

Republic of Iraq
Ministry of Higher Education
and Scientific Research
University of Kerbala
College of Engineering



Design and Analysis of a Powered Ankle Joint Prosthesis

A Thesis

*submitted to the College of Engineering / University of Kerbala
in Partial Fulfillment of the Requirements for the Degree of Master of
Science in Mechanical Engineering
(Applied Mechanics)*

By

Mohammed Fahad JASIM

(B.Sc. 2017)

Supervised by


Prof.Dr Muhsin J. JWEEG

Assist.Prof.Dr. Murtadha A. ALHER

April 18, 2022

EXAMINATION COMMITTEE CERTIFICATION


We certify that we have read the thesis entitled "**Design and Analysis of a Powered Ankle Joint Prosthesis**" and as an examining committee, we examined the student (**Mohammed Fahad Jasim**) in its content and that in our opinion, it meets the standard of a thesis and is adequate for the award of the Degree of Master of Science in Mechanical Engineering / Applied Mechanics.

Signature: 

Name: **Prof. Dr. Muhsin J. Jweeg**

Date: 18/ 4 / 2022

(Supervisor)

Signature: 

Name: **Prof. Dr. Jumaa Salman Chiad**

Date: 24/ 4 / 2022


(Member)

Signature: 

Name: **Assist. Prof. Dr. Murtadha A. Alher**

Date: 18/ 4 / 2022


(Supervisor)

Signature: 

Name: **Asst. Prof. Dr. Mohsin Noori Hamzah**

Date: 24/ 4 / 2022

(Member)

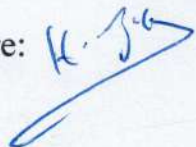
Signature: 

Name: **Prof. Dr. Ahmed Abdullah Hassan Al-Rajihy**

Date: 26/ 4 / 2022

(Chairman)

Approval of the Department of
Mechanical Engineering


Signature: 

Name: **Assist. Prof. Dr. Hayder Jabber
Kurji**

(Head of Mechanical Engineering Dept.)

Date: 26/ 4 / 2022

Approval of Deanery of the college of
Engineering -University of Kerbala

Signature: 

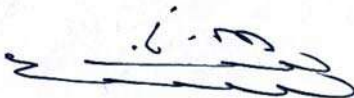
Name: **Prof. Dr. Laith Shakir Rasheed**

(Dean of the College of Engineering)

Date: 28/ 4 / 2022

Supervisors' Certification

We certify that this thesis entitled “**Design, Analysis and Manufacturing of an Ankle Joint Powered Prosthesis**” was prepared by **Mohammed Fahad Jasim** under our supervision at the Mechanical Engineering Department / College of Engineering / University of Kerbala, in partial fulfillment of the requirements for the Degree of Master of Sciences in Mechanical Engineering/ Applied Mechanics.



Signature:

Prof. Dr. Muhsin J. JWEEG

Date: 30 / 11 / 2021



Signature:

Asst. Prof. Dr. Murtadha A. ALHER

Date: 25 / 11 / 2021

Linguistic Certification

I certify that this thesis entitled "**Design, Analysis and Manufacturing of an Ankle Joint Powered Prosthesis**" was prepared by **Mohammed Fahad Jasim** under my linguistic supervision. It was amended to meet the style of the English language.

Signature: 

Linguistic Supervisor *Asst. prof. Dr. Hayder J. K. urji*
College of Engineering
University of Kerbala

Date: *6/3* / 2021

“The Ink of a scholar is holier than the blood of a martyr.”

Prophet Muhammad(peace be upon him)

Abstract

Artificial limbs are an essential device in the life of amputees. The demand for artificial limbs was increased drastically due to the spread of wars, catastrophes, and road accidents. The conventionally available prostheses are passive. The passive prosthesis does not produce a positive network, unlike the natural limbs, which have a positive net work by the muscles-tendon system. Powered prosthesis tries to mimic the function of the natural limb through the use of the powered system. This work aims to investigate, design, and build a powered ankle-foot prosthesis. The optimal design of the powered ankle-foot prosthesis is explored in this work with its valid mathematical model and the design parameters variation. The maximum force that the actuator must exert to maintain a normal gait cycle is calculated in this work. This work covers the design of a powered ankle-foot prosthesis model using Solidworks software based on the chosen mathematical model.

The Gaussian-based mixture model is used to optimize the muscle contraction datasets, corrupted with noise to be applied to the regression process.

The highest regression performance will produce a more accurate angle prediction response of the powered ankle-foot prosthesis. Several Regression techniques were benchmarked to get the optimal regression approach to fit the ankle joint angle dataset. The electromyographical signals were filtered using root means square rectifier, low pass filter, and Gaussian smoothing filter. The proposed method that works based on the

Gaussian mixture model improved the performance of the regression process from 55% to 82% with the use of both the Lowpass filter and the root mean square rectification in the polynomial regression algorithm. The highest regression performance was achieved using Gaussian smoothing filter and the root means square rectifier by the polynomial regression to 84% and by K-nearest neighbor regression to 97%. This means that the created statistical model can predict the ankle joint angle based on the electromyographical signal with 97% accuracy.

Ankle joint angle regression feature selection to get the most effective muscle on the plantarflexion and dorsiflexion movements were studied. The linear correlation coefficient is used to get the most influential muscle on the ankle joint angle. Both Lateral and Medial Gastrocnemius muscles produce the highest correlation coefficient with the ankle joint angle.

Acknowledgements

First, I would like to express my sincere gratitude and thanks to **ALLAH** (be glorious) for the guidance through this work and the blessings bestowed upon me. I would like to gratefully and sincerely thank my supervisors **Prof.Dr Muhsin J.Jweeg**, and **Asst.Prof.Dr.Murtadha A.Alher** for their invaluable help, advice, guidance, understanding, and encouragement during this work. I sincerely thank all the staff of the Mechanical Engineering Department of the University of Kerbela for their support and teaching.I thank my family for their support which made me as ambitious as I wanted, especially my talented brothers, **Dr.Najah Fahad Ghalyan**, and **Dr.Ibrahim F. Jasim Ghalyan**. I hope they always are well and able to get their satisfaction. Finally, and most importantly, I would like to thank all my friends. Their continuous help, support, and encouragement enabled me to complete this thesis.

Contents

Abstract	iii
Acknowledgements	v
1 Introduction	1
1.1 Background	1
1.2 Biomechanics of Ankle Joint	3
1.2.1 Gait Cycle	3
1.2.2 Ankle joint Motion	5
1.2.3 The Importance of the Ankle Joint in the Gait Cycle	7
1.3 Prosthetic Legs	10
1.3.1 Lower Limb Prosthesis Components	10
1.4 Electromyography	11
1.5 Machine Learning	14
1.6 Objectives	16
1.7 Research Contribution	18
1.8 Thesis Structure	19
2 Literature Review	21
2.1 Introduction	21
2.2 Passive and Powered Prosthesis Comparison	21
2.3 Development of Powered Ankle Prosthesis	23
2.3.1 Energy Storage and Return Prosthesis	23

2.3.2	Semi-Active Below Knee prosthesis	23
2.3.3	Active Ankle Foot Prosthesis with Series-elastic Actuator	24
2.4	Human Gait Cycle Classification	26
2.5	Ankle Joint Angle's Regression and Control Strategies . . .	28
2.6	Concluding Remarks	30
3	Theoretical Work and Modeling	31
3.1	Introduction	31
3.2	Ankle Design and Mathematical Models	32
3.2.1	Model 1	32
3.2.2	Model 2	34
3.3	Proposed Model Force Calculations	37
3.3.0.1	Sine Law Trial	39
3.3.0.2	Fixed Crank Distance Trial	39
3.4	Gait Cycle Classification	43
3.4.1	Support Vector Machine(SVM)	43
3.4.2	K-Nearest Neighbor(KNN)	43
3.4.3	Logistic Regression	44
3.5	Ankle Joint Regression	45
3.5.1	Linear Regression	46
3.5.2	Polynomial Regression	49
3.5.3	K-Nearest Neighbors Regression	53
3.6	Regression Models Evaluation	55
3.6.1	Mean Squared Error (MSE)	55
3.6.2	Root Mean Square Error	56
3.6.3	Relative Squared Error (RSE)	57
3.6.4	Coefficient of Determination	57

3.7	Feature Selection	58
3.8	Ankle Joint Angle Formulation	60
3.9	Orientation Measurement	61
3.9.1	Rotation Matrix	62
3.9.2	Single IMU's Orientation Measurement	65
3.10	EMG Signal Processing	69
3.10.1	Median Filter	71
3.10.2	Root Mean Square Filter	72
3.10.3	Fast Fourier Transform	72
3.10.4	Convolution Theorem	78
3.10.5	Digital Filters	81
3.10.6	Butterworth Filter	81
3.10.7	Gaussian Smoothing Filter	83
3.11	Ankle Joint Regression Optimization	86
4	Experimental Work	89
4.1	Introduction	89
4.2	Data Acquisition System	89
4.2.1	NodeMCU	90
4.2.2	IMU (MPU6050)	91
4.2.3	Ankle Joint Angle Orientation Measurement	92
4.2.3.1	Double IMUs' Orientation Measurement	96
4.2.3.2	Ankle Joint Angle Visualization.	98
4.2.4	OpenBCI	102
4.2.5	Enclosures Designs	103
4.3	Powered Ankle-Foot Prosthesis Modelling and Prototype Manufacturing	108
4.4	Below Knee Prosthetic Design	109

4.5	3D Printed Powered Ankle-Foot Prosthetic prototype	121
4.6	Regression Experiments	123
5	Results and Discussions	133
5.1	Introduction	133
5.2	Human Gait Cycle Classification Results	134
5.3	Human Gait Cycle Classification Features Selection	138
5.4	Ankle Joint Regression Based on Electromyographical Signals	143
5.5	Ankle Joint Regression Optimization	151
5.6	Ankle Joint Regression Feature Selection	161
6	Conclusions and Recommendations	165
6.1	Conclusions	165
6.2	Recommendations for Future Work	167
	References	169
A	Matlab and Python Code for Theoretical Work	181
A.1	Linear Actuator Displacement Against Ankle Joint Angle Matlab Plot Code	181
A.2	Ankle Joint Vs Motor Joint angle variation Matlab Plot Code	181
A.3	Ankle Joint Vs Motor Joint angle variation Matlab Plot Code(Constant Crank Length Approach)	182
A.4	the natural range of Ankle Joint Vs Motor Joint angle variation Matlab Plot Code.	183
A.5	the maximum force exerted by the linear actuator for the natural range of ankle joint angles.n Matlab Plot Code.	184
A.6	Lead screw length variation with the change of Z dimension(vertical distance from ankle joint to motor joint) given $J =$ $36.65, l = 60.14$. python Plot Code.	185

A.7	RMS python function..	186
A.8	Low Pass Filter with Butterworth's order variation to plot figure 3.42	187
A.9	Data Matching Python Code	188
A.10	Data Trimming	191
A.11	Data Visualization Linear Regression and Feature Scaling . .	191
A.12	Plotting EMG data in Frequency Domian Using FFT	193
A.13	LowPass Filter	194
A.14	Feature Selection Using Correlation Matrix with Heatmap and Information gain - mutual information	194
A.15	Gaussian Distribution Mean of a Set of Data	196
A.16	Optimized Data Regressions	196
B	Arduino for Experimental Work	207
B.1	Programming NodeMcu-32s to be Server, Access Point with OTA	207
B.2	Arduion Program for Collecting a Single MPU6050 Orientation While enabling Access Point, Local Server and OTA	209
B.3	Arduion Program for Collecting a Multiple MPU6050 Orientation While enabling Access Point, Local Server and OTA Using Single Line Connection	212
B.4	Unity3D Input Controller Script	216
B.5	Unity3D Manager Script	217
C	Extra	221
C.1	Surface EMG Appropriate Installation Position for Both Sagittal and Frontal Plans	221

C.2 Surface EMG Appropriate Installation Position for Both
Sagittal and Frontal Plans 224

List of Figures

1.1	Amputatoinis	3
1.2	Normal Gait Cycle[7].	4
1.3	Normal Gait Cycle[8].	4
1.4	Ankle Joint Movements[9].	6
1.5	Calf Muscle Group.	7
1.6	Muscles of the Anterior[10].	8
1.7	Mechanical Specification of Human Lower Limb During Normal Gait Cycle [9].	9
1.8	Below Knee prosthesis [14].	10
1.9	Electromyography Signal [17].	12
1.10	Surface EMG Installation [17].	13
1.11	Motor Unit [17].	14
1.12	Inductive Reasoning [20].	15
1.13	Objectives.	17
1.14	Objectives.	18
3.1	Model 1.	33
3.2	Circular motion of a linear actuator tip	34
3.3	Model 2.	35
3.4	Model 2.	36
3.5	Model 2 - Linear Actuator Displacement Against Ankle Joint Angle Plot.	37
3.6	Model 2 -Linear Actuator Force Calculation.	38

3.7	Model 2 -Ankle Joint Vs Motor Joint angle variation.	40
3.8	Ankle Joint Vs Motor Joint angle variation(Constant Crank Length Approach).	41
3.9	Natural range of Ankle Joint Vs Motor Joint angle variation.	42
3.10	Maximum force exerted by the linear actuator for the natural range of ankle joint angles.	42
3.11	Linear Model.	45
3.12	Logistic Regression Model.	45
3.13	Scattered Points.	46
3.14	Python Code for Linear Regression.	49
3.15	Polynomial Regression Error Function [21].	50
3.16	Polynomial Regression of Various Orders [21].	50
3.17	Error Value for Various orders [21].	51
3.18	Python Code for Polynomial Regression.	53
3.19	kNN Error Function of the Training Set.	54
3.20	kNN Error Function of The Test Set.	54
3.21	Python Implementation of KNN Regression.	55
3.22	Sciket Learn Man Square Error.	56
3.23	Sciket learns Root Mean Square Error.	56
3.24	Coefficient of Determination in python.	58
3.25	Correlation Cases.	59
3.26	The Two IMUs Installation.	60
3.27	Zero Angle Joint Angle.	61
3.28	Two-dimensional rotation case.	62
3.29	Frame o and 1 illustration.	63
3.30	Rotation of frame 1 around frame o's z-axis with θ angle. . .	64
3.31	Rotation of frame 1 around x-axis with ϕ angle.	65
3.32	EMG Signal Processing.	70

3.33	Suitable Interval of a signal for Median Filter.	71
3.34	Raw EMG with RMS of EMG signal.. . . .	72
3.35	amplitude and phase in frequency domain [55].	74
3.36	Signal sampling [56].	74
3.37	Two Points flow graph.	76
3.38	4-Points DFT Representation in The Flow Graph.	77
3.39	8-Points FFT Representation in The Flow Graph using decimation in time method[58].	79
3.40	Signal Filtering.	81
3.41	Low Pass, Bandpass, High Pass, Notch Filters Represented in frequency domain [60].	83
3.42	Butterworth LPF variation with order.	84
3.43	EMG Signal Filtered with Different Frequencies using HPF & LPFn [60].	85
3.44	impulse response of a GSF having $\sigma = 1$	86
3.45	Solus Muscle Contractions at 10 deg Plantarflexion of Ankle Joint.	87
4.1	NodeMCU ESP-32S GPIO Pins [62].	90
4.2	NodeMCU ESP-32S Access point usage.	91
4.3	MPU6050 Pins.	92
4.4	NodeMcu-32s with MPU6050 Connection.	93
4.5	MPU6050 Enclosure.	94
4.6	MPU6050 Enclosure[65].	94
4.7	MPU6050 and NodeMcu-32s Connection.	95
4.8	Power Bank.	96
4.9	NodeMcu-32s as Access point.	96
4.10	Orientation Illustration.	97

4.11 MPU6050's Installation inside a shoe.	98
4.12 Two MPU6050's Connection with NodeMcu-32s.	99
4.13 Two MPU6050s Illustration.	99
4.14 Two MPU6050's Connection with NodeMcu-32s.	100
4.15 Unity3D UI - Angle Joint Angle Visualization Project.	101
4.16 Data Acquisition Setup.	102
4.17 Foot's IMU Setup.	102
4.18 Ankle Joint Angle Visualization.	103
4.19 Ankle Joint Angle Visualization.	104
4.20 Data Acquisition system Installation.	105
4.21 Open-BCi Ganglion board.	106
4.22 Open-BIC GUI.	106
4.23 Data Acquisition Enclosure.	107
4.24 MPU6050 Enclosure.	107
4.25 MPU6050 Enclosure.	109
4.26 Motor Case.	109
4.27 Motor Pulley.	110
4.28 Lead Screw Pulley with Motor Case.	111
4.29 Motor Cap with Motor Case.	112
4.30 Motor Joint Axle with Motor Case.	112
4.31 Designed Pylon.	113
4.32 Assembled Pylon.	113
4.33 Connector.	114
4.34 Prosthetic Foot.	115
4.35 Prosthetic Foot.	115
4.36 Ankle Joint Axle.	116
4.37 Lead Screw Holder.	116
4.38 Ball Bearings.	117

4.39	Ankle Joint Ball Bearing.	117
4.40	Electronics Enclosure.	118
4.41	Nuts.	119
4.42	SolidWorks Design Model.	120
4.43	3D Printed Powered Ankle-Foot Prosthesis.	121
4.44	3D Printed Powered Ankle-Foot Prosthesis.	122
4.45	3D Printed Powered Ankle-Foot Prosthesis.	122
4.46	3D Printed Powered Ankle-Foot Prosthesis.	123
4.47	EMG Signal Shape Variation Due to Electrodes Location [69].	124
4.48	EMG Electrodes Placement and Electronics Enclosure Position.	128
4.49	Developed Application for Experiments Management. . . .	129
4.50	Ankle Joint Saved File Sample.	129
4.51	OpenBCI EMG Saved File Sample.	129
4.52	Standing.	130
4.53	Rising.	130
4.54	Electrodes Placement.	131
4.55	Down-sampled Raw Data.	131
5.1	Stance phase (A) and Swing phase (B) for a single cycle. . .	135
5.2	Classification technique performance for Raw EMG, EMG with Median filter, and EMG with RMS filter.	138
5.3	Recorded Lower Limb Muscles Activity for Both Stance and Swing Phases.	140
5.4	Recorded Lower Limb Muscles Activity for Both Stance and Swing Phases.	142
5.5	Ankle Joint Regression Experiment(Row data).	144
5.5	Ankle Joint Regression Experiment(Raw data).	144

5.6	Muscles Contractions with RMS Rectification.	145
5.6	Muscles Contractions with RMS Rectification.	145
5.7	Regression technique performance for Raw EMG and Rectified Dataset with RMS Filter.	146
5.8	Actual-Predicted Ankle Joint Angles.	146
5.9	Tibialis Anterior muscle activity both in the time domain and frequency domain.	147
5.10	LPF with $f_c = 2HZ$, and $n = 2$ for all recorded muscles. . .	148
5.10	Application of LPF with $f_c = 2HZ$, and $n = 2$ for all recorded muscles.	149
5.11	LPF with $f_c = 2HZ$, and $n = 2$, with RMS Rectification of 10 Window Size for All Recorded Muscles.	149
5.11	Application of LPF with $f_c = 2HZ$, and $n = 2$, with RMS Rectification of 10 Window Size for All Recorded Muscles. .	150
5.12	Regression technique performance for Raw EMG and Rectified Dataset with RMS Filter.	150
5.13	Recorded Muscles Contractions with Corresponding Ankle Joint Angles.	152
5.14	Proposed Method Illustration.	153
5.15	Application of The Proposed Method with The Raw Dataset.	154
5.16	Application LPF with The Proposed Method.	154
5.16	Application LPF with The Proposed Method.	155
5.17	Application LPF Then RMS Rectification with The Proposed Method.	155
5.17	Application LPF Then RMS Rectification with The Proposed Method.	156
5.18	Regression technique performance for Raw EMG and Rectified Dataset with RMS Filter.	156

5.19 Recorded Muscles Contractions with Its corresponding Ankle Joint Angles.	157
5.20 Gaussian smoothing($\sigma = 5$) Filter Application with Gaussian Averaged Dataset.	157
5.20 Gaussian Smoothing($\sigma = 5$) Filter Application with Gaussian Averaged Dataset.	158
5.21 The Gaussian Smoothing($\sigma = 5$) Filtered Dataset the RMS Rectified.	158
5.21 The Gaussian Smoothing($\sigma = 5$) Filtered Dataset the RMS Rectified.	159
5.22 Regression technique performance for GSF of The Averaged Dataset.	160
5.23 Regression technique performance for GSF and RMS Rectification of The Averaged Dataset.	161
5.24 Correlation Coefficient of Four Muscles Contractions With Ankle Joint Angle.	162
5.25 Gastrocnemius Muscle.	163
5.26 Regression performance for GSF and RMS Rectification of The Averaged Dataset of Lateral Gastrocnimius Muscle and The Four Muscles.	163
C.1 Surface EMG Appropriate Installation Position for The Frontal Plan [17].	222
C.2 Surface EMG Appropriate Installation Position for The Sagittal Plan [17].	223

List of Symbols

$*$	Convolution operator
A_o	Maximum Gain
E	Error
J	Distance between ankle joint to crank joint in vertical direction
N	Sample points number
P	Probability
R	Rotation Matrix
Z	vertical distance of ankle joint to motor joint
i	Imaginary number
k	Ankle joint to motor joint distance
l	Crank length
n	Filter order
r	Linear Correlation coefficient
r^2	Coefficient of Determination
y	Instantaneous linear actuator displacement
\bar{y}	Sample points mean
α	Angle between K and N sides
β	Angle between N and J sides
θ	Ankle joint angle
λ	motor angle
μ	Gaussian Mean
ω	angular frequency

σ Standard deviation

σ^2 Gaussian Variance

ω_0 Angular cut-off frequency

List of Abbreviations

<i>BPF</i>	Bandpass Filter
<i>DFT</i>	Discrete Fourier Transformation
<i>DIT</i>	Decimation in time
<i>DT</i>	Decision Tree
<i>ECG</i>	Electrocardiography
<i>EEG</i>	Electroencephalography
<i>EMG</i>	Electromyography
<i>FFT</i>	Fast Fourier Transformation
<i>FT</i>	Fourier Transformation
<i>GRF</i>	Ground Reaction Force
<i>GSF</i>	Gaussian Smoothing Filter
<i>GUI</i>	Graphical User Interface
<i>HPF</i>	Highpass Filter
<i>IMU</i>	Inertial Measurement Unit
<i>kNN</i>	k-Nearest Neighbors
<i>LPF</i>	Lowpass Filter
<i>LR</i>	Logistic Regression
<i>MSE</i>	Mean Square Error
<i>RF</i>	Random Forest
<i>RMS</i>	Root mean square
<i>RSE</i>	Relative Square Error
<i>SVM</i>	Support Vector Machine

var Variance

Thanks to Allah first for all his gifts.

*To the one who promised to fill the world with the right,
hope this research work could serve his way, to our beloved
imam, Al-Imam Al-Mahdy (peace be upon him).*

*To the one who was supportive of me in this world and, to
the soul of my father (Fahad Jasim).*

*To the one who is the reason for my existence in life (my
dear mother).*

*To the one who filled my life with happiness and pleasure,
my beloved wife.*

*To the candles that burn to light up to others to everyone who
taught me a letter (my teachers).*

*My supportive and inspiring brothers, especially, Falah,
Najah, and Ibrahim.*

*To everyone who helped and encouraged me, my colleagues,
and my friends. I dedicate this humble research, hoping the
Almighty finds acceptance and success.*

Chapter 1

Introduction

1.1 Background

Recently the number of amputees increased rapidly due to the drastic increment of wars, terrorist explosions, catastrophes, road accidents, severe diseases, and trauma that required amputation. The American statistical studies showed that the number of amputees was about 1.6 million in 2005. In 2017, 57.7 million people were living with limb amputation due to traumatic causes worldwide [1]. The prediction studies projected that the number of amputees in the USA will be increased to 3.6 million in 2050 [2]. Most of these limb losses are in the lower limb [3]. The number of amputees has increased during the last four decades in Iraq because of the wars, mine lands and terrorist explosions. According to DW, the number of amputations in Iraq in 2011 was about 100k [4]. As just taken the battles with Daesh in Mosul(2016-2017) as a case, statistics by the United Nations reported 4,800 amputees during these battles [5].

Limb loss significantly impacts amputees' physiological and psychological effects. Several approaches to solve amputees' transportation, such as crutches and wheelchairs. The most used method to solve this issue is the usage of prostheses. A **Prosthesis** is an install of an artificial instrument that replaces the missing biological part of a body. The

objective of a prosthesis is to restore the standard capability of the missing biological piece [6].

Losing any part of the human body is considered as a physical disability, but the loss of mobility greatly influences the human being. Therefore, the lower limb prosthesis is widely studied. Lower limb prosthesis function varies from standing to walking to running. Restoring the aforementioned intuitive abilities will make amputees have a higher life quality. Lower limb prostheses' affordable and improved performance will give amputees a better life, but they have tremendous business demand.

Lower limb prostheses are classified according to the type of amputation into Hemipelvectomy, Hip disarticulation, Above knee amputation(AK), Through the knee amputations, Below knee amputations(BK), Ankle disarticulation, Partial foot amputation, as shown in figure 1.1.

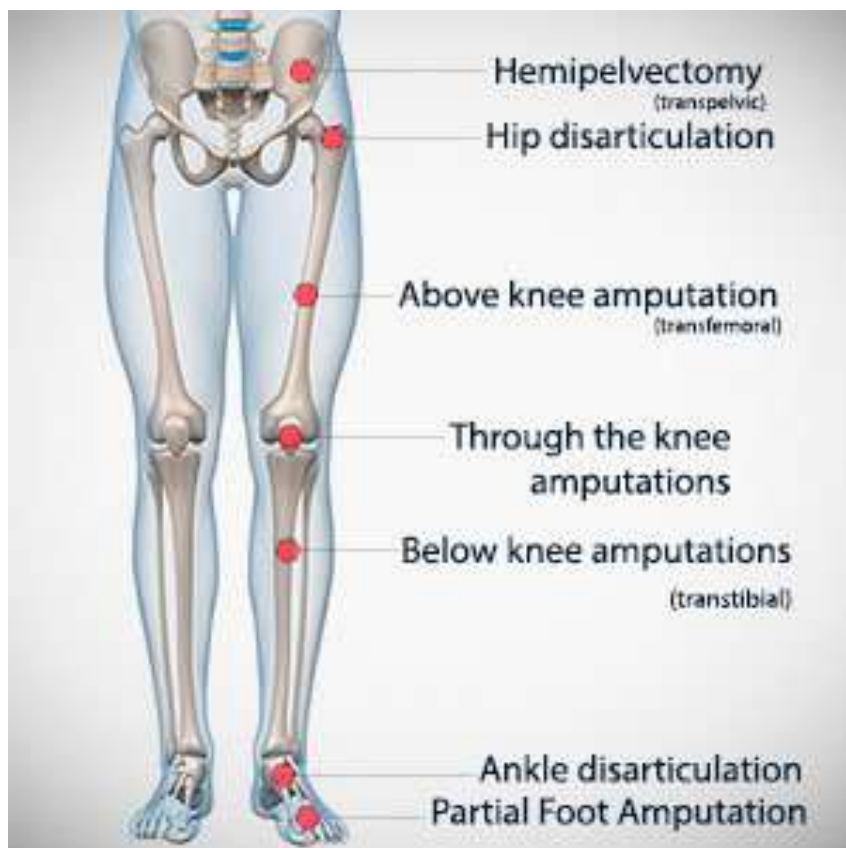


FIGURE 1.1: Lower limb amputation types.

1.2 Biomechanics of Ankle Joint

1.2.1 Gait Cycle

The gait cycle refers to the locomotion of the human body by the movement of the lower limbs. The most available gait cycle for a human being is the normal gait cycle, as shown in figure 1.2. It is considered the perfect and intuitive gait for humans.

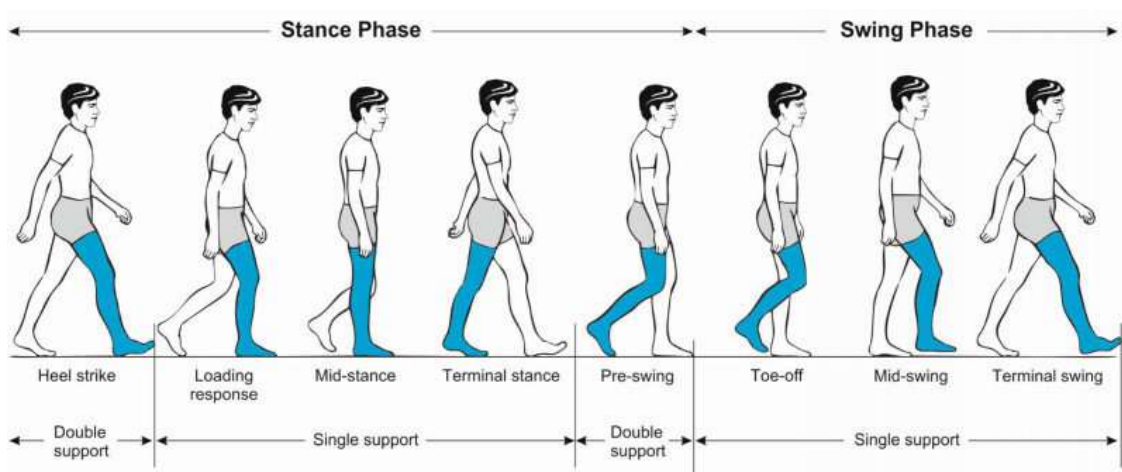


FIGURE 1.2: Normal Gait Cycle[7].

The goal of the gait cycle is the forward progression of the centre of mass of a body. The gait cycle can be defined as the period between the same two consecutive events for one leg. The gait cycle can be divided into a **double support period** and a **single support period**. In **double support period** both limbs are in contact with the ground. Only one limb is in contact with the ground in a **single support period**. The gait cycle also can be divided into the **Stance** and **Swing** periods. In the stance period, the leg is in contact with the ground. In the swing period, the leg is off the ground. The length of a gait cycle is called **stride**. The stance period has 60% of the stride, while the swing period has 40% ,as illustrated in the figure 1.3. The stance period includes the following events:

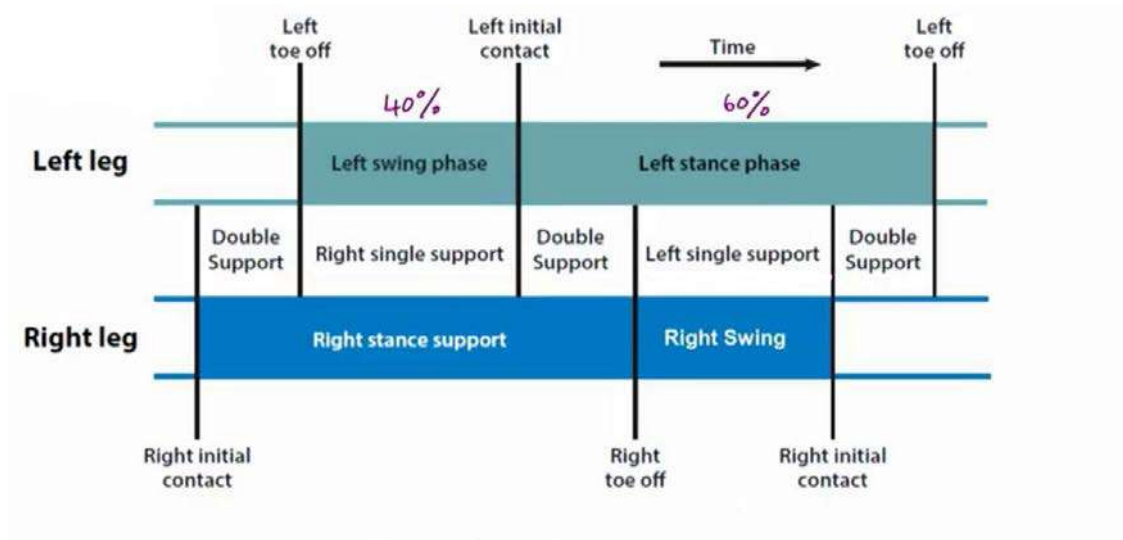


FIGURE 1.3: Normal Gait Cycle[8].

Normal gait cycle phases are as shown in figure 1.2,

- **Heel Strike (Initial contact):** event at which the foot contacts the ground.
- **Load response:** Initial double support event when the leg takes the weight.
- **Mid-stance:** The body moves forward through the supporting limb with load moving to the forefoot.
- **Terminal-stance:** The last single support period ends with opposite initial contact.

The swing period includes the following events:

- **Pre-swing:** swing preparation through which load is moving to the contralateral limb.
- **Initial-swing:** first one-third of the swing period, ends with foot adjacent.

- **Mid-swing:** middle one-third of the swing period, ends with the vertical tibia.
- **Terminal-swing:** the last one-third of the swing period, the knee extends for the next initial contact position.

1.2.2 Ankle joint Motion

Rotation of foot about the ankle joint in the sagittal plane called Dorsiflexion (DF) and Plantarflexion (PF), as shown in figure 1.4. Inversion and Eversion are other ankle movements in the frontal plane. This study will focus on the movements in the sagittal plane (Dorsiflexion and Planterflexion).

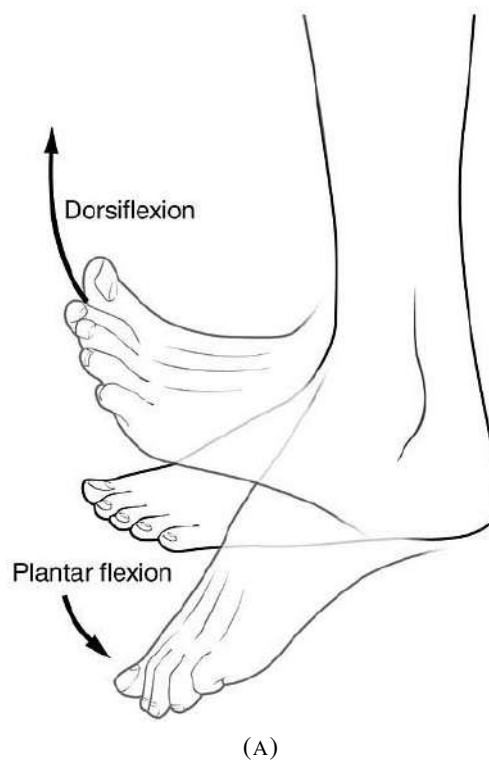


FIGURE 1.4: Ankle Joint Movements[9].

The muscle responsible for the plantar-flexion movement is mainly by the **Gastrocnemius** muscle (plantarflexion prime mover) with the aid of the **Soleus** muscle (located directly beneath the Gastrocnemius muscle) shown in figure 1.5. As mentioned earlier, the combination of the muscles is called

the **Calf** muscle group. Calf Muscle group connected to the skeleton bones by Achilles tendon. Minor muscles contribute to the plantar-flexion movement are Peroneus Longus, peroneus Brevis, Flexor Digitorum Longus, Flexor Hallicus Longus, and Tibialis posterior.

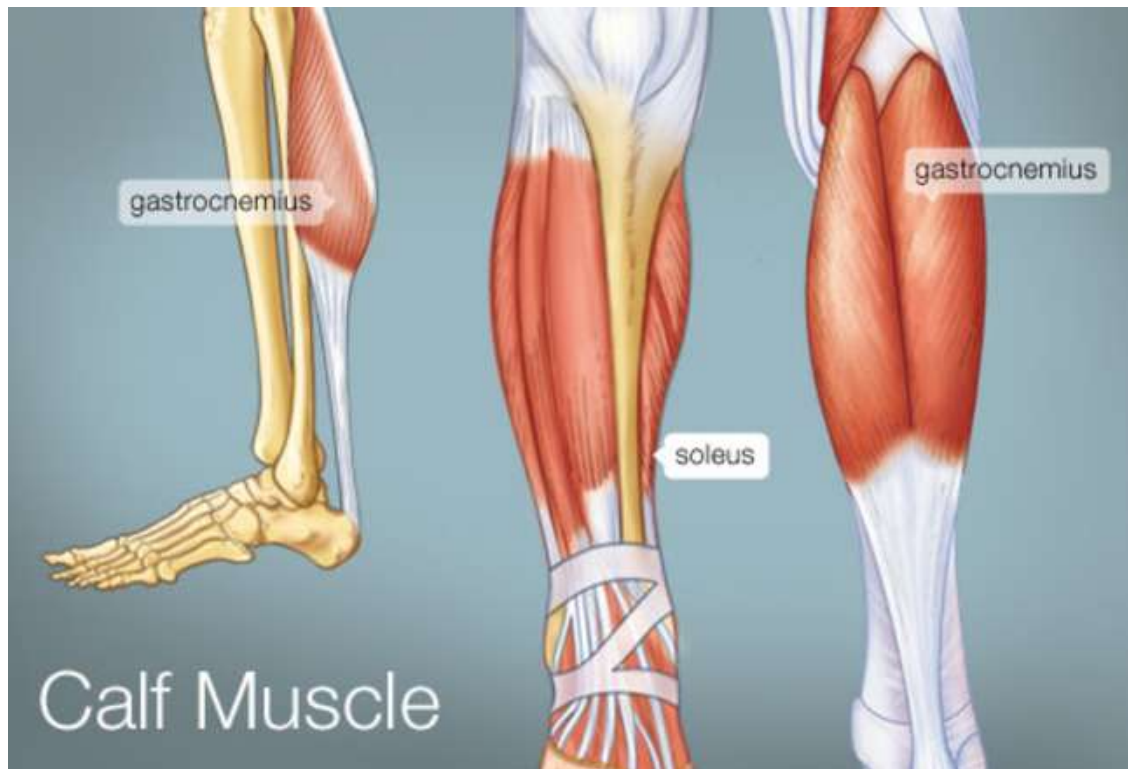


FIGURE 1.5: Calf Muscle Group.

Dorsi-flexion movement, on other hand, is done by the **Tibialis Anterior** muscle (it is also responsible for inversion movement). The Tibialis Anterior muscle is located on the lateral portion of the Tibia bone, and it is connected to the mid-portion of the foot, as shown in figure 1.6. Three additional minor muscles contribute to the Dorsi-flexion movement Extensor Digitorum Longus, Peroneus Tertius, and Extensor Hallucis Longus.

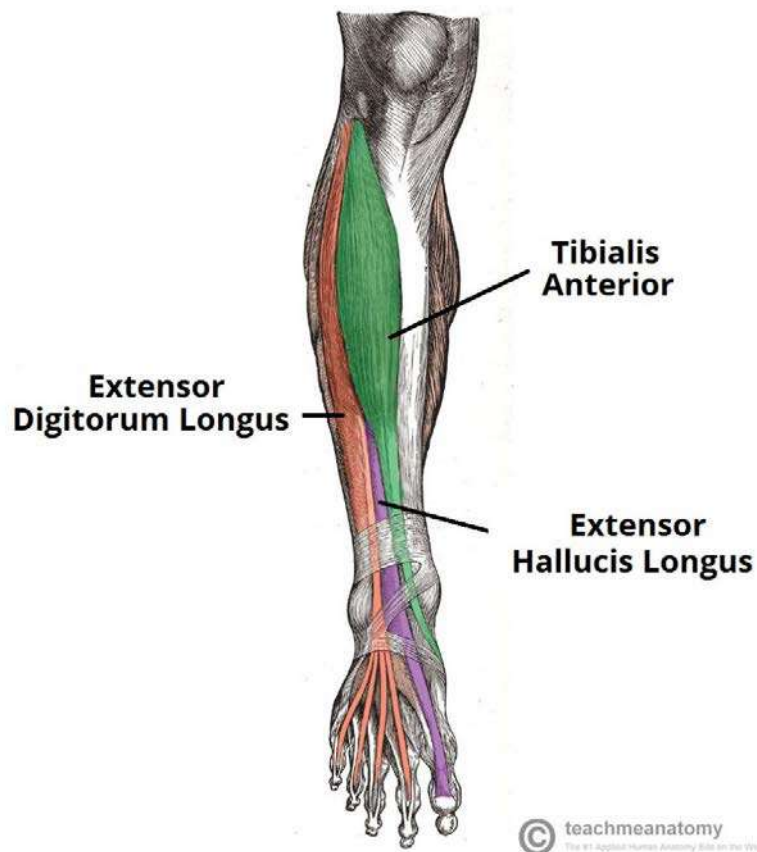


FIGURE 1.6: Muscles of the Anterior[10].

1.2.3 The Importance of the Ankle Joint in the Gait Cycle

Figure 1.7 represent joints angle, internal joints moment, and joints power versus gait cycle stride of the lower limb joints (hip, knee, ankle), respectively. Typically ankle joints pass through several events after the middle stance phase, such as foot flat, maximum DF (at terminal stance), and toe-off (which separate between stance and swing periods). The amount of work and power (done by) at the ankle joint is higher than both knee and hip joints, as shown in figure 1.7(B) and figure 1.7(C). The reason of high powered done by the ankle joint is the rising and progression of the body are majorly done by this joint.

All figures concentrated on the sagittal plane since most of the movements take place in this plane. Ankle angle is between the tibia and

the normal standing position of the foot [11].

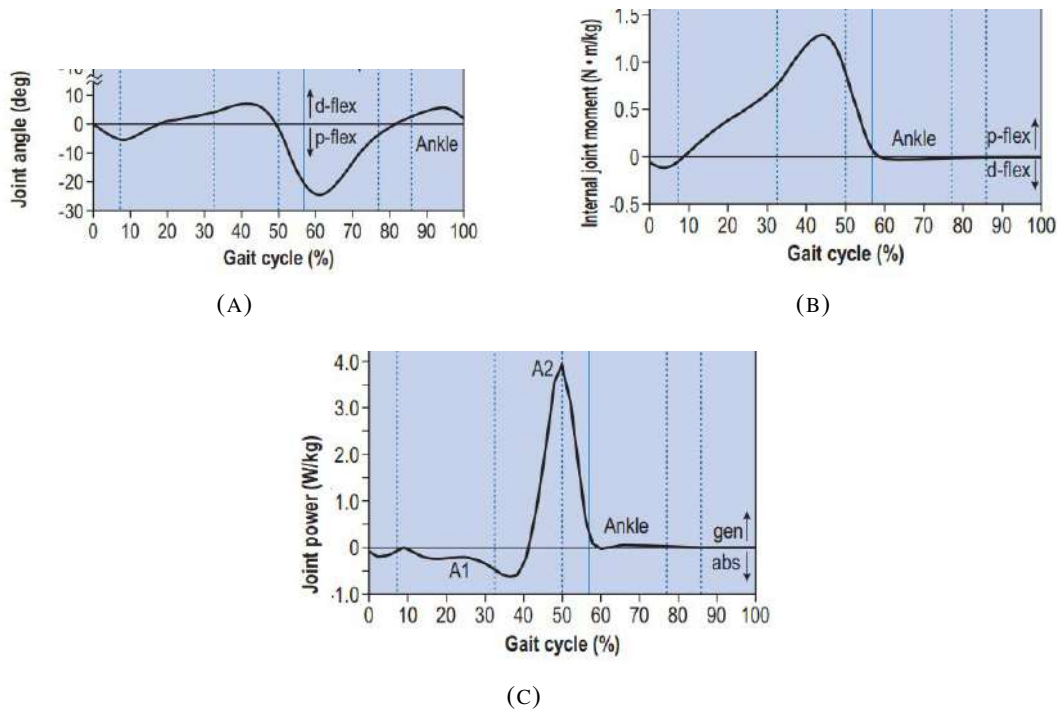


FIGURE 1.7: Mechanical Specification of Human Lower Limb During Normal Gait Cycle [9].

