**Department of Electrical, Computer, and Software Engineering**

**Part IV Research Project**

Compendium Report

Project Number: 77

Millimeter Wave Radar based Human Activity Recognition

Beck Busch

Samuel Mason

16/10/2023

## Declaration of Originality

This report is my own unaided work and was not copied from nor written in collaboration with any other person.

Name: Beck Busch

Name: Samuel Mason

**ABSTRACT:** Human activity recognition is an essential area of study, with numerous consequences for areas of technology such as human computer interaction and safety monitoring. We aimed to improve this technology by developing a new system pipeline capable of accurately detecting sequences of multiple activities. Over the course of this project we researched, designed, developed, and tested our approach. The following report seeks to outline and describe this process.

**GitHub Repository:** [github.com/BeckBusch/mmWave-HAR](https://github.com/BeckBusch/mmWave-HAR)

## Table of Contents

# 1. Introduction

The focus of our research project was the advancement of mmWave human activity recognition. Through preliminary research we quickly discovered that mmWave is an excellent candidate for HAR, since it is theoretically able to match visual detection methods in functionality while reducing the risk to user's privacy. Applications such as aged care monitoring or human computer interaction need to respect the users right to privacy, and many consumers are resistant to technology that involves installing a camera in their private spaces.

Over the course of this project we worked on several areas, such as:

- Researching gaps in existing literature
- Investigating mmWave radar hardware
- Developing custom processes for manipulating and compressing the radar data
- Designing procedures for the capture and handling of data
- Writing a novel ML algorithm

In this report we aim to explain the lifecycle of our project, and detail the work that we undertook.

## 2. Literature Review Reflection

In our literature review we set out to identify gaps in the existing mmWave HAR research. There were several key areas that we identified, including the detection of multi-activity sequences and detecting actions in real time instead of waiting until an activity is completed before inferencing happens. There were also several areas that we identified for our continued research, including the performance of different model variations, data segmentation methods, and how the data is compressed and modified along the pipeline.

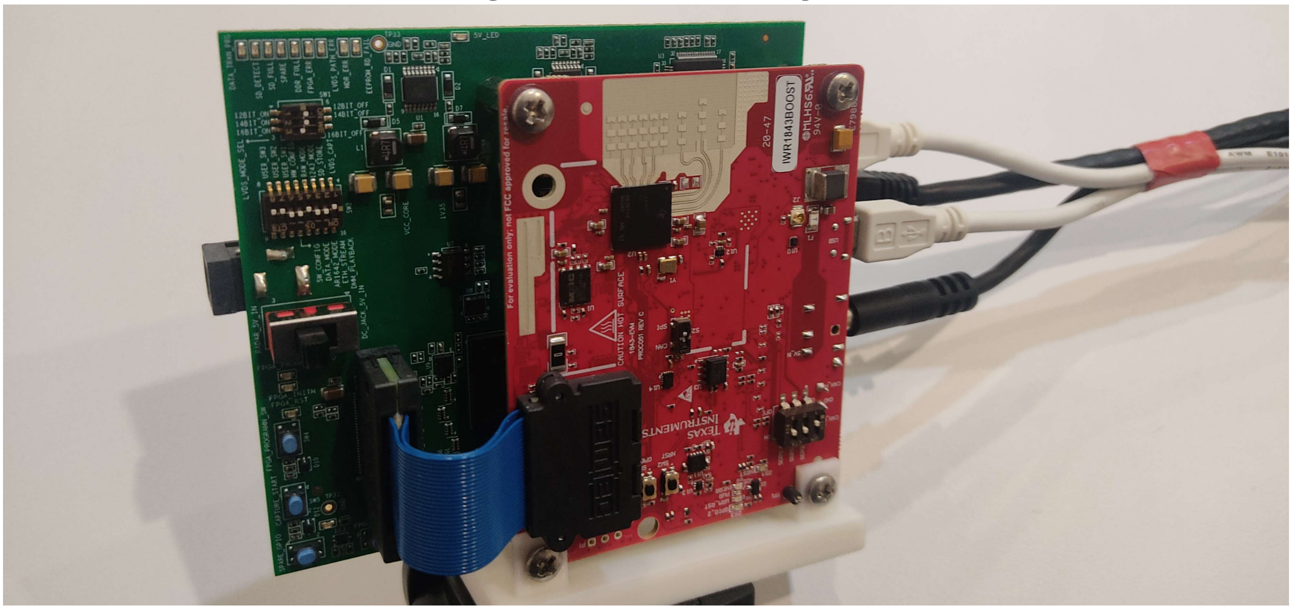
Immediately following the literature review, we had set out to develop an application capable of activity segmentation for complex activities, in real-time. However, the project scope was later reduced to only focus on activity segmentation, due to the difficulties in achieving real-time recognition with a novel method, as well as keeping the project direction more focused.

