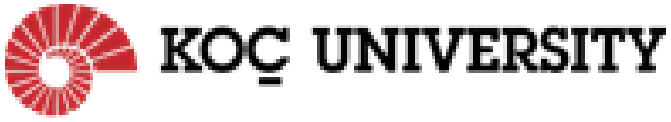

College of Engineering

ELEC 491 – Electrical Engineering Design Project

Final Report

GESTURE RECOGNITION WITH DOPPLER RADARS

Participant information:

Osman Raşit Kültür **37250**

Emre Kurtoğlu **61230**

Project Advisor:

Prof. Hakan Ürey

08.06.2018

Table of Contents

Abstract

The project aims to recognize gestures with Doppler Radars (DRs) which operate at 10.5 GHz. To do that, we had to amplify the output signals, sample and format them in the Micro Controller Unit (MCU) and train the system in MATLAB so that the system can recognize the gestures which are trained so.

Firstly, we placed DRs around each other, then we got the measurements from DRs. After getting measurements, we amplified the DRs' IF (Intermediate Frequency) output signals with special op-amp circuits. After practical measurements were done, data were feed into MCU's Analog to Digital Converter (ADC) Ports.

Secondly, we implemented a Handshake Routine (HR) between the MCU and MATLAB with an Interrupt Service Routine (ISR). Since the DRs we used are Continuous Wave Radars, there is always be available data on the MCU, but we needed to get data which are sampled at a certain rate and length. ISR provided that the data are sampled at 1KHz and formatted into 700 elements long arrays. HR made the formatted data acquisition between the MCU and MATLAB whenever MATLAB needs data. In this way, we were able to create our datasets in MATLAB.

Thirdly, we made a Digital Signal Processing (DSP) and Machine Learning (ML) on the datasets in MATLAB. We firstly extracted the unique features of the dataset of the determined gestures. These features were the data locations of maximum frequency of the signals (FFT), the data locations of maximum value of the envelopes of the signals (Envelope) and the Root-Mean-Square value of the signals (RMS). Then classifying the features of the gestures by using the Support Vector Machine (SVM) Classification Algorithm, the system became a successful hand gesture recognizer. After that, we came up with a simple Graphical User Interface (GUI) which takes test data from the user and runs prediction function with the trained model and returns the predicted gestures.

1. Introduction

In this project, we aimed to design a customizable setup to distinguish different gestures. By doing so, we can assign desired functions such as remote control to specified gestures. Since it is an editable setup, user may identify his/her own gestures and gestures as he/she like.

In the history, people always tried to increase the comfort in daily life. To achieve that they mostly used technological tools such as remote controllers, smart products etc. As we can see from the graph below, average number of remote controllers in a room is more than 5 which is a big number.

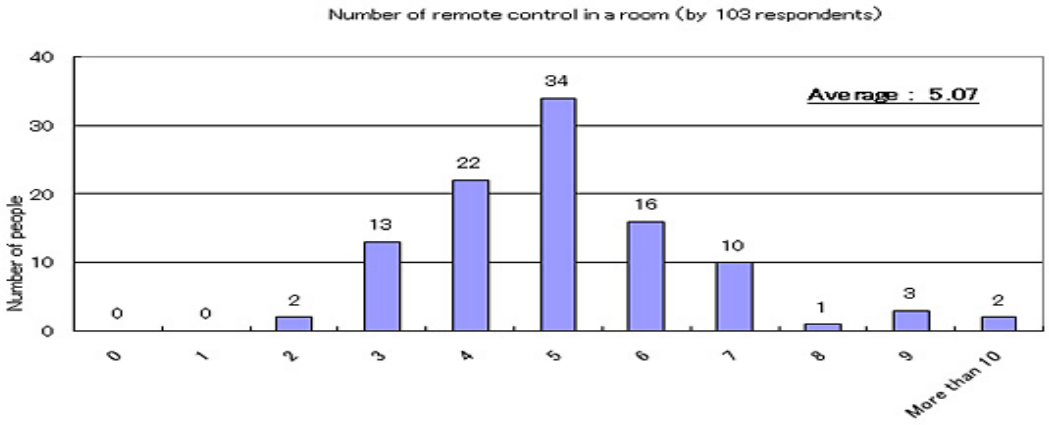

Figure 1: Number of remote controller in a room

From the graph we can understand that people do care about their comfort and our project is designed to contribute this welfare. By using our project, we will be able to control most of the devices in a room with assigned gestures.