1.3 Prosthetic Legs

Prosthesis manufactured to improve life quality or restore a missing function or appearance [12]. Prosthesis types are as follows:

- Limb prostheses include:
 1. Arm prostheses include above or below the elbow, hand, and finger
 2. Leg prostheses have above or below the knee, foot, and toe.

1.3.1 Lower Limb Prosthesis Components

Most of the prosthetic legs have the same components and as shown in figure 1.8) [13]:

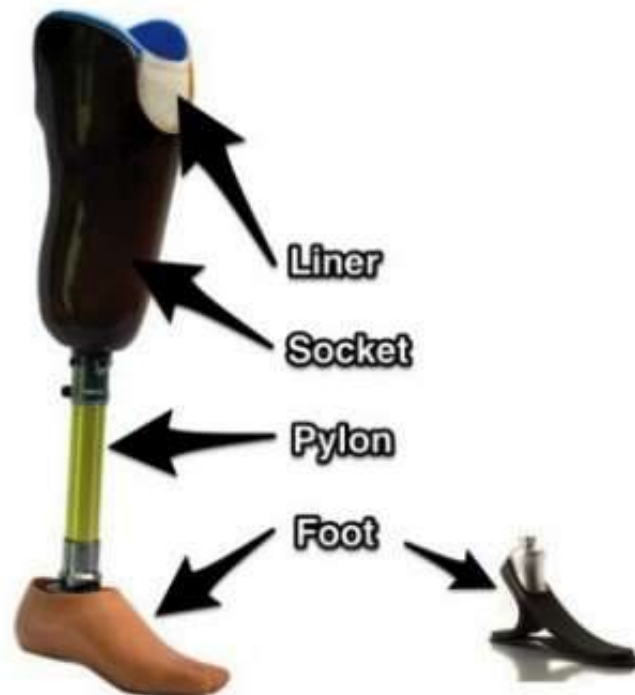


FIGURE 1.8: Below Knee prosthesis [14].

- **Socket**

A plastic or composite material element links a prosthesis to a residual limb part. It should be comfortable and provide a good weight distribution around the residual limb [13].

- **Pylon (shank) and Connectors**

Metal or composite material tube is used to link the socket to the knee joint and ankle joint to the knee joint AK prosthesis or to attach the socket to the ankle joint (BK prosthesis). Connectors are used to connect tubes to sockets or joints. Angle and distance between socket and foot can be adjusted in modular connector type [13].

- **Foot**

Ground attacking element. It should absorb ground reaction forces and grant an equilibrium state during the stance period. A study found more than 16-foot types. The difference between foot types is in the biomechanical durability, and one of the most known problems is their short life (range between 32-16 months) [15].

- **Knee and Ankle Joint**

They are used to replace the functions of a biological joint. They may require special connectors to connect the other components. The ankle joint may be replaced by a flexible plate member or a single-axis joint [13].

1.4 Electromyography

Electromyography (EMG) is a measurement of the electrical activity of a muscle. A device that measures electrical activity is called an electromyograph, and it produces data (or graph) as time domain signal called an electromyogram, as shown in figure 1.9 [16].

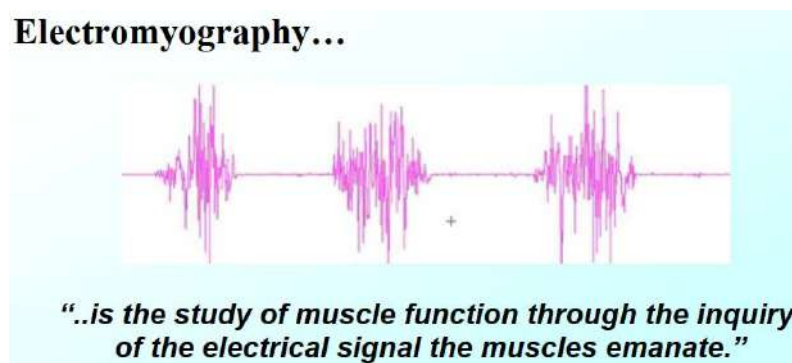


FIGURE 1.9: Electromyography Signal [17].

EMG is less effective in measuring the activity of subcutaneous muscles, unlike superficial muscle, where it is not possible to neglect the effect of

superficial muscles. Fat has a tremendous impact on the EMG signal. The Motor unit is all the muscle fibres connected to a single motor neuron. Motor neurons release a neuron transmitter and fire action potential in the muscular cells (fibres) [18].

Two kinds of EMG devices are used Intramuscular EMG and Surface EMG. **Intramuscular EMG** could be achieved by using various types of electrodes. The most commonly used needle inserted inside muscle under consideration with a surface electrode as a reference or two needles to measure the electrical potential. Intramuscular EMG can measure the electrical activity of a low-level muscle. **Surface EMG** uses more than one electrode (at least two electrodes) or a matrix of plenty of electors. The limitations of surface EMG are that it can measure the electrical activity of only superficial muscles, and it is highly affected by the tissue above muscles. It is recommended to position the electrodes on the belly of the muscle (along its longitudinal center) since it is situated in the middle between the motor unit and tendon effect point, as shown in figure 1.10 [19] (Appendix C.1 for the appropriate surface EMG installation position).

The amplitude of surface EMG signals (sEMG) varies from V to a few mV range. The record produced by EMG is called an **electromyogram**. Myoelectric signals are formed by physiological variation in the state of the muscle fibre membrane. Myoelectric signals are formed by physiological variation in the state of the muscle fibre membrane. To instantiate a contraction, a neuron generates a small electrical potential on the surface of the muscle fibre tissue and produces a waveform. The waveform travels the muscle fibre length and is known as **Action Potential (AP)**. **The Motor unit** is composed of a single motor neuron and all of the muscle cells it stimulates, as shown in figure 1.11.



FIGURE 1.10: Surface EMG Installation [17].

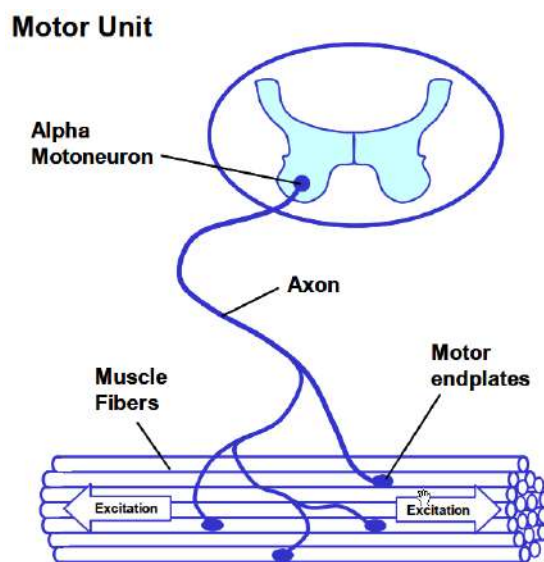


FIGURE 1.11: Motor Unit [17].

The number of muscle cells within a motor unit determines the degree of movement when the motor unit is stimulated. Motor units vary in size, small motor units are used for precise, small movements as motor units that control eyes; large motor units are used for gross movement. The recruitment process stimulates as large as possible motor units to increase muscle contraction.

Action Potential (AP) is the induced voltage difference across a muscle

fibre. However to produce a muscle contraction the neural cell produces a voltage difference on the surface of the muscle fiber, which produces a polarization on the muscle and produces a signal.

1.5 Machine Learning

Machine learning is a subset of artificial intelligence in computer science that often uses statistical techniques to give computers the ability to "learn" with data without being explicitly programmed.

Machine learning algorithms are divided as follows:

1. Supervised Learning

The machine-learning task of learning a function that maps an input to an output based on examples (input-output pairs). It infers a function from labelled training data consisting of training examples.

Supervised learning emerges from a rational principle called Induction reasoning. Inductive reasoning consists of constructing the axioms from observing the supposed consequences of these hypotheses scientists like physicists observe natural phenomena then postulate the law of nature. Inductive reasoning shortly means you move from specific observations to the general rule, for example, the sun rises today and yesterday and before yesterday and so on, therefore it will rise tomorrow. Induction can be illustrated as shown in figure 1.12.

Supervised learning mainly comprises of two approaches, **classification**, and **regression**. Classification is the process of labelling data based on pre-fed labelled data. Regression is the process of mapping output to a pre-trained data set consisting of both pair input-output. Both classification and regression will be explained in detail with the most used algorithms in chapter three [21].

Induction: Given a cause and an effect, induce a rule.

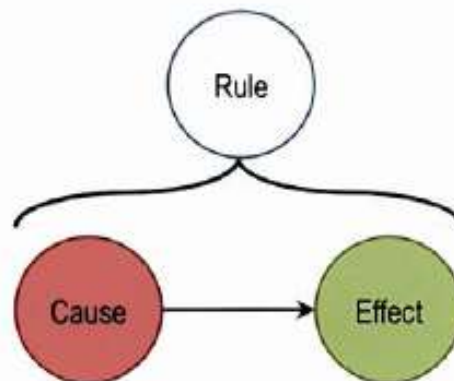


FIGURE 1.12: Inductive Reasoning [20].

2. Unsupervised Learning

Derive some structure from data by looking at the relationship between the input. Unsupervised learning is about creating divisions [21].

3. Reinforcement Learning

The area of machine learning is concerned with how software agents ought to take action in an environment to maximize some notion of cumulative reward [21].

1.6 Objectives

This research aims to design and analyze a below knee powered artificial limb that can mimic the human ankle, and model the EMG signals acquisition from the calf muscles group, in order to predict the ankle-foot prosthesis response. The objective can be broken into two procedures to achieve that objective, Modeling and Prototyping.

1. Modelling

Modelling is done by collecting the activity of the muscles affecting the ankle joint (EMG signals). Filtering the EMG signal has a significant

impact on the purity of the signal; it is crucial in removing noise or bypassing the effect of near muscles. A statistical model of muscle activity and ankle angle is created; a synchronized acquisition system must be designed to develop such a model. The system collects data on muscle activity with an ankle angle simultaneously.

2. Prototyping

Prototyping is done by making a biomechanical analysis of the ankle joint and trying to formulate a mathematical model of the ankle joint, inducing the effect of engineering elements (actuator, transmitter, springs....etc.). Applying standard gait cycle specification on the pre-create ankle joint mathematical model is enough to find prosthetic engineering elements, as shown in figure 1.14.

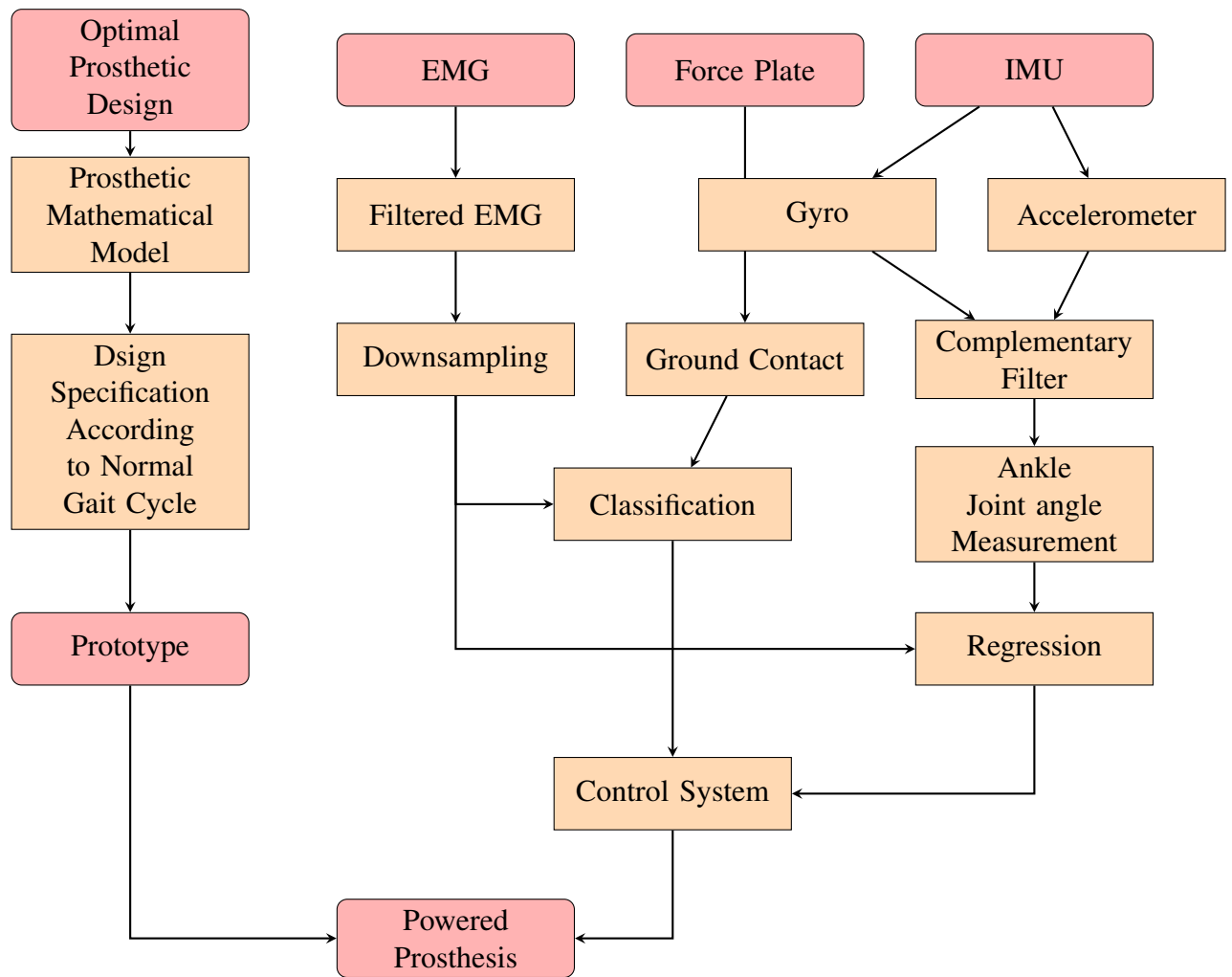


FIGURE 1.14: Objectives.

1.7 Research Contribution

This work introduces a new design of powered ankle-foot prosthesis with its mathematical model and the variation between its design parameters. The classification of the normal gait cycle is achieved with high accuracy. Regression of the ankle joint angle based on EMG signals accompanied with noise is modelled. A proposed hypothesis refined the EMG signals accompanied with noise assumed that EMG signals can be modelled as a Gaussian mixture model.

1.8 Thesis Structure

This work analyses, designs, and manufactures powered ankle-foot prostheses based on EMG signals. The layout of the thesis consists of:

- **Chapter One**

An introduction about the importance of artificial limbs and statistics about the increment numbers of amputation cases. A brief introduction about the gait cycle and lower limb movement is presented. A historical introduction about artificial limbs and their parts and types is covered as well in this chapter. Electromyographical signal is highlighted in this chapter.

- **Chapter Two**

In this chapter, a research survey about the inability of the passive prosthesis to replace the natural limb properly is explained. The emergence of the powered prosthesis and its ability to mimic the intact limb functionality and type of powered prosthesis is covered in this chapter. Human gait cycle classification and ankle joint regression literature are presented in this chapter.

- **Chapter Three**

The proposed design model with its mathematical model is explained in this chapter. It contains the supervised classification algorithms with several simple EMG filtering techniques. The regression algorithms and their evaluation methods with their mathematical background and their programming implementation are also covered in this chapter. The Gaussian distribution and signal filtering in the frequency domain with Lowpass, highpass, bandpass with Fourier

transformation are described. The proposed method for optimizing the EMG-angle regression process is highlighted in this chapter.

- **Chapter Four**

This chapter explains the ankle joint angle and the EMG data acquisition system electronics, connection, installation, programming, and computer interface. The proposed powered ankle-foot design is modelled using SolidWorks software in this chapter. This chapter explains regression experiments to model the ankle joint angle based on the calf muscles group activity.

- **Chapter Five**

This chapter investigates the maximum linear force exerted by the linear actuator of the proposed design. The result of using several classification algorithms with a set of filtering techniques is illustrated in this chapter. The regression algorithms performance with a group of filtering configurations using the proposed optimization method is also benchmarked.

- **Chapter Six**

The main conclusions and future intended research from this work are listed in this chapter.

Chapter 2

Literature Review

2.1 Introduction

Artificial limb marketing increased exponentially due to wars, catastrophes, traumas, accidents, etc. As a result of humanity's needs and the marketing demands, the development of lower limbs became incredibly fast and hard to catch, especially in the past three decades.

This chapter demonstrates the advantages of the powered prosthesis, the adverse passive prosthesis, the available commercial powered prosthesis. This chapter also presents the literature related to the ankle foot-powered prosthesis, gait cycle classification, ankle-joint regression, powered angle foot prosthesis control strategies.

2.2 Passive and Powered Prosthesis Comparison

Most of the available artificial lower limbs are passive; the following studies show the inability of the passive prosthesis to replace the lower limb's function.

Winter and Sienko, 1988 [22] studied the moment and power for (8) lower limb amputees, wearing five SACH feet, two uniaxial, and 1 Greissinger prostheses, a passive prosthesis manufactured by the Ottobock

company. EMG was documented for 3 of the SACH foot amputees case. Grissinger stores the highest energy, about 30%, to the push-off state. The findings showed that the highest metabolic rate consumed in the case of a passive prosthesis is because of the passive nature of its components and its inability to provide adequate work during the stance period.

Bateni and Olney, 2002 [23] analyzed the differences in the gait cycle kinematics and kinetic between the healthy and amputee subjects. The gait cycle experiment parameters were measured using a force plate with a motion capture system to measure the kinematics of subjects, linear velocity, relative angle, net moment, power, and work at all measured at the main lower limb joints. The intact joint limb of the amputees had a large hip extension, knee flexion, and ankle dorsiflexion. They explained the reason behind the most asymmetric walking pattern for amputees is attributed to passive components. The passive components force the amputee to lift the weight of the prosthesis instead of pushing the body up forward. The amputees show a lower power generated at push-off compared with the non-disabled. The deficiencies could be overcome in the case of imposing active components.

Hansen et al., 2004 [24] investigated the change in the slope of the curve moment versus ankle angle for various speeds. The ankle angle was measured using a motion capture system, and the angle moment was recorded using a force plate. 6HZ cut-off frequency for Butterworth bi-directional filter used for filtering the markers trajectory of the motion capture system. They collected their data from 24 natural limbs. They concluded that a spring-damper system could replace the ankle joint for a slow walking speed, while argument components should be used for moderate to fast speeds. They also showed that positive work should be exerted in the intact ankle joint, especially during the stance phase, and a

passive component cannot do this, but an argument system should be used.

2.3 Development of Powered Ankle Prosthesis

2.3.1 Energy Storage and Return Prosthesis

Energy storage and return (ESR) prosthetic feet are the most common energy storage and return prostheses in the market. It has a leaf spring foot with a fixed ankle joint, mainly manufactured from a carbon fibre composite material. The energy is stored during the heel strike and can be exposed during the push-off state. **Gardiner et al., 2016 [25]** showed that this prosthesis is considered a lightweight type and can be used with high safety; it has a low price and is widespread in markets. ESR can be used for various activities such as walking or even running. The difference between these prostheses is the spring stiffness. This prosthesis is the active component that produces positive work during the stance phase. Hence it is irresponsible to rely on ESR feet to restore the natural activity of intact limbs. They showed that COP(cost of transportation) in ESR is only 2.7% higher than SACH foot which is not so high.

2.3.2 Semi-Active Below Knee prosthesis

In this type of prosthesis, the powered components do not directly affect the ankle joint but affect the whole gait cycle. These types of prosthesis have a considerably lower weight of 1.2-1.4 kg and have a higher comfortability than passive prosthesis and significant battery life to be extended to more than 24 hr [26].

Chas A Blatchford & Sons Ltd. [27] developed **Elan** ankle. Elan angle foot has a microprocessor controlling two micromotors that operate two hydraulic cylinders with an adjustable damping ratio during DF and PF.

Ottobuck developed a semi-active ankle-foot prosthesis called **Triton smart** foot [28]. In this prosthesis, the ankle joint rotation is locked or released according to the gait cycle phase and incrementally. **Brochure, 2017** [29] showed that the whole system could be controlled using an embedded microprocessor, and the communication with the user can be done wirelessly. The user can adjust the heel height and ankle joint lock status.

Össur, 2006 [30] developed **Propiro foot**. The ankle in this type of prosthesis is linked to an active actuator driven by a DC motor and controlled by a microcontroller. The actuator can not provide enough power to push the body forward during the stance phase. However, the semi-active ankle is used to lift the foot during the start of the swing phase to improve the amputee gait cycle and provide more safety and good appearance of the walking cycle; also, it can work fine during stairs ascending/descending.

2.3.3 Active Ankle Foot Prosthesis with Series-elastic Actuator

Biomechatronics Group corporate with MIT media lab developed **BIOM** powered ankle-foot prosthesis [31]. BIOM is the first commercially available prosthesis that generates positive work during the stance phase.

The prosthesis uses a 200W DC brushless motor, with a timing belt and ball screw transmission. BIOM is supplied with a rechargeable battery which can ensure walking for 4-5 km.

Au and Herr, 2008 [32]] Explained the series elastic actuator principle, which for BIOM prosthesis, where the actuator is directly connected to a leaf spring foot. They showed that the series elastic actuator principle saves

the body from excessive strikes and protects the motor and the transmission system from shocks. They also studied the energetic cost of transport (COT) amputee using BIOM and it was almost the same as health leg and performed much better than passive prosthesis. BIOM is supplied with a rechargeable battery which can ensure walking for 4-5 Km. They also showed that each step using BIOM consumes 20 J from the battery. BIOM powered prosthesis has the maximum ankle torque at 47% of the gait cycle and its value about 120 Nm.

Au, Weber, and Herr, 2009 [33] Studied the gait cycle for BIOM prosthesis. They showed that the gait cycle is divided into six phases: **controlled plantarflexion** at heel strike, **controlled dorsiflexion** at midstance, **powered plantarflexion** at the terminal stance, and three sub-phases for the swing phase.

They showed, the required phase is triggered by recognition (classification) using four sensors; Force transducer at heel, force transducer at the toe, an encoder to measure the ankle joint rotation, and the ankle torque is measured by a potentiometer consisting of a linear spring displacement and stiffness [33].

Ferris et al., 2012 [34] made a comparative evaluation of BIOM powered prosthesis with a passive prosthesis and an intact limb. BIOM shows a higher power generated at the terminal stance from a passive prosthesis and even more than a healthy leg. The range of motion was more extensive than passive but still lower than the non-amputee subject. The step length in BIOM is more significant than a passive prosthesis, and the asymmetry almost vanishes due to the high power.

Rouse et al., 2015 [35] had gone far from daily walking. BIOM is used for an amputated dancer. The BIOM powered prosthesis has some adjustments directed for dancing, such as a leg-to-leg interface. However,

BIOM performs a good experience, but some challenges yield out. The range of motion(ROM) is lower than the healthy leg specialized control system, and step recognition for a dancing purpose is required. BIOM is used to trigger a particular behaviour at a specific phase, such as Powered Plantarflexion. Still, a model of muscle/s contraction and ankle joint angle would make it more generalized and has more natural behavior .

2.4 Human Gait Cycle Classification

Classification is dividing a dataset according to a pre-feed labelled dataset. Machine learning algorithms are used to achieve this task. In the case of a powered prosthesis, it is crucial to recognize the state of the human body during the gait cycle, which affects the behaviour of the prosthesis by implicating the state of the control system. This work aims to find whether the body is in the stance or swing phase based on the EMG signal of the lower limb muscles and whether it is possible to combine it with the regression model to have the optimal behaviour of the ankle-foot prosthesis.

Joshi, Lahiri, and Thakor, 2013 [36] studied the classification of eight gait cycle steps. Eight gait cycles steps are Heel contact, loading response, midstance, terminal stance, pre-swing, initial swing, mode swing, and terminal swing. Electromyographical data and hip angular rotation are used to find the pattern by which a classification process can be done. This work is utilized in controlling exoskeleton orthotic devices. The Electromyographical data was acquired for four muscles, namely: Hamstring, Getrocnimus, Quadriceps, Tibialis anterior. A marker is adhesive to the thigh to rack the flexion-extension of the hip joint and label the EMG data with present gait phases. Linear discrimination analysis is the algorithm used to classify the labelled data. The classification accuracy

was tested with and without Bayesian Information Criteria as a segmentation algorithm. The highest accuracy was concluded at the 8th subject's gait cycle was about 53.5% before applying Bayesian information criteria, and 84.16% after applying Bayesian information criteria. The classification accuracy increased after applying the BIC segmentation algorithm but still had a low accuracy compared with the demanding accuracy of operating an artificial limb and avoiding a walking failure.

Ziegler, Gattringer, and Mueller, 2018 [37] invited a new method for normalizing the EMG signal, called by the author by weighted signal difference(WSD). However, the EMG signal ranges from micro-volt to volts. Therefore, it is a great idea to normalize the signal. They highlighted the classification of the human gait cycle into stance and swing phases according to a labelled data of the EMG signal of seven lower limb muscles of five subjects at average speed(2 m/s). They used the support vector machine(SVM) to classify the dataset. The result data of SVM accuracy is compared between the data of EMG resulted after applying the root mean square filter(RMS) and weighted signal difference(WSD). WSD normalization method yields a higher accuracy for all subjects, and the training time is reduced to five times(faster training time). SVM is known for its low accuracy and slow training time.

Derlatka and Bogdan, 2015 [38] used the ensembled K-nearest neighbor (KNN) method to classify the human gait cycle into five phases using the ground reaction force(GRF) data. The classified phases are initial contact(IC), loading response(LR), mid-stance(Midst), terminal stance(Tst) and pre-swing phase(pre-sw). The GRF was derived from 200 subjects(almost 3500 gait cycles). Ensemble learning is the process of modelling using multiple algorithms of the same data or the same algorithms with variable parameters. In regression, the ensembled

regression uses regression such as linear regression, decision tree, K-nearest neighbour, etc., with the same featured data and uses a mathematical operator such as mean to find the best fitting model. Classification is done by using votes to choose the correct classes (majority voting). They showed that the ensemble learning gives more general decisions, lower error, and less overfitting. Dynamic time wrapping algorithm (DTW) was used to find the best alignment between the three components of GRF calculated from the force plate (x, y, and z - components). They tested a hypothesis weak classifiers can produce a high classification quality using ensemble learning, and this was proved by using KNN with multiple Kernel parameter values.

2.5 Ankle Joint Angle's Regression and Control Strategies

Lambrecht and Kazerooni, 2009 [26] used series of finite-state behavior strategies to control the powered prosthesis. The designed prosthesis worked passively with gait cycle phases and generated power when needed. The gait cycle is divided into four modes: stance, pre-swing, swing flexion, and swing extension. The classification between the modes mentioned above was done using observed elementary criteria. The swing flexion to swing extension transition was done when the knee angular speed had a negative velocity. The transition from swing extension to stance was done when the axial load passed a certain threshold, etc. A limitation of this work is that the classification is done with specific criteria that may change for each amputee, unlike the supervised learning approach, which creates a general model that can operate with a wide range of amputees specifications.

Huang, Kuiken, Lipschutz, et al., 2008 [39] used EMG signals to classify several movements such as level-ground, ambulation, jump over an obstacle, ascending and descending stairs. Each mode is divided into sub-phases, such as the gait cycle is divided into heel strike and toe-off. The powered prosthesis follows the trajectory based on the state of the body, such as being in pre or post of toe-off or heel strike phases. The EMG data were filtered by a high pass filter of 25HZ cut-off frequency to eliminate the relative movement of the electrodes artefacts. A Confusion matrix is used to evaluate the performance of the used algorithms. Simple linear discriminate analysis(LDA) and artificial neural network classifiers were used to classify the movement mode and phases. Both LDA and ANN showed a low performance ranging from less than 10% at pre-toe off phase and rise to a maximum of less than 60% at the full stride cycle phase.

Both **Lambrecht and Kazerooni, 2009 [26]** and **Huang, Kuiken, Lipschutz, et al., 2008 [39]** use the finite state approach, and the prosthesis follows a particular trajectory for a specific phase. The finite-state method must be restricted from the pre-defined conditions, unlike the human joints that act for uncounted states. Continuous control is the final approach to emulating the human joints and remove the pre-defined state's restriction to perform a specific action.

Dey et al., 2019 [40] aim to find a continuous estimation of the ankle joint angle and moment based on the angle and angular velocity of the knee and hip joint with its corresponding ground reaction force(GRF). A motion capture system was used to get the angle and angular speed of the hip and knee joints. A low pass filter using the Butterworth filter was used with a 6 Hz cut-off frequency to smooth the data and remove high-frequency noise. They used support vector regression(SVR) to build a statistical model that predicts a continuous estimation of the ankle joint's angle and moment. As

mentioned before, hip and knee joint angles and angular velocities during the gait cycle are input to the SVR algorithms. The corresponding ankle joint's angle and the moment are the target variables. The performance using the coefficient of determination (R^2) and the root mean square error (RMSE) was calculated for all sequences. It is noted that the SVR performance increased when the number of features increased, which means that the ankle joint has a high correlation with these features. A limitation of this work is the difficulty of using a motion capture system with the real amputee-prosthesis case to get the input features, which makes the EMG approach more feasible with the power ankle-foot prosthesis.

2.6 Concluding Remarks

As discussed before, a passive prosthesis has less ability to function similar to the natural limb due to its inability to provide positive power. Several powered prosthesis designs are reviewed in 2.3; This work proposes a new, simple, more comfortable, affordable, and feasible powered ankle-foot prosthesis design. Most of the implemented control strategies are based on finite state control where the prosthetic joints follow a particular trajectory according to a pre-defined mode, unlike the human joints that act for uncounted states. This work could combine classification and ankle joint angle regression to produce the optimal continuous control for a transtibial powered prosthesis.

Chapter 3

Theoretical Work and Modeling

3.1 Introduction

This chapter concerns the design of powered ankle-foot prostheses with their mathematical models and validity. The powered-ankle foot prosthesis mainly includes the following components: pylon, foot, ankle joint, motor, and crank. This chapter investigates the amount of force that the linear actuator should exert and the mathematical relationship between the ankle and motor joints angle. The prototype design according to the selected model and its parts will be explained here. Several simple electromyographical filtering techniques will be described here as well. EMG signal processing techniques are explained in detail with the following sections: Fast Fourier transformation, Convolution theorem, and digital filters. This chapter describes the classification algorithms applied to estimate if the body is in stance or swing phase. It also illustrates the ankle joint formulation and rotation matrix used in the angle measurements sensors. Furthermore, this chapter describes the re-regression algorithms used to build a statistical model of the ankle joint angle according to the lower limb muscles activity. Finally, it demonstrates the Gaussian distribution and its ability to optimize the regression process, which is considered as an essential contribution of this work.

3.2 Ankle Design and Mathematical Models

Deriving a mathematical model for the ankle joint is crucial since it directly influences the parametric design dimensions and gives an excellent image of how an extent may affect others. The mathematical model also plays a valuable part in the design of the control system. Mathematical models can influence other strategies such as material and suitable motor selection.

This work only deals with a single degree of freedom related to the main two motions of the ankle joint, namely Plantarflexion and Dorsiflexion. The calculations focus on the maximum linear displacement of a linear actuator and its relationship with ankle angle. The relationship between the ankle joint angle and motor joint angle was investigated. In addition, the force exerted by the motor is calculated.

The following models show the trails to design and derive the suitable ankle joint with a valid mechanism and clear mathematical model.

3.2.1 Model 1

This model is created as the first trail to illustrate the main components of a below-knee prosthesis. The objective is to find an equation for the linear actuator as the ankle joint angle function. As shown in Figure 3.1 variable y must be seen as the ankle joint angle (θ).

The linear actuator can be easily derived for this linkages system and as follows :

$$y = l \sin(\theta) \quad (3.1)$$

$$y_{total} = y_{doris} + y_{planter} \quad (3.2)$$

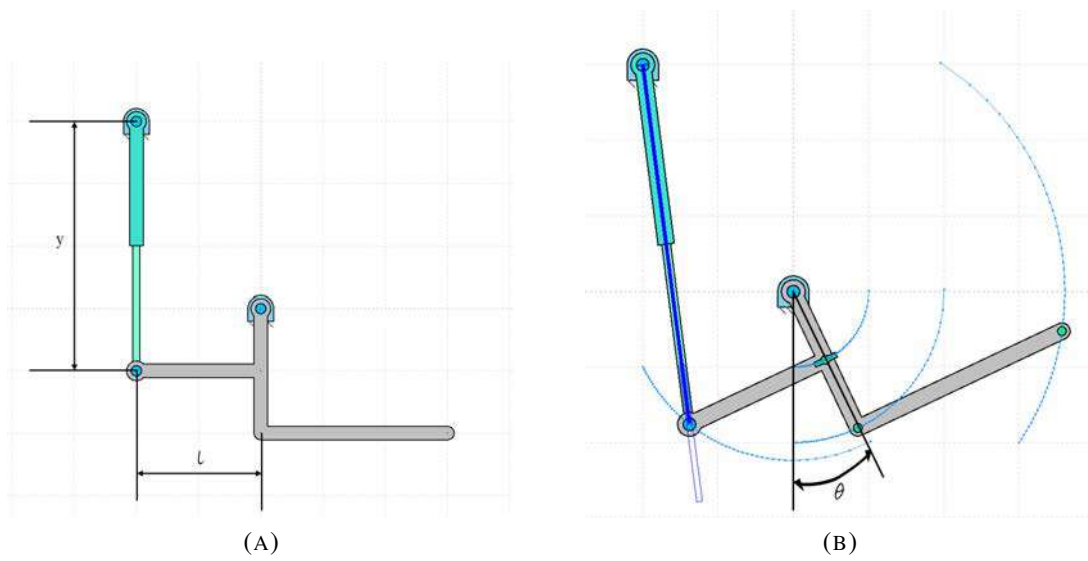


FIGURE 3.1: Model 1.

where l : crank length.

θ : ankle joint angle.

y : linear actuator displacement.

3.2.2 Model 2

After a focus on this model, it yields that the linear actuator is not constrained vertically, and this result from the motion of the crank in a circular path around the ankle joint, as shown in figure 3.2.

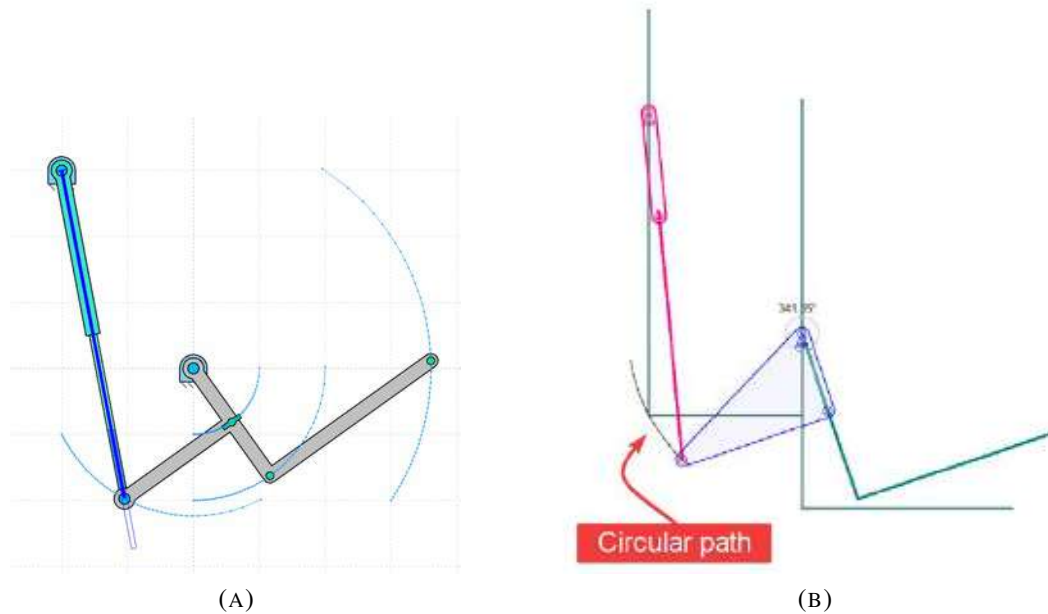


FIGURE 3.2: Circular motion of a linear actuator tip

Since the linear actuator is not constrained along the vertical axis, a new joint introduced, namely the motor joint, as shown in figure 3.3.

Now, parameters that must be derived is the linear actuator displacement as a function of the ankle joint angle(θ) and the force that the linear actuator must apply as a function of the ankle joint moment. The focus is on the ankle joint angle and moment, since these parameters, are considered as knowns and can be easily found from the gait cycle specification. The gait cycle specification is normalised chiefly in terms of subject mass.

Figure 3.4 shows the linear actuator displacement can be easily found using Cosine law.

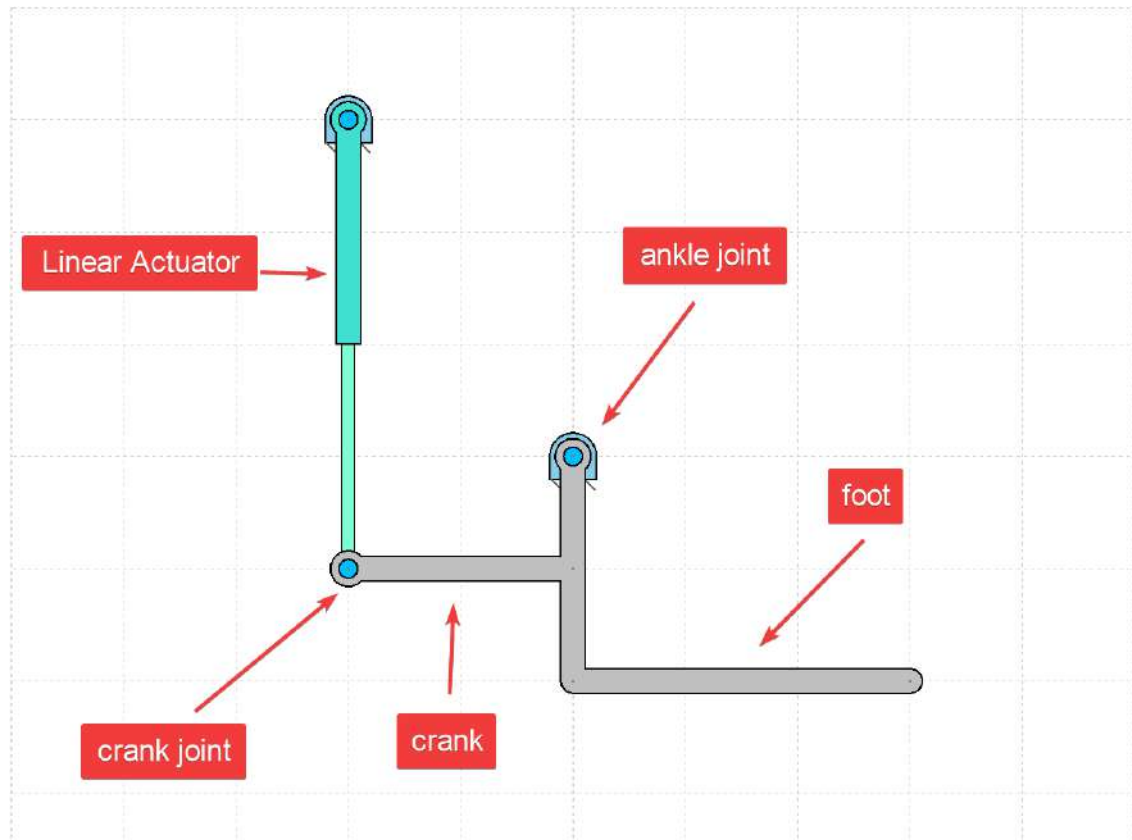


FIGURE 3.3: Model 2.

$$\beta = \tan^{-1} \frac{l}{J} = \text{constant} \quad (3.3)$$

$$\eta = \alpha + \beta + \theta = \text{constant} \quad (3.4)$$

$$\eta = 90 + \cos^{-1} \frac{l}{k} \quad (3.5)$$

$$\alpha = \eta - \beta - \theta \quad (3.6)$$

$$N^2 = J^2 + l^2 = \text{constant} \quad (3.7)$$

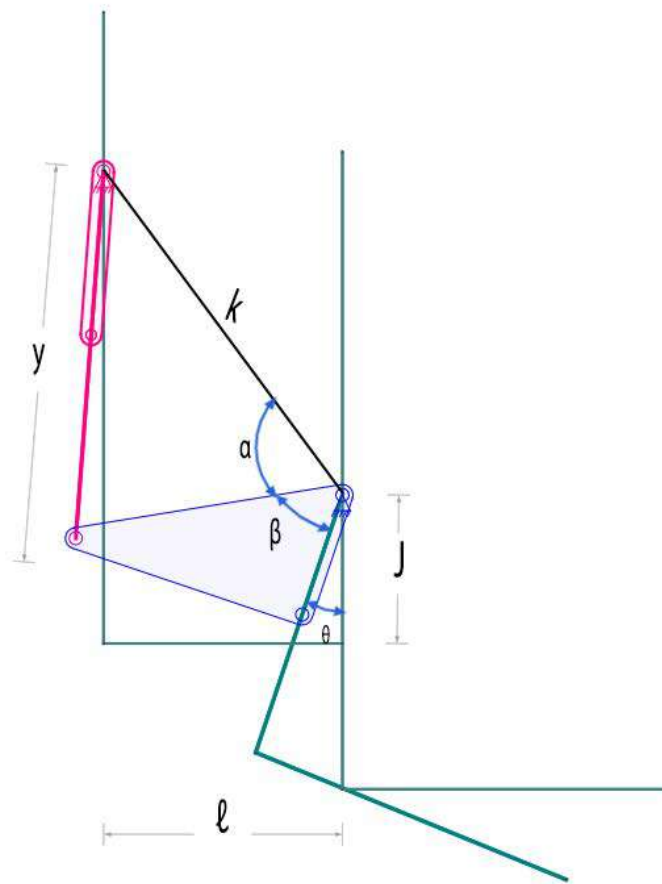


FIGURE 3.4: Model 2.

$$y^2 = N^2 + k^2 - 2NK \cos \alpha \quad (3.8)$$

$$y^2 = N^2 + k^2 - 2Nk \cos (\eta - \beta - \theta) \quad (3.9)$$

where l : Crank length.

k : Ankle joint to motor joint distance.

j : Distance between ankle joint to crank joint in vertical direction.

y : Linear actuator displacement.

