

[18] K. J. Lee, C. Park, and B. Lee, “Tracking driver’s heart rate by continuouswave Doppler radar,” in 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Aug 2016, pp. 5417–5420.

[19] D. Belgiovane and C. Chen, “Micro-Doppler characteristics of pedestrians and bicycles for automotive radar sensors at 77 GHz,” in 2017 11th European Conference on Antennas and Propagation (EUCAP), March 2017, pp. 2912–2916.

[20] Mahafza BR. Radar Systems Analysis and Design Using MATLAB. 2000.

[21] Bhatta NP, Geethapriya M. Closed Loop Control of Soft Switched Forward Converter Using Intelligent Controller IJCTA, https://www.researchgate.net/publication/317427496 (2016). (accessed September 7, 2020).

[22] Factors Affecting Radar Performance., https://www.britannica.com/technology/radar/-Factors-affecting-radar-performance. (accessed September 7, 2020).

[23] Frequency-Modulated Continuous-Wave Radar (FMCW Radar), https://www.radartutorial.eu/02.basics/Frequency%20Modulated%20Continuous%20Wave%20Radar.en.html (accessed September 7, 2020).

[24] Radar CI, Manager A, Rao S. The fundamentals of millimeter wave sensors the fundamentals of millimeter wave sensors 2. 2017.

[25] Design of an FMCW radar baseband signal processing system for automotive application, https://springerplus.springeropen.com/articles/10.1186/s40064-015-1583-5 (accessed September 3, 2020).

[26] SAS Institute Inc. Machine Learning, https://www.sas.com/en_us/insights/analytics/machine-learning.html (accessed October 2, 2020).

[27] Oravec M. Feature Extraction and Classification by Machine Learning Methods for Biometric Recognition of Face and Iris, http://code.google.com/p/biosandbox (2014).

[28] Feature Extraction Techniques, https://towardsdatascience.com/feature-extraction-techniques-d619b56e31be (accessed October 17, 2020).

[29] Support Vector Machines (SVM) — An Overview, https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989 (accessed October 21, 2020).

[30] SVM: Feature Selection and Kernels, https://towardsdatasc ence.com/svm-feature-selection-and-kernels-840781cc1a6c#:~:text=The%20main%20objective%20in%20SVM,a%20two%2Ddimensional%20plane). (accessed October 22, 2020).

[31] Random Forest Simple Explanation, https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d (accessed October 18, 2020).

[32] Which one to use - RandomForest vs SVM vs KNN, https://discuss.analyticsvidhya.com/t/which-one-to-use-randomforest-vs-svm-vs-knn/2897/3 (accessed October 16, 2020).

[33] Understand the Fundamentals of the K-Nearest Neighbors (KNN) Algorithm, https://heartbeat.fritz.ai/understand-the-fundamentals-of-the-k-nearest-neighbors-knn-algorithm-533dc0c2f45a (accessed October 13, 2020).

[34] Convolutional Neural Network, https://se.mathworks.com/solutions/deep-learning/convolutional-neural-network.html (accessed October 19, 2020).

[35] Kingma DP, Ba J. Adam: A Method for Stochastic Optimization, http://arxiv.org/abs/1412.6980 (2014).

[36] Y. le Cun. Generalization and Network Design Stratyegies.

[37] Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Computation 1997; 9: 1735–1780.

[38] Beginner's Guide to RNN & LSTMs, https://medium.com/@humble_bee/rnn-recurrent-neural-networks-lstm-842ba7205bbf (accessed October 1, 2020).

[39] Li W, Tan B, Piechocki RJ. Non-contact breathing detection using passive radar. In: 2016 IEEE International Conference on Communications, ICC 2016. Institute of Electrical and Electronics Engineers Inc., 2016. Epub ahead of print July 12, 2016. DOI: 10.1109/ICC.2016.7511389.

[40] R. Boulic, N. M. Thalmann, and D. Thalmann, "A global human walking model with real-time kinematic personification," The Visual Computer, vol. 6, pp. 344–358, Nov 1990.

[41] Matlab, Backscatter Radar signal from pedestrian, 2020. [Online]. Available: https://se.mathworks.com/help/phased/ref/backscatterpedestrian.html.

[42] Backscatter Radar signal from bicyclist, 2020. [Online]. Available: https://se.mathworks.com/help/phased/ref/backscatterbicyclist.html.

[43] User's Guide mmWaveICBoost and Antenna Module, www.ti.com (2018).

[44] User's Guide of mmWave Demo Visualizer, www.ti.com (2017).

[45] mmw Demo Data Structure v0.1 xWR14xx/xWR16xx SDK v1.0.x.x.

APPENDIX: MATLAB FUNCTION TO GENERATE RANGE PROFILE, RANGE-AZIMUTH HEATMAP, RANGE-DOPPLER HEATMAP.