## 3. Research and Testing

### 3.1. Radar Setup

The first step of our process was to research the available mmWave radar systems, and decide on the best one for our uses. We decided on the IWR1843 due to the high speed recording capabilities, and the number of TX/RX antennas. The IWR1843 is included in the IWR1843BOOST development module, which includes an array of functionalities on top of the actual radar chip. We also decided to utilise the DCA1000EVM data streaming board, allowing us to record the raw ADC data from the radar, without it being preprocessed by the IWR1843BOOST.

*Figure 1: Radar Hardware Setup*




### 3.2 Extraction Script

To create the data extraction script, we first had to configure the premade processing script to function with our hardware setup. This involved modifying the algorithms to function based on a single TX antenna. The other main focus was selecting a beamforming operation that would give us the best results. While we considered range-doppler, we decided that the more “photographic” results of the range-angle transformation would provide the best input to our model.

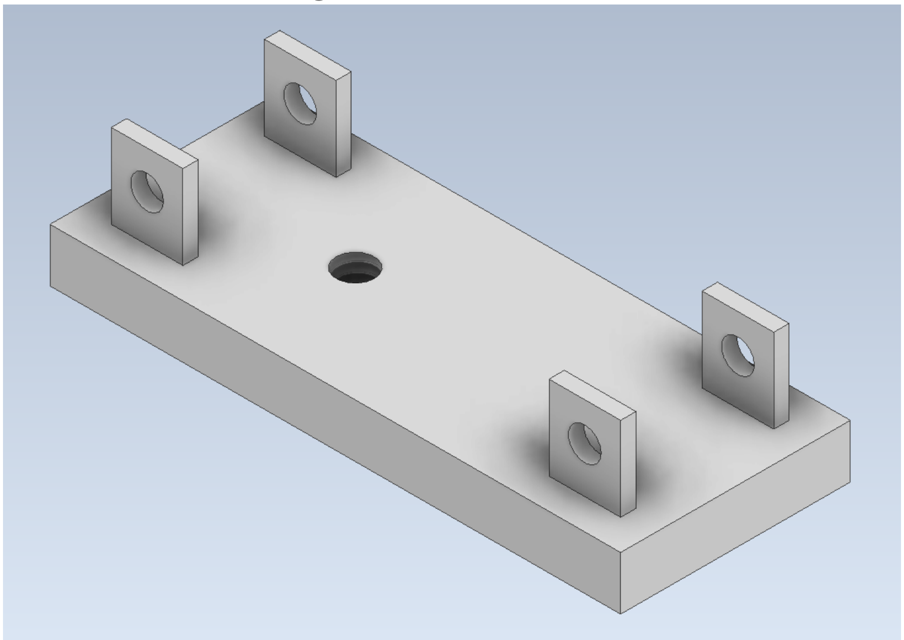
### 3.3. ML Models

The chosen model architecture was derived from findings in the literature review, wherein CNN-LSTM based approaches to the HAR problem typically saw better results than those using other machine learning models. Both 2D and 3D CNN-LSTM have been used in the past, here we opted for a 2D CNN-LSTM model due to slightly reduced complexity, with similar results able to be achieved.