To achieve this goal, we used DRs as the sensors. DRs has a transceiver antenna circuit on top of it. What DRs do is, basically, measuring the frequency shift of the transceived signal. If there is any moving object in its range (20 meter), doppler shift occurs.

2. System Design

2.1. Hardware Design

In our setup we used 4 HB100 as DRs and 1 amplifier circuit for each HB100. An overview of HB100 DR is shown in the figure 2.

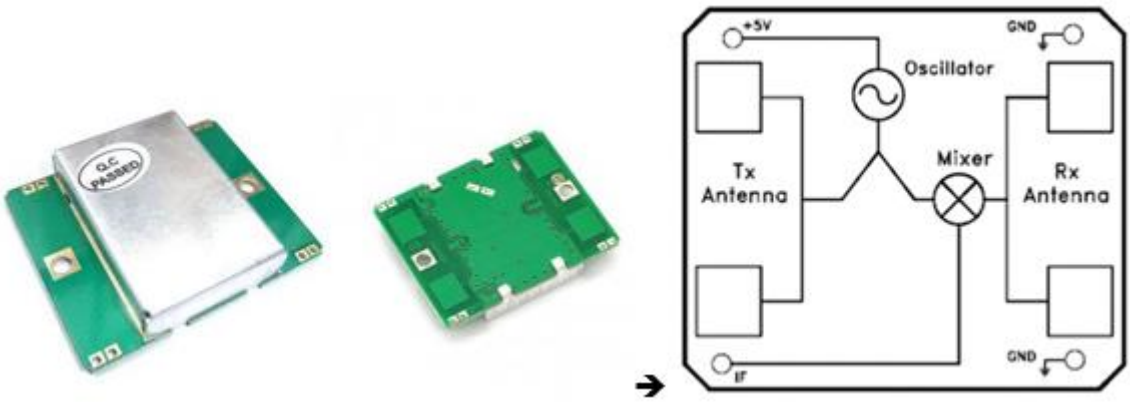

Figure 2: Overview of HB100 Doppler Radar

We used amplifier circuits to amplify IF (Intermediate Frequency) signal of the DRs. After hooking up the DRs to amplifier circuit boards, we assembled boards to hardboards which are shaped as a cubic box with open floor and ceiling. Therefore, we got 2 parallel DRs for right and left, and 2 parallel DRs for up and down. That allowed us to identify direction of the gesture precisely. The overview of the system is shown in the figure 3 as follows.

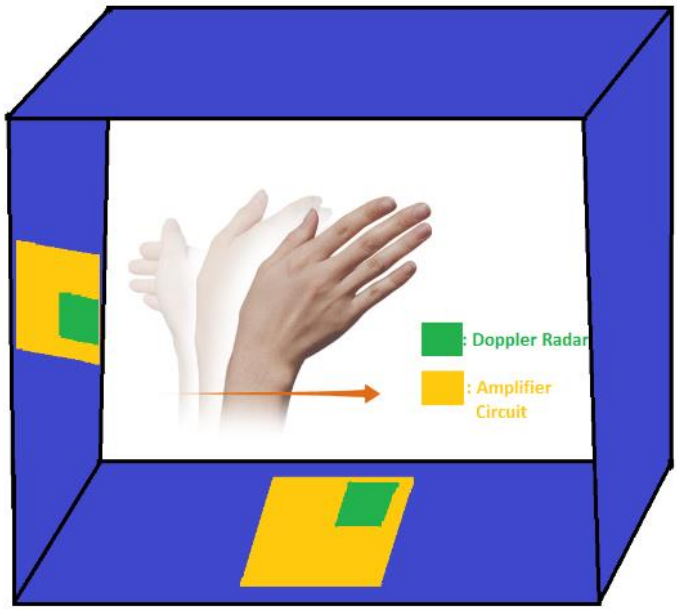

Figure 3: Overview of the system

Since HB100 is a very sensitive DR, we broke one of them by making it touch with soldering iron accidentally. Wiring was one of the most difficult parts of the building amplifier boards. We used copper pertinax boards to fix electronic components and had a lot of short circuits, while soldering cables to components so we had to rebuild some boards again, and we completed all the boards eventually. After completing setup, we connected amplified DR outputs and power inputs to our MCU board, Arduino Mega 2560.