A relationship between the linear actuator and the ankle joint was plotted

to verify this equation using Matlab programming language, as shown in figure 3.5(Appendix A.1). The plot shows a reasonable behaviour where the linear actuator displacement decreases with the increment of the ankle joint angle. Figure 3.5 can help determine the linear actuator's right length; if the rest dimensions are available, in 4.4, the rest of the dimensions are calculated based on the available lead screw length.

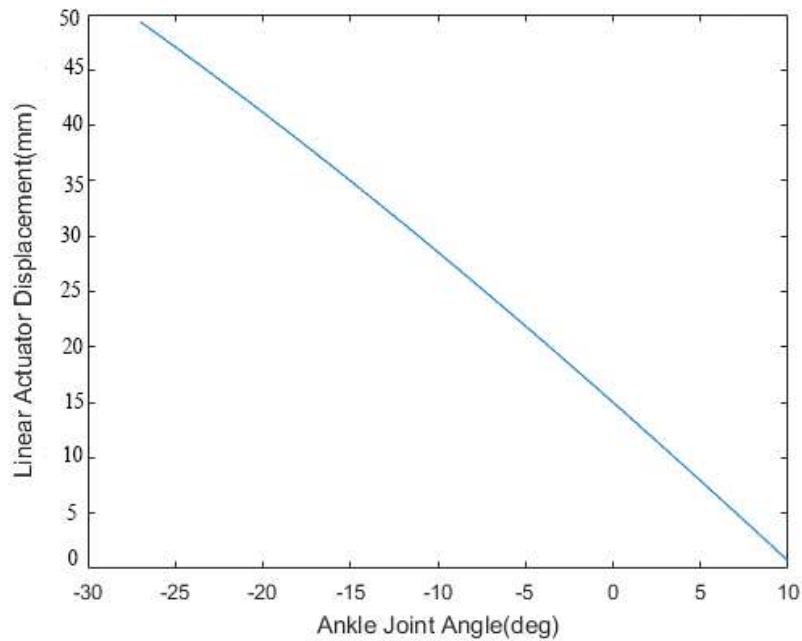


FIGURE 3.5: Model 2 - Linear Actuator Displacement Against Ankle Joint Angle Plot.

3.3 Proposed Model Force Calculations

The force exerted by the linear actuator is significant since it can specify the required motor's type and power. From figure 3.6 we could calculate the force F as follows.

$$M = (F \sin \lambda)Z + (F \cos \lambda)l \quad (3.10)$$

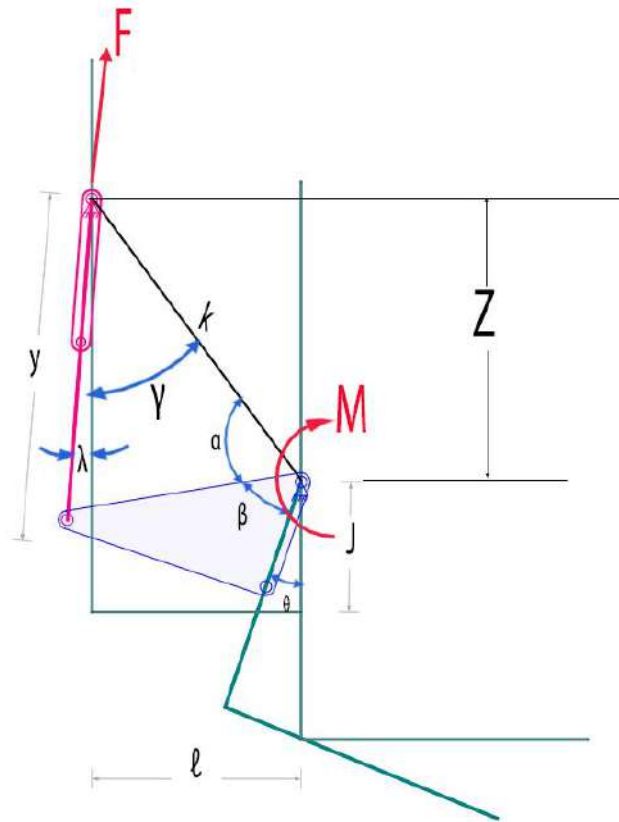


FIGURE 3.6: Model 2 -Linear Actuator Force Calculation.

$$F = \frac{M}{Z \sin \lambda + l \cos \lambda} \quad (3.11)$$

As shown in the equation 3.11, F is changing with the Ankle joint moment M and motor joint angle λ .

Several trials were attempted to find the validity of equation 3.11. A key point used to check the validity of this equation; this checkpoint is: both ankle joint angle and motor joint angle should equal zero at a neutral position (i.e. neither plantarflexion nor dorsiflexion). The importance of finding the motor angle is that the maximum force that the motor must exert can be calculated, and the motor angle only depends on the ankle joint angle. The followings are trails for checking the validity of equation 3.11:

3.3.0.1 Sine Law Trial

As shown in figure 3.6, N is assumed to be the ankle joint to crank joint distance. A sine law can be applied to the triangle with the following sides; N , Y , k , and as follows.

$$\frac{y}{\sin \alpha} = \frac{N}{\sin \gamma} \quad (3.12)$$

$$\gamma = \lambda + \sin^{-1} \frac{l}{k} \quad (3.13)$$

$$\frac{y}{\sin \alpha} = \frac{N}{\sin(\lambda + \sin^{-1} \frac{l}{k})} \quad (3.14)$$

$$\lambda + \sin^{-1} \frac{l}{k} = \sin^{-1} \frac{N \sin \alpha}{y} \quad (3.15)$$

$$\lambda = \sin^{-1} \frac{N \sin \alpha}{y} - \sin^{-1} \frac{l}{k} \quad (3.16)$$

$$\alpha = \eta - \beta - \theta \quad (3.17)$$

$$\alpha = 90 + \cos^{-1} \frac{l}{k} - \sin^{-1} \frac{l}{N} - \theta \quad (3.18)$$

$$\lambda = \sin^{-1} \left(\frac{N}{y} \sin 90 + \cos^{-1} \frac{l}{k} - \sin^{-1} \frac{l}{N} - \theta \right) - \sin^{-1} \frac{l}{k} \quad (3.19)$$

To test the equation 3.19 λ should be equal zero when θ equals zero. A plot between both λ and θ to illustrate the last equation.

As shown in figure 3.7, the condition(θ and λ both must equal zero at the start) which was not achieved from this trail.

3.3.0.2 Fixed Crank Distance Trial

This approach is more straightforward than the previous one(i.e. sine law) and has more promising results.

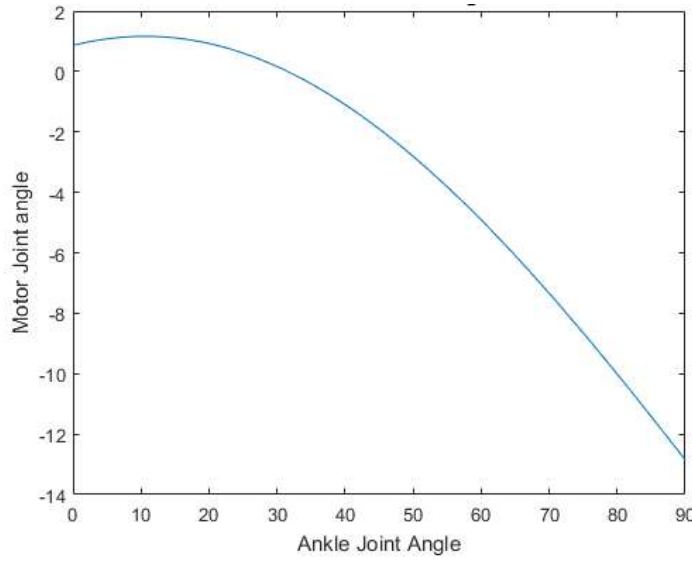


FIGURE 3.7: Model 2 -Ankle Joint Vs Motor Joint angle variation.

From figure 3.6 we could find:

$$l = J \sin \theta + l \cos \theta - y \sin \lambda \quad (3.20)$$

$$y^2 = K^2 + N^2 - 2NK \cos \alpha \quad (3.21)$$

where:

$$\gamma + \alpha + \beta + \theta = \pi \quad (3.22)$$

$$\alpha = \pi - \gamma - \beta - \theta \quad (3.23)$$

$$\beta = \tan^{-1} \frac{l}{J} \quad (3.24)$$

$$K = \sqrt{Z^2 + l^2} \quad (3.25)$$

$$N = \sqrt{J^2 + l^2} \quad (3.26)$$

Now, λ can be easily found from equation 3.20 and as following

$$\lambda = \sin^{-1} \frac{J \sin \theta + l \cos \theta - l}{y} \quad (3.27)$$

Figure 3.8 shows the achievement of the condition i.e. θ and λ start at zero (note, the code for this plot can be found in the appendix A.3)

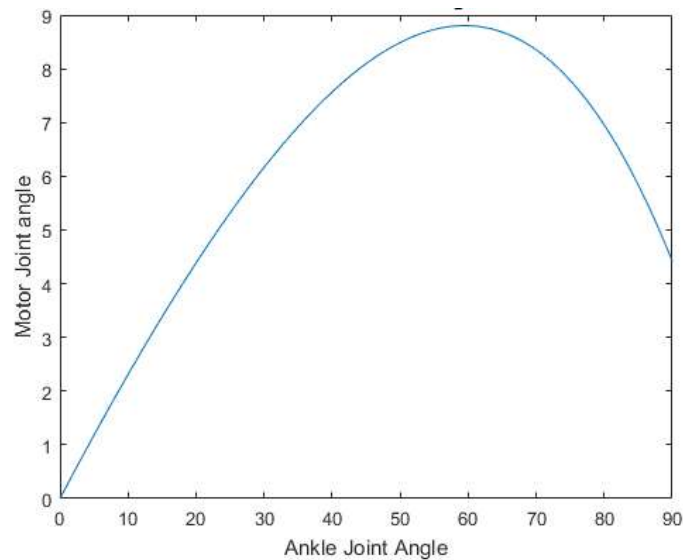


FIGURE 3.8: Ankle Joint Vs Motor Joint angle variation(Constant Crank Length Approach).

According to figure 1.7, the gait cycle specification, the ankle joint range of motion varied from -27(plantarflexion) to 10(dorsiflexion). A plot shown in the figure 3.9 illustrates the variation of the motor joint to ankle joint angles in the actual range of motion(code in appendix A.4).

Maximum internal ankle joint moment can be estimated from figure 1.7 to less than 1.5 N.m/kg. Figure 3.10 shows the linear actuator's force for the ankle joint range of motion; as shown in figure 3.9, the range of motion of the motor joint starts at -7 deg and ends at 2.3 deg. Figure 3.10 shows that the maximum force that the linear actuator should exert is 900 N which is considered a high force; this can be interpreted by observation figure 1.7. Figure 1.7 shows that the ankle joint experience the highest power among all lower limb joints.

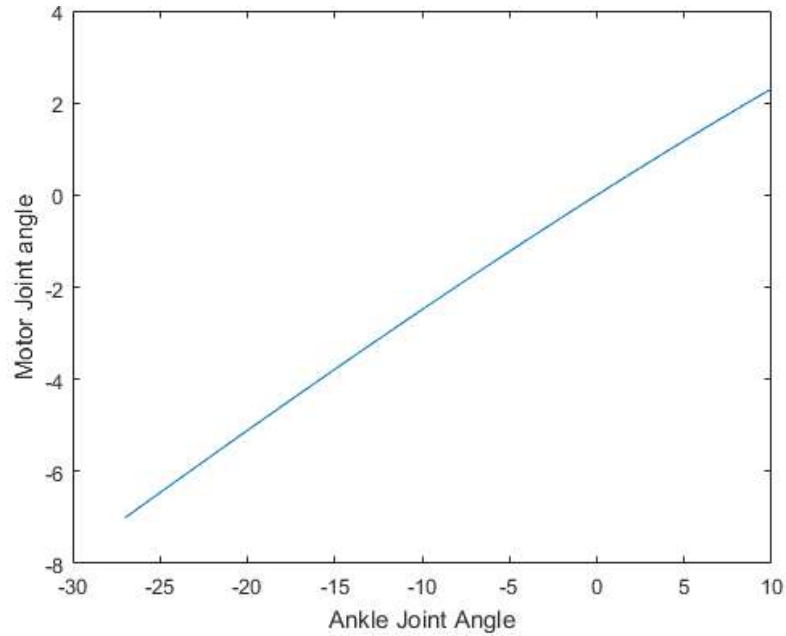


FIGURE 3.9: Natural range of Ankle Joint Vs Motor Joint angle variation.

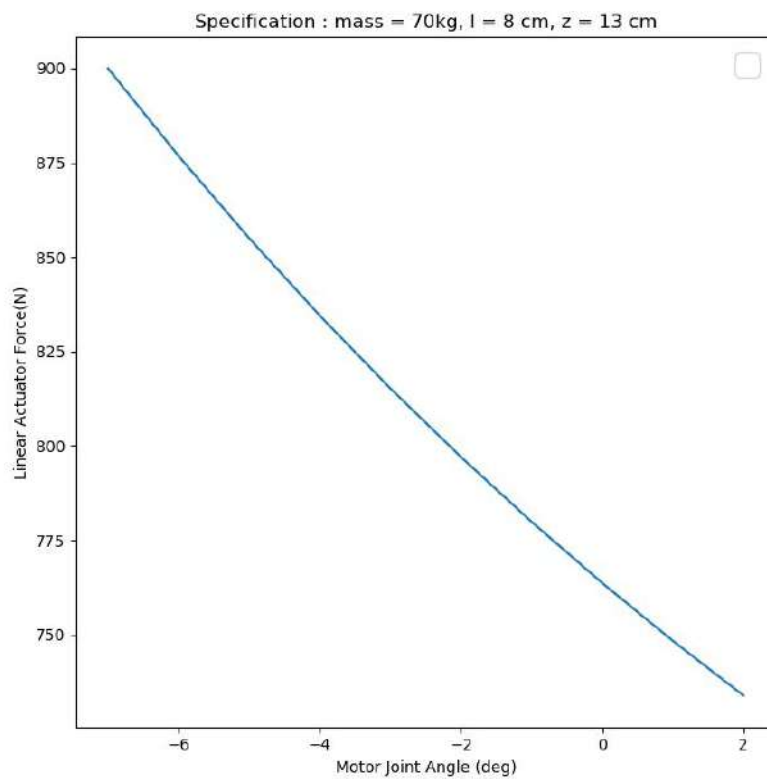


FIGURE 3.10: Maximum force exerted by the linear actuator for the natural range of ankle joint angles.

3.4 Gait Cycle Classification

Classification is the method of dividing an unsorted data set that is not yet classified into a process called training based on previous examples from a separate data set already labelled. Compared to regression, which gives continuous output, classification generates a discrete output (labels).

Many classification techniques are available in the pattern recognition field; four well-known classification algorithms were used in this work, as outlined below. In machine learning literature [21], more information on general statistical modelling and classification can be found.

3.4.1 Support Vector Machine(SVM)

SVM attempts to find the optimum line dividing data within separate labels for a collection of training data $D = \{(x_1, y_1) \dots (x_n, y_n)\}$ by minimizing the distance between support vectors, the vectors separating the different data labels. It is possible to formulate SVM as follows [37]:

$$\min_{w, \epsilon_i} J(w, \epsilon_i) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \epsilon_i \quad (3.28)$$

subject to

$$y_i \varphi(w^T x_i) \geq 1 - \epsilon_i \quad (3.29)$$

Where the SVM parameters are w and ϵ_i and the constant vector is c . The Lagrange classifier can be used to solve these equations.

3.4.2 K-Nearest Neighbor(KNN)

This method finds the closest point defined and averages the number of classes there, then marks the unknown end under consideration over a certain distance as the highest repeatable class. It's possible to formulate

K-NN as follows [41]

$$\hat{y} = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i \quad (3.30)$$

The expected K-NN label is \hat{y} , N_k is a vector of the neighbouring points, and the N vector is equal to k in length. It is then possible to add multiple approaches to find the distances from the unknown point to its neighbouring points, as follows [42]:

$$d(x_i, x_j) = \sqrt{\sum_{j=1, i \neq j}^k (x_i - x_j)^2} \quad \text{Euclidean distance} \quad (3.31)$$

$$d(x_i, x_j) = \sum_{j=1, i \neq j}^k |x_i - x_j| \quad \text{Manhattan distance} \quad (3.32)$$

$$d(x_i, x_j) = \left(\sum_{j=1, i \neq j}^k (x_i - x_j)^p \right)^{\frac{1}{p}} \quad \text{Minkowski distance} \quad (3.33)$$

3.4.3 Logistic Regression

A linear model can be used for a data set as shown in Figure 3.11(A), where the information is labelled either 0 for a specific range of x or 1 for a reset range; however, the results should be either 1 or 0 for a binary classification problem, and thus the upper and lower sections (marked with blue circles in the figure) can be trimmed as shown in Figure 3.11(B). The governing equations for logistic regression is as follows [43].

$$y = b_0 + b_1x \quad \text{Line equation} \quad (3.34)$$

$$p = \frac{1}{1 + e^{-y}} \quad \text{Sigmoid Function} \quad (3.35)$$

$$\ln \frac{p}{1-p} = b_0 + b_1 x \quad \text{Logistic Regression} \quad (3.36)$$

The plot is shown in Figure 3.12 results in applying of the sigmoid function to the linear model.

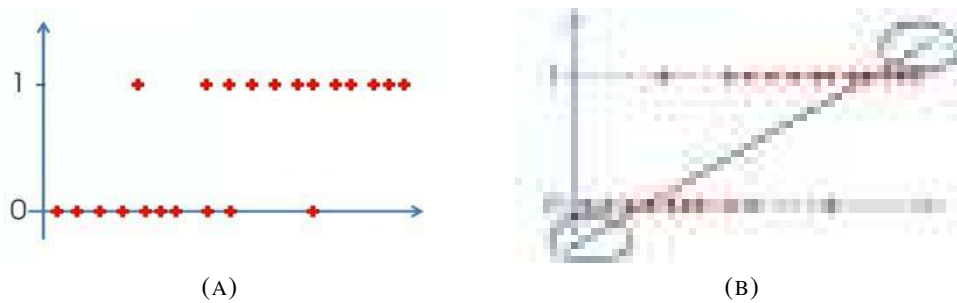


FIGURE 3.11: Linear Model.

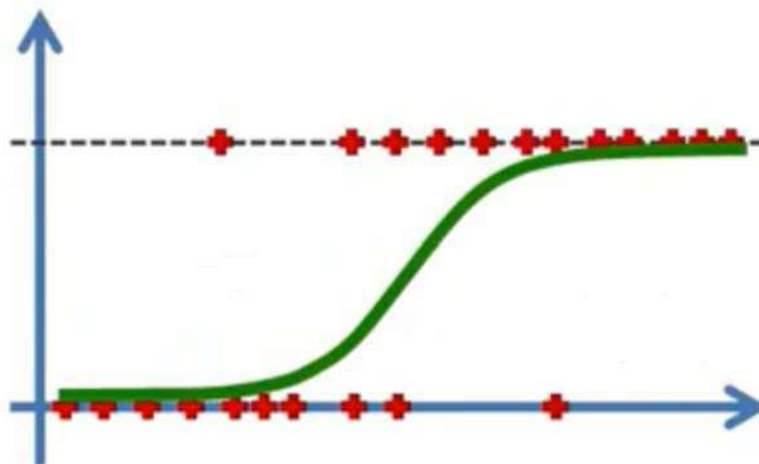


FIGURE 3.12: Logistic Regression Model.

Where p = prediction probability, permitting the setting of a threshold (such as 0.5) where y is equal to 1, if p has crossed this point, and otherwise it is equal to zero [44].

3.5 Ankle Joint Regression

Regression is a subset of supervised learning where it aims to predict one or more of a continuous dependent variable (Target variable) based on

D-dimensional input variables. The learning process is done by a set of $\{x_n\}$ observations with their target variable set $\{t_n\}$, and the objective is to find the value of the target variable for a non-given input variable. This work used regression to predict the ankle joint angle (which is considered a target variable) based on the lower limb muscles activity. The prediction of the ankle joint angle could mainly affect the manufactured powered ankle-foot prosthesis response. The number of muscles under consideration is four; therefore, the problem of building the required statistical model is of 4th order. The following is an explanation of the used regression algorithms:

3.5.1 Linear Regression

Linear Regression is as considered the a simplest regression type where you fit a line to a set of input data and target vector observations. Consider we have N observation of independent x variable such that $x = \{x_1 \dots x_n\}$, and their corresponding dependent target variable t such that $t = \{t_1 \dots t_n\}$, as shown in figure 3.13.

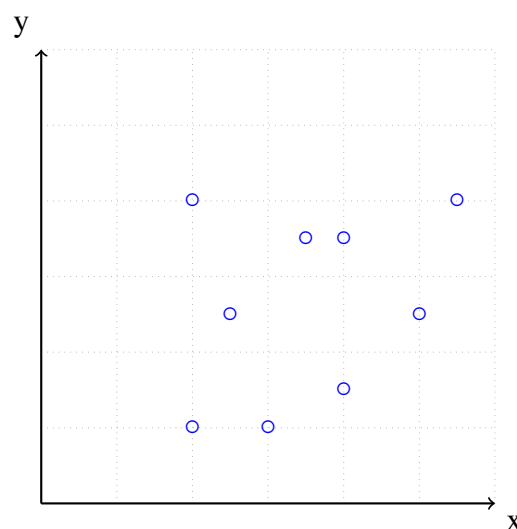


FIGURE 3.13: Scattered Points.

The goal is to estimate (with a specific error) the value of predicted \hat{t} for a new input \hat{x} data. The simple linear regression uses the equation of a line (equation 3.37) to fit the given data set.

$$y(x) = w_0 + w_1x \quad (3.37)$$

where w_0 and w_1 are constants.

The only unknown parameters in equation 3.37 are w_0 , and w_1 coefficients. w_0 , and w_1 parameters can be founded using the **minimized error function** approach that measures the amount of the misfitting between the target variable t_n and the input data x_n [45].

$$E_2(F) = \sqrt{\frac{1}{n} \sum_{k=1}^n (f(x_k) - y_k)^2} \quad (3.38)$$

The following approach uses the root mean square error to obtain the required coefficients. After Applying the error function for both the given data set (X_k and y_k) with the predicted value, we got equation 3.39.

$$E_2(F) = \sum_{k=1}^n (f(x_k) - y_k)^2 = \sum_{k=1}^n (Ax_k + B - y_k)^2 \quad (3.39)$$

The minimal error function can be used by applying the local minima approach; equation 3.39 is differentiated with respect to each coefficient, as shown in the following derivation:

$$\frac{\partial E}{\partial A} = 0 : \sum 2(Ax_k + B - y_k)(x_k) = 0 \quad (3.40)$$

$$\frac{\partial E}{\partial B} = 0 : \sum 2(Ax_k + B - y_k)(1) = 0 \quad (3.41)$$

Reformulation of Equation 3.40, and 3.41 into equations 3.42, and 3.43 respectively.

$$A \sum x_k^2 + B \sum x_k = \sum y_k x_k \quad (3.42)$$

$$A \sum x_k + Bn = \sum y_k \quad (3.43)$$

equations 3.42, and 3.43 can be formed as in a matrix form.

$$\begin{bmatrix} \sum x_k^2 & \sum x_k \\ \sum x_k & n \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} \sum y_k x_k \\ \sum y_k \end{bmatrix} \quad (3.44)$$

Equation 3.44 can be solved as a system of equations using linear algebra.

Assume: $C = \begin{bmatrix} \sum x_k^2 & \sum x_k \\ \sum X_k & n \end{bmatrix}$, $D = \begin{bmatrix} A \\ B \end{bmatrix}$, $E = \begin{bmatrix} \sum y_k x_k \\ \sum y_k \end{bmatrix}$.

Therefore,

$$C \times D = E \quad (3.45)$$

As known that the multiplication of an array with its inverse, produce an identity matrix.

$$I = C \times C^{-1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \longrightarrow \text{IdentityMatrix} \quad (3.46)$$

Multiplying both sides of equation 3.45 with C^{-1} .

$$I \times D = E \times C^{-1} \quad (3.47)$$

Linear Regression is simply approached using programming language; in python, the Sciket-learn library performs linear regression, as shown in figure 3.14.

```
# Fitting Linear Regression to the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the Test set results
y_pred = regressor.predict(X_test)
```

FIGURE 3.14: Python Code for Linear Regression.

3.5.2 Polynomial Regression

The generalization of linear regression is a polynomial regression, where a data set is fitted into a polynomial equation, as shown in equation 3.48 [21].

$$y(x) = w_0 + w_1x + w_2x^2 + \dots + w_mx^m \quad (3.48)$$

Where m is the order of the polynomial equation, equation 3.48 is related to a linear model, where even the equation is a non-linear function of the independent variable of x . However, it is still linear in terms of the unknown coefficients. Similar to linear regression, the least square errors could be used as the error function criteria to obtain the required coefficients with the specified polynomial order, as shown in equation 3.49.

$$E(w) = \frac{1}{2} \sum_{n=1}^N (y(x_n) - t_n)^2 \quad (3.49)$$

The order of the polynomial equation specifies how flexible the line is. The illustration of least square error is as shown in figure 3.15.

Figure 3.16 shows an example of using a polynomial regression of the same dataset with a different order m value.

Figure 3.16 shows that as the increment of m order, the fitting with the training data set will give the least error when $m = 1, 2$ it gave a poor fitting, as $m = 3$ the fitting get better, at the highest $m = 9$ the error goes to zero.

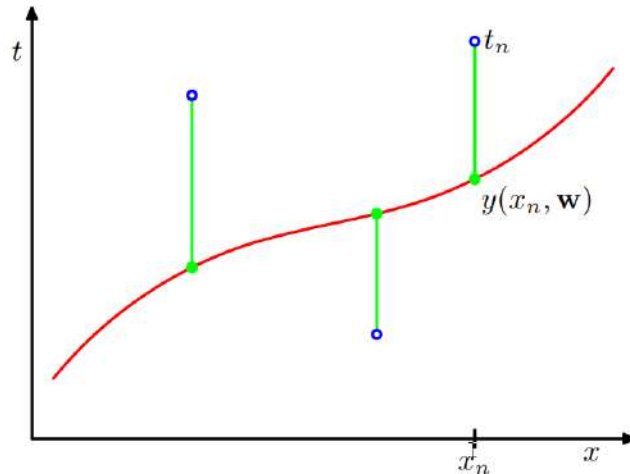


FIGURE 3.15: Polynomial Regression Error Function [21].

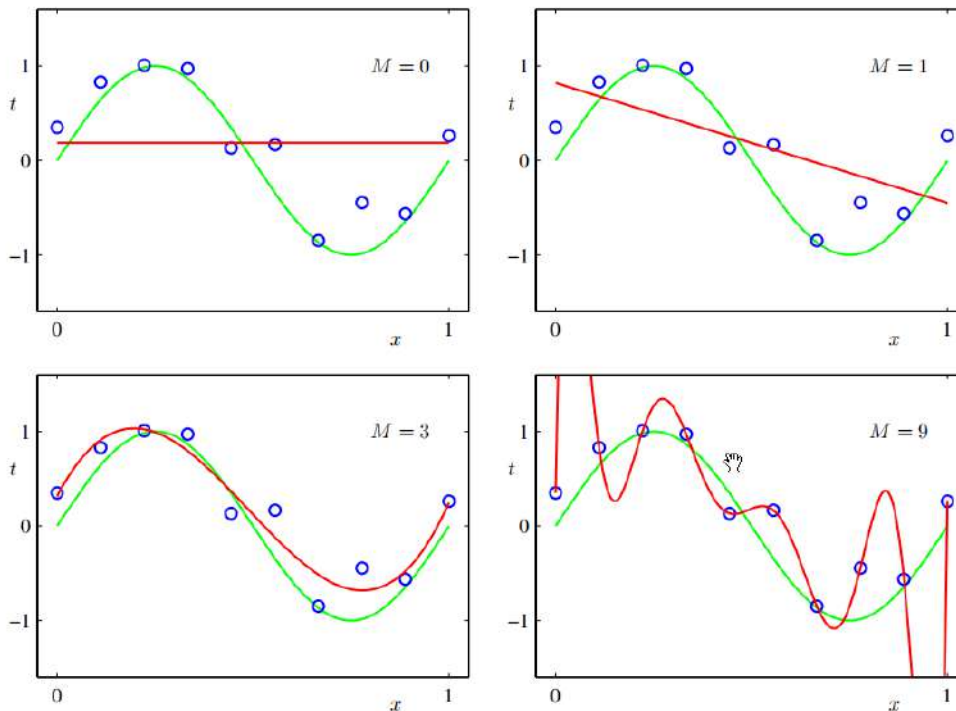


FIGURE 3.16: Polynomial Regression of Various Orders [21].

However, as the order of the polynomial regression goes higher, the fitting gets better. Still, on the other hand, a problem of overfitting is yielded, where at $M = 9$ the error function gives the heights with the test set. This is because the built model could not provide a general description of the data;

the error for various orders for both training and testing sets is as shown in figure 3.17.

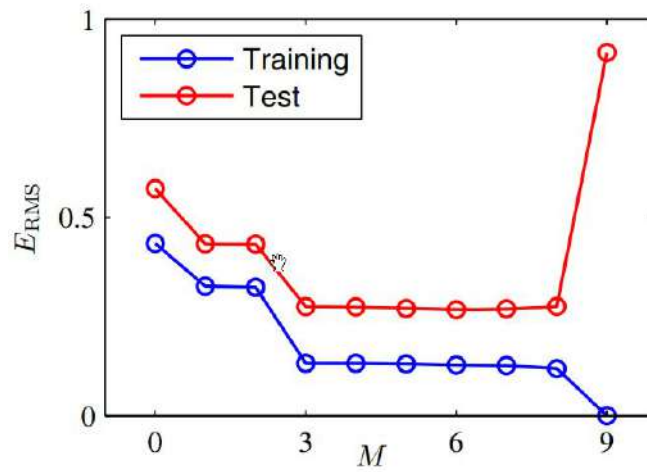


FIGURE 3.17: Error Value for Various orders [21].

The generalization of linear regression and coefficient estimation using the minimized error function for the polynomial regression is shown in the following derivation:

$$y(x) = F(w_0, w_1, w_2 \dots w_m) \quad (3.50)$$

Where the error function is as follows:

$$E(w_0, w_1, w_2 \dots w_n) = \sum (y(w_0, w_1, w_2 \dots w_n) - y_k)^2 \quad (3.51)$$

Differentiation for each variable's coefficient.

$$\frac{\partial E}{\partial w_j} = 0, \text{ where } j = 0, 1, 2, 3 \dots M \quad (3.52)$$

Assume a polynomial regression of second order $M = 2$.

$$y(x, w) = w_0 + w_1x + w_2x^2 \quad (3.53)$$

Therefore, the error function is as following:

$$E = \sum_{k=1}^N (y(x, w) - y_k)^2 = \sum_{k=1}^N (w_0 + w_1 x + w_2 x^2 - y_k)^2 \quad (3.54)$$

Where N is the number of the sample points. Differentiation concerning ω_0 .

$$\frac{\partial E}{\partial w_0} = \sum_{k=1}^N 2(w_0 + w_1 x + w_2 x^2 - y_k) \times 1 = 0 \quad (3.55)$$

Differentiation with respect to ω_1 .

$$\frac{\partial E}{\partial w_1} = \sum_{k=1}^N 2(w_0 + w_1 x + w_2 x^2 - y_k) \times x = 0 \quad (3.56)$$

Differentiation with respect to ω_2 .

$$\frac{\partial E}{\partial w_2} = \sum_{k=1}^N 2(w_0 + w_1 x + w_2 x^2 - y_k) \times x^2 = 0 \quad (3.57)$$

Equations 3.55, 3.56, and 3.57 can be rearranged into equations 3.58, 3.59, and 3.60 respectively.

$$w_2 \sum_{k=1}^N x_k^2 + w_1 \sum_{k=1}^N x_k + w_0 \sum_{k=1}^N 1 = \sum_{k=1}^N y_k \quad (3.58)$$

$$w_2 \sum_{k=1}^N x_k^3 + w_1 \sum_{k=1}^N x_k^2 + w_0 \sum_{k=1}^N x_k = \sum_{k=1}^N y_k x_k \quad (3.59)$$

$$w_2 \sum_{k=1}^N x_k^4 + w_1 \sum_{k=1}^N x_k^3 + w_0 \sum_{k=1}^N x_k^2 = \sum_{k=1}^N y_k x_k^2 \quad (3.60)$$

Equations 3.58, 3.59, and 3.60 can be formulated in a matrix form.

$$\begin{bmatrix} \sum x_k^2 & \sum x_k & n \\ \sum x_k^3 & \sum x_k^2 & \sum x_k \\ \sum x_k^4 & \sum x_k^3 & \sum x_k^2 \end{bmatrix} \begin{bmatrix} w_2 \\ w_1 \\ w_0 \end{bmatrix} = \begin{bmatrix} \sum y_k \\ \sum y_k x_k \\ \sum y_k x_k^2 \end{bmatrix} \quad (3.61)$$

Equation 3.61, can be solved using the same approach introduced in linear regression. Figure 3.18 shows a python code to fit the learning set to a polynomial regression and predict a test set.

```
# Fitting Polynomial Regression for the dataset
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree = 4)
X_poly = poly_reg.fit_transform(X_train)
lin_reg_2 = LinearRegression()
lin_reg_2.fit(X_poly, y_train)

# Predicting the Test set results
y_pred = lin_reg_2.predict(X_test)
```

FIGURE 3.18: Python Code for Polynomial Regression.

3.5.3 K-Nearest Neighbors Regression

K-Nearest Neighbors Regression is considered one of the simplest regression algorithms to comprehend. K-Nearest Neighbors Regression has high effectiveness. The K-Nearest Neighbors algorithm can be used for supervised machine learning subsets (classification and regression). The K-Nearest Neighbors Regression uses the average of the nearest points to predict the point under consideration, as shown in the following equation [46]:

$$y(x) = \frac{1}{N} \sum_{k=1}^N y_k \quad (3.62)$$

Where N is the number of neighbour points, the nearest points are the points that have less distance to the unknown end. Several methods to measure the distance between points, as shown in equations 3.31, 3.32 and 3.33.

Factor k is the main parameter in the KNN algorithm, where the increment of the k factor will increase the error function for the training set, as shown in figure 3.19. the error rate at $K=1$ is always zero for the training sample. This is because the closest point to any training data point is itself.

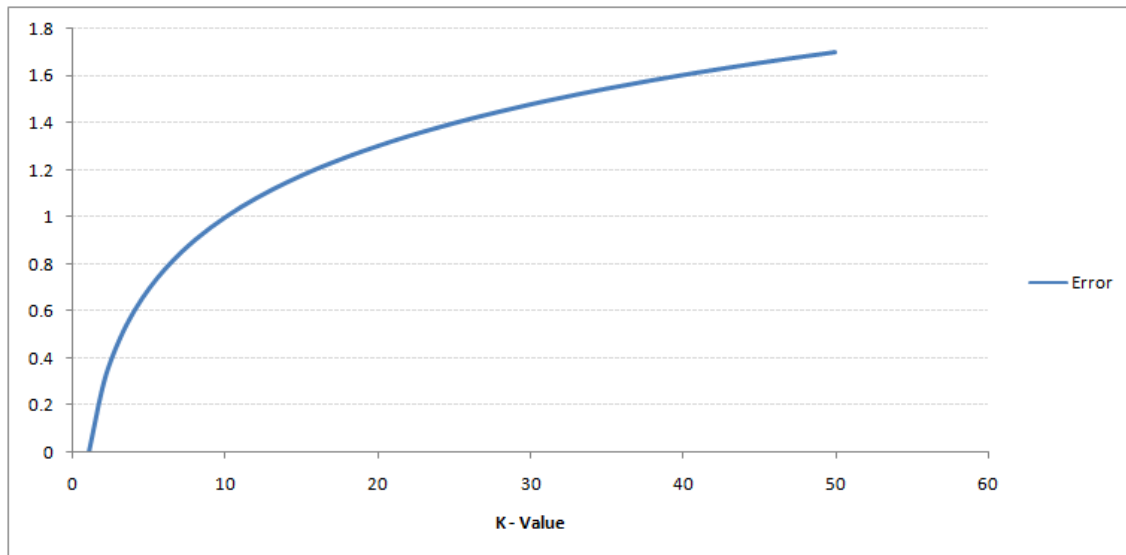


FIGURE 3.19: kNN Error Function of the Training Set.

On the other hand, a lower k -factor will produce an overfitting problem; this can be seen by the error function of the test set, as shown in figure 3.20.

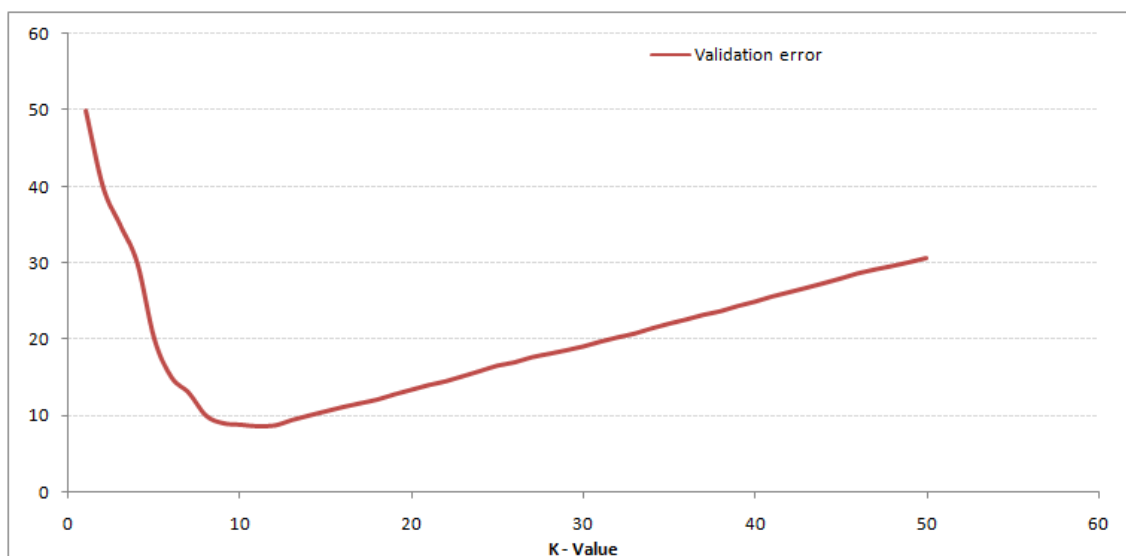


FIGURE 3.20: kNN Error Function of The Test Set.

Therefore, the test's optimal k can be measured ; $k = 9$ is considered the best k-factor for the given data set. Figure 3.20 is known as the elbow curve, which determines the best k-factor value. Figure 3.21 shows Python code to perform KNN regression.

```
from sklearn import neighbors
model = neighbors.KNeighborsRegressor(n_neighbors = k)
model.fit(x_train, x_train)
```

FIGURE 3.21: Python Implementation of KNN Regression.

3.6 Regression Models Evaluation

The measurement of the regression algorithms performance is essential since it indicates the ability of the algorithm under consideration to give the correct prediction with the unseen dataset. The algorithm's performance is used to benchmark a set of regression algorithms and pick the highest version.

3.6.1 Mean Squared Error (MSE)

Mean squared error is used to measure the difference between the predicted and actual data and as shown in equation [47].

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_{ai} - y_{pi})^2 \quad (3.63)$$

Where y_{ai} :Actual value.

y_{pi} : Predicted value.

n : Sample size.

The square is used to prevent negative errors and inflate the minor errors. Mean square error can be easily found in Sciket-learn python library as

shown in figure 3.22. MSE was used to compare between models of the same units.

```

1 # example of calculate the mean squared error
2 from sklearn.metrics import mean_squared_error
3 # calculate errors
4 errors = mean_squared_error(expected, predicted)

```

FIGURE 3.22: Sciket Learn Man Square Error.

3.6.2 Root Mean Square Error

Root mean squared error used to measure the error rate of a regression model and it can be calculated as shown in equation 3.64 [47]..

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_{ai} - y_{pi})^2}{n}} \quad (3.64)$$

Where y_{ai} :Actual value.

y_{pi} : Predicted value.

n : Sample size.

RMSE compares models whose errors are measured in the same units. The same library can be used as a mean squared error to find this metric but with setting square to false in the parameters, as shown in figure 3.23 [47]..

```

1 # example of calculate the root mean squared error
2 from sklearn.metrics import mean_squared_error
3 # calculate errors
4 errors = mean_squared_error(expected, predicted, squared=False)

```

FIGURE 3.23: Sciket learns Root Mean Square Error.

3.6.3 Relative Squared Error (RSE)

Unlike RMSE, the relative squared error (RSE) can be compared between models whose errors are measured in different units [47].

$$RSE = \frac{\sum_{i=1}^n (y_{ai} - y_{pi})^2}{\sum_{i=1}^n (y_{ai} - \bar{y})^2} \quad (3.65)$$

Where y_{ai} :Actual value.

y_{pi} : Predicted value.

n : Sample size.

$$\bar{y} = \frac{\sum_{i=1}^n y_i}{n}$$

3.6.4 Coefficient of Determination

This method is used to find the percentage (accuracy) of variation not described by the regression line. The coefficient of Determination(r^2) equals the total variation that is not represented by the regression line(i.e. squared error) over the total variation of the target variable and as shown in equation 3.66, and can be applied in python as shown in figure 3.24 [47].

$$r^2 = 1 - \frac{\sum_{i=1}^n (y_{ai} - y_{pi})^2}{\sum_{i=1}^n (y_{ai} - \bar{y})^2} \quad (3.66)$$

Where y_{ai} :Actual value.

y_{pi} : Predicted value.

n : Sample size.

$$\bar{y} = \frac{\sum_{i=1}^n y_i}{n}$$

```

1 from sklearn.metrics import r2_score
2 sklearn.metrics.r2_score(y_true, y_pred)

```

FIGURE 3.24: Coefficient of Determination in python.