The main Matlab operation has been done for getting range profile, range-azimuth heatmap, range-doppler heatmap, after radar configuration as the following code:

```
function [dataOk, frameNumber, detObj] =
readAndParseData68XX(DATA_sphandle, ConfigParameters)

    OBJ_STRUCT_SIZE_BYTES = 16;
    BYTE_VEC_ACC_MAX_SIZE = 2^16;
    MMWDEMO_UART_MSG_DETECTED_POINTS = 1;
    MMWDEMO_UART_MSG_RANGE_PROFILE   = 2;
    MMWDEMO_UART_MSG_NOISE_PROFILE   = 3;
    MMWDEMO_UART_MSG_DETECTED_POINTS_SIDE_INFO  = 7;
    MMWDEMO_UART_MSG_AZIMUT_STATIC_HEAT_MAP = 4;
    MMWDEMO_RANGE_DOPLER_HEATMAP = 5;
    maxBufferSize = BYTE_VEC_ACC_MAX_SIZE;
    NUM_ANGLE_BINS = 64;

    detObj = [];
    frameNumber = 0;

    persistent byteBuffer
    if isempty(byteBuffer)
        byteBuffer = zeros(maxBufferSize,1);
    end

    global bytevec_log;
    bytevec_log = [];


    persistent byteBufferLength
    if isempty(byteBufferLength)
        byteBufferLength = 0;
    end

    persistent magiNotOkCounter
    if isempty(magiNotOkCounter)
        magiNotOkCounter = 0;
    end

    magicOk = 0;
    dataOk = 0;

    bytesToRead = get(DATA_sphandle,'BytesAvailable');
    if (bytesToRead ~= 0)

        % Read the Data Serial Port
        [bytevec, byteCount] = fread(DATA_sphandle, bytesToRead,
'uint8');
```

```
         % Check if the buffer is not full, and then add the
data to the buffer:
        if(byteBufferLength + byteCount < maxBufferSize)
            byteBuffer(byteBufferLength+1:byteBufferLength +
byteCount) = bytevec(1:byteCount);
            byteBufferLength = byteBufferLength + byteCount;
        end

    end

    % Check that the buffer is not empty:
    if byteBufferLength > 16
        byteBufferStr = char(byteBuffer);

        % Search for the magic number inside the buffer and
check that at least one magic number has been found:
        startIdx = strfind(byteBufferStr', char([2 1 4 3 6 5 8
7]));
        if ~isempty(startIdx)

            % Check the position of the first magic number and
put it at
            the beginning of the buffer
            if length(startIdx) >= 2
                if startIdx(end-1) > 1
                    byteBuffer(1:byteBufferLength-(startIdx(1)-
1)) = byteBuffer(startIdx(1):byteBufferLength);
                    byteBufferLength = byteBufferLength -
(startIdx(1)-1);
                end
            else
                if startIdx(1) > 1
                    byteBuffer(1:byteBufferLength-(startIdx(1)-
1)) = byteBuffer(startIdx(1):byteBufferLength);
                    byteBufferLength = byteBufferLength -
(startIdx(1)-1);
                end
            end
            if byteBufferLength < 0
                byteBufferLength = 0;
            end

            totalPacketLen = sum(byteBuffer(8+4+[1:4]) .* [1 256
65536 16777216]');
            if ((byteBufferLength >= totalPacketLen) &&
(byteBufferLength ~= 0))
                magicOk = 1;
            else
                magicOk = 0;
            end
        end
    end


    if (magicOk == 1)
```

```
        %%%%% HEADER
        word = [1 256 65536 16777216]';
        idx = 0;
        magicNumber = byteBuffer(idx + 1:8);
        idx = idx + 8;
        Header.version = dec2hex(sum(byteBuffer(idx+[1:4]) .*
word));
        idx = idx + 4;
        Header.totalPacketLen = sum(byteBuffer(idx+[1:4]) .*
word);
        idx = idx + 4;
        Header.platform = dec2hex(sum(byteBuffer(idx+[1:4]) .*
word));
        idx = idx + 4;
        Header.frameNumber = sum(byteBuffer(idx+[1:4]) .* word);
        frameNumber = Header.frameNumber;
        idx = idx + 4;
        Header.timeCpuCycles = sum(byteBuffer(idx+[1:4]) .*
word);
        idx = idx + 4;
        Header.numDetectedObj = sum(byteBuffer(idx+[1:4]) .*
word);
        idx = idx + 4;
        Header.numTLVs = sum(byteBuffer(idx+[1:4]) .* word);
        idx = idx + 4;
        Header.subFrameNumber = sum(byteBuffer(idx+[1:4]) .*
word);
        idx = idx + 4;


        %%%%% TLV

        % Analyze each of TLV (Type length value) messages:
        for tlvIdx = 1:Header.numTLVs
            word = [1 256 65536 16777216]';
            % First, analyze the TLV header (TLV type and
length):
            tlv.type = sum(byteBuffer(idx+(1:4)) .* word);
            idx = idx + 4;
            tlv.length = sum(byteBuffer(idx+(1:4)) .* word);
            idx = idx + 4;

            % Check that the TLV message is of the right type
(Detected objects):
            switch tlv.type
                case MMWDEMO_UART_MSG_DETECTED_POINTS
                    detObj =[];

                    if tlv.length > 0
                        % Extract the raw data for all the de-
tected points
                        bytes = byteBuffer(idx+(1:Header.num-
DetectedObj*OBJ_STRUCT_SIZE_BYTES));
                        idx = idx + Header.num-