After building the system, we controlled all the connections and DR outputs. They were perfectly working, and system was ready for next step. Schematic of an amplifier circuit is shown in the Figure 4 below. In the amplifier circuits, 2 op-amps with gains of 10 were used to amplify the signals.

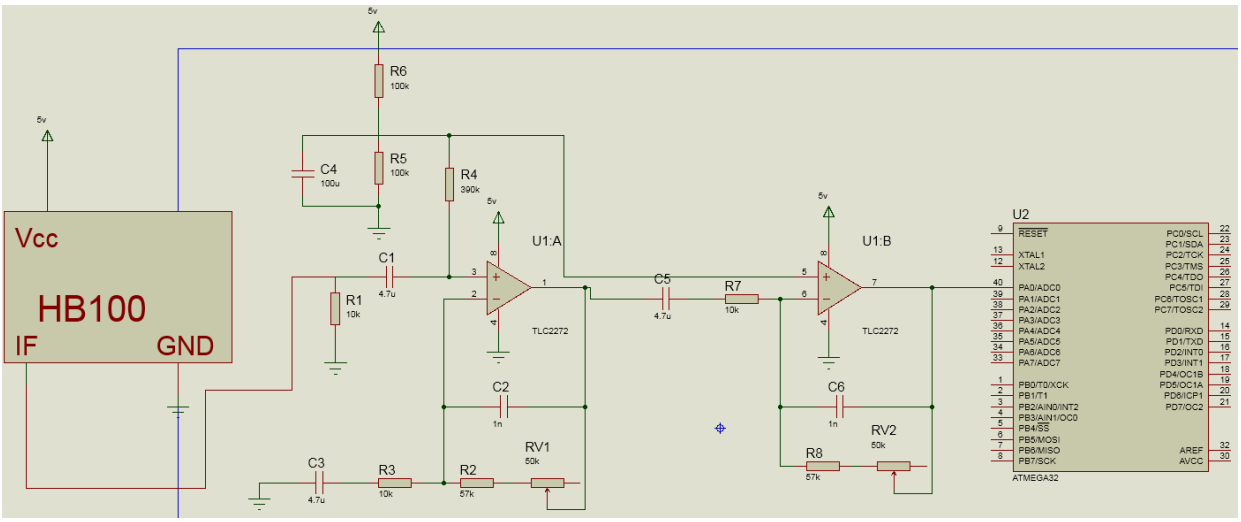

Figure 4: Amplifier Circuit

2.2. Embedded System Part

After getting the data from amplifier circuits into Arduino Mega Board (MCU) which uses ATmega2560 microcontroller, we needed to sample the data at a reasonable rate. The CPU speed of the MCU is 16 MHz. We used Timer1 Interrupt with 1 KHz clock setups by changing the prescalers to sample the data at 1 KHz and 230400 Baud Rate to transmit the formatted data. Timer1 register parameters are shown in the figure 5 as follows.

TCCR1B – Timer/Counter 1 Control Register B

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

CSn2	CSn1	CSn0	Description
0	0	0	No clock source. (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/1 (No prescaling
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on Tn pin. Clock on falling edge
1	1	1	External clock source on Tn pin. Clock on rising edge

Figure 5: Timer1 register parameters of ATmega2560

After having 1 KHz sampling rate, we implemented an ISR along with the communication with MATLAB. ISR allowed a data formatting such that it samples ADC outputs at 1 KHz and puts the samples into 700 elements long 4 arrays (1 for each ADC). When the specific trigger is received from MATLAB, MCU sends the formatted data to MATLAB. In this way, we have collected 50 observations for 7 gestures (right, left, up, down, clockwise, counter clockwise and constant).

2.3. Machine Learning Part

After saving datasets as mat files into MATLAB's workspace, we extracted features of every observation. These features were the data locations of maximum frequency of the signals (FFT), the data locations of maximum value of the envelopes of the signals (Envelope) and the Root-Mean-Square value of the signals (RMS). What makes a gesture different from others is that it has unique FFT, Envelope and RMS features. For further clarification, FFT and RMS features for a gesture are shown in the figure 6 as follows.