3.7 Feature Selection

Feature selection is a preprocessing step in the machine learning field. Feature selection reduces the number of features required to build the statistical model and get the most valuable features on the built model. Reducing the redundant features will increase the created model performance, and the time required to produce the statistical model will be reduced. Feature selection is broadly divided into three approaches: wrapper model [48], filter model [49], and hybrid model [50]. The wrapper model uses all possible feature elimination cases and measures the performance with each case; the case of features with the highest model performance is considered the best feature to build the model. The wrapper model is a time-consuming process, and it is increased drastically with the increment of the dataset. The filter method reduces the number of features independently from the chosen model based on several criteria such as distance, consistency, and information. The linear correlation method is used in this work to select the best-correlated features. Correlation is a known measure that measures how well the used variables are reduced. The correlation coefficient equals ± 1 if the used variables are fully correlated and 0 if the used variables are not connected. The correlation coefficient for the two variables is as shown in equation 3.67 [51].

$$r = \frac{n \sum_{i=1}^n x_i y_i - \sum_{i=1}^n x_i \sum_{i=1}^n y_i}{\sqrt{[n \sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2][n \sum_{i=1}^n y_i^2 - (\sum_{i=1}^n y_i)^2]}} \quad (3.67)$$

To have a good understanding of what the correlation coefficient value means? Consider a linear function such as $y(x) = x$. Such a function has a correlation coefficient equal to a positive one where y has a fully positive correlation with x , as shown in figure 3.25(A).

Thus, fully negative correlation if $y(x) = -x$, as shown in figure 3.25(B). If $y(x) = c$, a zero correlation between y and x variables, as shown in figure 3.25(C).

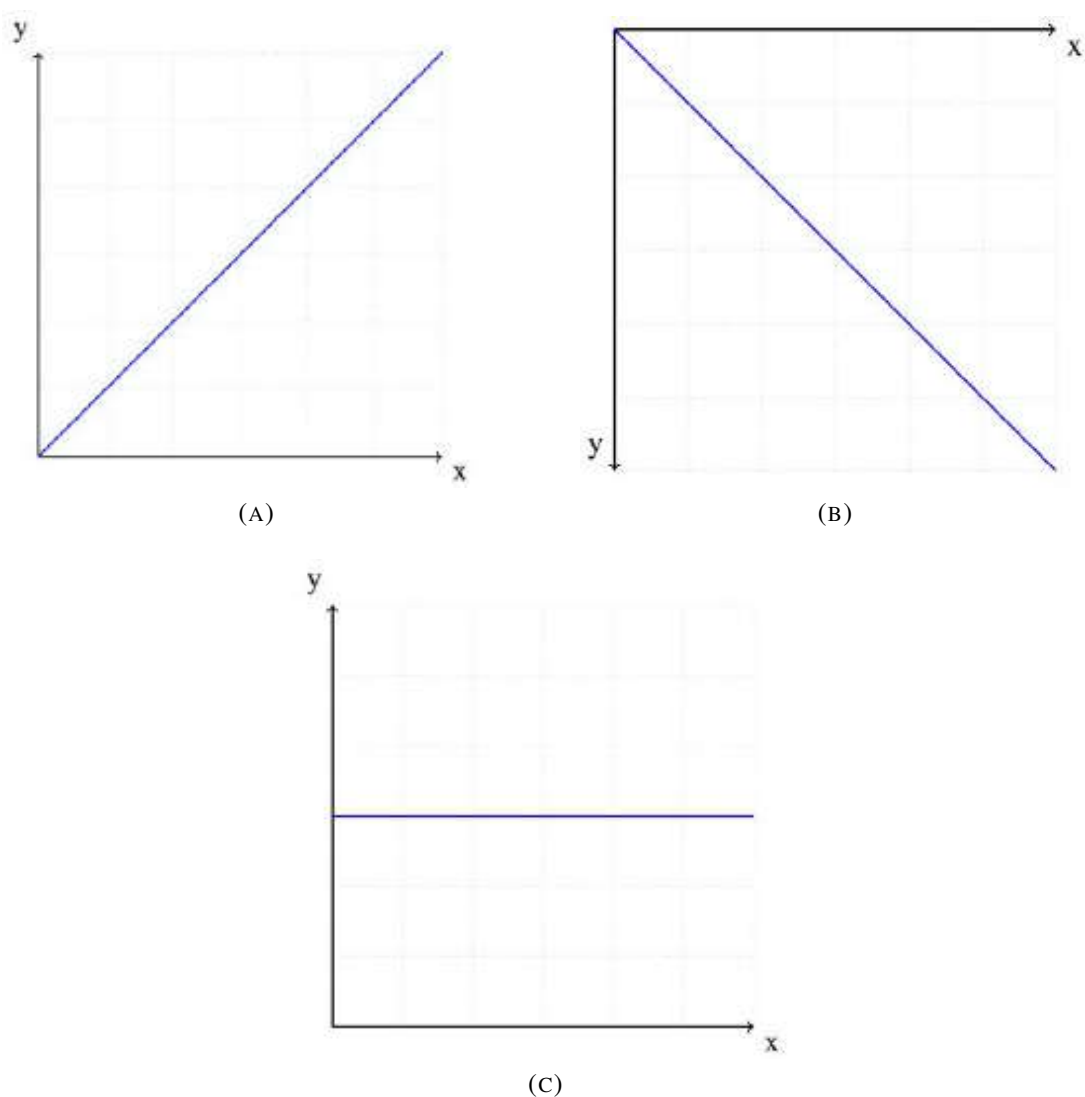


FIGURE 3.25: Correlation Cases.

3.8 Ankle Joint Angle Formulation

The ankle joint angle is produced by the movement of the shank and the foot. In this work, the angle's sign defines whether the joint state is in plantarflexion or dorsiflexion. Therefore, if the sign is positive, the ankle joint state is considered dorsiflexion, and if it is negative, it is considered plantarflexion. The limb's sensors are installed, as shown in figure 3.26(A).

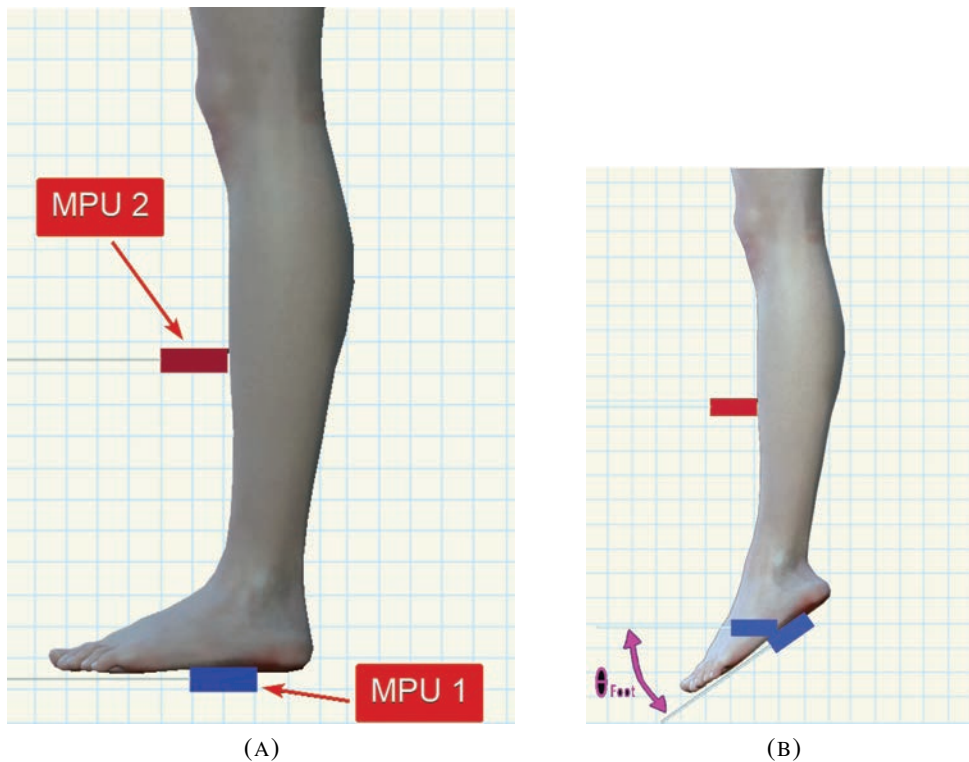


FIGURE 3.26: The Two IMUs Installation.

Both sensors should measure a zero angle at the normal standing position. The sensors in figure 3.26(A) are positioned to measure zero angles. Therefore, the relative ankle joint angle can be calculated by subtracting the foot from the shank angle as shown in equation 3.68.

$$\theta_{ankle} = \theta_{foot} - \theta_{shank} \quad (3.68)$$

Equation 3.68 will give a zero degree at the start and negative when the foot

is in plantarflexion as shown in figure 3.26(B) and positive when the foot is in dorsiflexion.

The movement shown in figure 3.27 gives zero ankle joint angle since only the knee joint is flexed and both the foot and shank have the same angle, therefore according to equation 3.68, the ankle joint angle will be zero.

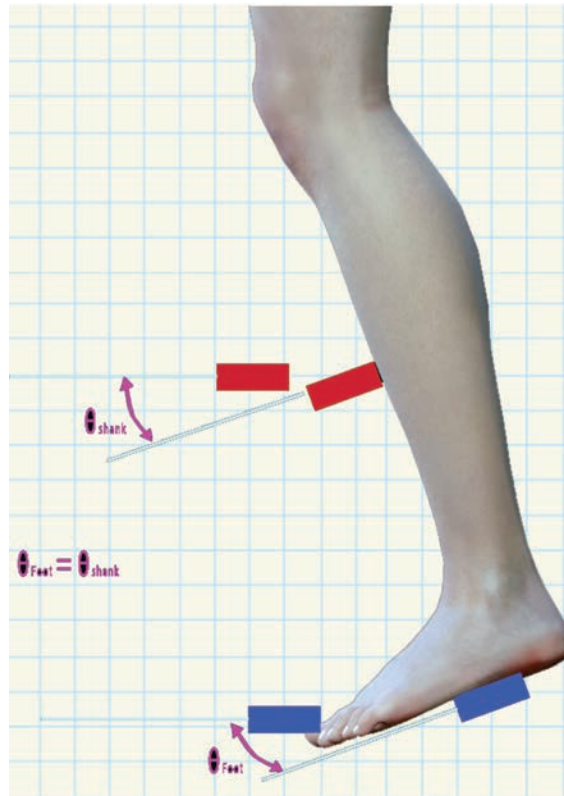


FIGURE 3.27: Zero Angle Joint Angle.

3.9 Orientation Measurement

The ankle joint angle is essential since it represents the target variable in the regression operation. The ankle joint angle is measured using double IMUs, one adhesive to the foot and the other adhesive to the shank. IMUs measure both the linear acceleration and angular velocity in 3 directions. The rotation measures the IMU's orientation based on the linear acceleration.

3.9.1 Rotation Matrix

Rotation matrix elements represent the projection of the new frame axes onto the old one. For a two-dimensional rotation case, the rotation matrix is as follows.

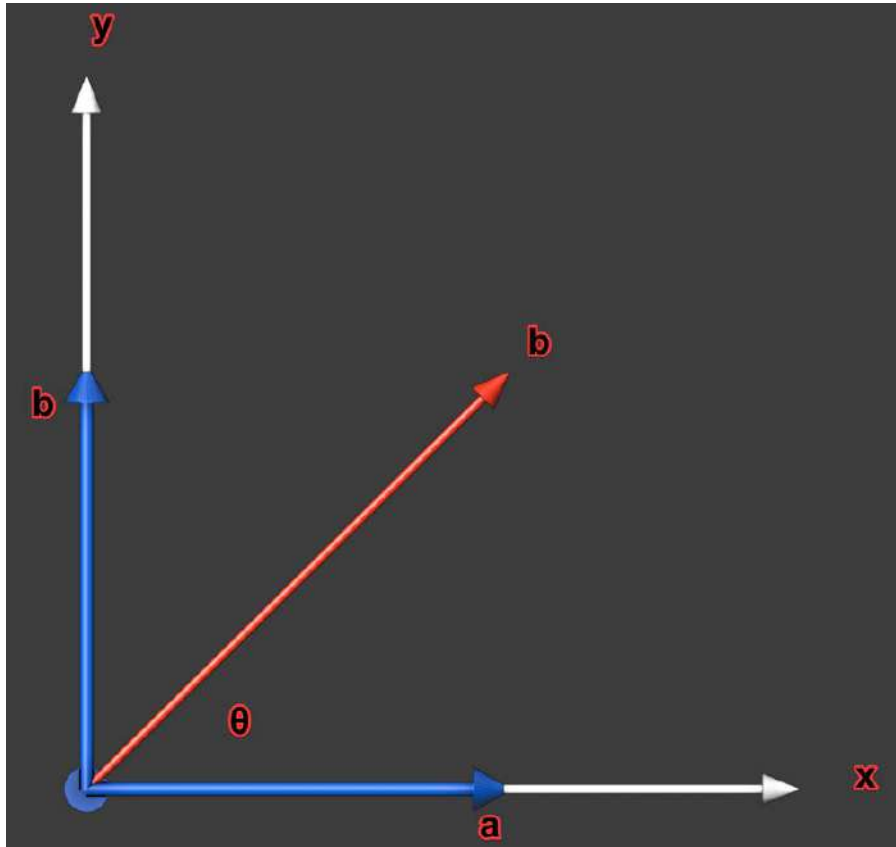


FIGURE 3.28: Two-dimensional rotation case.

$$R = \begin{bmatrix} a \\ b \end{bmatrix}$$

The same approach can be applied with a three-dimensional frame; the rotation matrix represents the projection of the new frame over the old frame. Suppose frame 1 is a rotation of frame 0 around the z-axis with θ as shown in both figure 3.29 and figure 3.30. the rotation matrix for frame 1 rotated relative to frame 0 is called "Z-Rotation Matrix" as long the rotation is around Z-axis.

$$R_1^0 = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The above matrix will inform us of all the projections for any angle θ rotated around Z-axis.

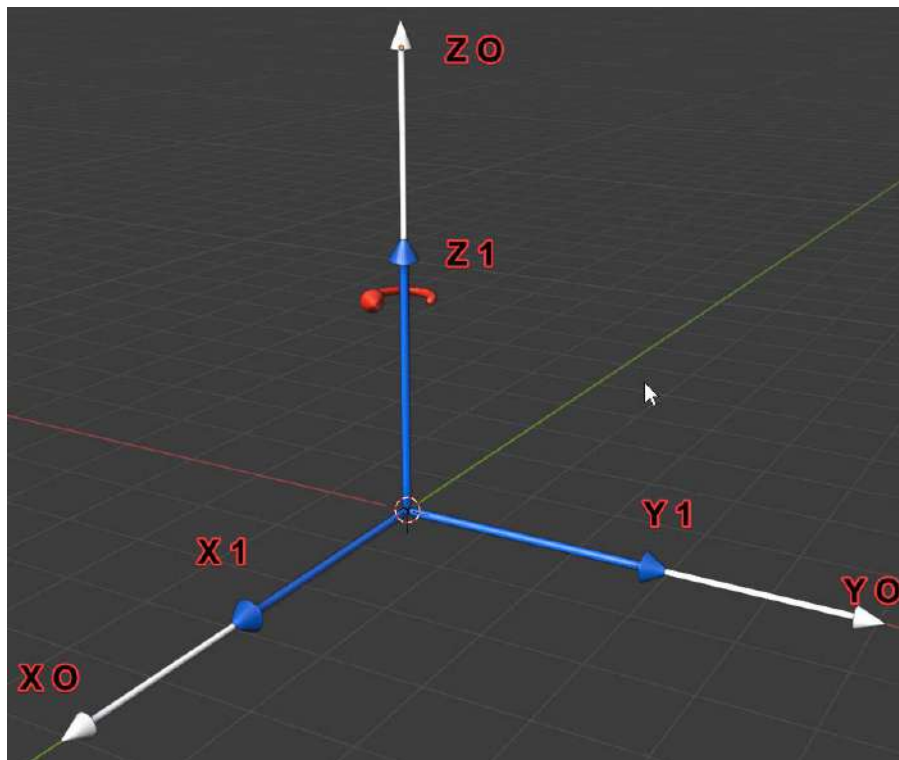


FIGURE 3.29: Frame 0 and 1 illustration.

Now, we can find all rotation matrices around global axes using the same approach.

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

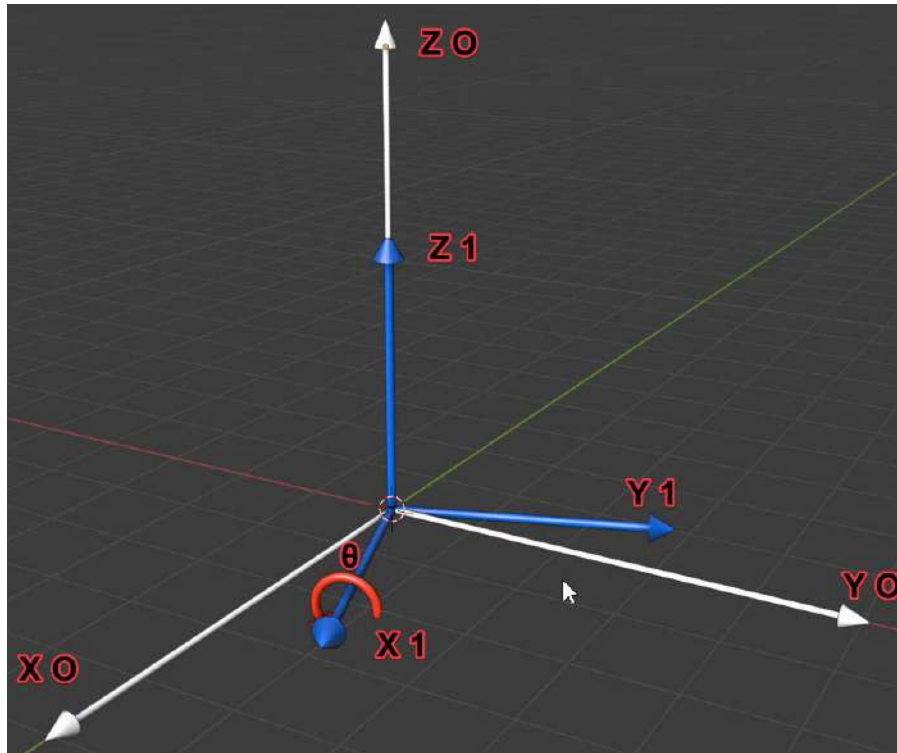


FIGURE 3.30: Rotation of frame 1 around frame 0's z-axis with θ angle.

$$R_z = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Now, suppose frame one is returned around x 1 with angle ϕ as shown in figure 3.31, in such case, the rotation of frame two with respect to frame zero can be found using the **Euler angles** equation and as follows.

$$R_n^0 = R_1^0 \cdot R_2^1 \cdot R_3^2 \cdot \dots \cdot R_n^{n-1} \quad (3.69)$$

The rotation matrix helps determine the tilting angle of an accelerometer and as explained in 4.2.3.

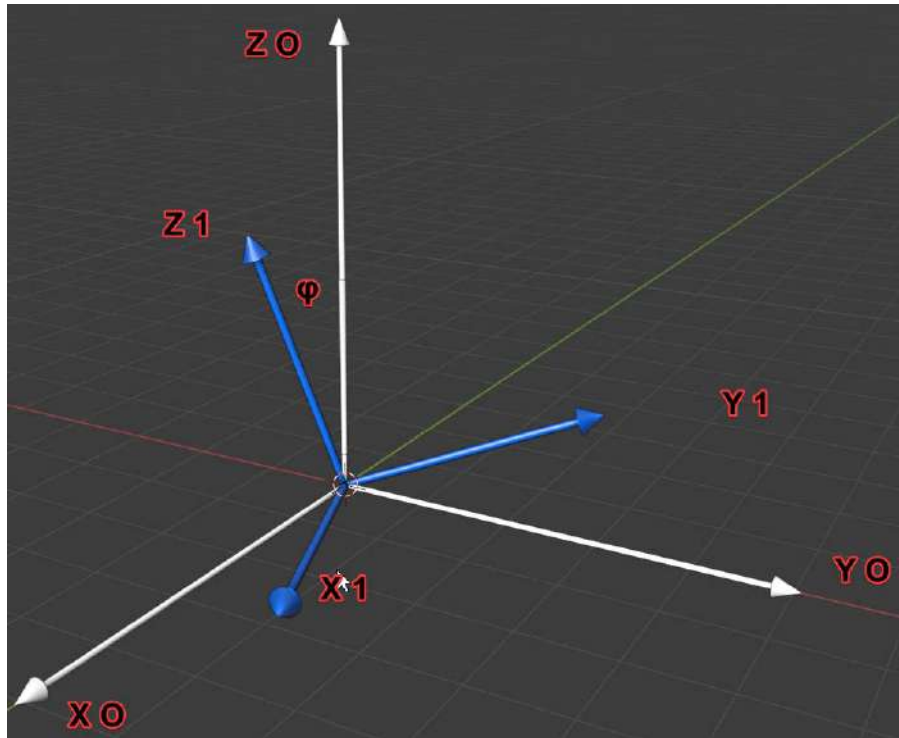


FIGURE 3.31: Rotation of frame 1 around x-axis with ϕ angle.

3.9.2 Single IMU's Orientation Measurement

As mentioned earlier in section 4.2.1, the ESP32 can be programmed as an access point. This section explains how ESP32 is programmed and connected to MPU6050 to obtain the ankle joint angle. As mentioned before in section 4.2.2, MPU6050 can measure both the linear acceleration via the accelerometer and the angular speed via gyroscope over x,y, and z. MPU6050 tilting can be calculated from the gyroscope by applying a single integration with respect to time.

$$\theta = \int \frac{d\theta}{dt} dt \quad (3.70)$$

In programming, this is done by multiplying the gyroscope between every two loops of measurement(Δt). The gyroscope measures the angular time change rate; since the gyroscope measures the angular speed, the angle

difference between the current reading and previous reading will be added.

$$\theta_2(t + \delta t) = \theta_1(t) + \Delta\theta \quad (3.71)$$

$$\Delta\theta = \theta \cdot \delta t \quad (3.72)$$

Suppose this approach is used to collect the sensor's tilting. In that case, a problem is occurring primarily. If there is a small error or deviation during each reading, this error will accumulate and give far away readings. In addition, the gyroscope does not have a fixed reference frame.

MPU6050's accelerometer measures the linear acceleration in terms of gravity, where if it is set horizontally, it will calculate the acceleration as (0, 0) and 1g for all x, y, and axes, respectively. Will be redistributed between other axes. The resultant vector generally will have 1g along the gravity vector. The three-dimensional acceleration concerning the earth frame can be expressed as follows.

$$a = R \cdot g \quad (3.73)$$

Where R is a rotation matrix and as explained in section 3.9.1, g is the gravitational acceleration in all three axes

$$g = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

The rotation matrix could yield six rotation possibilities R_{xyz} , R_{yxz} , R_{xzy} , R_{yzx} , R_{zxy} , and R_{zyx} . Only the first two possibilities R_{xyz} , and R_{yxz} can be used, since the rest possibilities have three unknowns θ_x , θ_y , and θ_z , unlike R_{xyz} , and R_{yxz} which include only θ_x , and θ_y as unknown when multiplied

by gravitational acceleration vector and as follows

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = R_x R_y R_z \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.74)$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x \\ 0 & \sin \theta_x & \cos \theta_x \end{bmatrix} \begin{bmatrix} \cos \theta_y & 0 & \sin \theta_y \\ 0 & 1 & 0 \\ -\sin \theta_y & 0 & \cos \theta_y \end{bmatrix} \begin{bmatrix} \cos \theta_z & -\sin \theta_z & 0 \\ \sin \theta_z & \cos \theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.75)$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} \cos \theta_y \cos \theta_z & \cos \theta_y \sin \theta_z \\ \cos \theta_z \sin \theta_y \sin \theta_x - \cos \theta_x \sin \theta_z & \cos \theta_x \cos \theta_z + \sin \theta_y \sin \theta_x \sin \theta_z \\ \cos \theta_x \cos \theta_z \sin \theta_y + \sin \theta_x \sin \theta_z & \cos \theta_x \sin \theta_y \sin \theta_z - \cos \theta_z \sin \theta_x \\ -\sin \theta_y \\ \cos \theta_y \sin \theta_x \\ \cos \theta_y \cos \theta_x \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.76)$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} -\sin \theta_y \\ \cos \theta_y \sin \theta_x \\ \cos \theta_y \cos \theta_x \end{bmatrix} \quad (3.77)$$

Therefore roll can be easily found as following

$$\frac{a_x}{\sqrt{a_y^2 + a_z^2}} = \frac{\sin \theta_y}{\cos \theta_y} \quad (3.78)$$

$$\theta_y = \tan^{-1} \frac{a_x}{\sqrt{a_y^2 + a_z^2}} \quad (3.79)$$

The above procedure used R_{xyz} possibility to obtain a roll. Now, let us try the R_{yxz} possibility.

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = R_y R_x R_z \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.80)$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} \cos \theta_y & 0 & \sin \theta_y \\ 0 & 1 & 0 \\ -\sin \theta_y & 0 & \cos \theta_y \end{bmatrix} \begin{bmatrix} \cos \theta_z & -\sin \theta_z & 0 \\ \sin \theta_z & \cos \theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x \\ 0 & \sin \theta_x & \cos \theta_x \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.81)$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} \cos \theta_z \cos \theta_y - \sin \theta_y \sin \theta_x \sin \theta_z & \sin \theta_z \cos \theta_y + \sin \theta_y \sin \theta_x \cos \theta_z & \\ -\cos \theta_x \sin \theta_z & \cos \theta_x \cos \theta_z & \\ \cos \theta_y \sin \theta_x \sin \theta_z & -\cos \theta_z \cos \theta_y \sin \theta_x + \sin \theta_z \sin \theta_y & \\ -\sin \theta_y \cos \theta_x & & \\ \sin \theta_z & & \\ \cos \theta_y \cos \theta_x & & \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (3.82)$$

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} -\sin \theta_y \cos \theta_x \\ \sin \theta_x \\ \cos \theta_y \cos \theta_x \end{bmatrix} \quad (3.83)$$

Therefore pitch can be easily found as following

$$\frac{a_y}{\sqrt{a_x^2 + a_z^2}} = \frac{\sin \theta_x}{\cos \theta_x} \quad (3.84)$$

$$\theta_x = \tan^{-1} \frac{a_y}{\sqrt{a_x^2 + a_z^2}} \quad (3.85)$$

This work is only dealing with planterflexion and dorsiflexion movements. Therefore, one angular calculation will be needed(either roll or pitch), and this will be decided according to the MPU6050 enclosure design, which is discussed in section [4.2.5](#).

3.10 EMG Signal Processing

Electromyography (EMG) is an experimental application for enhancing, recording, and inspecting myoelectric signals generated by the variation of the membrane of physiological muscle fibers excited by a single or multi-motor unit [17]. These EMG signals are used in several applications and techniques, including medical research, recovery, and sports science. They have also been described as providing a potential source of artificial limb control, especially about lower limb prostheses. Their human motion classification and regression usage can also be crucial in developing prostheses and orthosis. Raw EMG signals, however, are usually distorted by noise. This noise primarily results from surrounding muscle action, causing "cross-talk" and ECG bursts, or relative electrode movement concerning the muscle under consideration; external noise sources are also a concern.

Noisy signals may lead to undesirable output in a prosthesis, electromechanical delay, or poor classification and regression. Thus, EMG signals should be refined by applying physical or mathematical filtering techniques. Using two filtering methods, the Median filter and Root Mean Square (RMS) filter, this study aims to obtain a better signal and distinguish between them based on analyzing the output of different classification algorithms.

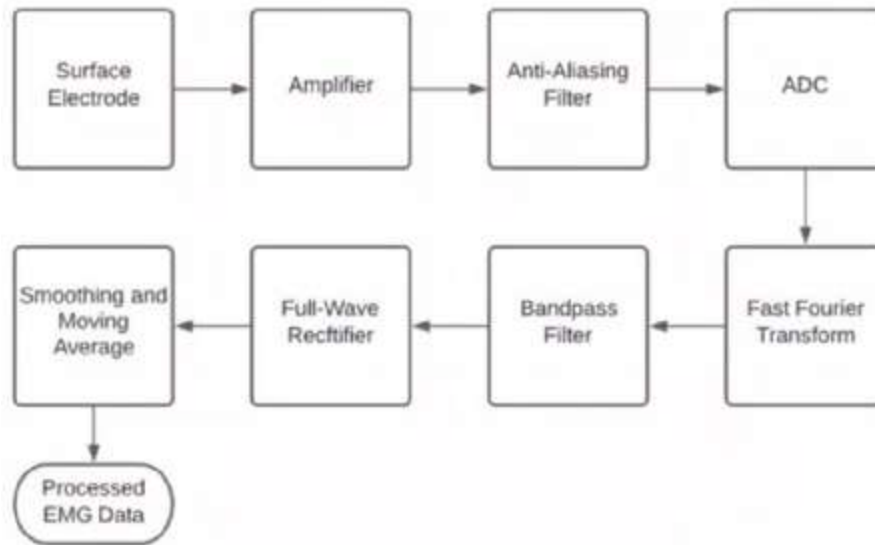


FIGURE 3.32: EMG Signal Processing.

Figure 3.32 shows the stages for processing the EMG signals. Firstly, an instrumental amplifier is used to increase the amplitude of the EMG signal to a specific gain (such as 100), which gives a big signal that is easier to analyze and understand. Anti-aliasing is done by using a lowpass filter (analogue) to remove the effect of any unwanted potential signals; this process is implemented in hardware. Analog to digital converter is done using a microcontroller for sampling the signal; a 16-bit or 8-bit sample frequency is used to convert the signal into digital form, to make it easily processed using programming languages. Fast Fourier transformation (FFT) converts the data from time-domain to frequency-domain to digital filtering. The bandpass filter removes the high frequencies yielding from the sensors' imperfection and low-frequency noise generated from electrodes shifting on the patient muscles. Inverse fast Fourier *transformation* converts back the filter data into time-domain. The Full-Wave Rectifier stage rectifies the negative signal values. The rectification is done by taking the absolute value of each sample point. Smoothing and moving average are done using a moving window operation as explained in 3.10.1.

3.10.1 Median Filter

Random nature of the noise applied to EMG signals. The median filter is a type of non-linear filter most commonly used for image filtering to deal with scattered light applied to an image from various environmental sources. It uses a travelling filter window, where the filter is added around the point under consideration for a certain range (in images, pixels are considered the points in question). For instance, if we have a set of $s = \{1, 2, 3, 4, 5\}$, the median is the middle value of a set, then the median of s is 3. the window size could be formulated as follows [52]:

$$N = 2k + 1 \quad (3.86)$$

k is the number of samples around the consideration point, N should always be odd to ensure that the discussion point remains an actual number in the middle. To add a median to a sample of 20 points with a window size of 7, $N = 7$, and to apply the equation 3.86, $k = 3$, let's consider a signal of 100 simple points. In Python, with the **Medfilt** method, can be implemented

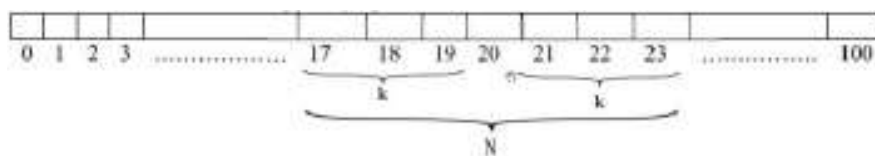


FIGURE 3.33: Suitable Interval of a signal for Median Filter.

using a single class in a **scipy** library. Therefore, the mean filter was used to spread the overshoot over all the windows specified, whereas the median filter was used to eliminate this overshoot [53].

3.10.2 Root Mean Square Filter

For the same median filter window size as in equation 3.86, the root mean square can be formulated as follows [52]:

$$RMS = \sqrt{\frac{1}{N} \sum_{i=n-k}^{n+k} x_i^2} \quad (3.87)$$

Where $n = 0 \dots$ final sample, and N is the window size. Unfortunately, RMS filter for the raw EMG signal, as seen in Figure 3.34 is not natively available in the Python signal library. A Python function was thus developed for this work, as shown in Appendix A.7.

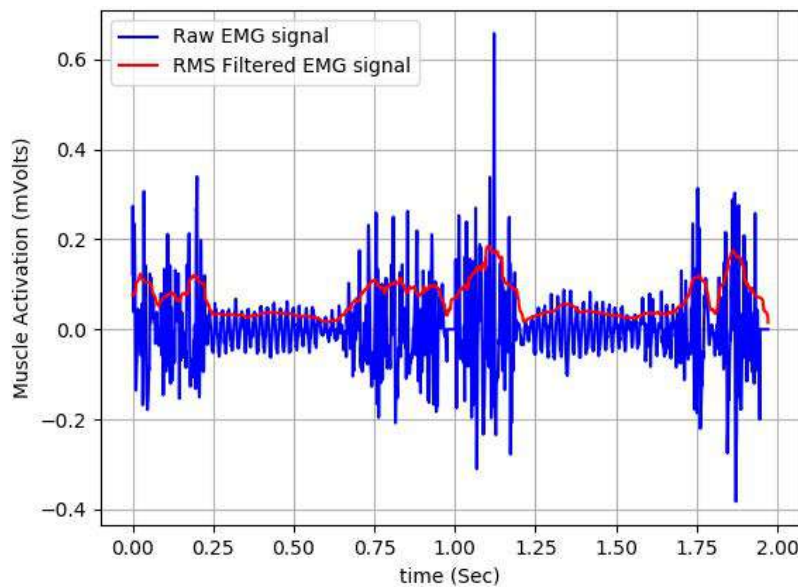


FIGURE 3.34: Raw EMG with RMS of EMG signal..

3.10.3 Fast Fourier Transform

Fourier series takes any periodic function in time and decomposes it into a constant (a_0) and a sum of a series of sinusoids; each sinusoid can is defined

by its frequency(k) and as shown in equation 3.88 [54].

$$f(t) = \frac{1}{2}a_0 + \sum_{k=1}^{\infty} a_k \cos 2\pi kt + b_k \sin 2\pi kt \quad (3.88)$$

The method of finding the Fourier series coefficient is by using the Fourier transform. Fourier transform represents a function (i.e. input data) in the frequency domain spatial. On the contrary, inverse Fourier transform is used to express a frequency-domain function into ordinary measurement spatial, such as time domain, Fourier transform, and its inverse can be described as follows [54].

$$F(\omega) = \int_{-\infty}^{\infty} f(x)e^{-i\omega x} dx \quad (3.89)$$

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} f(\omega)e^{-i\omega x} d\omega \quad (3.90)$$

where $i = \sqrt{-1}$ and $e^{i\theta} = \cos \theta + i \sin \theta$

Amplitude or Phase can be plotted at every frequency or across the entire spectrum, and it can be called the frequency domain representation of a signal, as shown in figure 3.35.

In practice, it is impossible to apply an integration within an infinite range, and the analogue to digital converter (ADC) can not handle a continuous function. Currently, no sensor can measure with zero time in-between records. Therefore it is reasonable to take samples of discrete points on a function (signal) running from zero time to time = N. signal sampling can be imagined as shown in figure 3.36.

To conduct Fourier transform on a discrete set of samples we have to use a discrete Fourier transform, as shown in equation 3.92.

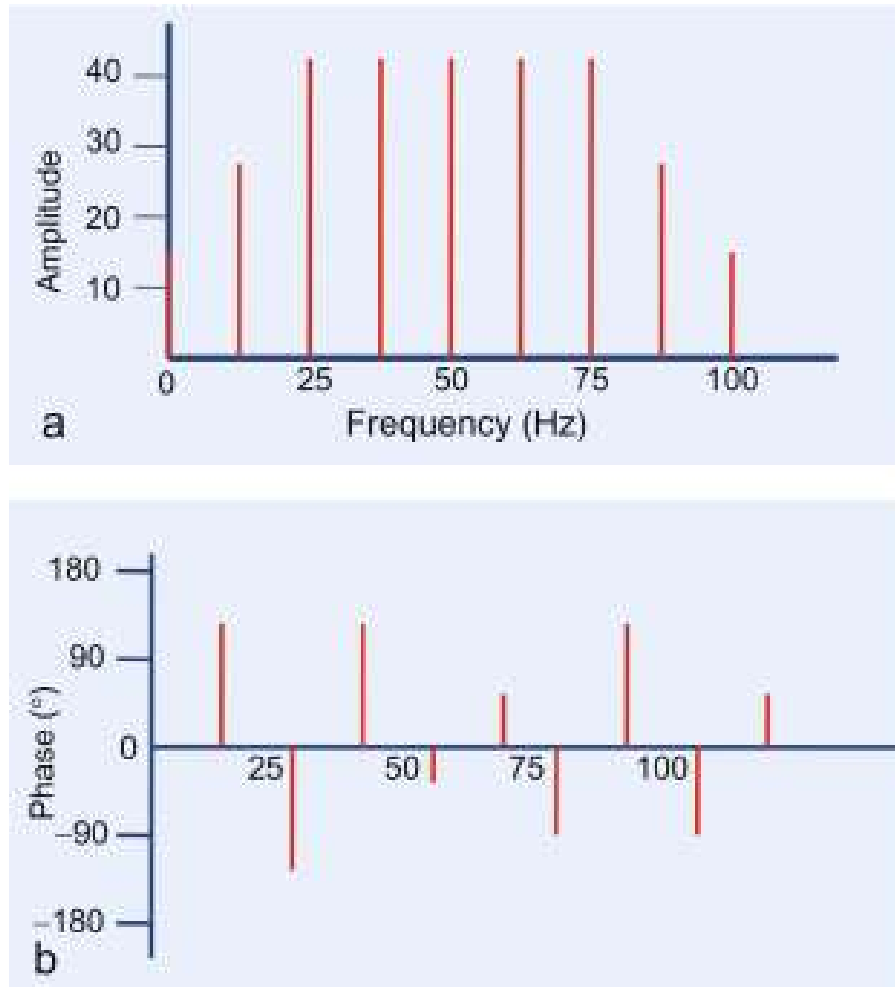


FIGURE 3.35: amplitude and phase in frequency domain [55].

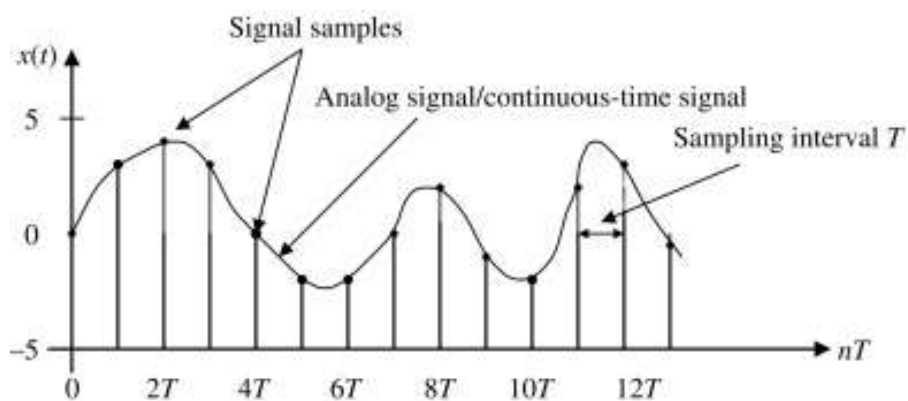


FIGURE 3.36: Signal sampling [56].

$$F(\omega) = \int_{-\infty}^{\infty} f(x)e^{-i\omega x} dx \quad \text{Continuous} \quad (3.91)$$

$$F_k = A_k = \sum_{n=0}^{N-1} a_n e^{\frac{-i2\pi Kn}{N}} \quad \text{Discrete} \quad (3.92)$$

$$A_k = \sum_{n=0}^{N-1} a_n W_N^{kn} \quad (3.93)$$

where $W_N = e^{\frac{-2\pi i}{N}}$

$a_n =$ is sample value at index n

$N =$ Sample size

$K = 0 \dots N - 1$

Inverse discrete Fourier transform a_n can be represented as follows

$$A_n = \frac{1}{N} \sum_{k=0}^{N-1} A_k W_N^{-kn} \quad (3.94)$$

An example of applying a DFT for two points is as follows

$$N = 2, W_2 = e^{-i\pi} = -1$$

$$A_k = \sum_{n=0}^1 a_n \cdot (-1)^{kn} = a_0(-1)^{k \cdot 0} + a_1(-1)^{k \cdot 1}$$

$$A_k = a_0 + (-1)^k a_1$$

$$A_0 = a_0 + a_1$$

$$A_1 = a_0 - a_1$$

For four sample points DFT is as follows

$$N = 4, W_4 = e^{\frac{-i\pi}{2}} = -i$$

$$A_k = \sum_{n=0}^3 a_n \cdot (-i)^{kn} =$$

$$(-i)^{k(0)} a_0 + (-i)^{k(1)} a_1 + (-i)^{k(2)} a_2 + (-i)^{k(3)} a_3$$

$$A_k = a_0 + (-i)^k a_1 + (-i)^{2k} a_2 + (-i)^{3k} a_3$$

$$A_k = a_0 + (-i)^k a_1 + (-1)^k a_2 + (i)^k a_3$$

note : $(-i)^2 = (-1)^2 \cdot (i)^2 = -1$, while $-(i)^2 = 1$
--

$$A_0 = a_0 + a_1 + a_2 + a_3$$

$$A_1 = a_0 - ia_1 - a_2 + ia_3$$

$$A_2 = a_0 - a_1 + a_2 - a_3$$

$$A_3 = a_0 + ia_1 - a_2 - ia_3$$

The upper equations can be expressed in a matrix form

$$\begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (3.95)$$

Another expression can be formulated as follows

$$A_0 = (a_0 + a_2) + (a_1 + a_3)$$

$$A_1 = (a_0 - a_2) - i(a_1 - a_3)$$

$$A_2 = (a_0 + a_2) - (a_1 + a_3)$$

$$A_3 = (a_0 - a_2) + i(a_1 - a_3)$$

If we consider the addition of two arrows in the following flow graph is as follows

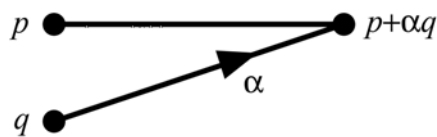


FIGURE 3.37: Two Points flow graph.