### 3.4. Radar Mount

To hold the radar at a constant position during testing and recording we used a lightweight camera tripod. To attach the radar system to this tripod we modelled and 3D printed a mounting bracket. The mount was designed to take the place of the existing mounting system, attaching at the same points with the same screws.

*FIgure 2: Radar Mount Model*



# 4. Development

## 4.1. Model Development

The initial development stage of the machine learning model(s) was relatively straightforward, the goal was to get the model running and able to make inferences. Once this was completed, hyperparameter tuning was carried out. This process was largely trial and error. Hyperparameters that were modified were the kernel size, number of epochs, number of hidden layers and hidden layer nodes, as well as the number of selected frames for the CNN-LSTM+S model. It was found that for the CNN-LSTM+S model, fewer hidden layer nodes were required compared to the baseline model with the addition of new data. This is likely because the internal model representation can still be accurate even with fewer hidden nodes, due to fewer frames being included in each sequence compared to the baseline. Data augmentation was carried out to increase the size of the training data set for the CNN-LSTM+S model. Our results show that the training process is more stable with the addition of augmented data, and that the model can achieve a higher overall accuracy (compared to using unaugmented data), suggesting that the inclusion of shorter sequences aids in the identification of sequences of all lengths.

## 4.2. Data Collection

Prior to training the model, we needed to collect data from the research team. We decided on 5 actions to be performed: Clapping, Walking, Waving, Jumping Jacks, and Standing still. We collected data from three people, with each participant performing each action 20 times. The data was recorded in the Intelligent Environments Lab, 405.760C, with the participant standing approximately 1.5 metres away from the radar. In the case of the walking activity, the participant walked backwards and forwards across a 3 metre line parallel to the plane of the radar, at the same 1.5 metre distance. For all other activities, these were performed in front of the radar, without any horizontal motion.

# 5. Presentation Reflection

### 5.1. Seminar Presentation

At the time of our seminar presentation we had finished researching and testing the radar devices, and had begun to design the machine learning models. We explained several areas of our project:

- The background of HAR, and the possible applications of this technology.
- Alternative technologies to mmWave, such as visual sensors and CSI, and their drawbacks.
- How millimetre wave radar functions, and the way we are using it in our project.
- Our proposed system pipeline and ML model.

We showcased sample mmWave heatmap data, as well as photos of the physical setup used during data collection.

### 5.2. Exhibition Day Presentation

During the exhibition day we will be showcasing the functionality of our system, by inviting guests to perform actions in front of the radar and attempting to identify both their actions and the transition between two activities. We aim to describe the key elements of our system, the data extraction, frame selection, multiple pass approach, and the CNN-LSTM model.

#### *5.2.3. Display Day Poster*

Our poster was designed with two main goals. The first goal was to serve as a reference for our explanations, and the second goal was to provide an in-depth but digestible description of our system and project. To accomplish these goals, we described various parts of our project in bullet pointed text. We made key words bold to draw the reader's eye across the main points of each line. We also included a hand-drawn diagram of our system pipeline that details each technical aspect of our project. While a hand-drawn diagram may seem out of place in such a presentation, we decided that it was a worthy compromise between clinical design and sufficient explanation or detail. The diagram will serve as a visualisation of the more complex algorithms, with drawn pictures able to explain their function.

## 6. Conclusion

Overall we are quite happy with the outcome of this project. As outlined in our reports, we managed to achieve some very promising results when comparing our novel approach to the baseline model. We were able to create a system that attempts activity sequence detection with an entirely new approach. Frame selection ensures that activities can be recognised regardless of their duration, by extracting relevant frames from the data instead of analysing the entire sequence. As well as abstracting the temporal dimension of the data, the frame selection process compresses the data, dramatically reducing the time taken to run the CNN-LSTM model. This enables our multi-pass system to run at similar speeds to existing approaches. By attaching the "sequence detection" functionality to our multiple pass approach, we remove the need to train a model on every possible border case between two activities. This, if fully realised, would make implementation of this mmWave HAR easier and more efficient, and mmWave technology could see more real-world applications.