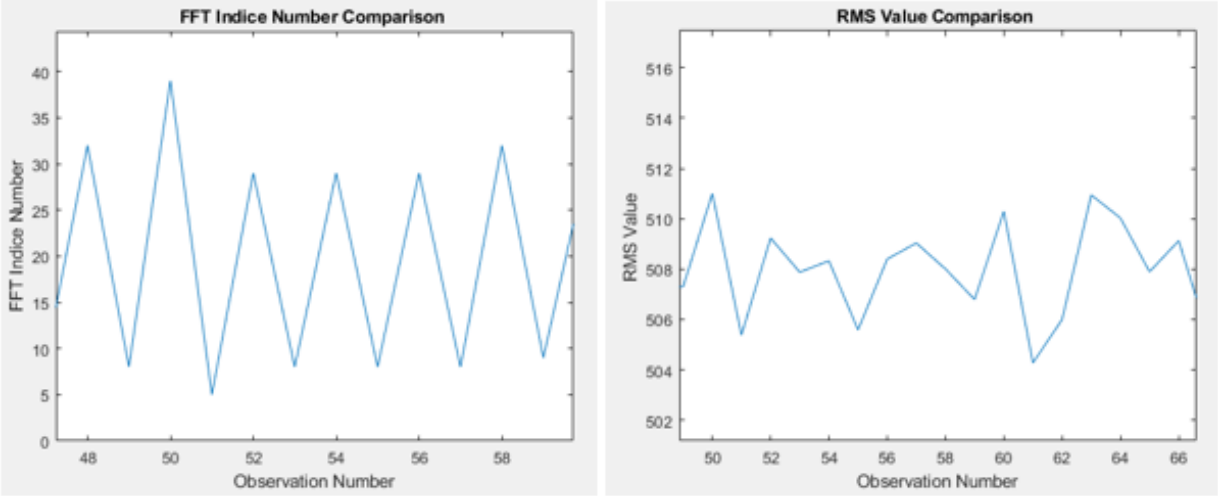

Figure 6: Example of two features of a gesture

After feature extraction, we created a table and filled it with these feature values and corresponding classes(gestures). For further clarification, a part of the table which uses the features as FFT, Mean and RMS is shown in the figure 7 as follows.

TestTable | T

200x4 table

	1 FFT	2 Mean	3 RMS	4 GESTURE
1	9	505.9120	506.9455	'Right_Left'
2	44	504.9300	507.0805	'Up_Down'
3	6	507.1680	508.3370	'Right_Left'
4	35	507.3380	510.1709	'Up_Down'
5	7	504.7460	506.3161	'Right_Left'
6	34	504.5360	507.5060	'Up_Down'
7	11	503.7900	505.2564	'Right_Left'
8	37	506.6880	508.4442	'Up_Down'
9	9	508.4920	509.4193	'Right_Left'
10	33	504.5140	506.4990	'Up_Down'
11	8	503.5740	504.6605	'Right_Left'
12	32	505.5100	508.7659	'Up_Down'
13	4	505.5960	506.7713	'Right_Left'
14	40	501.8000	504.8634	'Up_Down'
15	7	505.7860	506.5033	'Right_Left'
16	42	509.4920	512.9553	'Up_Down'
17	4	505.2380	506.1763	'Right_Left'
18	30	505.7040	508.2148	'Up_Down'
19	11	504.4200	505.4419	'Right_Left'
20	29	506.1200	509.2891	'Up_Down'

Figure 7: Example of a Feature-Gesture Table

Then we used MATLAB's Classification Learner Application to train the system. It contains many kinds of ML algorithms, but we used Medium Gaussian SVM Classification Algorithm which had the best accuracy with 96.9%. SVM works by placing a hyperplane between different classes with optimal distance. After training the system, we obtained a trained model and extracted it into MATLAB's workspace, and we were able to call our prediction function with this model whenever we need. A comprehensive summary of the software design is shown in the figure 8 as follows.