4-points DFT can be expressed as follows

In general, the DFT matrix should look like the following matrix:

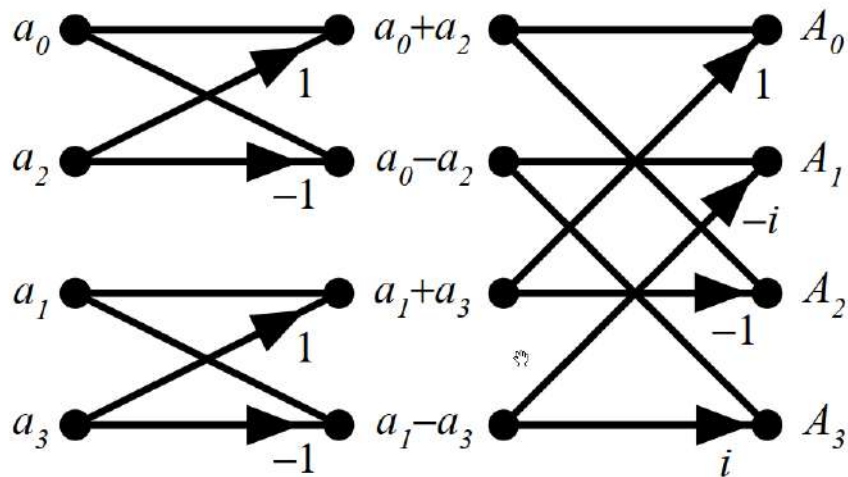


FIGURE 3.38: 4-Points DFT Representation in The Flow Graph.

$$\begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ \cdot \\ \cdot \\ \cdot \\ A_{(N-1)} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & W_N & W_N^2 & W_N^3 & \dots & W_N^{N-1} \\ 1 & \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \cdot & \dots & \cdot \\ 1 & W_N^{N-1} & \cdot & \cdot & \dots & W_N^{N-1^2} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \cdot \\ \cdot \\ \cdot \\ a_{(N-1)} \end{bmatrix} \tag{3.96}$$

For a discrete Fourier transform, the number of complex multiplications that should be applied equals N^2 and $N(N - 1)$ complex additions, which is considered big and takes a long time for large sample points. Fast Fourier transform(FFT) is an algorithm used to reduce the number of operations by reusing the pre-calculated operations, as shown in the following derivation.

Several FFT algorithms were developed; one of the methods is called **decimation in time(DIT)** [57], as shown in the following. Recall equation 3.93, iteration index n can be divided into odd and even as

follows:

$$n = 2r \quad (\text{even})$$

$$n = 2r + 1 \quad (\text{odd})$$

$$\text{where } r = 1, 2, 3, \dots, \frac{N}{2} - 1$$

$$A_k = \sum_{r=0}^{\frac{N}{2}-1} a_{2r} e^{-\frac{2\pi i k(2r)}{N}} + \sum_{r=0}^{\frac{N}{2}-1} a_{2r+1} e^{-\frac{2\pi i k(2r+1)}{N}} \quad (3.97)$$

$$\text{since } W_N = e^{-\frac{2\pi i}{N}}$$

$$A_k = \sum_{r=0}^{\frac{N}{2}-1} a_{2r} W_N^{k(2r)} + \sum_{r=0}^{\frac{N}{2}-1} a_{2r+1} W_N^{k(2r+1)} \quad (3.98)$$

$$A_k = \sum_{r=0}^{\frac{N}{2}-1} a_{2r} W_N^{k(2r)} + W_N^k \sum_{r=0}^{\frac{N}{2}-1} a_{2r+1} W_N^{k(2r)} \quad (3.99)$$

Equation 3.99 showed that each frequency domain amplitude could be computed using two DFT of half range ($N/2$). This is done by adding the odd and even indexes results and multiplying the odd results by W_N^k , the twiddle factor. This method is called radix-2 decimation in the time algorithm, where the data is divided into two groups. For 8 points (DIT) can be formulated as shown in figure 3.39 flow graph. The computation cost of DIT algorithm is as follows

$$2\left(\frac{N}{2}\right)^2 + N = \frac{N^2}{2} \text{ complex multiplies} \quad (3.100)$$

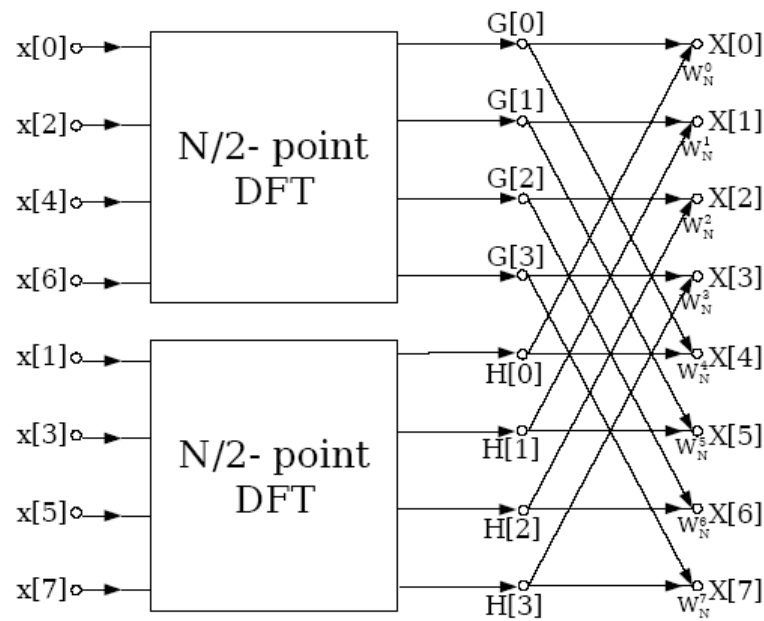


FIGURE 3.39: 8-Points FFT Representation in The Flow Graph using decimation in time method[58].

3.10.4 Convolution Theorem

The output for specific input and system can be calculated using the convolution theorem for any linear time-invariant system. The convolution theorem for a continuous system can be expressed in equation 3.101, where the output is the integration of the multiplication of the input signal and the system [59].

$$\text{Convolution} : (f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(y - \tau)d\tau \quad (3.101)$$

For a digital system, the output is the sum of the whole domain of the multiplication of the input signal and the system, as shown in equation 3.102.

$$y(n) = \sum_{-\infty}^{\infty} x(k)h(n - k) \quad (3.102)$$

Mathematically, convolution produces an output function of two input functions affected by all previous input values. Convolution theorem is

used for several applications such as signal processing, image processing, statistics, and differential equations.

Convolution theorem utilized with digital filters, where both the signal and the filter transfer function are in the frequency domain. The signal is originally in the time domain, but it should be transformed into a frequency domain. The digital filters are originally represented in the frequency domain. Using Convolution theorem, the filtered signal is simply a product of the multiplication of the original signal and the digital filter transfer function.

The following is an example of using the convolution theorem to apply an impulse system to a digital signal :

$$\begin{aligned}
 k &= -2 & -1 & 0 & 1 & 2 \\
 x(k) &= & x_{-1} & x_0 & x_1 & x_2 \\
 h(k) &= h_{-2} & h_{-1} & h_0 & h_1 & \\
 h(-k) &= & h_1 & h_0 & h_{-1} & h_{-2} \\
 \\
 y(0) &= x_{-1}h_1 + x_0h_0 + x_1h_{-1} + x_2h_{-2}
 \end{aligned}$$

Note that the sum of the signal ($x(k)$) and the system ($h(k)$) orders, should be equal to n order. To find $y(1)$, $h(10k)$ should be founded.

$$\begin{aligned}
 k &= -2 & -1 & 0 & 1 & 2 \\
 x(k) &= & x_{-1} & x_0 & x_1 & x_2 \\
 h(1-k) &= & & h_1 & h_0 & h_{-1} \\
 \\
 y(1) &= x_0h_1 + x_1h_0 + x_2h_{-1}
 \end{aligned}$$

Now, to find $y(-1)$ then $h(-1-k)$ should be founded and as shown in following:

$$\begin{aligned}
 k &= -2 \quad -1 \quad 0 \quad 1 \quad 2 \\
 x(k) &= \quad \quad x_{-1} \quad x_0 \quad x_1 \quad x_2 \\
 h(k) &= h_{-2} \quad h_{-1} \quad h_0 \quad h_1 \\
 h(-1-k) &= h_1 \quad h_0 \quad h_{-1} \quad h_{-2} \\
 y(-1) &= x_{-1}h_0 + x_0h_{-1} + x_1h_{-2}
 \end{aligned}$$

The number of sample points for the output signal is $(N = N_1 + N_2 - 1)$, therefore, for the upper x signal the possible output signal is seven, namely $\{y(-3), y(-2), y(-1), y(0), y(1), y(2), y(3)\}$.

3.10.5 Digital Filters

In signal processing, filters are used to remove unwanted parts such as noise or extract valuable components of a signal.



FIGURE 3.40: Signal Filtering.

3.10.6 Butterworth Filter

Butterworth is an electrical engineer who came with two transfer functions that would suitably describe the behavior of a particular filter. Several digital Butterworth filters such as low pass filter, high pass filter, the band-pass filter, notch filter will be attended in this work.

Low pass filter (LPF) attenuates frequencies above a certain chosen cut-off frequency(stopband) and passes the frequencies below the cut-off frequency(passband). LPF is mostly applied before the analogue to the digital conversion process in EMG signals to remove the sampling errors.

High pass filter (HPF) attenuate frequencies below a specifically chosen cut-off frequency (stopband) and pass the frequencies above the cut-off frequency (passband), HPF is used to remove slow changes such as those errors due to motion artefacts.

Butterworth described both LPF and HPF as shown in the equations 3.103 & 3.104 respectively [60].

$$|H(\omega)| = \sqrt{\frac{A_o}{1 + (\frac{\omega_o}{\omega})^{2n}}} \quad (3.103)$$

$$|H(\omega)| = \sqrt{\frac{A_o}{1 + (\frac{\omega}{\omega_o})^{2n}}} \quad (3.104)$$

where $H(\omega)$ is Filter gain(Normalized)

A_o is the Max gain in the passband

ω_o is the lower cut-off frequency (HPF) or upper cut-off frequency (LPF)

ω is the angular frequency of the input signal

n is Butterworth's filter order (integer)

Both **Bandpass(PBF)** and **Notch** filters are combinations of LPF & HPF. Bandpass and Notch filters have upper and lower cut-off frequencies. PBF is used to pass frequencies at a specific range and attenuate the rest, unlike notch filter where it removes filters at a typical annoying narrow band, figure 3.41 shows all the aforementioned filters.

The slope steepness determines the filter quality in transition from the stopband to the passband. Slope steepness is measured in Decibel(dB) per octave. Octave is the distance between the cut-off frequency(f_c) and its double value($2f_c$) in LPF and ($\frac{1}{2}f_c$) in HPF, all in the frequency domain. Increasing the Butterworth filter order increases the steepness of the filter.

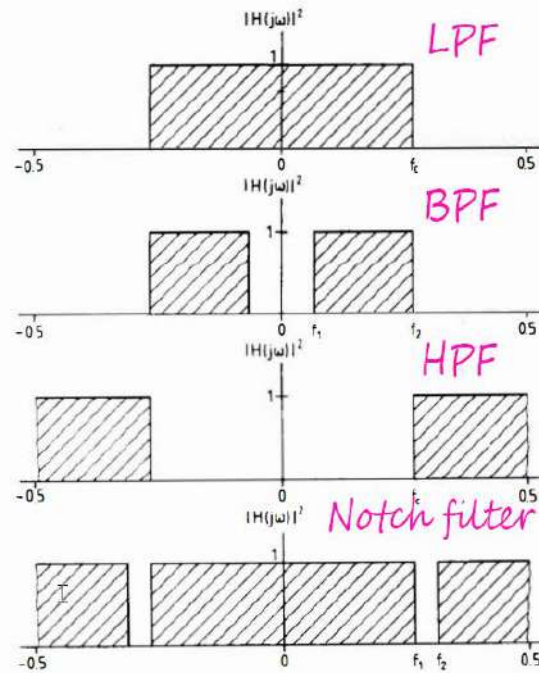


FIGURE 3.41: Low Pass, Bandpass, High Pass, Notch Filters Represented in frequency domain [60].

Therefore, a cascade filter may need to be used by connecting multiple filters in series. The Butterworth filter's order (n) defines the steepness of the transition between stop and passbands. The increment of n increases the steepness of the transition in the frequency domain, as shown in figure 3.42.

Figure 3.43 shows the effect of LPF and HPF on EMG signals with different cut-off frequencies.

3.10.7 Gaussian Smoothing Filter

Gaussian smoothing filter(GSF) can be described by impulse response, as shown in equation 3.105 [61].

$$g(x(t)) = \frac{e^{\frac{-x(t)^2}{2\pi\sigma^2}}}{\sqrt{2\pi\sigma^2}} \quad (3.105)$$

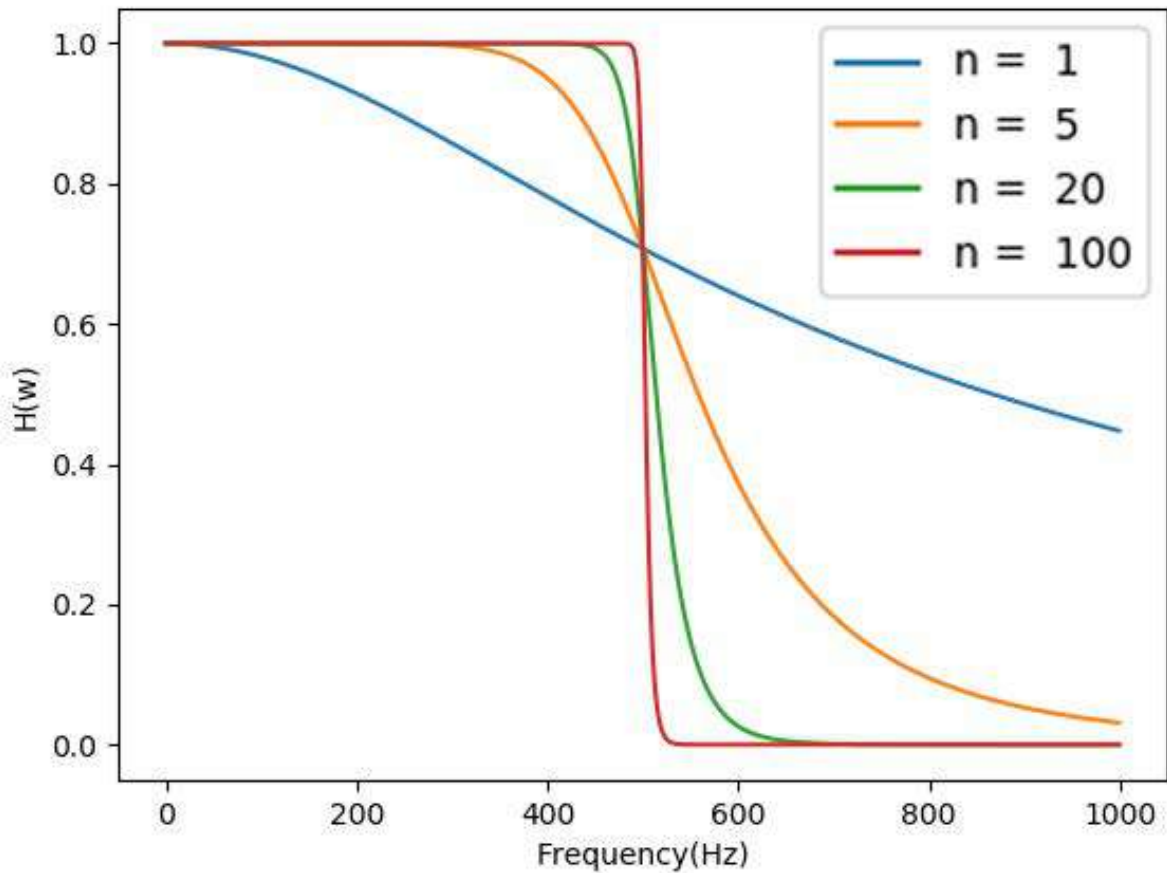


FIGURE 3.42: Butterworth LPF variation with order.

$x(t)$ is a signal sample point, and σ is the standard deviation. As discussed in 3.10.4, the convolution theorem can find the filtered signal utilizing the Gaussian function, as shown in equation 3.106.

$$\hat{x}(t) = x(t) * g(x(t)) \quad (3.106)$$

Where $*$ donates the convolution operator and $\hat{x}(t)$ is the output signal. The discrete Gaussian kernel is used by applying a convolution operator with fixed window size, as shown in equation 3.107.

$$\hat{x}(n) = \sum_{k=-m}^m x(n)g(n-k) \quad (3.107)$$

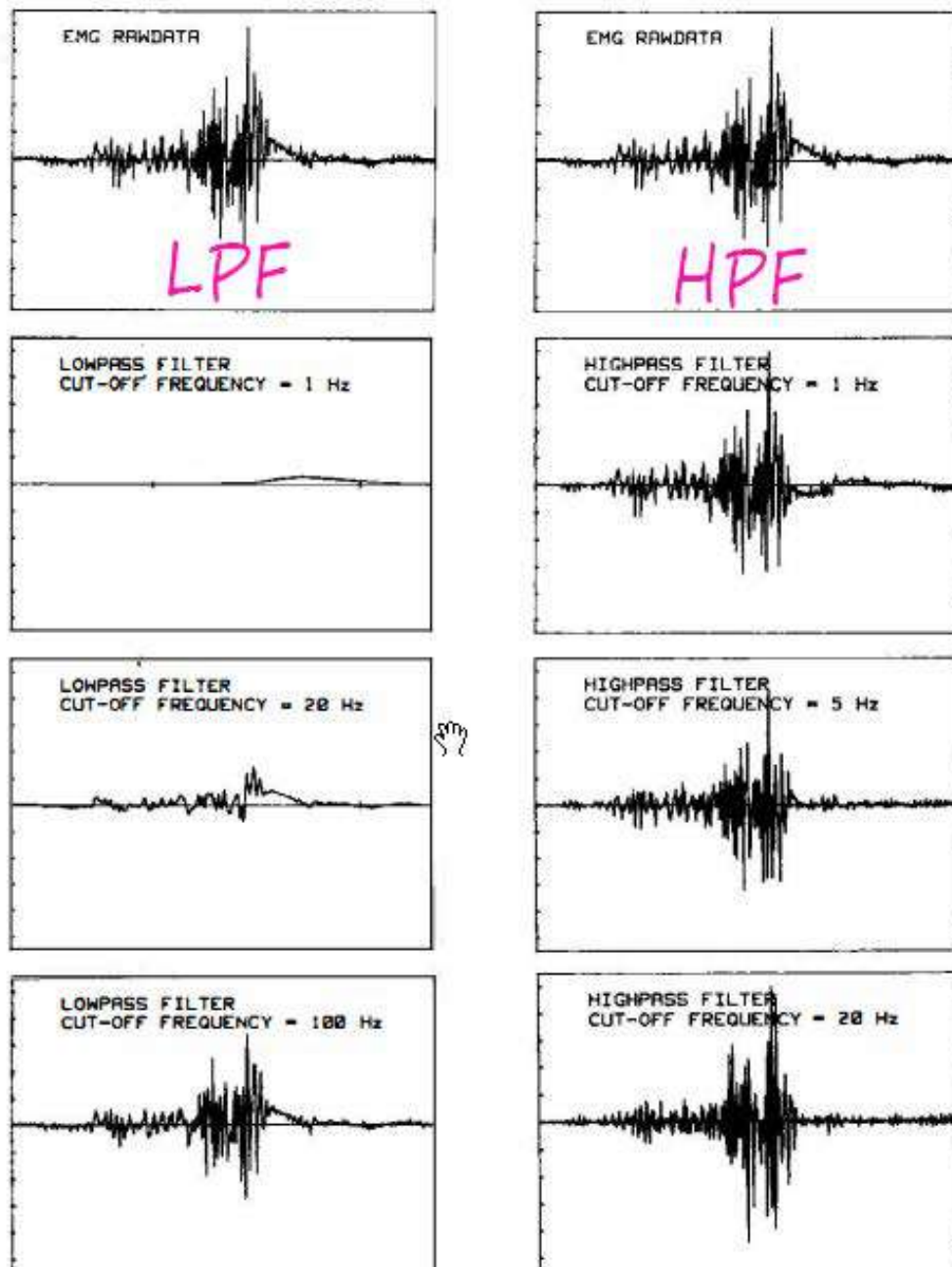


FIGURE 3.43: EMG Signal Filtered with Different Frequencies using HPF & LPF [60].

where $2m$ is the window size.

Gaussian smoothing filters can minimize rises and falls. Gaussian smoothing filter is considered the ideal time-domain filter. Figure 3.44 shows a Gaussian smoothing filter (GSF).

The Gaussian smoothing filter is almost like the mean filter, but here a

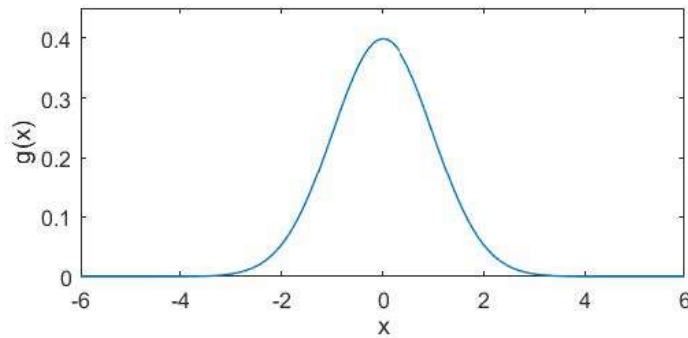


FIGURE 3.44: impulse response of a GSF having $\sigma = 1$.

wight is multiplied by the adjacent points, and the wight is reduced as the points get farther, wherein the mean filter, the distribution is assumed as uniform.

3.11 Ankle Joint Regression Optimization

One of the primary goals of this work is to find the model that governs the relationship between the ankle joint angle and the muscles contractions assuming there is a correlation between the two variables. Due to the data acquisition error, noise introduction, and human non-perfect movement in the regression experiments may give a random muscle contraction within a specific range for each ankle joint angle. In this work, the randomness was solved by assuming that the muscle contractions have a Gaussian distribution at each ankle joint angle(using a probability distribution due to the randomness). Therefore this problem can be overcome by finding the distribution parameters. The expectation(C.14) is used to find the most frequently received muscle contraction at a certain angle. The variance(C.18) would measure the uncertainty value of the estimated muscle contraction value. Figure 3.45 shows the Solus muscles contractions at 10 ankle joint angles for one regression experiment.

Figure 3.45 proves that the muscles contractions have a Gaussian distribution.

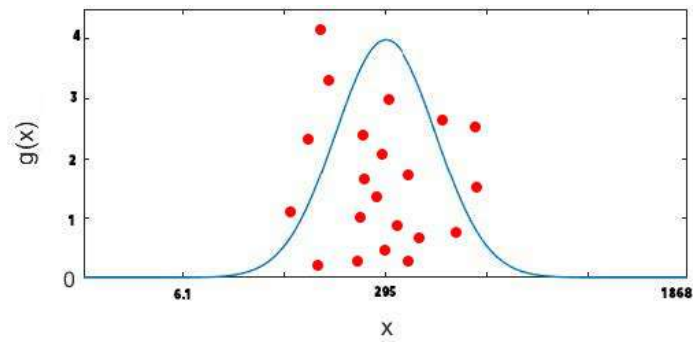


FIGURE 3.45: Solus Muscle Contractions at 10 deg Plantarflexion of Ankle Joint.

Chapter 4

Experimental Work

4.1 Introduction

In this chapter, the model proposed in chapter 3 is further investigated. A CAD design of model 2 3.2.2 was modelled using solid work, and an explanation of each part was introduced. Furthermore, the CAD design is manufactured as a prototype using a 3D printer to test the required control strategy in the future work. The data acquisition system for building the muscle activity and ankle joint regression model are explained in detail in this chapter. This chapter also explains connecting the data acquisition electronics and their programming methodology in detail. The last sections of this chapter presents the regression experiments and the used method for synchronizing the EMG-angle dataset. A video is attached with the electronic copy of the thesis to illustrate the designed CAD model and the regression experiments.

4.2 Data Acquisition System

The following Components and tools are used in the regression process require the measurement of the ankle joint and corresponding lower limb

muscle activity. Biosensing is also used as a main component in the created powered prosthesis.

4.2.1 NodeMCU

The NodeMCU ESP-32S is one of the production boards that NodeMcu generated to test the ESP-WROOM-32 module. It is based on a single chip of the ESP32 microcontroller that supports Wifi, Bluetooth, Ethernet and Low Power. NodeMCU ESP-32s has 28 digital input/output pins, and 8

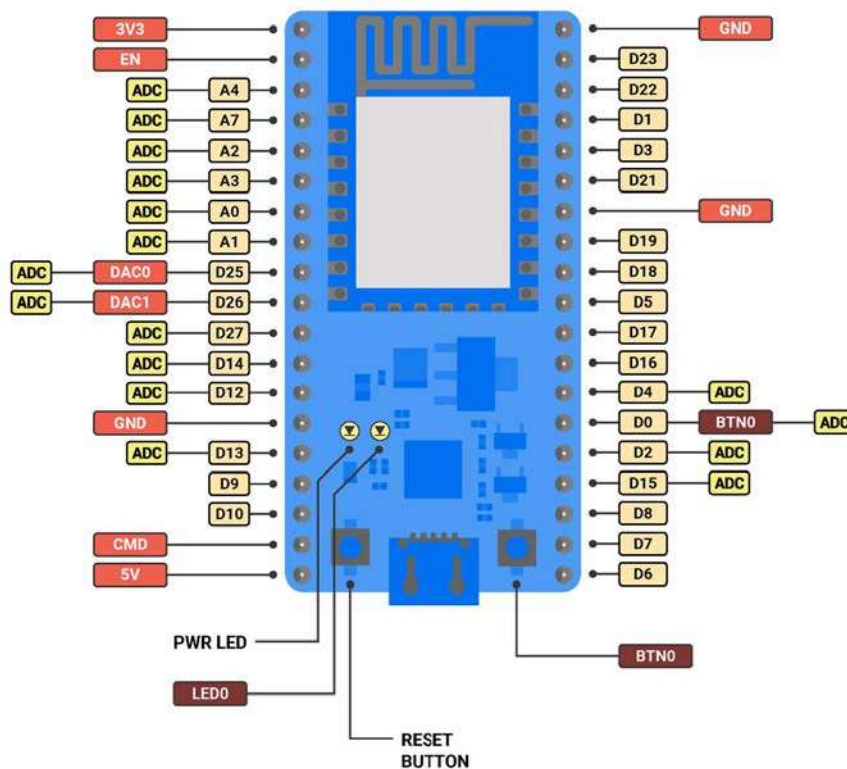


FIGURE 4.1: NodeMCU ESP-32S GPIO Pins [62].

Analog Input Pins with 2 Analog Outputs Pins. Power is supplied to the NodeMCU ESP-32S via the USB Micro B on-board adapter or directly through the 'VIN' pin. Automatically, the power source is chosen. Power is supplied to the NodeMCU ESP-32S via the USB Micro B on-board adapter or directly through the 'VIN' pin. Automatically, the power source is chosen. The system can run on a 6 to 20 volt external supply. The voltage

regulator can overheat and harm the system if more than 12V is used. 7 to 12 volts is the recommended range. The NodeMCU ESP-32S sold with a serial-to-USB chip on board that enables programming. It is possible to program NodeMcu-32s using various languages: Lua, Python, Java, Ruby, etc. In this work, NodeMcu-32s are programmed using the Arduino-C programming language. In this work, NodeMcu-32s is programmed to be a local server and access point to send data to a personal computer, as shown in figure 4.2. Over air Transmission(OTA) was enabled to program NodeMCU-32s wirelessly. Code for programming NodeMcu-32s as a local server, access point with Enabling OTA can be found in B.1.

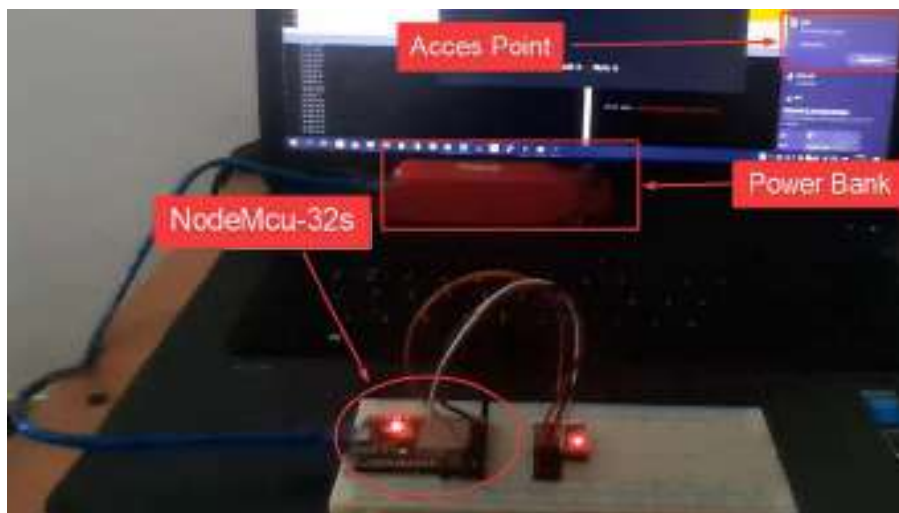


FIGURE 4.2: NodeMCU ESP-32S Access point usage.

4.2.2 IMU (MPU6050)

IMU is a type of equipment that responds to or detects parameters of physical motion, including acceleration, rotation, or shift of location [63]. IMUs mainly include accelerometers to measure linear acceleration and gyroscopes to measure angular speed. IMUs may include, in some cases, a magnetometer. The MPU6050 consists of a three-axis accelerometer and a three-axis gyroscope and is a micro-electro-mechanical device (MEMS). It

allows one to calculate the speed, orientation, inertia, displacement and other characteristics of motion [64]. This board uses I^2C protocol for the Arduino interface. The key advantage of the MPU6050 is that it can conveniently be combined the accelerometer and gyro. MPU6050 includes eight pinouts as shown in figure 4.3. MPU6050's pins function is as shown in table 4.1. This board is used to measure the orientation of the ankle joint and will be explained in ankle joint angle measurement 4.2.3.

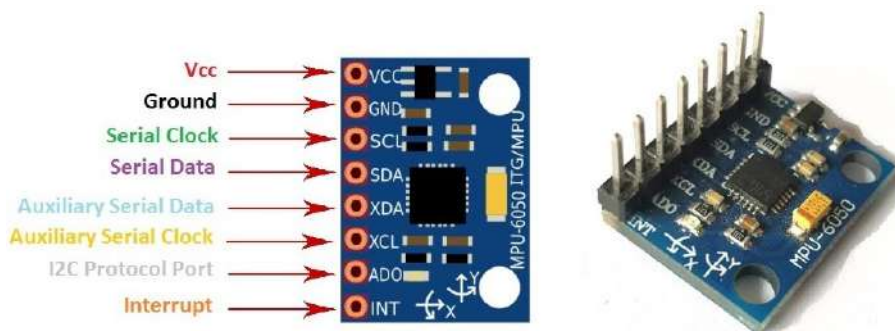


FIGURE 4.3: MPU6050 Pins.

TABLE 4.1: MPU6050 pinout discription

Pin	Pin Name	Description
1	VCC	Supply Voltage(3-5V)
2	GND	Ground
3	SCL	I^2C Serial Clock pulse
4	SDA	I^2C Serial Data transfer
5	XDA	other interfaced other I^2C (Auxiliary Serial Data)
6	XCL	the interfaced other I^2C (Auxiliary Serial Clock)
7	AD0	Gives new I^2C address for more than one MPU6050
8	int	Indicate if data is available(interrupt)

4.2.3 Ankle Joint Angle Orientation Measurement

The ankle joint angle is essential since it represents the target variable in the regression operation. The ankle joint angle is measured using double IMUs, one adhesive to the foot and the other adhesive to the shank. IMUs measure

both the linear acceleration and angular velocity in 3 directions. The rotation measures the IMU's orientation based on the linear acceleration.

The MPU6050 should be connected with NodeMcu-32s as shown in figure 4.4.

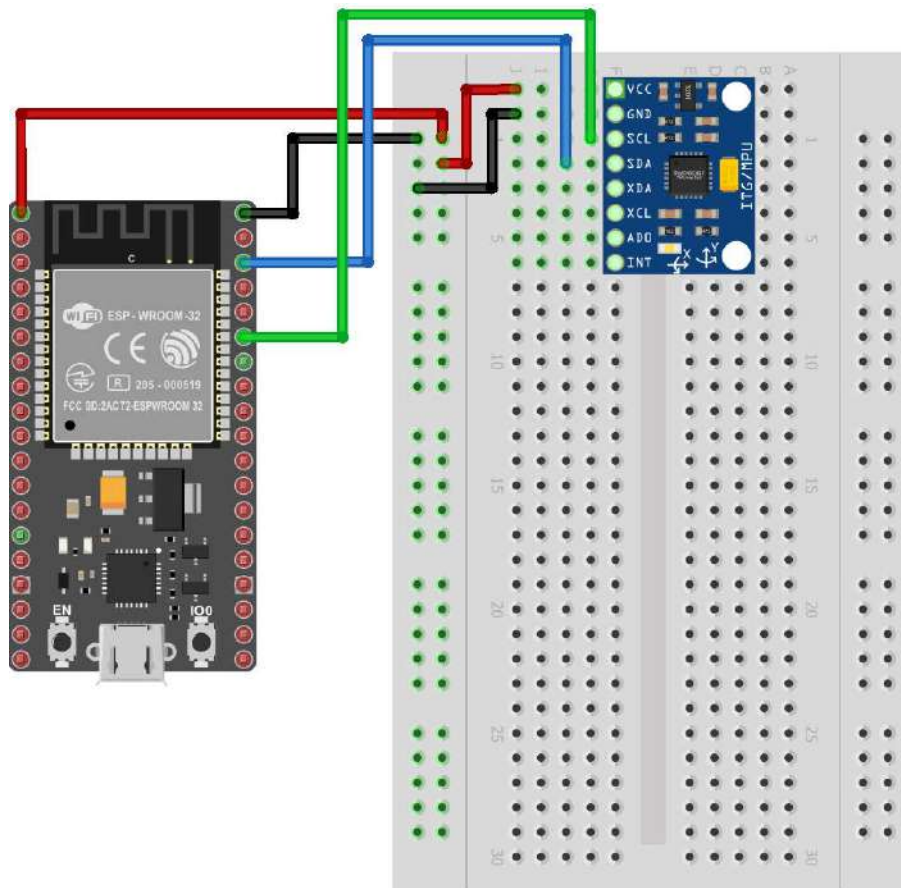


FIGURE 4.4: NodeMcu-32s with MPU6050 Connection.

NodeMcu-32s programmed to collect IMU's orientation using accelerometer data and according to equations 3.79 and 3.85. The result is as shown in figure 4.5. The result is more accurate than the gyro results, since the accelerometer work according to gravity acceleration reference. The orientation measurement was accurate using an accelerometer, but it was shaky due to low accelerometer frequency.

Therefore, a new approach should be used to obtain better results. A

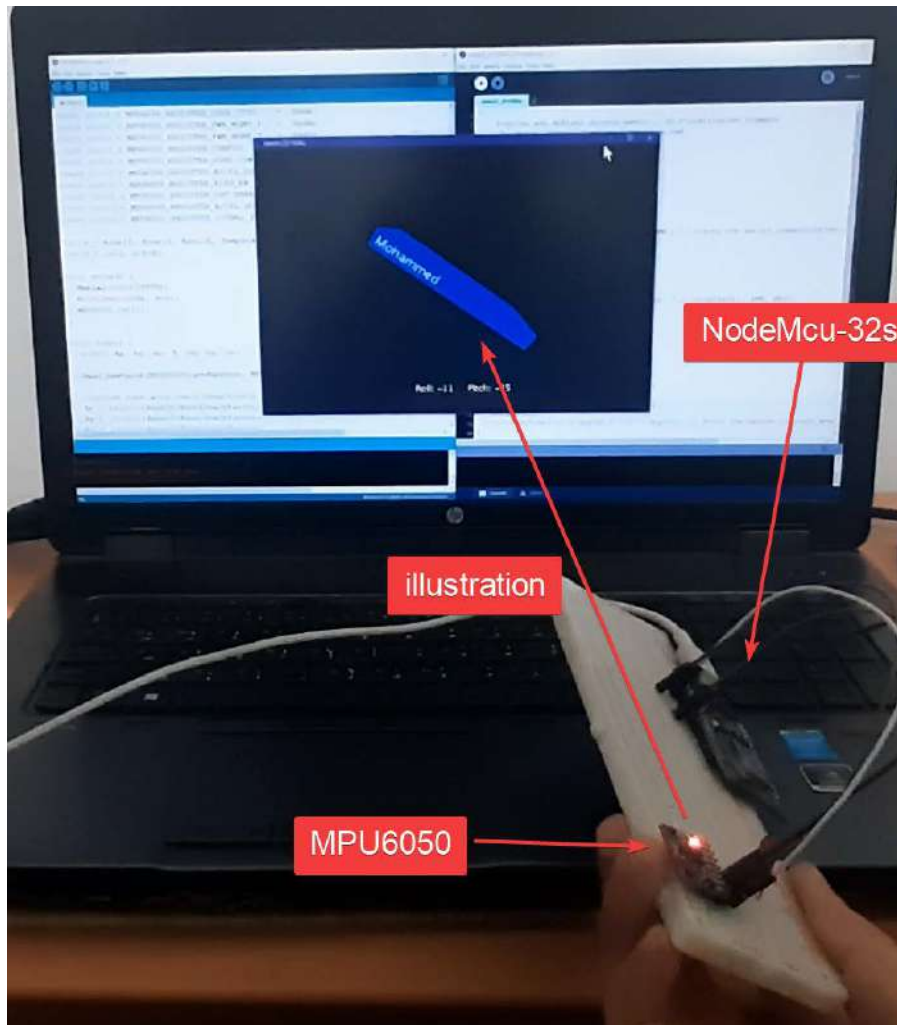


FIGURE 4.5: MPU6050 Enclosure.

combination of accelerometer and gyroscope is used to get a better result, as shown in figure 4.6. A complementary filter is used in this case to combine the reading from both the accelerometer and gyroscope with a factor k .

It is essential to calibrate the sensor before operating a task. The NodeMcu-32 is programmed to process a procedure to eliminate offset in all six readings. This procedure is done by putting the MPU6050 horizontal at the awake time, and then the program will automatically collect 500 readings for both accelerometer and gyroscope in all three axes. These 500 readings are considered as offsets. The final code for ESP32 Appendix B.2 makes NodeMcu-32s function as an access point and a local server while

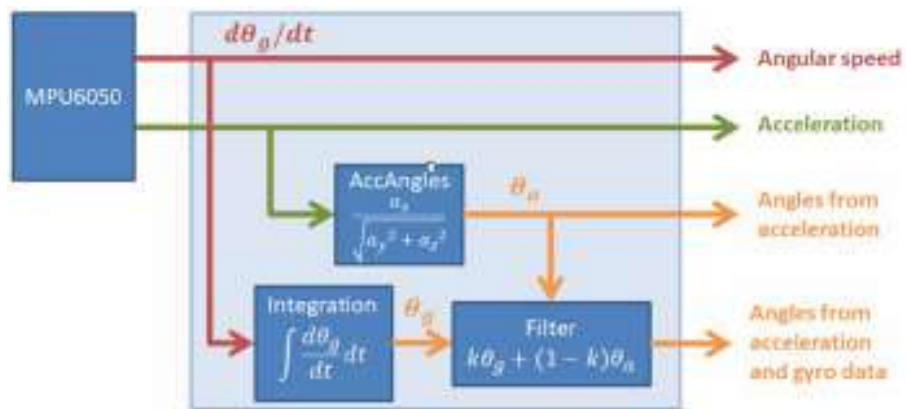


FIGURE 4.6: MPU6050 Enclosure[65].

enabling and as shown in figures 4.7, 4.8, 4.9, and 4.10, . The code used the MPU6050_light library for collecting sensor orientation, which gave reasonable results with a single MPU6050, but it requires some .cpp course code adjustment when using double MPU6050.

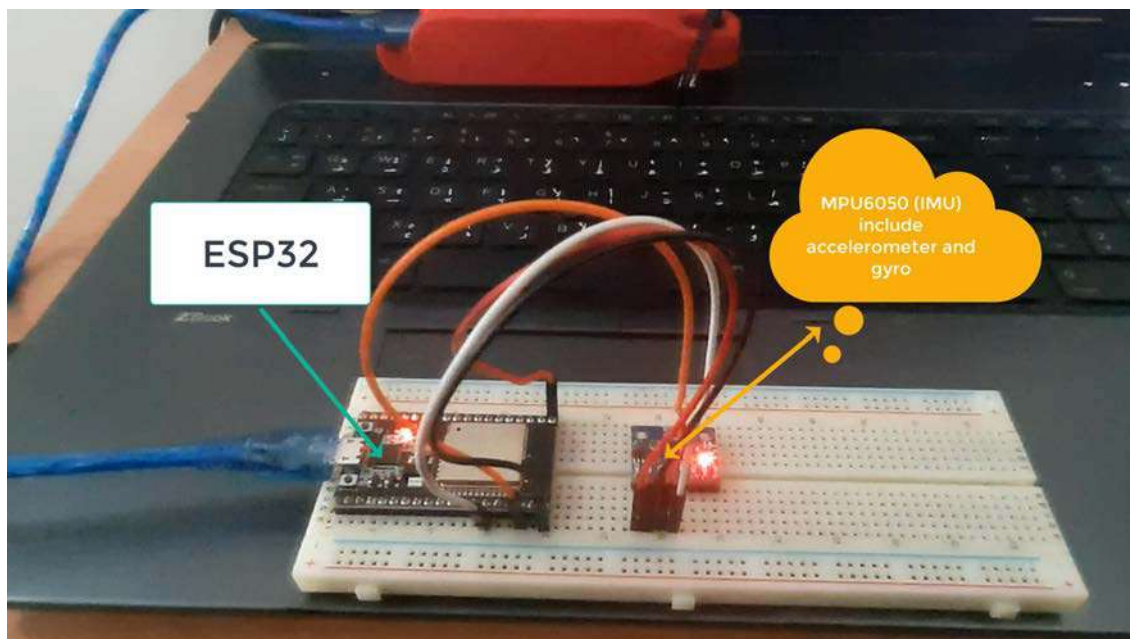


FIGURE 4.7: MPU6050 and NodeMcu-32s Connection.



FIGURE 4.8: Power Bank.