## 7. Project Replication Information

- mmWave Studio
  - mmWave Studio is a program by Texas Instruments used to program and operate the radar boards used in this project. This program requires the MATLAB runtime engine to be installed, specifically version 8.5.1 to work with our hardware.
  - UNI FLASH is also required to upload the firmware images to the IWR1843BOOST
  - The network device used to interface with the DCA1000EVM needs to have custom IPV4 rules set | STATIC IP: 192.168.33.30 | SUBNET MASK: 255.255.255.0

The full process for setting up the radar environment is outlined in the Environment Setup Steps document, found in \Documentation\System Documentation\

- MATLAB Studio
  - Matlab Studio is used to run the data extraction script. The Radar Toolbox needs to be installed. This will usually be installed alongside mmWave Studio.
- Python
  - Python 3.11.3 was used for the scripts in this project.
  - Install necessary dependencies for each of the python scripts. These can be installed with pip install <package_name> in the command line on windows. (Requires that pip for python be installed first).
  - Packages that were used in this project are: pytorch, numpy, sklearn.preprocessing, sklearn.utils, enum, joblib, easygui, os, shutil, csv, pandas

## 8. Submitted File List

| File Name | Description |
|---|---|
| Main\Data Collection\ FileManagement.py | Python script that relocates the radar binaries from the mmWave Studio program directory to the SSD storage drive. |
| Main\Data Collection\ mmStudioSetup.xml | XML file used to configure mmWave studio with our chosen settings for radar capture. |
| Main\Data Extraction\ FullProcessRangeAngle.m | MATLAB script that converts the raw radar binaries into a sanitised, compressed, and cropped CSV file of data. |
| Main\Data Extraction\ rawDataReader.m | Modified version of a script provided by Texas Instruments that converts raw radar binaries into matlab matrices. |
| Main\Data Extraction\ FinalCapture\setupExport.setup.json | File exported from mmWave Studio containing information about the radar setup and recording metadata. |
| Auxiliary\Radar Mount\ RadarMount.ipt | Autodesk Inventor design file for the radar mount. |
| Main\Classification Scripts\ CNN-LSTM.py | CNN-LSTM implements a 2D CNN-LSTM model, and is used as a baseline model to evaluate the performance of the novel model against. |
| Main\Classification Scripts\ CNN-LSTM+S.py | CNN-LSTM+S implements a 2D CNN-LSTM with the same network parameters as the baseline, but has additional processing for frame selection and can take input radar sequences of varying lengths. |
| Main\Classification Scripts\ model_eval.py | model_eval is an evaluation script that is used for testing single activity sequences, it can be used to observe salient points in activity data, and for making predictions using the saved CNN-LSTM+S model. |
| Main\Preprocessing Scripts\ data_augmenter.py | Performs resampling, to increase the size of the training data set, and improve the accuracy of the model. |
| Main\Preprocessing Scripts\ data_compressor.py | Reduces the size of the input data frames, from 143x35 to 7x11, in order to reduce training and processing time. |
| Main\Preprocessing Scripts\ renamer.py | Script to rename data source files to align with the common naming convention. |
| Main\Shared Resources\ reduced_data.csv | Our training data, which has been processed and compressed. This data is then augmented to enlarge the dataset before training.* |

| File Name | Description |
| --- | --- |
| Main\Trained Models\ CNN-LSTM+S_model.pth | Archived version of our pre-trained model for our novel approach. |
| Main\Trained Models\ CNN-LSTM_model.pth | Archived version of our pre-trained model for the baseline. |

*augmented_data.csv, which contains additional activity samples through a resampling process, was not included in the submission, due to the file size exceeding 100MB.

Other auxiliary files not listed in this table are outlined in a directory readme.md, present in every folder.