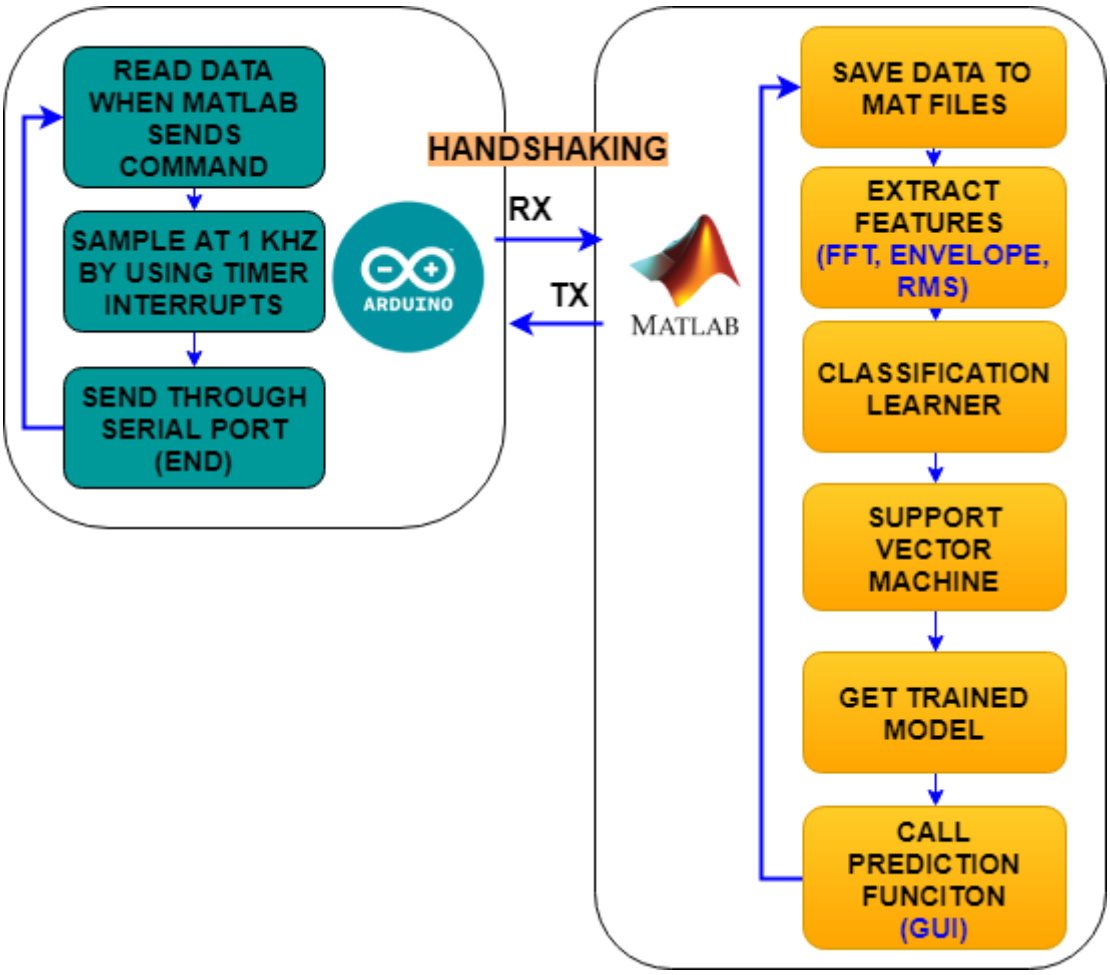

Figure 8: Software Design

After getting the trained model, we came up with a simple Graphical User Interface (GUI) which takes test data from the user and runs prediction function with the trained model and returns the predicted gestures. GUI also generates datasets when "Train Button" is pressed. Overview of the GUI is shown in the figure 9 as follows.