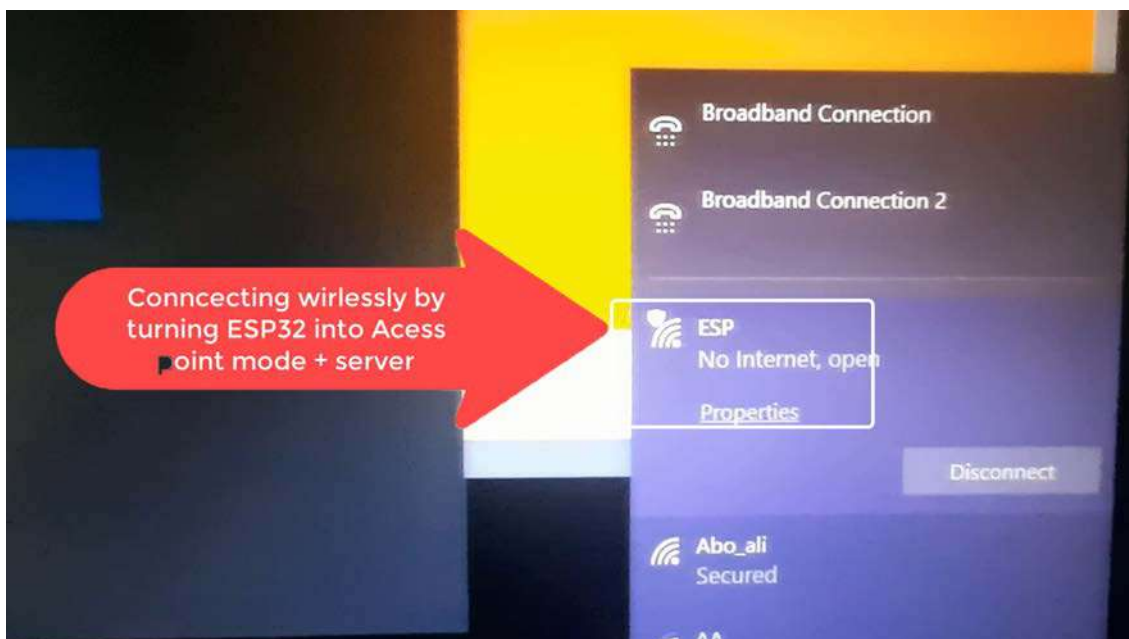


FIGURE 4.9: NodeMcu-32s as Access point.

4.2.3.1 Double IMUs' Orientation Measurement

Two IMUs needed to calculate the relative angle between the two sensors to measure the ankle joint angle. The first sensor should be situated inside a shoe to obtain the foot orientation, and the second one should be positioned on the shank to measure its orientation. The IMU installed inside the shoe is



FIGURE 4.10: Orientation Illustration.

shown in figure 4.11.

The second MPU6050 should be installed on the shank using the predesigned enclosure model as shown in figure 4.24. The first attempt has done by connecting both MPU6050s a long one line for both SDA and SCL pins and using I2c protocol as shown in figure 4.12. Using the AD0 pin for powering up the second MPU6050 will make its address equal 0x69, but the reading acquisition is still using the same line(same SDA and SCL). The code used for such connection is as shown in the appendix B.3. The code is done by creating a new Arduino library which is the same as the MPU6050_light library but with a different address. As shown in figure 4.13, both IMUs connected via a single line. Therefore, it was observed that the second MPU6050 readings are affecting the first one. A new trial approach should be attempted in such a case since the second MPU6050 reading is affecting the first one. Therefore the connection between the two MPU6050 and NodeMcu-32s was done across multiple lines, as shown in figure 4.14.



FIGURE 4.11: MPU6050's Installation inside a shoe.

4.2.3.2 Ankle Joint Angle Visualization.

As mentioned in section 3.9.2 the NodeMcu-32s is programmed to work as a TCP server. Therefore, the data should be received to be visualized in a specific form. Unity3D [66] is a free application used for game

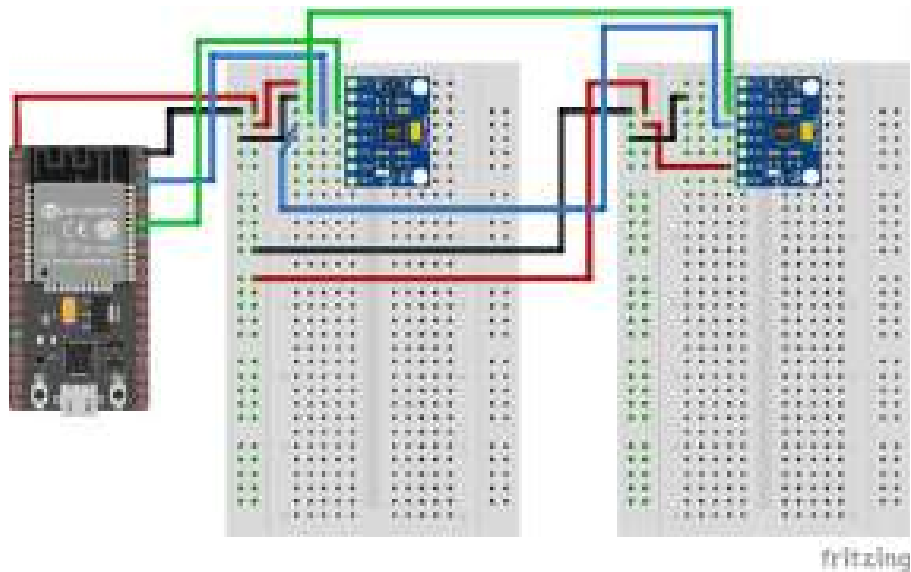


FIGURE 4.12: Two MPU6050's Connection with NodeMcu-32s.

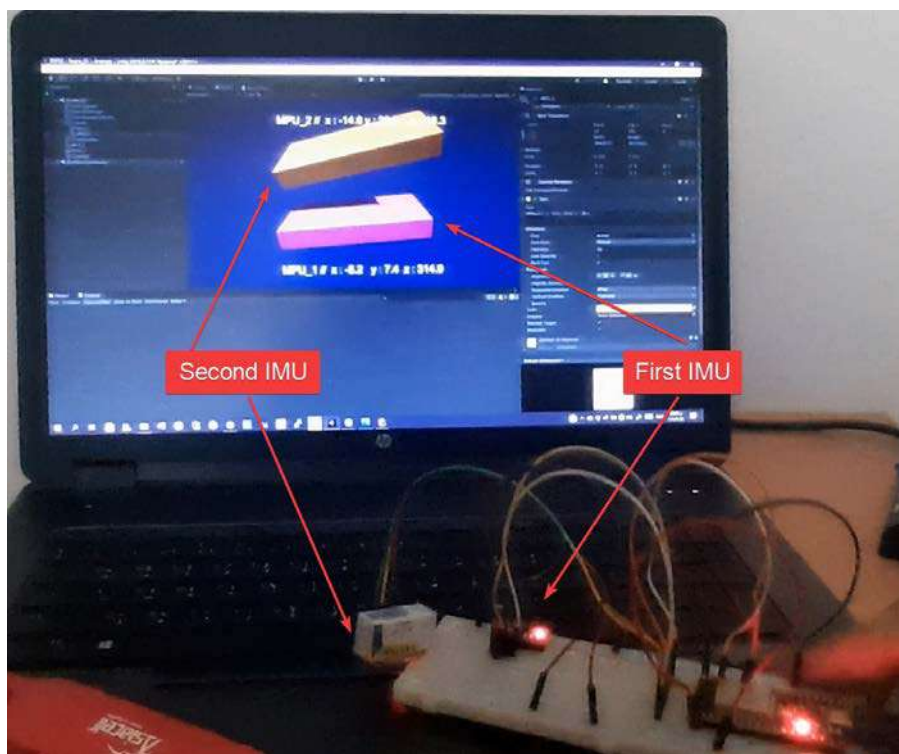


FIGURE 4.13: Two MPU6050s Illustration.

development, 2D and 3D Visualization and environment simulation. Unity 3D was used to visualize the ankle joint angle movement. Unity 3D is programmed using the C# programming language. Solid primitives or 3D models can be created/imported into unity3D. C# script is attached to 3D

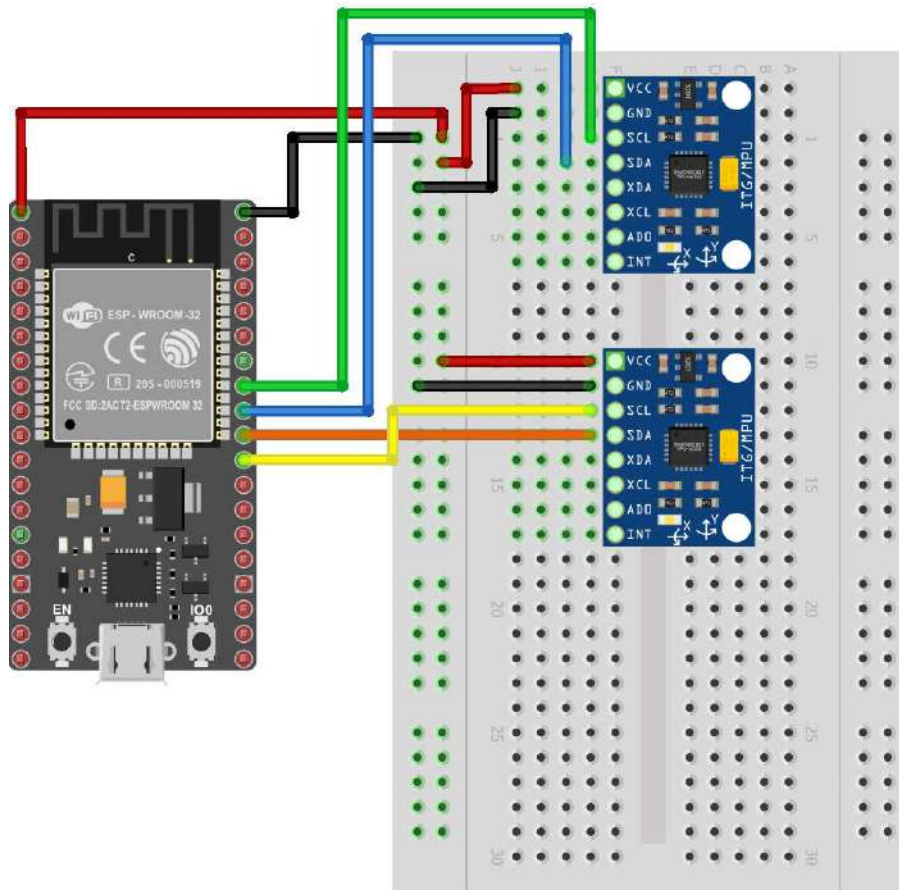


FIGURE 4.14: Two MPU6050's Connection with NodeMcu-32s.

models to perform a specific action. Primitives/3D models can be saved in a Unity 3D scene. Prefabs are saved objects with specific attached components such as mesh, script, material, animation, rigid body component, collider...etc. The created scene is a simple scene contacting 2 Cuboids representing the shank and foot with some lighting and post-processing effect, as shown in figure 4.15.

Unity 3D project has two scripts, *InputController.cs* and *manager.cs*. The *InputController.cs* script used to collect data from NodeNcu-32s TCP server socket and cast it to a suitable variable type to be understood by Unity3D (Data transferred across the network as a bucket, therefore it needs to be cast into float variable). The C# *manager.cs* script (Appendix B.5) is used to receive data from the *InputController.cs*. The

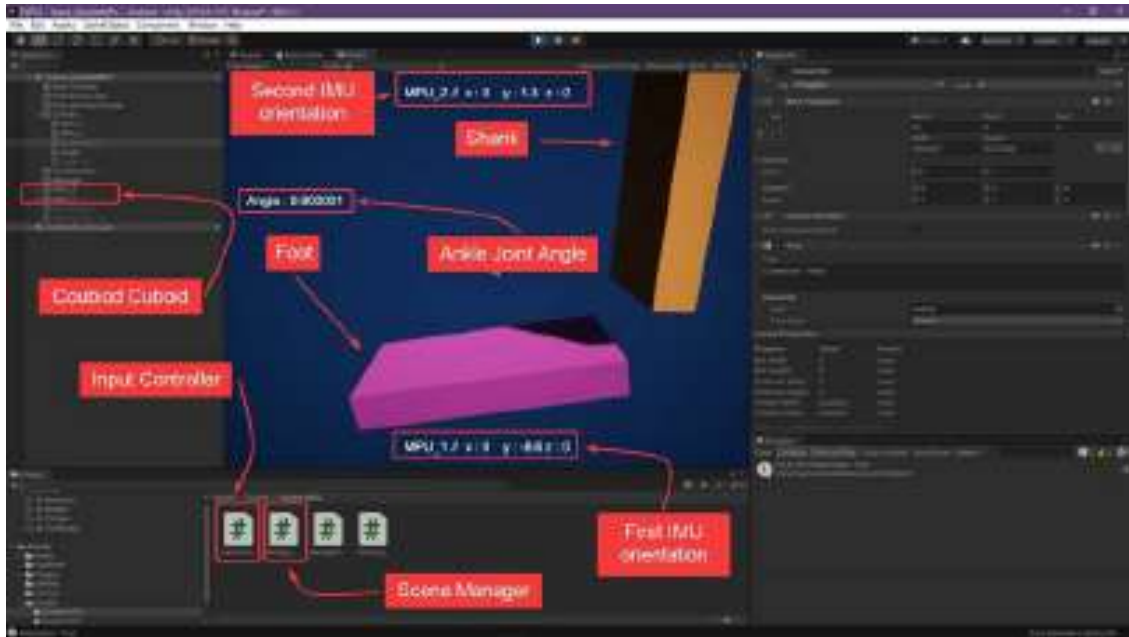


FIGURE 4.15: Unity3D UI - Angle Joint Angle Visualization Project.

manager.cs takes only the orientation around the y-axis and applies it to cuboid primitives, representing the foot and shank. In figure 4.15 text fields are added to show the orientation around all three axes. The whole unity 3D illustration project is uploaded to the **GitHub** platform and downloaded from [67] directory. Figure 4.16 shows the 3D printed model for the data

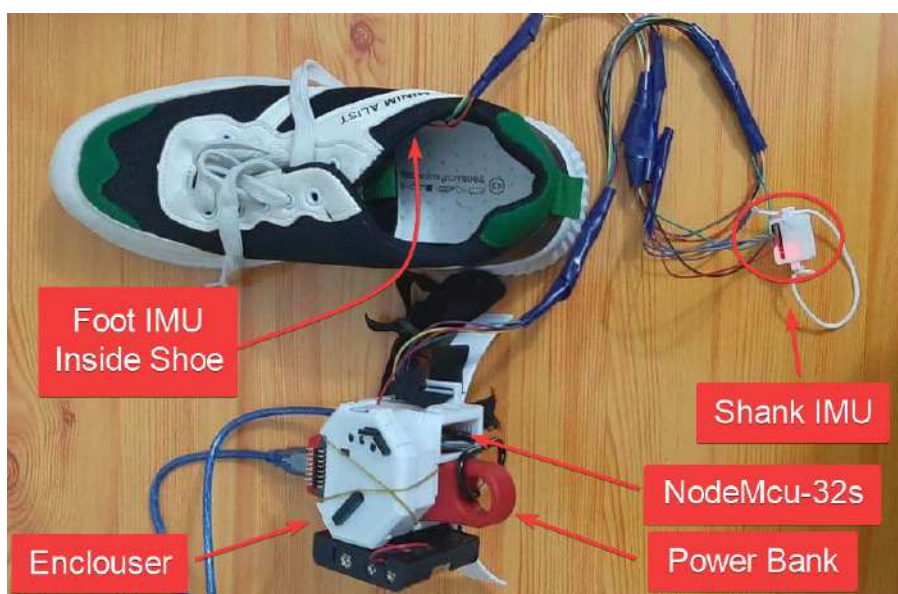


FIGURE 4.16: Data Acquisition Setup.

acquisition system enclosure. The enclosure functions as a case for OpenBCI, NodeMcu-32s, and the power bank for operating NodeMcu-32s. Two IMUs are wired to the NodeMcu-32s includes rubber for binding it around the shank, and one will be positioned inside a shoe, as shown in figure 4.17. The illustration of the physical change of the ankle joint inside



FIGURE 4.17: Foot's IMU Setup.

Unity3D is as shown in Figures 4.18, 4.19, and 4.20.

4.2.4 OpenBCI

OpenBCI stands for open-source brain-computer interface. The OpenBCI goal is to provide anyone with the computer tools necessary to sample the electrical activities of their body. OpenBCI includes several products for measuring biosignals. The signals that can be measured are EEG(Electroencephalography), Electromyography(EMG), and ECG(Electrocardiogram). OpenBCI has an accelerometer as an additional tool for decoding movement. This type of board can be selected according to the number of electrodes wanted to analyze data. Ganglion board(which is used in this work) has four channels, as shown in figure 4.21.



FIGURE 4.18: Ankle Joint Angle Visualization.

Cyton has eight input channels. The data is sent over Bluetooth Adapter in the case of windows and Linux operating systems. Open-BCI comes with its own GUI as shown in figure 4.22.

4.2.5 Enclosures Designs

The following models are used as enclosure components for the data acquisition system. The objective of the data acquisition system is to get the Electromyographical data of the calf muscle group corresponding to its ankle joint angle for building a statistical regression model. All the following models are designed using Solidworks 2020.

The philosophical core of this work is that if one can measure the EMG signal of several muscles corresponding to ankle joint angle(Regression



FIGURE 4.19: Ankle Joint Angle Visualization.



FIGURE 4.20: Data Acquisition system Installation.

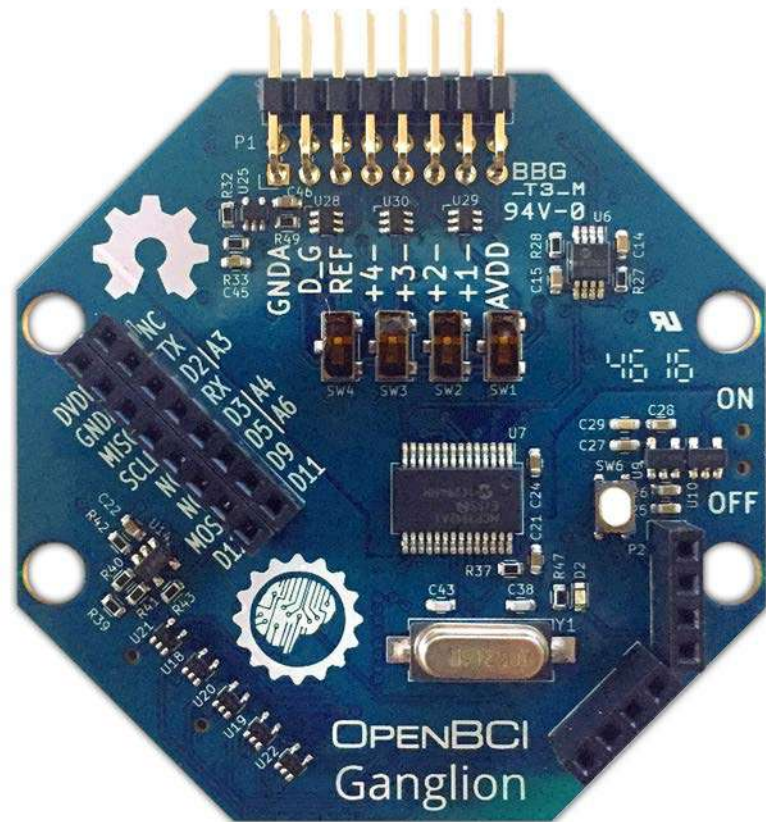


FIGURE 4.21: Open-BCi Ganglion board.



FIGURE 4.22: Open-BIC GUI.

model) for several intact limbs (several subjects), is it possible to apply this pattern to an amputated limb as future work? And if it is not, what is the deviation factor between intact and amputated limbs in the pre-created

statistical model.

The model shown in figure 4.23 is used as an enclosure for OpenBCI board(which is used to collect EMG signals of the Calf muscle group), ESP32(used to send the ankle joint angle data to a computer), and a power bank. A belt accompanies this model to bind it to a subject thigh. This model is designed to make the whole data acquisition system operate wirelessly. The model shown in figure 4.24 used as an enclosure for

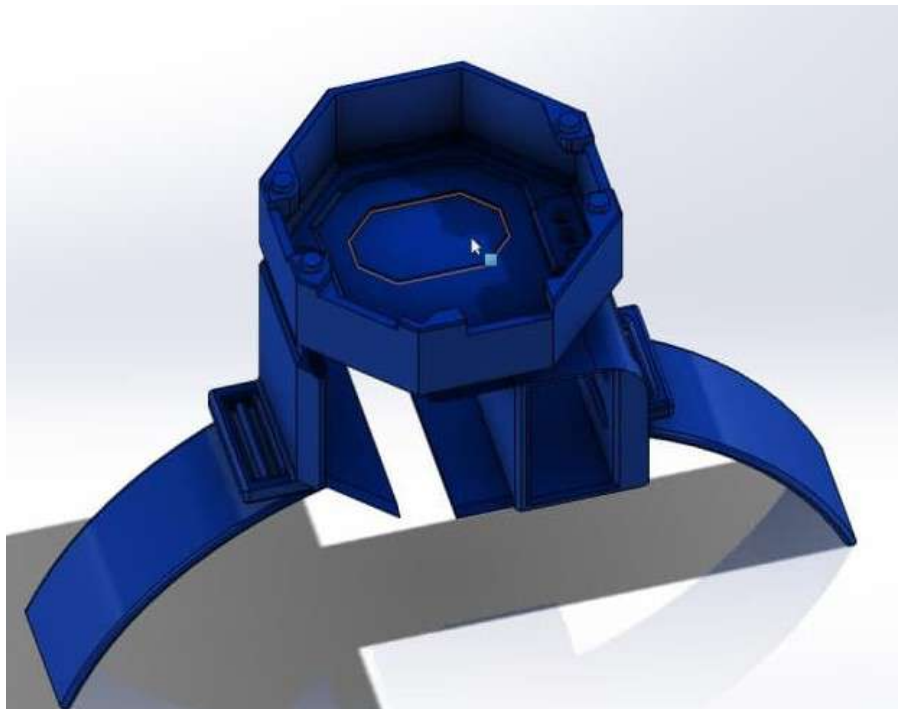


FIGURE 4.23: Data Acquisition Enclosure.

MPU6050 and it also accompanied by a belt to be bound on a subject shank.



FIGURE 4.24: MPU6050 Enclosure.

4.3 Powered Ankle-Foot Prosthesis Modelling and Prototype Manufacturing

A powered ankle-foot is modelled based on the derived mathematical model in section 3.2. The modelling is implemented using Solidworks 2020. Solidworks is considered as an industry-leading application in mechanical design and simulation. The model is manufactured as a prototype using a 3D printer machine.

3D printing is an additive method that involves building up layers of material to produce a three-dimensional object. 3D printing is also suitable for quick prototyping since it allows for the development of complicated, customized designs. Thermoplastics, such as acrylonitrile butadiene styrene (ABS), metals (including powders), resins, and ceramics are among the materials used in 3D printing [68]. 3D printers are used in several applications such as aerospace, automotive, medical, robotic, etc.

3D printers are used in several applications such as aerospace, automotive, medical, robotic, etc. 3D printers are available in several sizes and speeds. Endure 5 plus printer is used to manufacture the prototype

parts. Endure 5, as shown in figure 4.25. ABS filament is used as the filament material. ABS is known for its high hardness and temperature resistance than PLA filament.



FIGURE 4.25: MPU6050 Enclosure.

4.4 Below Knee Prosthetic Design

The final design was modelled using Solidworks 2020, and it includes the following parts:

- Motor Case

The motor case used to hold the used motor and rotated with the rotation of the motor joint. The motor case is as shown in figure 4.26. Figure 4.26 shows the i Nema 17 stepper motor installation inside the motor case.

- Motor Pulley

GT pulley of 20 teeth and 5 ϕmm diameter is used to link the motor to the rest of the device, the pulley and its place in the device is as shown in figure 4.27 (A), and 4.27 (B) respectively.

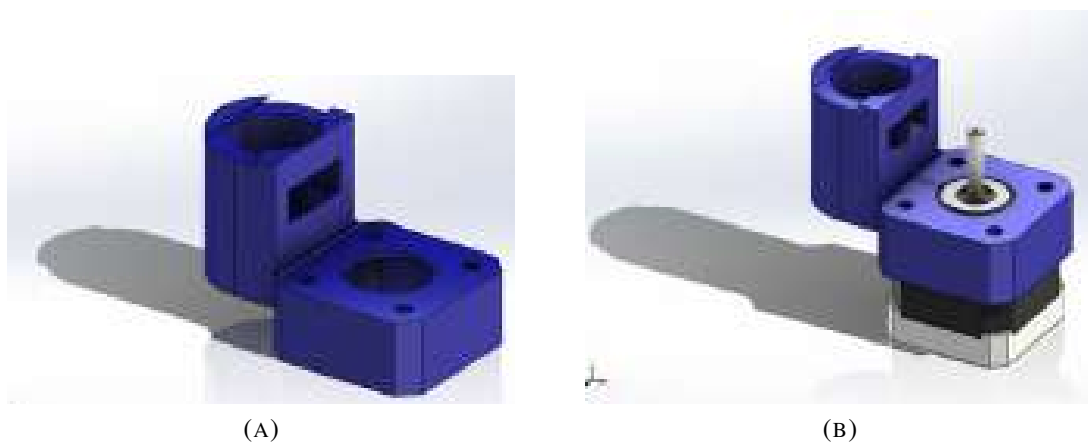


FIGURE 4.26: Motor Case.



FIGURE 4.27: Motor Pulley.

- Lead Screw

T8 Lead Screw of 8 mm diameter is used to transmit the motion to the designed ankle-foot prosthesis, as shown in figure ??

- Lead Screw Pulley

A modified GT pulley with a larger height and diameter is increased to hold the lead screw. The modified pulley has a couple of grooves to make the pulley holden inside the motor case and hold a bearing; the modified pulley is as shown in figure 4.28. Modified pulley, lead screw,

and the motor case can be assembled as shown in figure 4.28. A belt is used to link both pulleys (motor pulley and lead screw pulley).

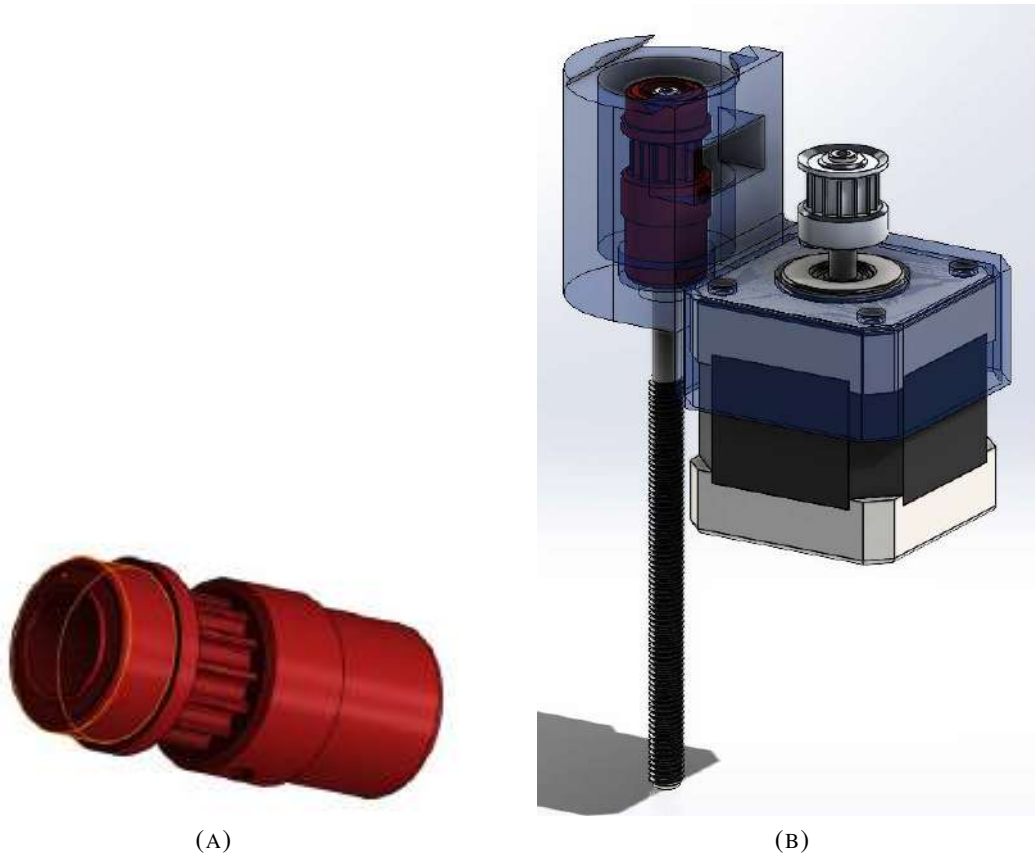


FIGURE 4.28: Lead Screw Pulley with Motor Case.

- Motor Cap

A motor cap is used to link the motor case with the device. The motor cap is considered as a part of the motor joint (discussed in 3.2.2). The motor cap has a slot to link it to the motor joint axle and keep the motor case rotating with the ankle joint rotation. The motor cap and its assembly with the rest parts are shown in figures 4.29 and 4.29, respectively.

- Motor Joint Axle

The motor joint axle is considered one of the main parts of the motor joint and links the motor case to the pylon of the powered prosthesis.

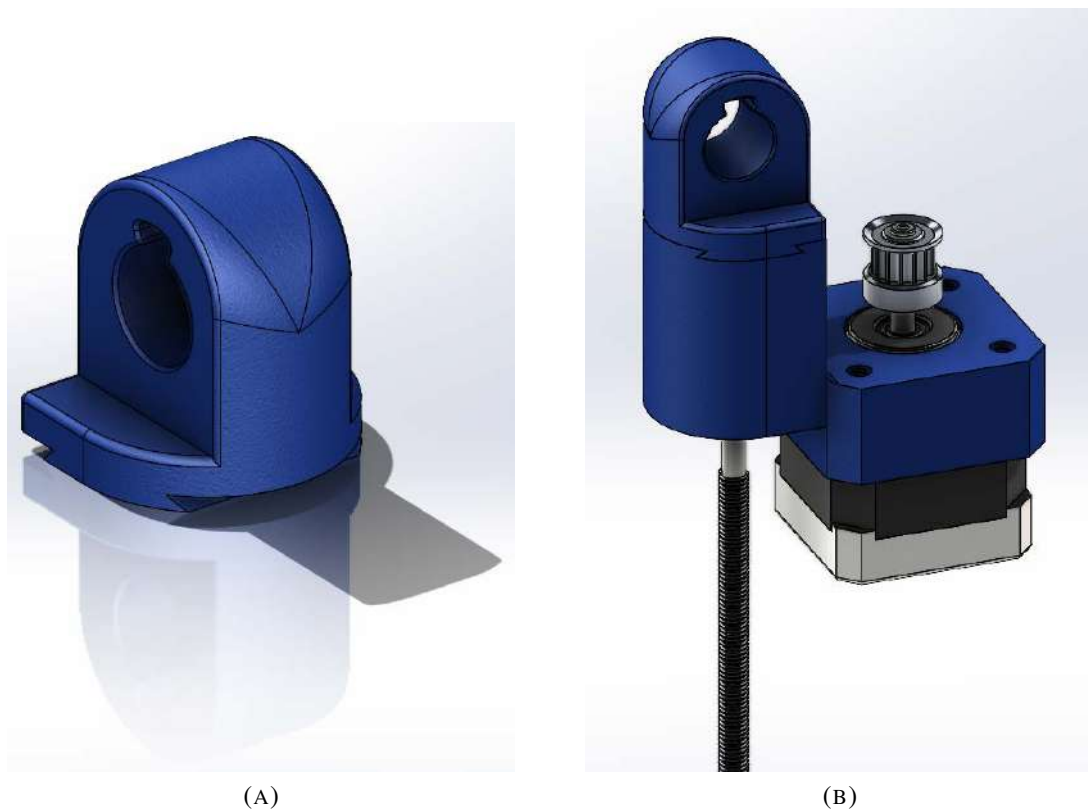


FIGURE 4.29: Motor Cap with Motor Case.

The motor joint axle is designed to have a key to make it firmly concentrated with the motion of the whole motor joint. Motor joint axle and its assembly as shown in figures 4.30 and 4.30, respectively.

- Pylon

A pylon is a part that links the foot or the ankle joint with the socket; figure 4.31 shows the pylon designed with a key-space to make the motor joint axle be able to pass through the pylon. The designed pylon also works as a shield for the motor, and it directly connects with the motor joint axle, as shown in figure 4.32.

- Connector

The connector links the pylon with the ankle joint, constructing a part of the ankle joint. The connector is connected with the pylon using a

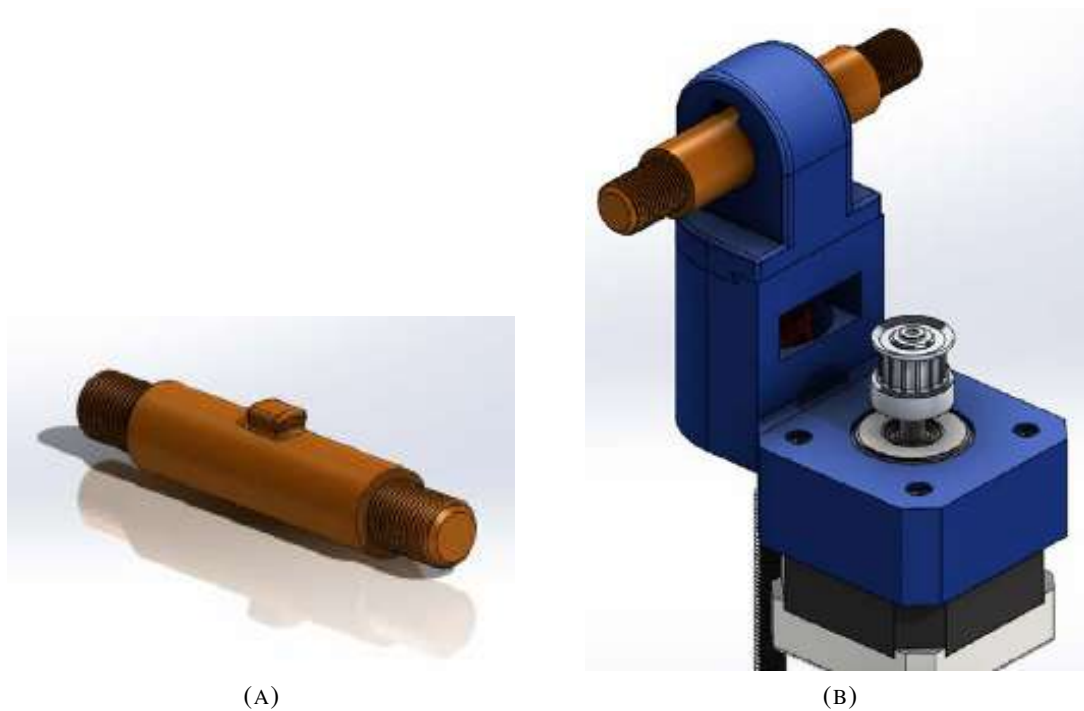


FIGURE 4.30: Motor Joint Axle with Motor Case.

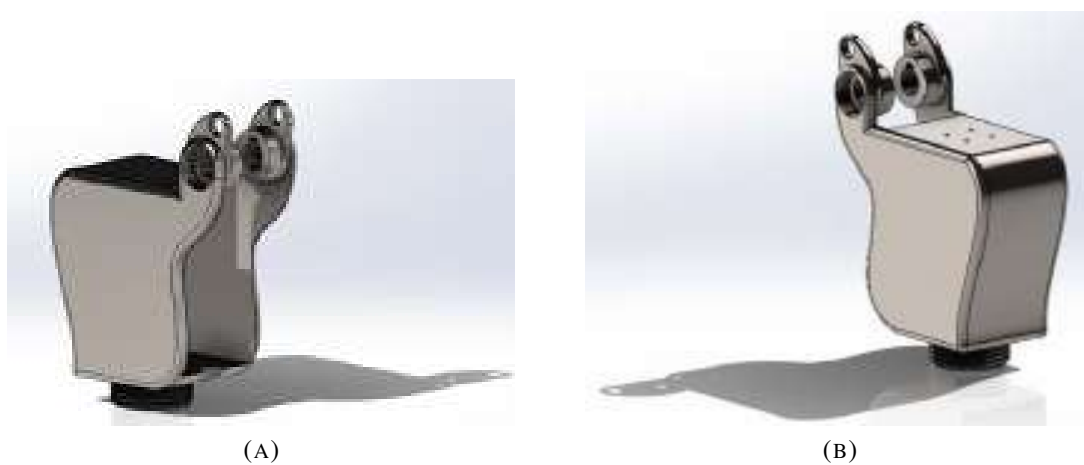


FIGURE 4.31: Designed Pylon.

threaded screw, and it has an inner hole channel for a space of the used electronics wires, as shown in figure 4.33.

- Foot

The foot is designed to have an ankle joint and has a joint a crank joint to link with the motor joint. The foot is designed to be a compact, simple, and sophisticated look. The foot is designed to hold a bearing

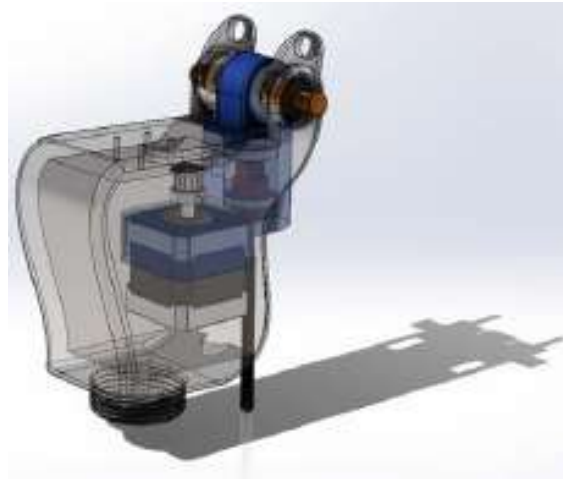
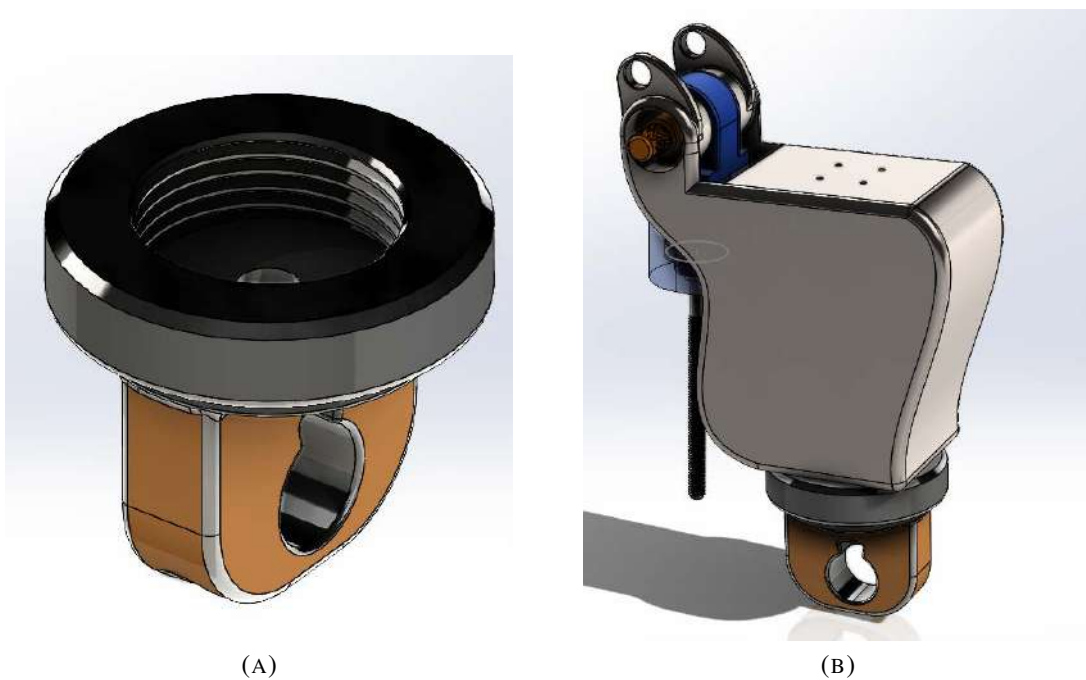


FIGURE 4.32: Assembled Pylon.



(A)

(B)

FIGURE 4.33: Connector.

inside and carry the foot during the ankle joint rotation, as shown in figure 4.34. The foot is designed to have a rectangular hole that holds the orientation sensor. The orientation sensor works as a feedback system that compares the rotation of the foot with the natural intact foot. The foot also includes a circular hole channel penetrate the whole foot to install the sensor wires, as shown in figure 4.35. The

foot has a fingers hold for cosmetics purposes.

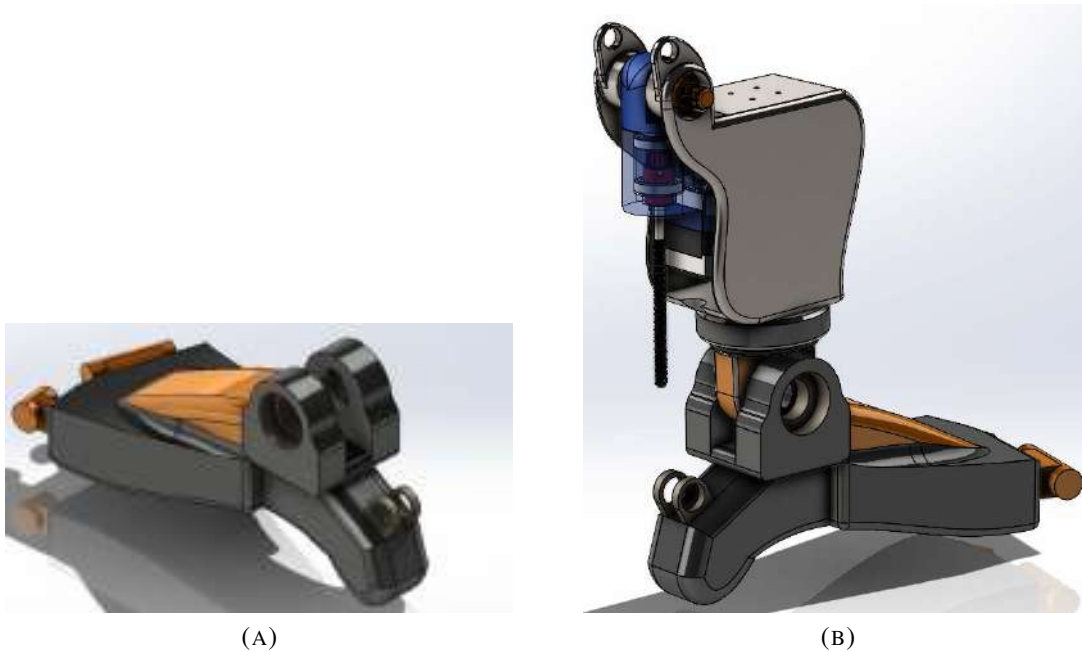


FIGURE 4.34: Prosthetic Foot.