DetectedObj*OBJ_STRUCT_SIZE_BYTES;
```

```
                        % Reshape the array to have the data for
each point
                        % (X,Y,Z,doppler) in each column
                        bytes = reshape(bytes,
OBJ_STRUCT_SIZE_BYTES, Header.numDetectedObj);


                        % Convert the byte matrix to float data
                        floatData = reshape(typecast(re-
shape(uint8(bytes), 1, []), 'single'),4,Header.numDetectedObj);


                        detObj.numObj = Header.numDetectedObj;
                        detObj.x = floatData(1,:);
                        detObj.y = floatData(2,:);
                        detObj.z = floatData(3,:);
                        detObj.doppler = floatData(4,:);

                    end

                case MMWDEMO_UART_MSG_DETECTED_POINTS_SIDE_INFO

                    if tlv.length > 0
                        bytes = byteBuffer(idx+(1:Header.num-
DetectedObj*4));
                        idx = idx + Header.numDetectedObj*4;

                        % Reshape the array to have the data for
each point
                        % (snr,noise) in each column
                        bytes = reshape(bytes, 4, Header.num-
DetectedObj);

                        % Convert the byte matrix to float data
                        floatData = reshape(typecast(re-
shape(uint8(bytes), 1, []), 'int16'),2,Header.numDetectedObj);
                        detObj.snr = floatData(1,:);
                        detObj.noise = floatData(2,:);

                        dataOk = 1;

                    end

                    case MMWDEMO_UART_MSG_RANGE_PROFILE
                    rp = byteBuffer(idx+(1:tlv.length));
                    idx = idx + tlv.length;
                    rp=rp(1:2:end)+rp(2:2:end)*256;
                    subplot(1,3,1)
                    plot(rp);
                    xlabel('Range (meters)');
                    title('RANGE PROFILE')


                case MMWDEMO_RANGE_DOPLER_HEATMAP
                    len = ConfigParameters.numDopplerBins  *
ConfigParameters.numRangeBins * 2;
```

```
                    rangeDoppler = byteBuffer(idx+(1:len));
                    idx = idx + len;
                    rangeDoppler = rangeDoppler(1:2:end) +
rangeDoppler(2:2:end)*256;
                    rangeDoppler = reshape(rangeDoppler, Con-
figParameters.numDopplerBins, ConfigParameters.numRangeBins);
                    rangeDoppler = fftshift(rangeDoppler,1);

                    rangeArray= (0:ConfigParameters.num-
RangeBins-1) * ConfigParameters.rangeIdxToMeters;
                    dopplerArrey = ConfigParameters.dopplerReso-
lutionMps * ConfigParameters.numDopplerBins / 2;

                    subplot(1,3,2)
                    imagesc(rangeArray,dopplerArrey,rangeDop-
pler);
                    set(gca,'YDir','normal')
                    xlabel('Range (meters)');
                    ylabel('Doppler (m/s)');
                    title('RANGE DOPLER HEATMAP');


                    case MMWDEMO_UART_MSG_AZIMUT_STATIC_HEAT_MAP
                    numBytes = 2 * 4 * ConfigParameters.num-
RangeBins * 4;
                    q = byteBuffer(idx+(1:numBytes));
                    idx = idx + numBytes;
                    q = q(1:2:end)+q(2:2:end)*2^8;
                    q(q>32767) = q(q>32767) - 65536;
                    q = q(1:2:end)+1j*q(2:2:end);
                    q = reshape(q, 2*4, ConfigParameters.num-
RangeBins);
                    Q = fft(q, NUM_ANGLE_BINS);  % column based
NUM_ANGLE_BINS-point fft, padded with zeros
                    QQ=fftshift(abs(Q),1);
                    QQ=QQ.';

                    QQ=QQ(:,2:end);
                    QQ=fliplr(QQ);
                    theta = asind((-NUM_ANGLE_BINS/2+1 :
NUM_ANGLE_BINS/2-1)'*(2/NUM_ANGLE_BINS));
                    range = (0:ConfigParameters.numRangeBins-1)
* ConfigParameters.rangeIdxToMeters;
                    subplot(1,3,3)
                    imagesc(theta, range, QQ, [0,max(QQ(:))]);
                    set(gca,'YDir','normal')
                    xlabel('Azimuth Angle [degree]');
                    ylabel('Range [m]');
                    title('AZIMUT STATIC HEATMAP');

                end
            end



            %Remove processed data
```

```
        if idx > 0
            shiftSize = Header.totalPacketLen;
            byteBuffer(1: byteBufferLength-shiftSize) =
byteBuffer(shiftSize+1:byteBufferLength);
            byteBufferLength = byteBufferLength - shiftSize;
            if byteBufferLength < 0
                %             fprintf('Error: bytevec_cp_len <
bytevecAccLen, %d %d \n', bytevec_cp_len, bytevecAccLen)
                byteBufferLength = 0;
            end
        end

    else
        magiNotOkCounter = magiNotOkCounter + 1;
```