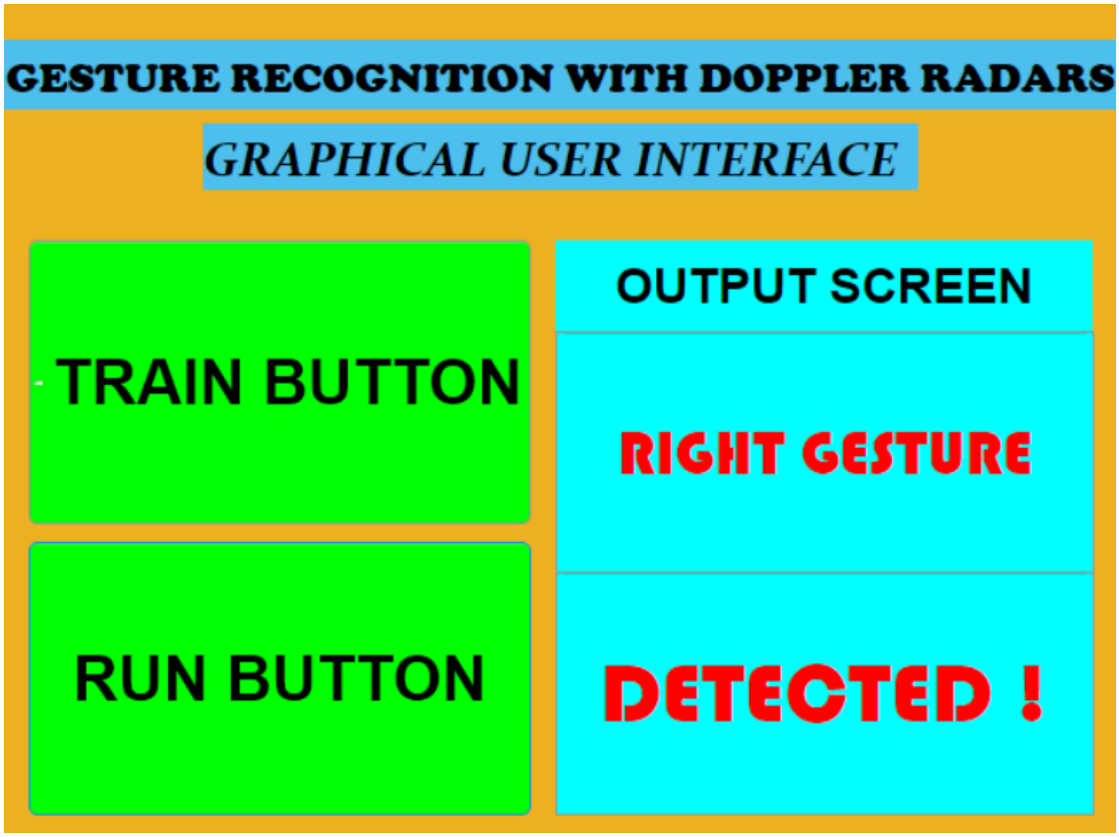

Figure 9: GUI design

3. Analysis and Results

3.1. Results

Generally, system predicted gestures correct but sometimes confusions occurred with clockwise and counter clockwise gestures because of the training data. Since we collected training data for 7 gestures (no gesture state, up, down, right, left, clockwise and counter clockwise) with fast movement, it recognized slowly performed clockwise or counter clockwise gestures as up, down, right or left. To overcome this, user should perform the gesture just as it is train.

3.2. Classification Accuracy Analysis

When training the system, we got most accurate result in Medium Gaussian SVM as shown in the figure 10 below.

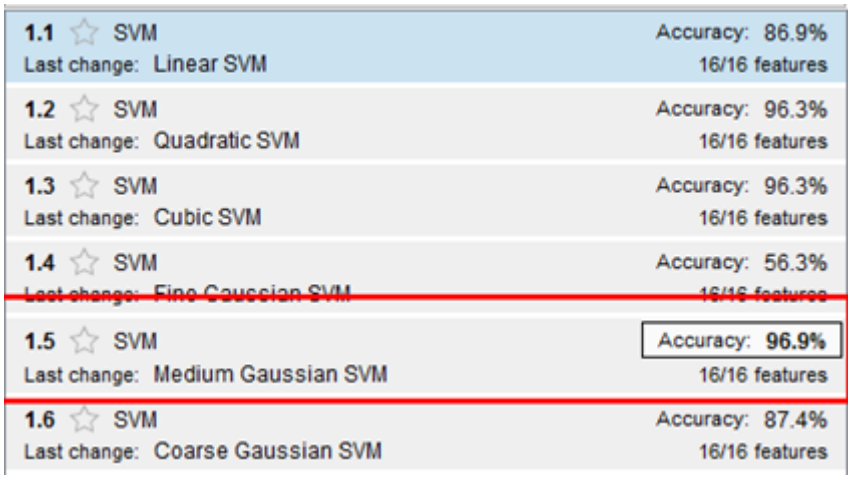

Figure 10: Accuracies of training results of the system

Prediction model of the Medium Gaussian SVM and its consistency with 7 gestures by looking at the FFT feature is given in the figure 11 as follows. As it seen from the different colors, FFT successfully distinguishes between gestures.