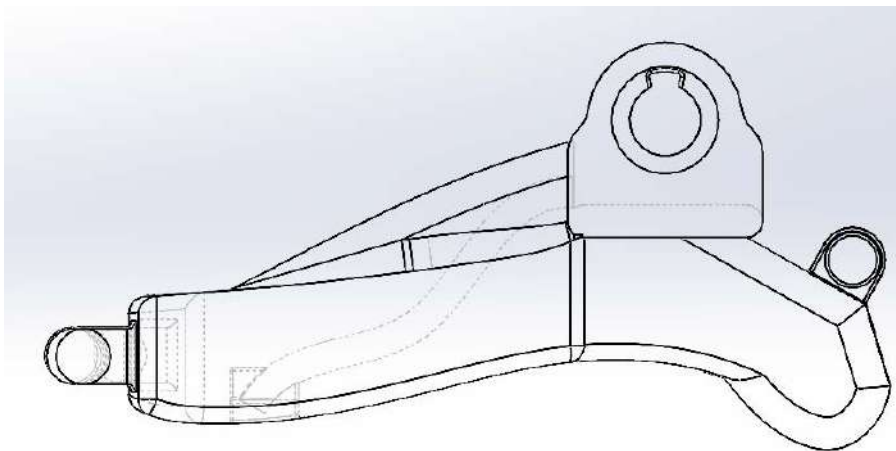


FIGURE 4.35: Prosthetic Foot.

- Ankle joint axle

The ankle joint axle is almost working, similar to the motor joint axle, and it works to link the connector to the foot and directly penetrate the bearings through the inner ring. The ankle joint axle is as shown in figure 4.36.

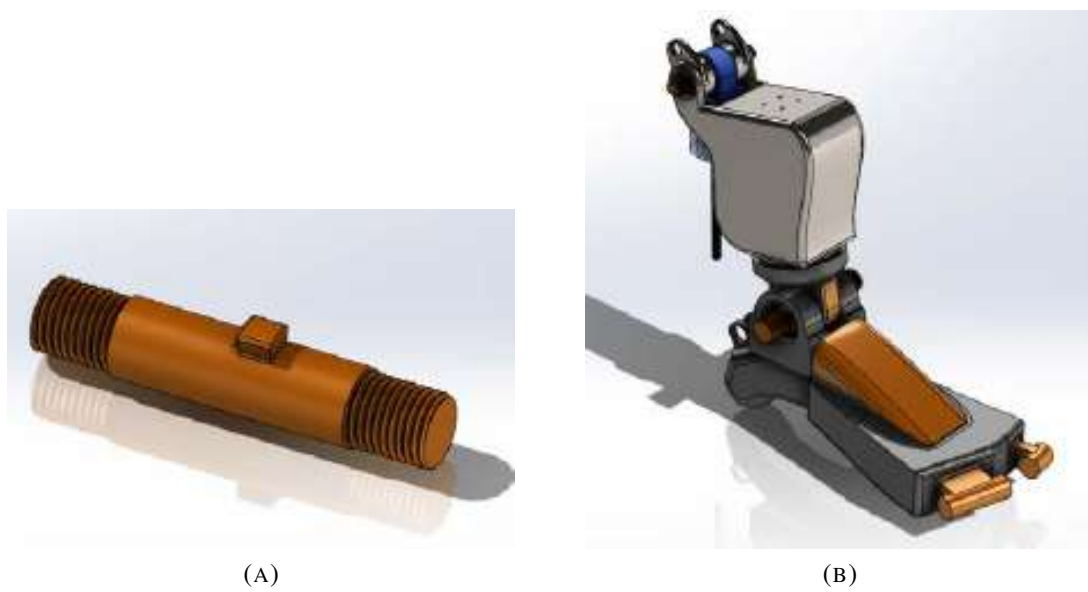


FIGURE 4.36: Ankle Joint Axle.

- Lead Screw Holder

the lead screw holder links the crank joint to the lead screw, as shown in figure 4.37.

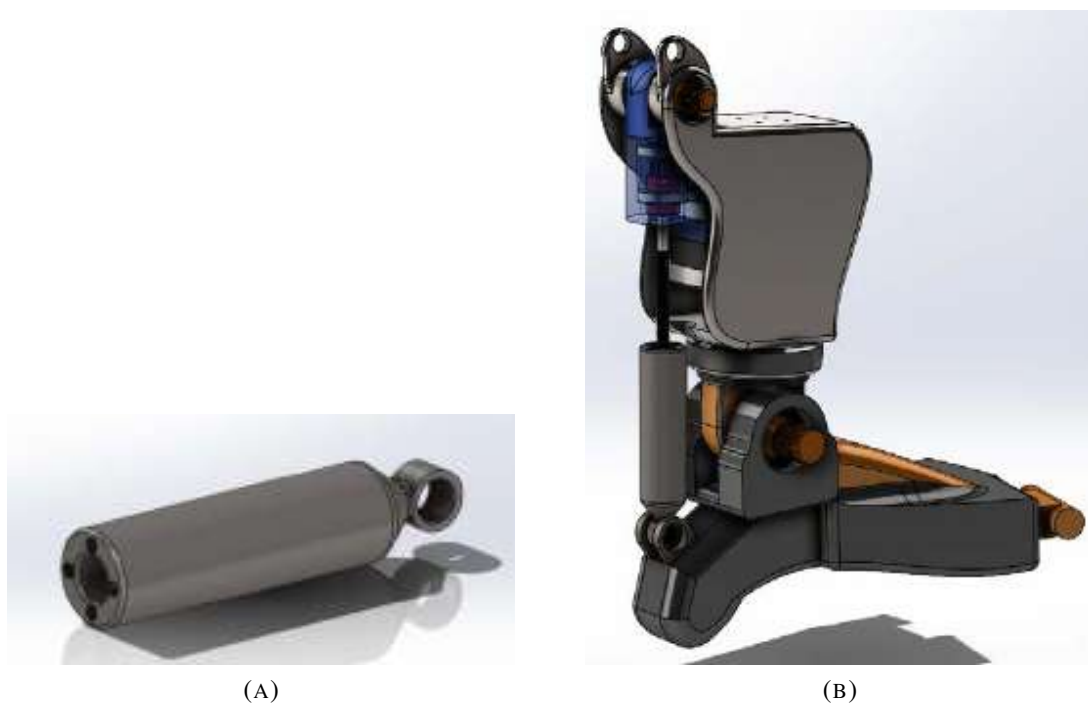


FIGURE 4.37: Lead Screw Holder.

- Bearing

SKF ball bearings are used to concentrate the motion of the joints into only a radial motion. Two SKF ball bearings were used: 6804 for the ankle joint and 6802 for the motor joint and hold the modified pulley, as shown in figures 4.38, and 4.39.

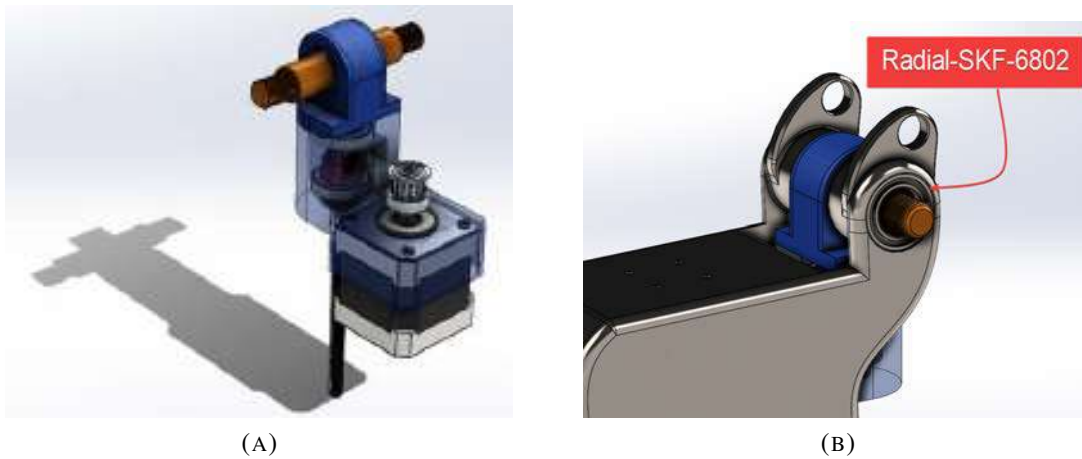


FIGURE 4.38: Ball Bearings.

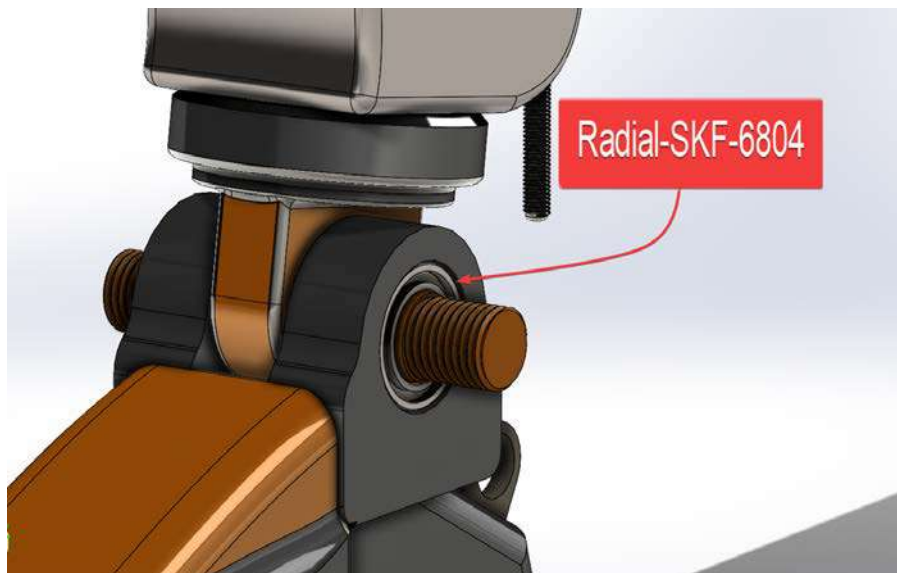


FIGURE 4.39: Ankle Joint Ball Bearing.

- Electronics Enclosure

The Electronics enclosure is linked with the pylon via bolt-nut. All electronics that operate the designed artificial limb, such as the

mainboard, battery, accelerometer ... etc, are placed inside the electronics enclosure, as shown in figure 4.40.

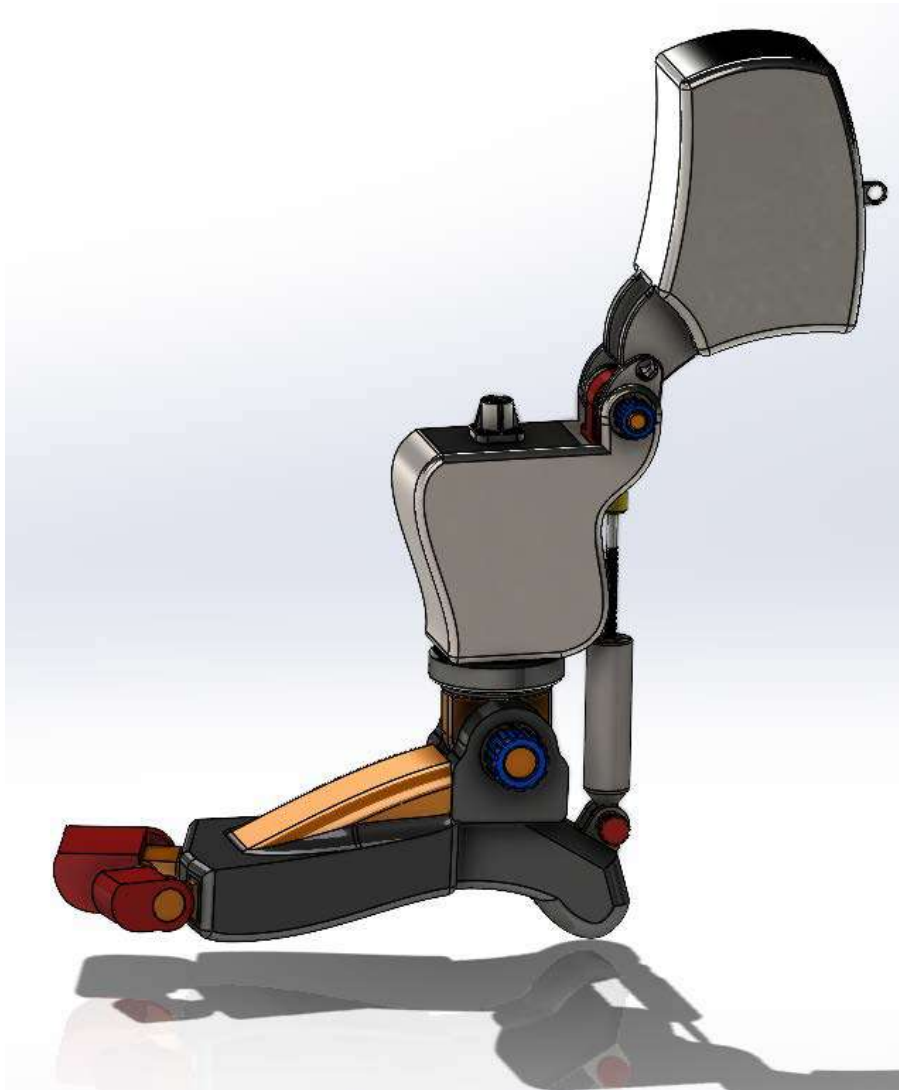
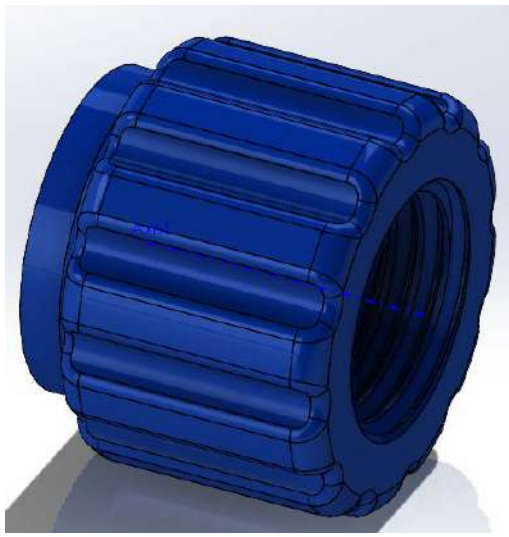


FIGURE 4.40: Electronics Enclosure.

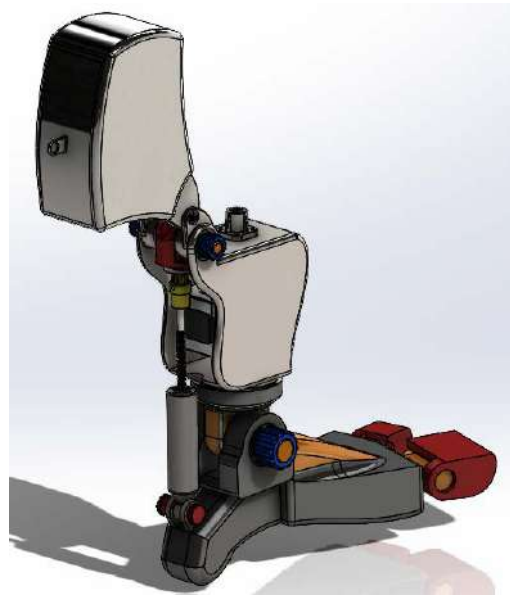
- Nuts

Two nuts are used to hold the ball bearings and keep joints compact. A nut with extrusion at one side is designed to keep pressure onto the bearings. The nuts are used in the ankle joint and motor joint, as shown in figure 4.41.

The final design modelled using Solidworks 2020 and as shown in figure 4.42. The Mechanism of ankle joint movement started from the



(A)



(B)

FIGURE 4.41: Nuts.

motor and moved to the lead screw using a belt. The lead screw moves the ankle joint using a crew and nut. The whole system device is of a single degree of freedom since it only depends on the ankle joint angle. The powered ankle-foot motion is as illustrated in video [72]. All parts with the complete model assembly can be downloaded from [73].

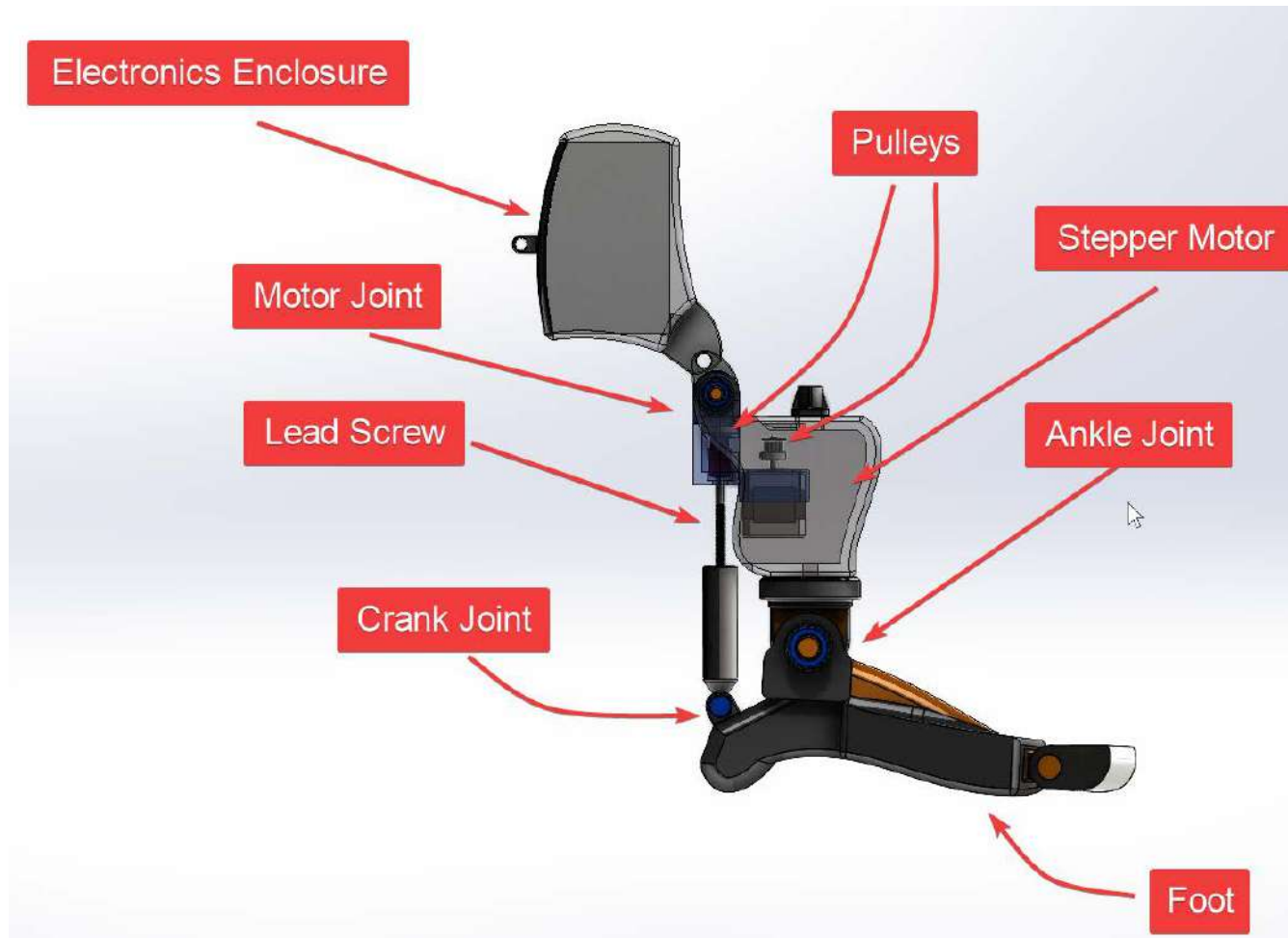


FIGURE 4.42: SolidWorks Design Model.

4.5 3D Printed Powered Ankle-Foot Prosthetic prototype

A powered ankle-foot prosthesis prototype is manufactured using a 3D printer, as shown in figures 4.43, 4.44, 4.45, and 4.46.

The prototype tests the generated statistical model using the regression and control systems. The designed model by SolidWorks (4.4) was printed using ABS 3D printer filament. ABS filament endures higher pressure, heat, and strength than PLA filament. ABS is more flexible than PLA filament and has higher temperature resistance.

The manufactured prototype was used to test any theoretical work and

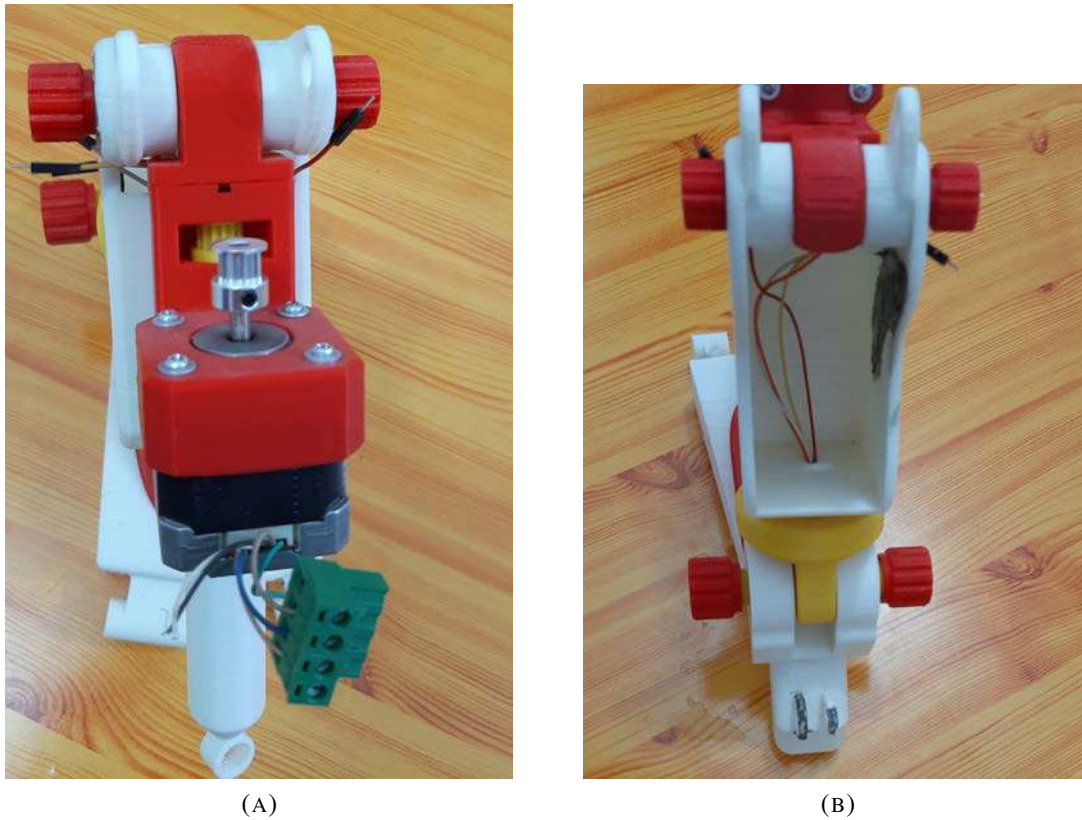


FIGURE 4.43: 3D Printed Powered Ankle-Foot Prosthesis.

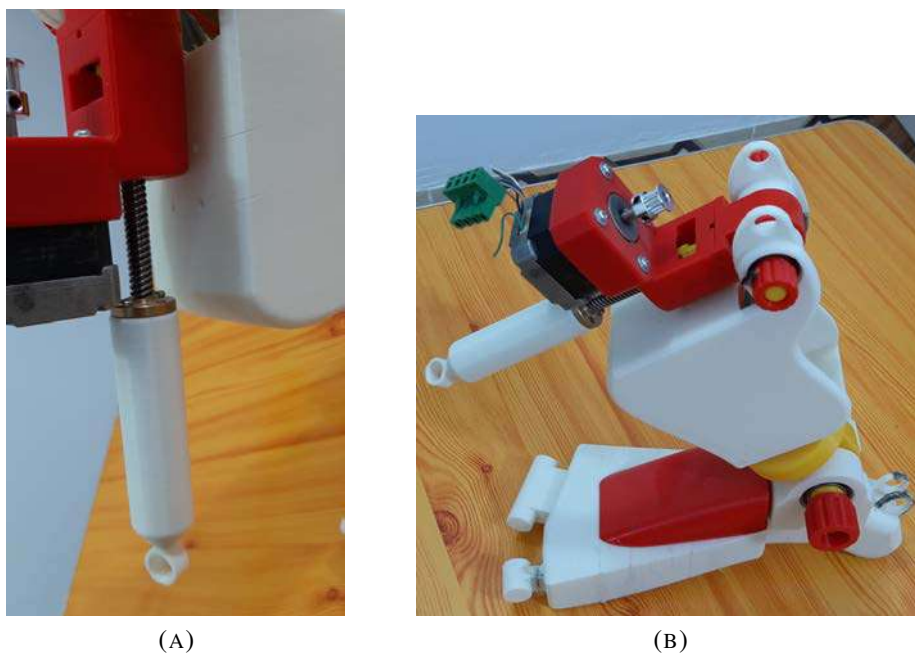


FIGURE 4.44: 3D Printed Powered Ankle-Foot Prosthesis.

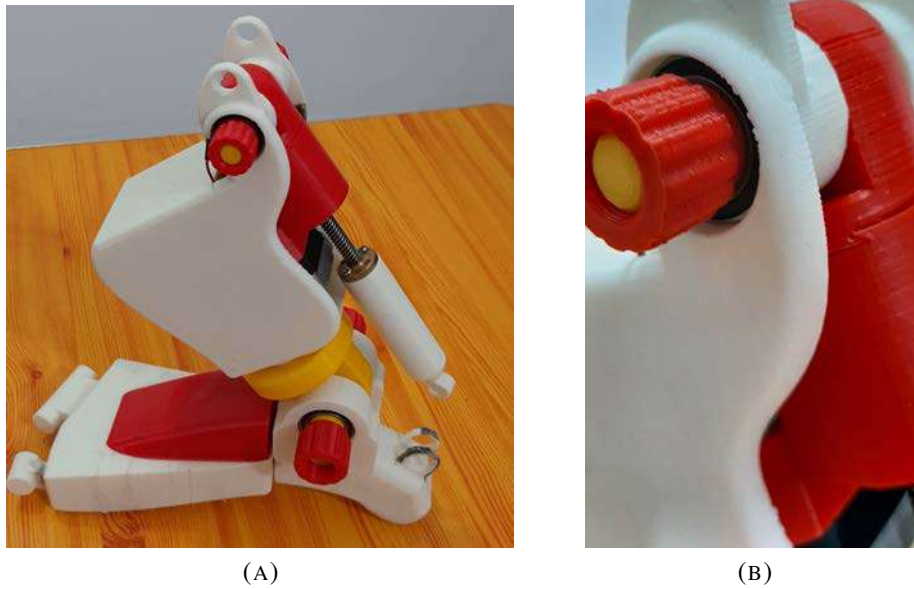


FIGURE 4.45: 3D Printed Powered Ankle-Foot Prosthetic.

measure how-2el deviated the prosthesis's behavior for both intact amputated limbs.

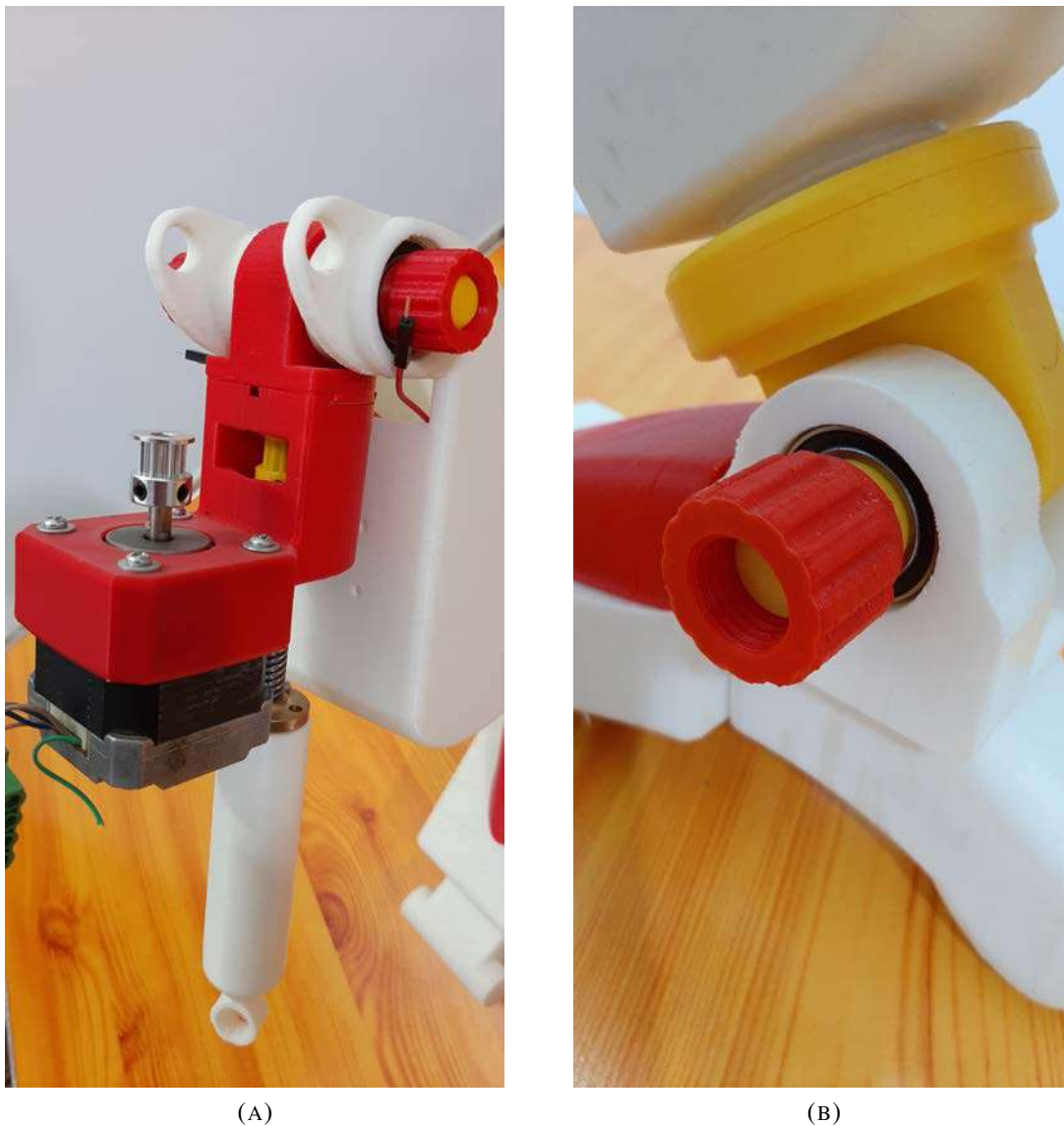


FIGURE 4.46: 3D Printed Powered Ankle-Foot Prosthesis.

4.6 Regression Experiments

Regression experiments aim to build a statistical model that can predict the response of the ankle joint relying on the leg muscles contraction. Four muscle activities (Tibialis Anterior, Medial Gastrocnimis, Lateral Gastrocnimies, and Soleus) were recorded with the measurement of the ankle angle explained in section 4.2.3.2. Before electrode installation, the skin should be cleaned, and leg hair should be shaved for all subjects in

order to prevent any separation that may cause any distortion in the muscles' activity signal. The electrodes were installed along the longitudinal centre of the muscle and at the belly of the muscle with 2 cm between the two electrodes to get the highest muscle activity signal amplitude [69]. The location of the electrodes can drastically affect the signal shape, as shown in figure 4.47. Appendix C.1 is a reference map for the placement of the electrodes for the major muscles. A ground electrode is used to normalize the signal acquisition from each muscle, the ground electrode placed on the knee bone in these experiments. The goal of using two electrodes is to compare the activity between each electrode and remove the noise.

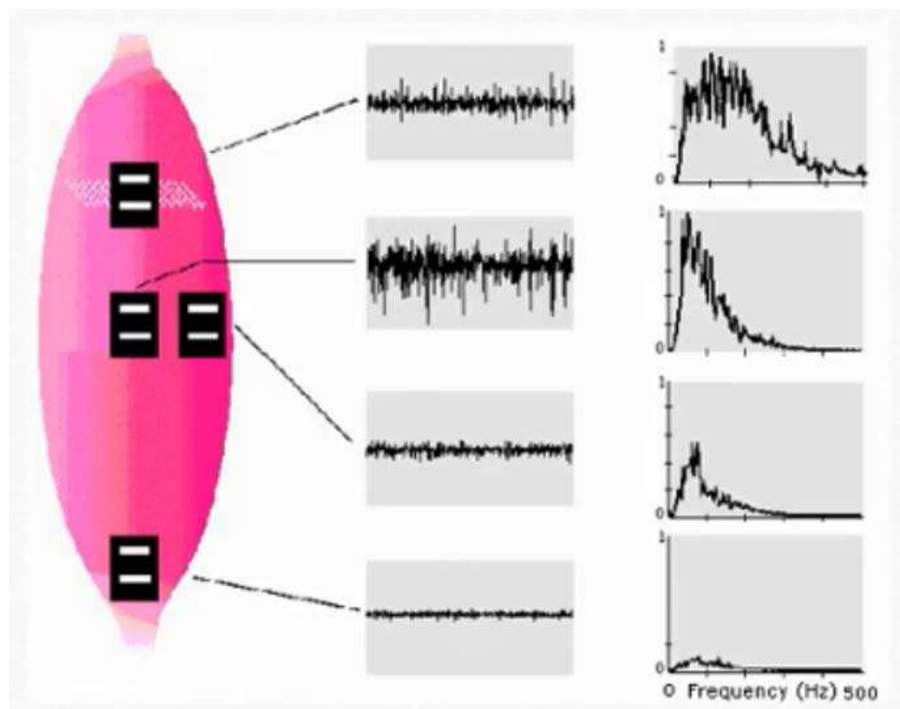


FIGURE 4.47: EMG Signal Shape Variation Due to Electrodes Location [69].

All electronics were mounted on the thigh, and wires are used to connect the electrodes to OpenBCI and to connect the two IMUs to the Nodemcu-32s and as shown in figure 4.48. The data is transmitted wirelessly either by

TABLE 4.2: Subjects Aspects

Subject	Gender	Age	Height(cm)	Weight(kg)	Origin	Date
1	Male	25	187	87	Iraqi	2021/4/17
2	Male	22	178	65	Iraqi	2021/4/17
3	Male	20	180	75	Iraqi	2021/4/17
4	Male	20	190	90	Iraqi	2021/4/17
5	Male	17	183	86	Iraqi	2021/4/17
6	Male	15	178	90	Iraqi	2021/4/17

Bluetooth in the case of OpenBCI or across wifi using node MCU-32s. As explained in section 4.2.3.2, the author developed a c# based application to visualize the ankle angle. The c# application is further developed to visualize and save the muscles contraction and the ankle angle, as shown in figure 4.49. The application can display and store the ankle angle and the data transmitted from OpenBCI for four channels. The full version of the application with its source code can be downloaded from **GitHub** [67]. The experiment's recorded data is stored in two .csv files, one for the muscle contractions and the ankle angle. Figures 4.50 and 4.51 show a sample of the two saved experiments files.

A sampling frequency rate mist-matching problem between the two sensors emerged. OpenBCI sends data at a rate of 200 Hz, whereas Node Mcu-32s sends data at a rate of 80-50 Hz. Both data frames should have the same size to fit it and build the required statistical model(regressor). As shown in figures 4.50 and 4.51, both data have a timestamp in epoch/UNIX(Measure the seconds for 1st January 1970, with microseconds as a decimal). OpenBCI data were downsampled using a custom python code, where the timestamp variable was used as crucial to match both data. Data matching and cutting the redundant and saving the matched data into a new .CSV file was done using code shown in appendix A.9. Six subjects were involved in the experiment and had the aspect shown in table 4.1.

In this experiment, the subjects were ordered to perform a body rise and

fall over the ankle joint plantarflexion and record the four aforementioned muscles with the ankle angle as shown in figures 4.54, 4.52, and 4.53. The entire experiment was recorded using a smartphone camera, with the record of the developed application screen for observing the start and the end of each experiment.

a published video [70] on YouTube by author, shows the entire experiment. The EMG data are downsampled to be matching the orientation data using the timestamp variable. The start and the end time of each experiment is recorded manually. Therefore all data before and after the experiment's movement are trimmed using a custom-made python code (Appendix A.10). The new matched and trimmed data that includes all four muscle activities with shank, foot, and ankle angle, ready to perform any digital or statistical modeling process, is exported into the .csv data frame. The newly created data frame can be downloaded from [71]. Figure 4.55 shows a sample of the final processed dataset. The ankle joint Ankle angle calculation from shank and foot rotation records were calculated according to the formulation in equation 3.8.

The whole data were visualized with respect to time, and the EMG data are plotted with respect to the ankle angle to observe a possible pattern. A simple linear regression was applied with the raw data, and the coefficient of determination(r^2) was used as a metric to evaluate the regressor performance. A feature scaling was performed on the EMG data to prevent any bias in any dimension; the python code for the above work(data plotting, linear regression and feature scaling) (Appendix A.11).

The EMG for the four-channel was plotted in the frequency domain using Fast Fourier Transformation. To find the signal's spectrum and the noise frequency range, the code for plotting the EMG signals in the frequency domain is shown in appendix A.12. Linear and Polynomial

regressions are used after applying a Low-pass filter of a specific cut-off frequency to get the best regression performance. The lowpass filter python's code (Appendix A.13).

Correlation Matrix with Heat-map and Information gain - mutual information methods for feature selection were used to get the most inflecting muscle on the ankle joint plantarflexion movement. Python code for applies the aforementioned feature above selection methods (Appendix A.14).

The proposed method 3.11 was used to optimize the ankle joint regression. The Gaussian distribution's mean(μ) is used to get the angle sample point representing multiple muscle contractions. The variance(σ^2) is used to estimate the uncertainty of the used mean sample point value. Python code for getting the mean of a set of muscle contractions at a certain angle (Appendix A.15). The new optimized data was modelled using linear regression, polynomial regression, K-nearest neighbour regression with a low-pass filter and Gaussian smoothing filter to get the best regression accuracy(r^2). Python code for the upper operations (Appendix A.16).



FIGURE 4.48: EMG Electrodes Placement and Electronics Enclosure Position.

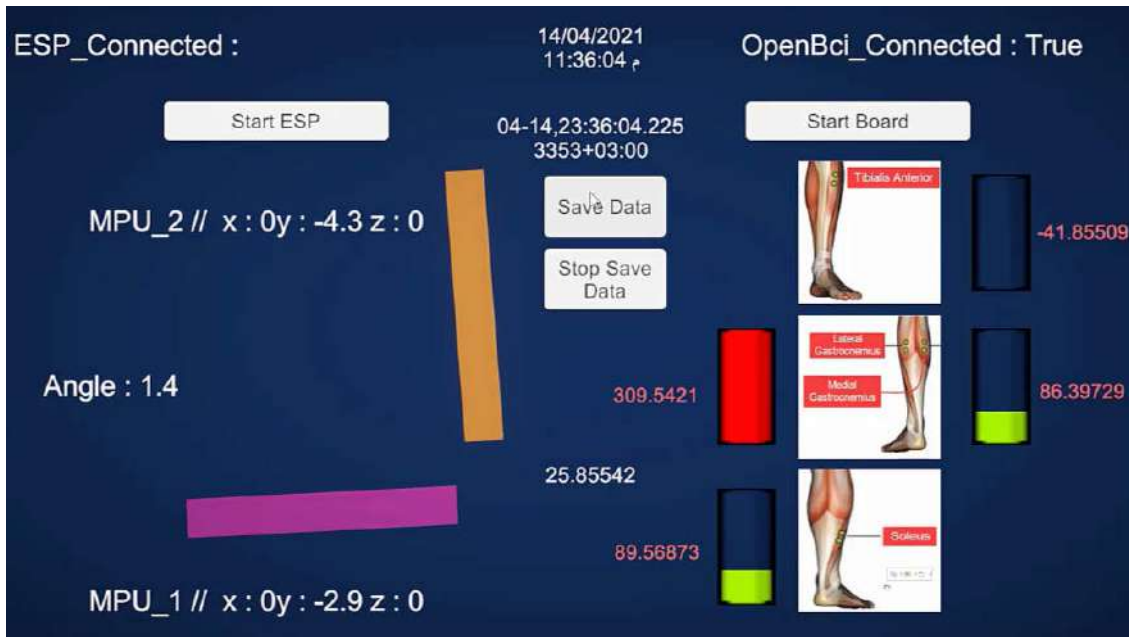


FIGURE 4.49: Developed Application for Experiments Management.

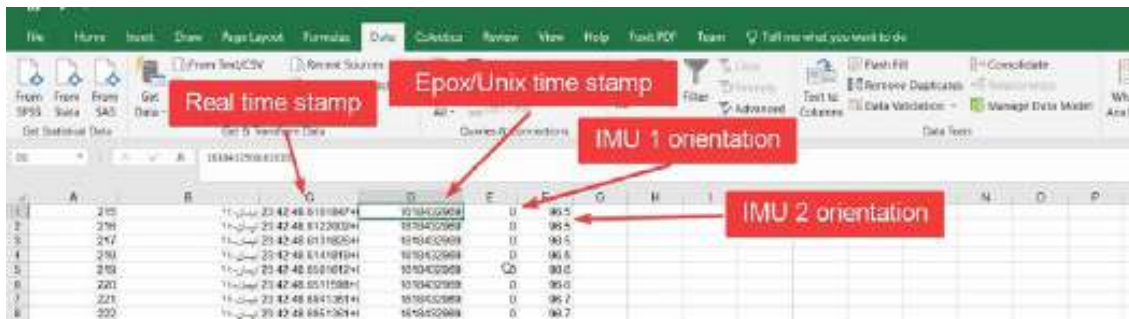


FIGURE 4.50: Ankle Joint Saved File Sample.

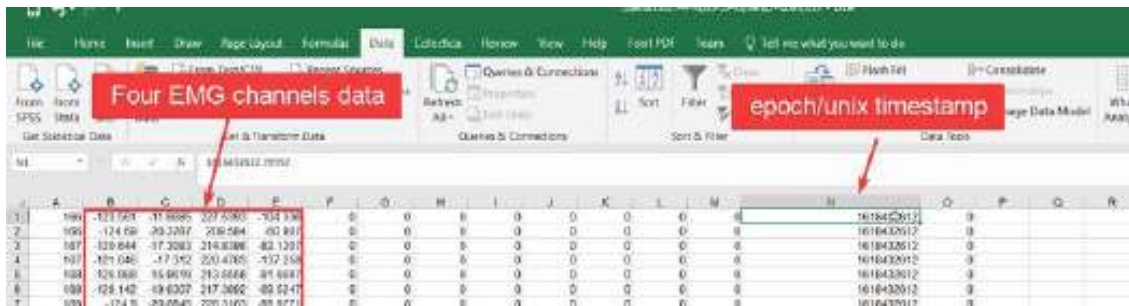


FIGURE 4.51: OpenBCI EMG Saved File Sample.