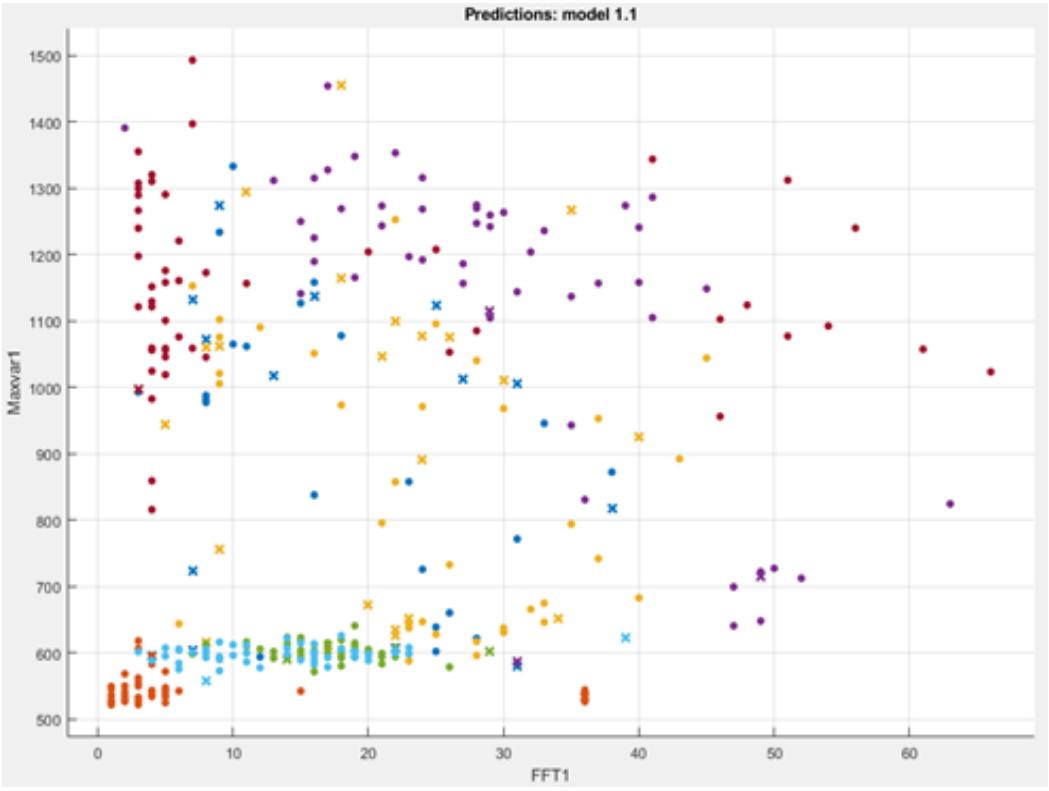

Figure 11: Prediction Model

The performance of the classification is shown in the confusion matrix in the figure 12 below. As we said, we took 50 observations for each 7 gestures. According to this Confusion Matrix, Up and Constant gestures (no gesture is performed) have 100% of accuracy. Whereas, Clockwise gesture has 46% of accuracy as lowest and Counter Clockwise gesture has 74% of accuracy.

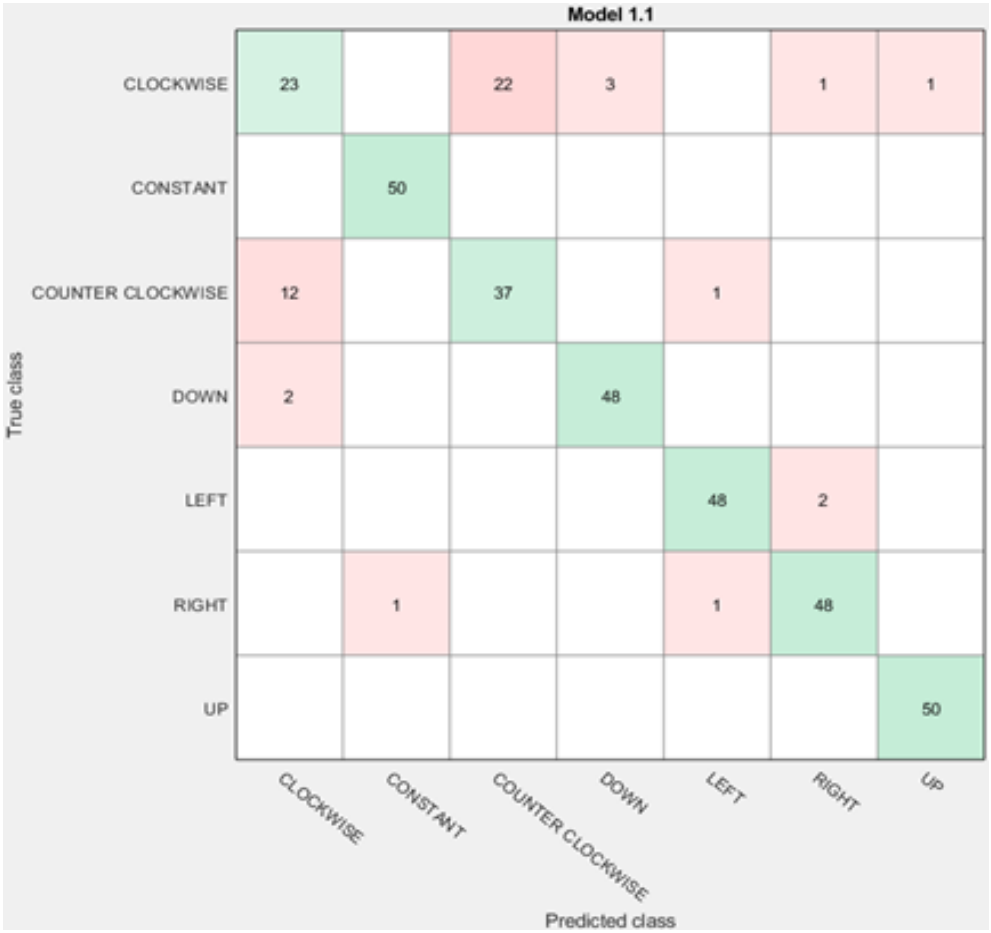

Figure 12: Confusion Matrix

4. Conclusions

In conclusion, it was a comprehensive project which is related to 3 disciplines of engineering: electronic design, embedded systems and machine learning. The amplifier circuits design and their integration with DRs made the DR output signals amplified up to an acceptable range for ADC ports of the microcontroller (which is the range of 0-5 Volts for the ATmega2560 microcontroller). By changing values of the timer interrupt registers to 1 KHz rate and implementing a handshake procedure with the MATLAB, DR data is sampled whenever MATLAB sends a trigger to MCU. Feature extraction and classification of the features made the system well trained for the 7 gestures. GUI is made easy to use the system.

As stated in the goal of the project, the system targeted to distinguishes between the specified gestures. As a result, the project is achieved its goal and the system can distinguish gestures successfully. To get the correct gesture detection, the gestures need to be performed just as they are trained, otherwise the performed gesture could be confused with other specified gestures in the trained model of the system and the system might yield a wrong output.

V. References

[1] Lingyun Ren, Nghia Tran, Farnaz Foroughian, Krishna Naishadham, Jean E. Piou, Ozlem Kilic and Aly E. Fathy 2013. Short-Time State-Space Method for Micro-Doppler Identification of Walking Subject Using UWB Impulse Doppler Radar,IEEE TRANSACTIONS ON MICROWAVE THEORY AND TECHNIQUES.

[2] R. I. A. Harmanny, J. J. M. de Wit and G. Prémel Cabic. 2017. Radar Micro-Doppler Feature Extraction Using the Spectrogram and the Cepstrogram. Proceedings of the 11th European Radar Conference.