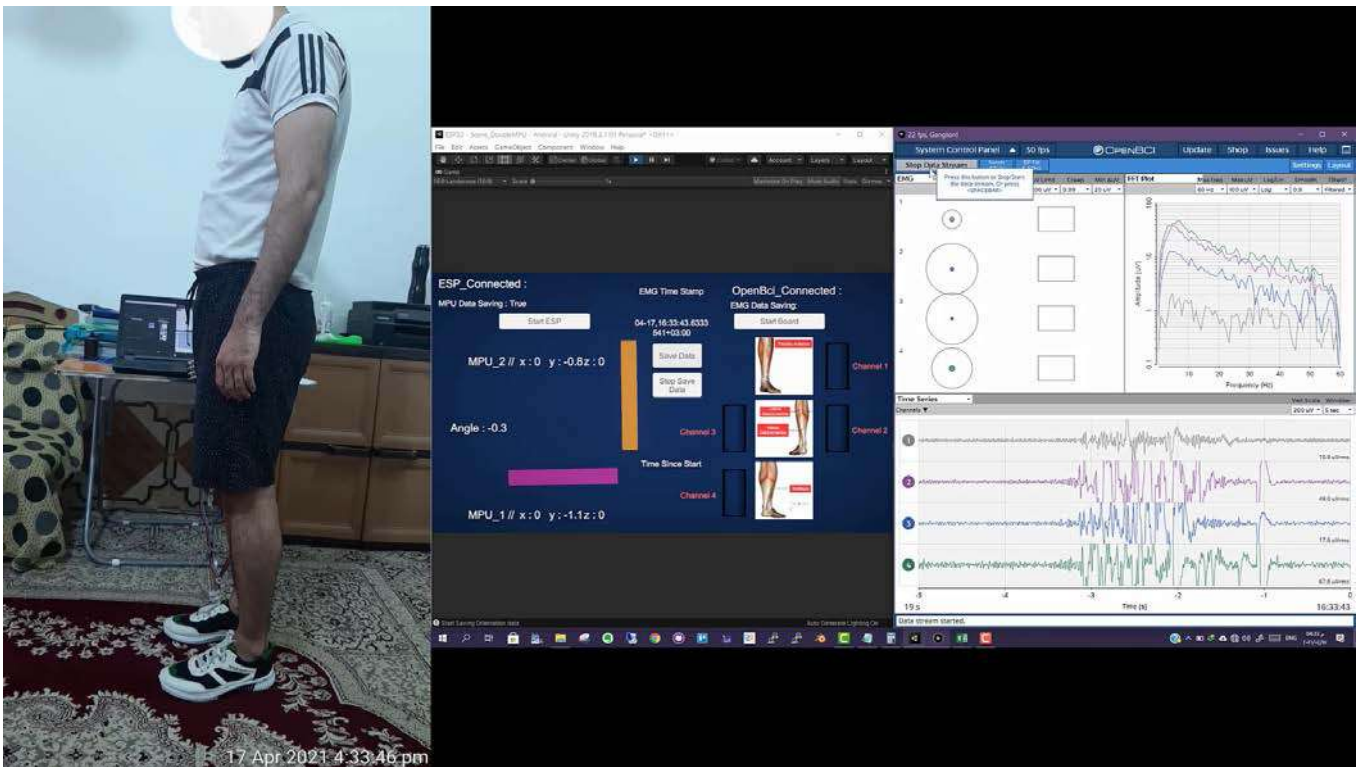


FIGURE 4.52: Standing.

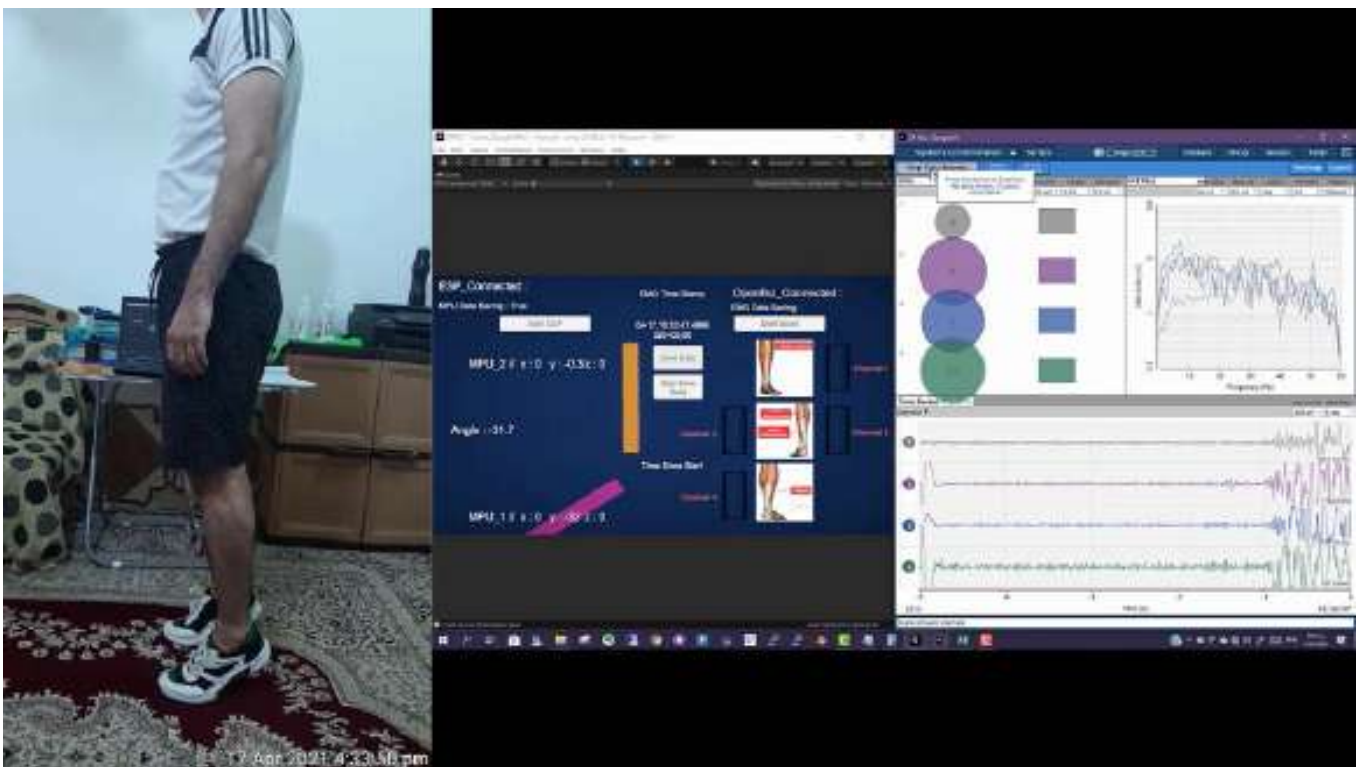


FIGURE 4.53: Rising.



FIGURE 4.54: Electrodes Placement.

	A	B	C	D	E	F	Formula Bar	H	I	J	K	L	M	N
		emg timestamp	esp timestamp	ch_1	ch_2	ch_3	ch_4	foot angle	shank angle	shank 90	ankle	ankle	index	
1	0	161866413.020000	161866413.0154000	-445.4407959	-142.8847504	-08.19481659	-69.33213806	-0.4	88.8	-1.2	0.8	0		
2	1	161866413.0510000	161866413.0483500	-425.9110413	-115.0524139	-71.51062775	-30.83360481	-0.4	88.8	-1.2	0.8	1		
3	1	161866413.0510000	161866413.0483500	-440.6761475	-127.3698818	-90.79354858	-54.50717163	-0.4	88.8	-1.2	0.8	2		
4	2	161866413.0800000	161866413.0833800	-414.1527711	-133.4228958	-87.5473175	-77.88902283	-0.4	88.8	-1.2	0.8	3		
5	3	161866413.0800000	161866413.0843400	-405.8876038	-139.6497345	-92.73455811	-93.81156464	-0.4	88.8	-1.2	0.8	4		
6	4	161866413.0800000	161866413.1203100	-404.7506714	-116.6717911	-83.83035278	-96.46137238	-0.3	88.7	-1.3	1	5		
7	5	161866413.1270000	161866413.1223200	-389.5152659	-124.8920388	-75.93119049	-84.43011475	-0.3	88.7	-1.3	1	6		
8	6	161866413.1560000	161866413.1483300	-397.8842163	-125.9054321	-81.11489269	-146.294317	-0.3	88.7	-1.3	1	7		
9	7	161866413.1800000	161866413.1802900	-386.3279114	-131.306015	-85.088591	-110.2578583	-0.3	88.7	-1.3	1	8		
10	8	161866413.1800000	161866413.1812900	-375.4507778	-137.2001038	-88.69836426	-119.9441986	-0.3	88.7	-1.3	1	9		
11	9	161866413.1800000	161866413.2152600	-399.7691345	-140.0798187	-91.81828308	-121.5972366	-0.4	88.7	-1.3	0.9	10		
12	10	161866413.2160000	161866413.2452500	-366.973938	-115.1272125	-80.74817657	-103.140831	-0.4	88.7	-1.3	0.9	11		
13	11	161866413.2460000	161866413.2462600	-357.9533081	-106.9031677	-87.9165802	-112.0230942	-0.4	88.7	-1.3	0.9	12		
14	12	161866413.2460000	161866413.2802300	-357.0183411	-129.1219177	-83.6316452	-142.1816408	-0.4	88.6	-1.4	1	13		
15	13	161866413.2770000	161866413.2622300	-372.3781128	-151.2882996	-100.3863983	-150.0204773	-0.4	88.6	-1.4	1	14		
16	14	161866413.3210000	161866413.3202100	-358.5255127	-114.4016724	-83.53814697	-100.6904144	-0.4	88.6	-1.4	1	15		
17	15	161866413.3360000	161866413.3302000	-383.1938782	-124.3490917	-83.39880916	-127.8653107	-0.4	88.6	-1.4	1	16		
18	16	161866413.3500000	161866413.3491900	-369.7901001	-119.9996443	-86.57868195	-112.9393992	-0.4	88.6	-1.4	1	17		
19	17	161866413.3800000	161866413.3801600	-340.6113892	-128.2095342	-45.7015623	-170.171051	-0.4	88.6	-1.4	1	18		
20	18	161866413.3800000	161866413.3811700	-377.0305481	-103.5110779	-104.6480898	-122.8501053	-0.4	88.6	-1.4	1	19		
21	19	161866413.4110000	161866413.4141500	-371.9218445	-99.01940259	-83.6727829	-120.5687037	-0.4	88.5	-1.5	1.1	20		
22	20	161866413.4560000	161866413.4491300	-353.5963135	-124.4844437	-92.01275635	-156.6887207	-0.4	88.5	-1.5	1.1	21		
23	21	161866413.4560000	161866413.4521300	-333.1802053	-81.43070984	-84.49655969	-139.7806244	-0.4	88.5	-1.5	1.1	22		

FIGURE 4.55: Down-sampled Raw Data.

Chapter 5

Results and Discussions

5.1 Introduction

This chapter will present and discuss the theoretical and experimental results. The proposed mathematical model of the powered ankle-foot prosthesis will be discussed with its implications here. The following sections will discuss human Gait cycle classification based on EMG signals using multiple classification algorithms and multiple filtering techniques results. Features selection of Human Gait cycle classification based on EMG signals to get the most effective muscle in the classification process, and its methodology is exposed. Ankle joint plantarflexion experiment data and plots are also illustrated in this chapter. The ankle joint regression with multiple algorithms and multiple filtering techniques that could improve the regression performance results is discussed in this chapter. The best parameter for each filter approach result is obtained based on supervised machine learning. Ankle joint regression feature selection using linear correlation results is exposed. The final section shows the results of the optimization of ankle joint regression using Gaussian distribution and its ability to improve the performance of the regression process.

5.2 Human Gait Cycle Classification Results

In this section, the results of multiple classification techniques (described in 3.4) with multiple filtering techniques (explained in 3.10.1) are used to classify a human gait cycle (defined in 1.2.1) into stance and swing phases based on supervised machine learning.

Human Gait cycle classification is essential in pre-disease diagnosis, operating a lower limb prosthesis, and a better understanding of the biomechanics of muscles.

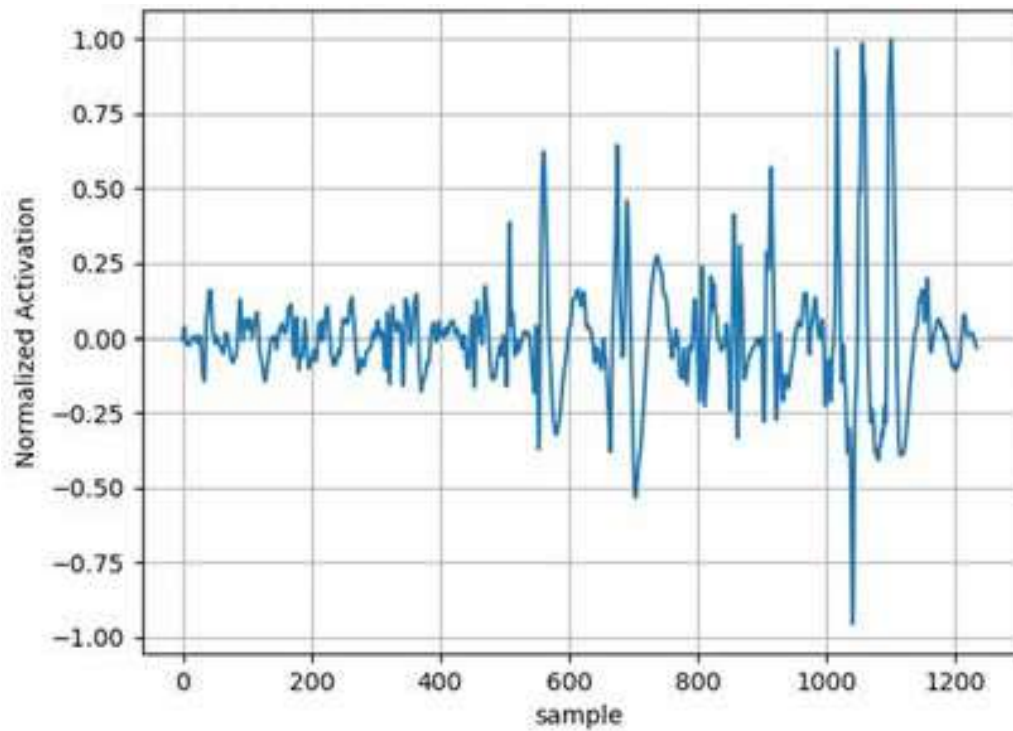
As explained in 3.4, classification is the process of labelling an unsorted dataset based on pre-labelled data through a process called training. As discussed in 3.4, classification deals with a discrete dataset, unlike regression, which produces continuous predictions of the dataset. The performance between five classification algorithms, namely: Support Vector Machine (SVM), Logistic Regression (LR), K-Nearest Neighbors (KNN), Decision Tree (DT), and Random Forest (RF), has measured (i.e., performance) to get the best algorithm for that deal with such case (i.e., human gait cycle classification).

Figure 5.1 shows the Soleus muscle activity for both stance and swing phases, respectively.

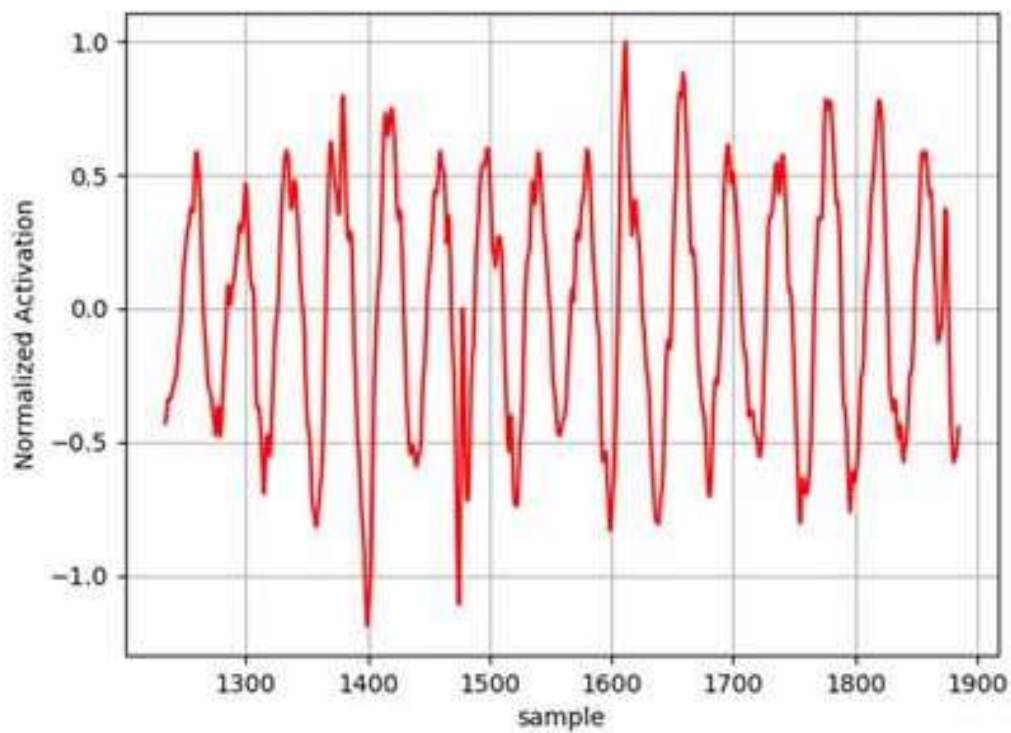
Binary classification is applied to classify signals shown in figure 5.1 in the context of a signal having both stance and swing phases, as shown in equation 5.1.

$$y_{phase}(t) = \begin{cases} 1 & x(t) \text{ in stance phase} \\ 0 & x(t) \text{ in swing phase} \end{cases} \quad (5.1)$$

where $x(t)$ is the activity of the selected muscle, thus $y_{phase}(y) \in R^n$, where n is the number of the independent variables (i.e. recorded muscles).



(A)



(B)

FIGURE 5.1: Stance phase (A) and Swing phase (B) for a single cycle.

In this work, seven lower limb muscles are used in the classification process; the recorded muscles are the Soleus muscle, Tibialis anterior muscle, Gastrocnemius Lateralis muscle, and Vastus Lateralis Rectus Femoris muscle, Biceps Femoris muscle, and Gluteus Maximus muscle.

All of the datasets utilized in this work are obtained from the **HuMoD** database [74] is an open database dedicated to analysing and recording human motion dynamics, which provides well-organized and documented datasets mainly about the lower limbs. The database contains three-dimensional motion tracking data collected using sophisticated cameras and markers attached to a participant jogging on a treadmill with a force plate beneath it recording the ground response force at 1,000 Hz. It also captures all lower limb contact events, recording 1 if the limb is in touch with the ground and 0 otherwise, providing an effective technique of labelling the data for classification purposes. Seven electromyography sensors are also placed on each leg muscle to record muscle activity. Participants conduct needed activities such as straight walking at various speeds, straight running, sideways walking, and kicking a soft football. The raw and filtered EMG data are provided, with the latter being produced using a root Mean square filter with a window size of 100. The Soleus muscle, Tibialis anterior muscle, Gastrocnemius Lateralis muscle, Vastus Lateralis muscle, Rectus Femoris muscle, Biceps Femoris muscle, and Gluteus Maximus muscle all provide EMG data. The data in this study is based on a female participant walking straight at 1.0 m/s (age 27, 161 cm tall, and 57.3 kg in mass). Mat format is used to store the data. One issue with this data set is that the EMG recording frequency (frame rate) is 2,000 Hz.

In contrast, the force plate operates at 1,000 Hz, necessitating data pre-processing to ensure that both sets of observations have similar reading

rates. This paper's pre-processing and subsequent categorization are done with Python. The information was then supplied not these zero vectors with even indices (Python starts with 0 indexing), forcing each in-between empty odd cell to its primary cell.

Electromyographical signals are mostly accompanied by noise, which leads to poor classification and incorrect prosthesis response. Median(x) and root mean square filters(x) filters were used to refine the EMG signals accompanied with noise to get a higher classification performance. A 60 gait cycle data of HuMoD databases is fed into the Sciket learning library(includes most machine learning algorithms). The classification and programming operations are done using Spyder IDE with a Core i7 processor of 2.40 GH CPU speed and 32GB RAM laptop using windows ten as an operating system.

The confusion matrix is used to measure the performance of the classification algorithms. The confusion matrix is based on the dataset's true and false predicted samples. The classification performance is calculated for the raw data set and after applying median and root mean square filters. The performance is shown in figure 5.2.

Figure 5.2 shows the classification performance for the Support Vector Machine, Logistic Regression, K-Nearest Neighbors(KNN), Decision Tree, and Random Forest classification algorithm. Figure 5.2 shows that the performance of all algorithms had increased with the application of filters. Median filters show the upper hand over the RMS filter since it removes outliers that may produce due to noise. K-Nearest Neighbors(KNN) performs better than other algorithms despite the used filter; this interprets that each phase's data has separated and clustered together.

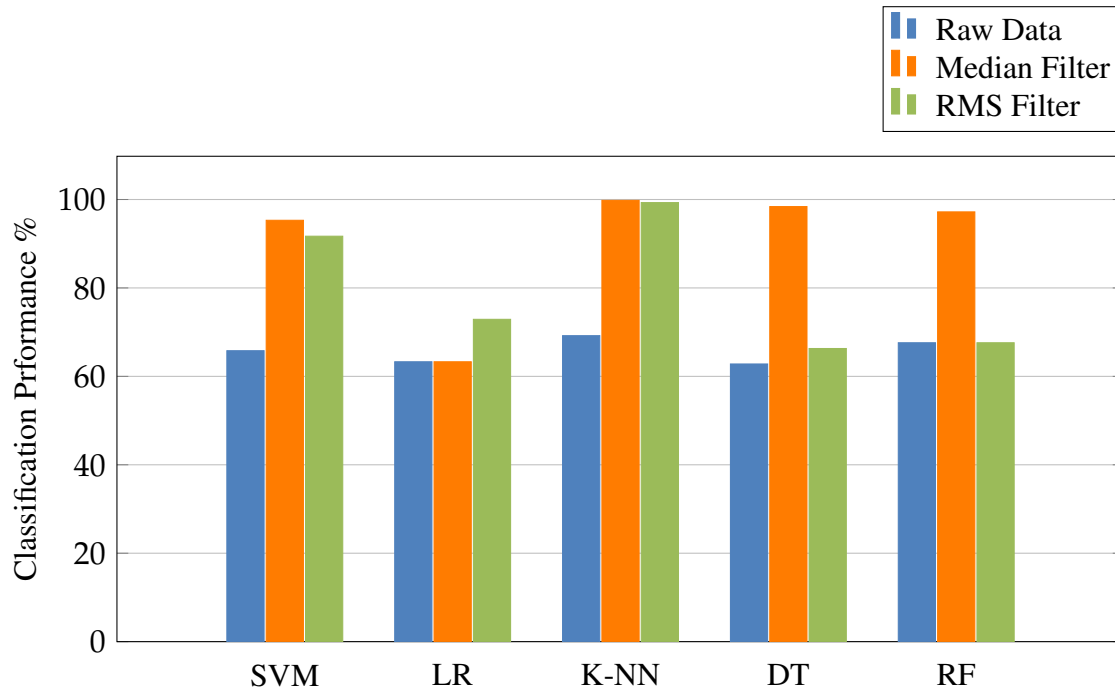


FIGURE 5.2: Classification technique performance for Raw EMG, EMG with Median filter, and EMG with RMS filter.

5.3 Human Gait Cycle Classification Features Selection

As discussed in section 3.7, feature selection selects the most influential variable on the build classification model. In this work, feature selection refers to the best muscle that can be used to distinguish between the stance and swing phases. Machine learning literature has several approaches to obtain this high influential variable. In this work, a simple graphical method is used in the feature selection topic. The used approach (graphical method) is made by plotting both the stance and swing phases of the seven recorded muscles and giving a different color for each phase; the muscle that has the highest activity separation between the classes is considered as the best muscle that can be used to distinguish between the gait cycle phases. Figure 5.4 shows the muscle activity for each phase.

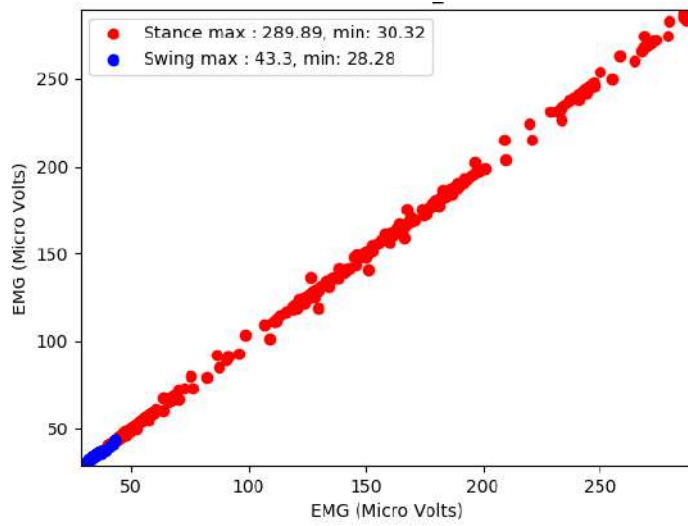
Figure 5.4 shows muscle contraction and the maximum to the minimum

TABLE 5.1: Muscles Abbreviations

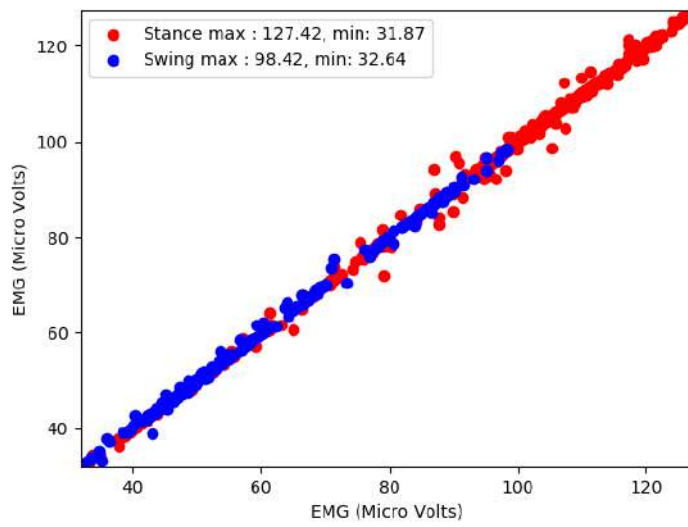
Abbreviation	Full muscle name
SOL	Soleus muscle
TIA	Tibialis Anterior muscle
GLS	Gastrocnemius Lateralis muscle
VSL	Vastus Lateralis muscle
RCF	Rectus Femoris muscle
BCF	Biceps Femoris muscle
GLX	Gluteus Maximus muscle

activities for both stance and swing phases, the abbreviations represent the muscles under consideration and shown in table 5.1.

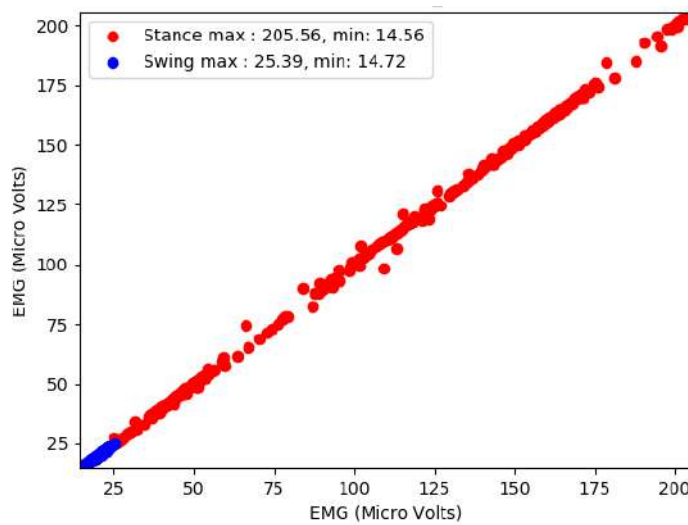
The **Rectus Femoris muscles** show the highest separation between the two classes. The Rectus Femoris muscle mainly affects the knee joint extension, and it works in swing phases while rising the whole limb. Thus, it has a higher swing phase activity, making it the best variable to discriminate the gait cycle phases.



(A)

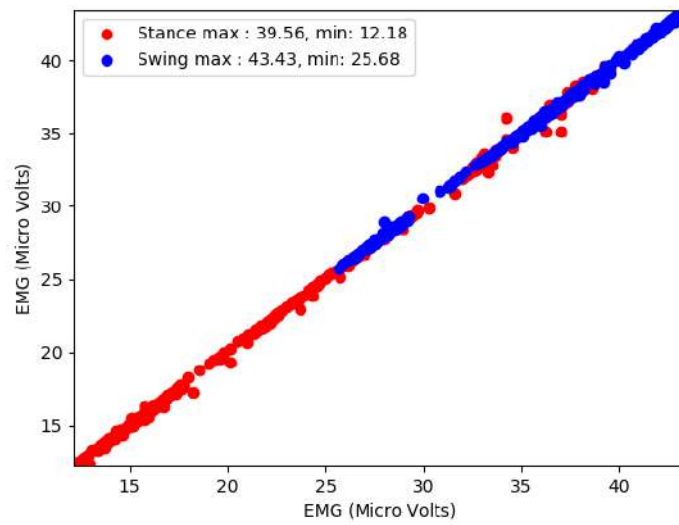


(B)

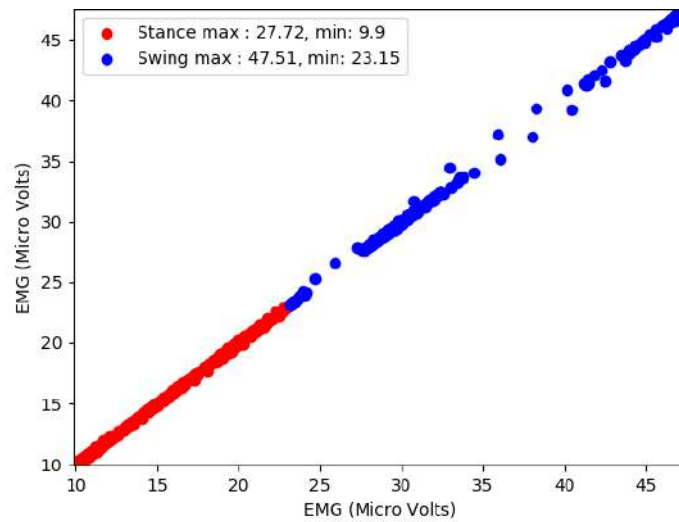


(C)

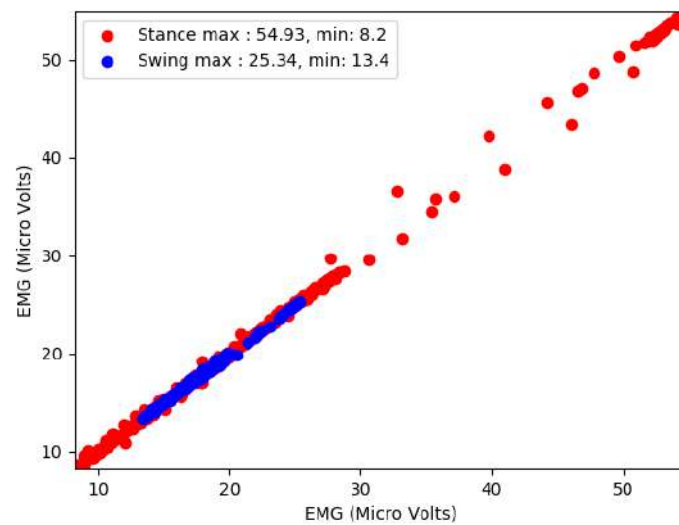
FIGURE 5.3: Recorded Lower Limb Muscles Activity for Both Stance and Swing Phases.



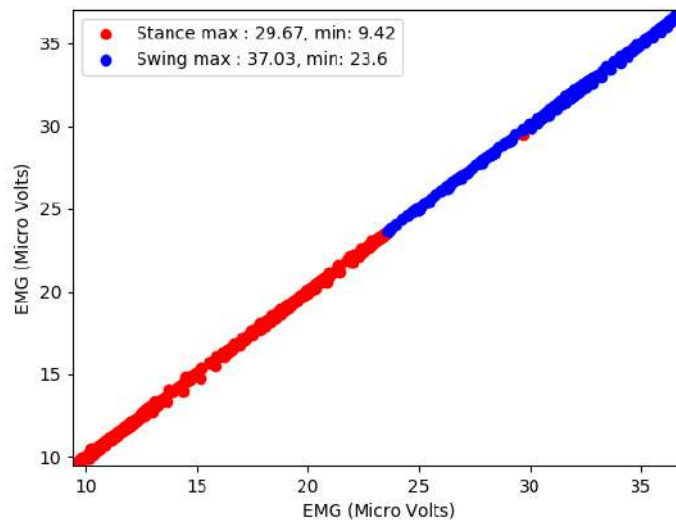
(A)



(B)



(C)



(A)

FIGURE 5.4: Recorded Lower Limb Muscles Activity for Both Stance and Swing Phases.

5.4 Ankle Joint Regression Based on Electromyographical Signals

Ankle joint regression is considered the essence of this work. As discussed in section 3.5, the ankle joint regression aims to find the optimal model to predict the response of the powered ankle-foot prosthesis. The electromyographical signals majorly drive the imaged powered ankle-foot prosthesis, and it operates mainly based on the regression model built in this section.

Figure 5.5 shows a plot of the whole data acquisition in the regression experiment (4.6). The x-axis for all sub-figures is the time in seconds. The y-axis in figure 5.5 represents the ankle joint orientation, Tibialis Anterior muscle, Medial Gastrocnimius muscle, Lateral Gastrocnimius muscle, and Solus muscle activities for eleven plantarflexion movements shown in the first segment of video [70].

The negative ankle joint angle shown in figure 5.5 represents the plantarflexion movement. In contrast, the negative signal sample points in muscle contraction do not have meaning, and a reverse installation of electrodes causes them. The negative sign of muscles contractions can be overcome using a root mean square rectification by applying equation 3.87 for all sample points, as shown in figure 5.6.

Figure 5.6 shows, the root mean square filter tends to smooth signals. The linear regression, polynomial regression, KNN regression performances for rectified EMG signals by RMS is as shown in figure 5.7.

The regression performance for all regression algorithms is still low; even RMS rectifier is applied; this means that the noise is still affecting the proper building of the statistical model, note that the performance is measured based on the r-square score(explained in 3.6.4). The best performance for KNN

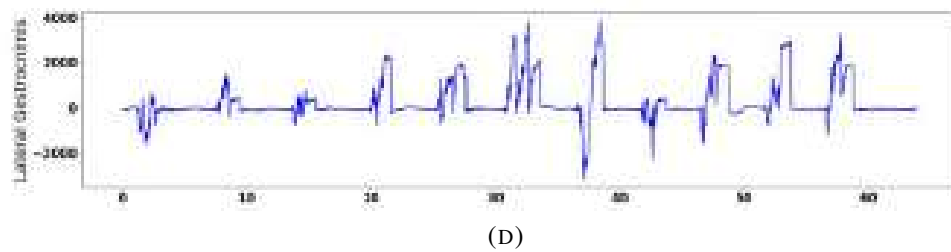
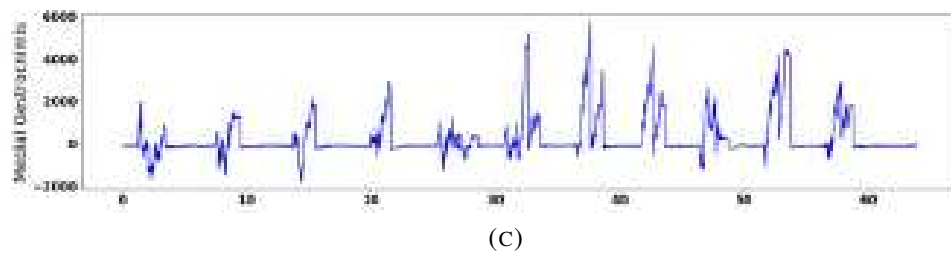
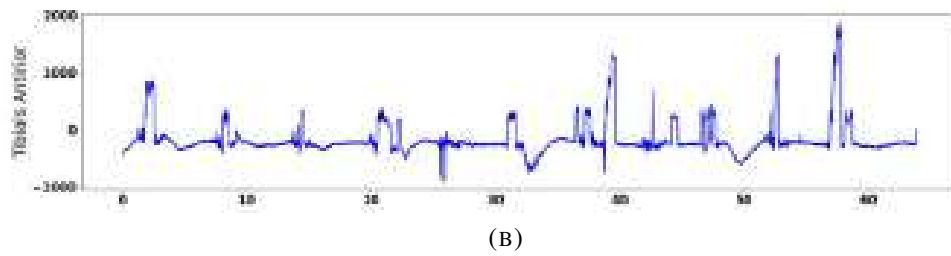
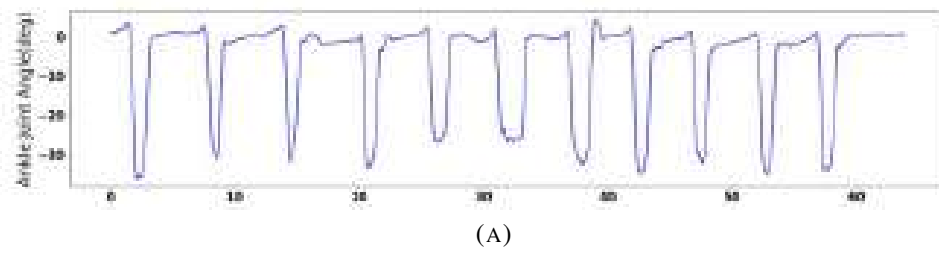


FIGURE 5.5: Ankle Joint Regression Experiment(Row data).

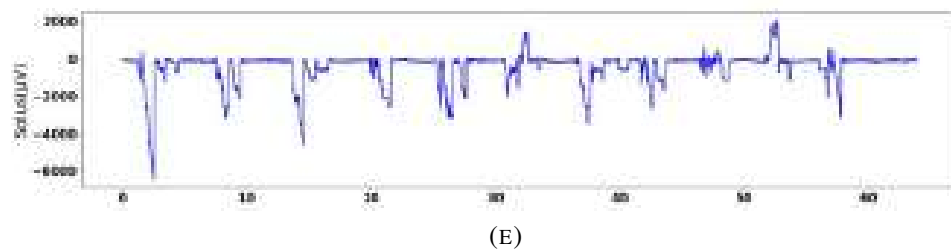


FIGURE 5.5: Ankle Joint Regression Experiment(Raw data).

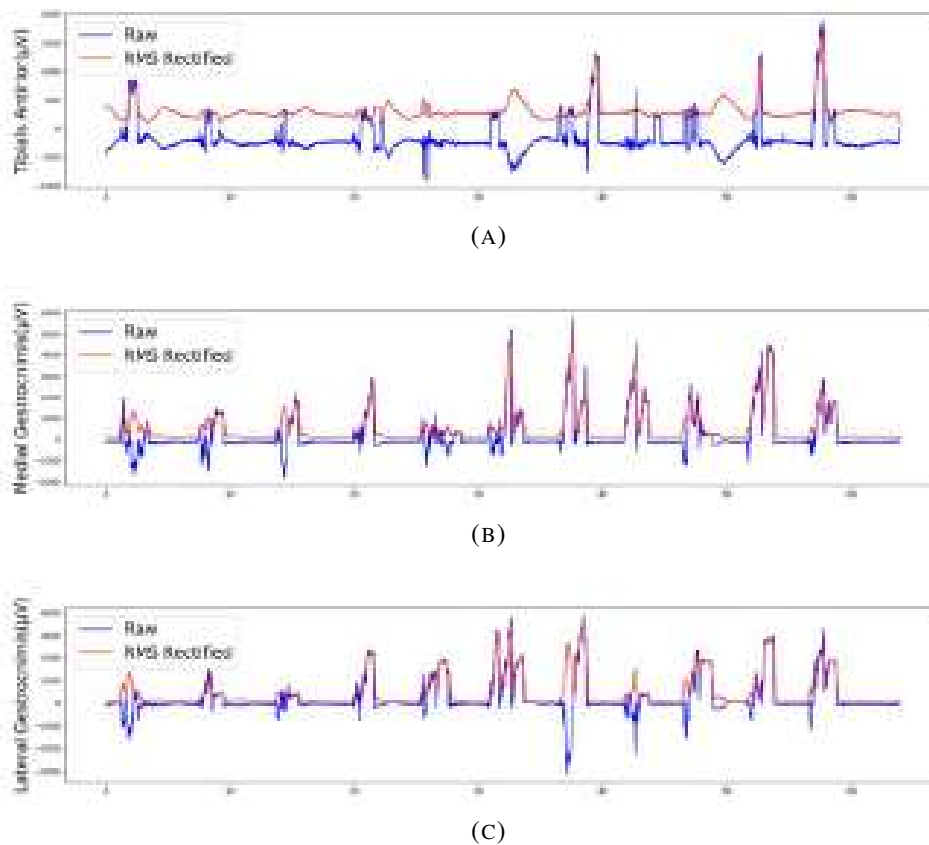


FIGURE 5.6: Muscles Contractions with RMS Rectification.

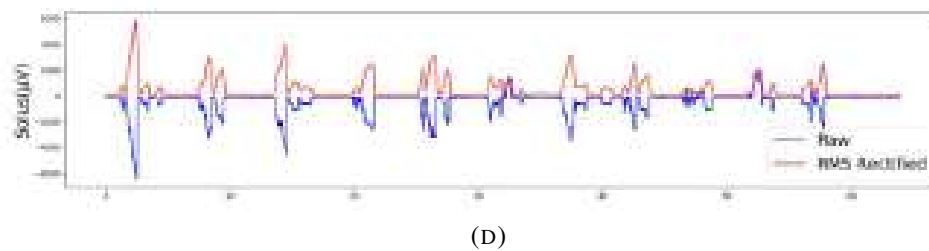


FIGURE 5.6: Muscles Contractions with RMS Rectification.

regression was recorded at k -value = 4 after the rectification process.

The regressor performance is measured by applying the test set of independent variables and comparing the predicted results with the dependent variable in the test set. Figure 5.8 shows the actual and the predicted values of the ankle joint angle for four muscles activities.

Figure 5.8 shows, most predicted angles are incorrect with a certain

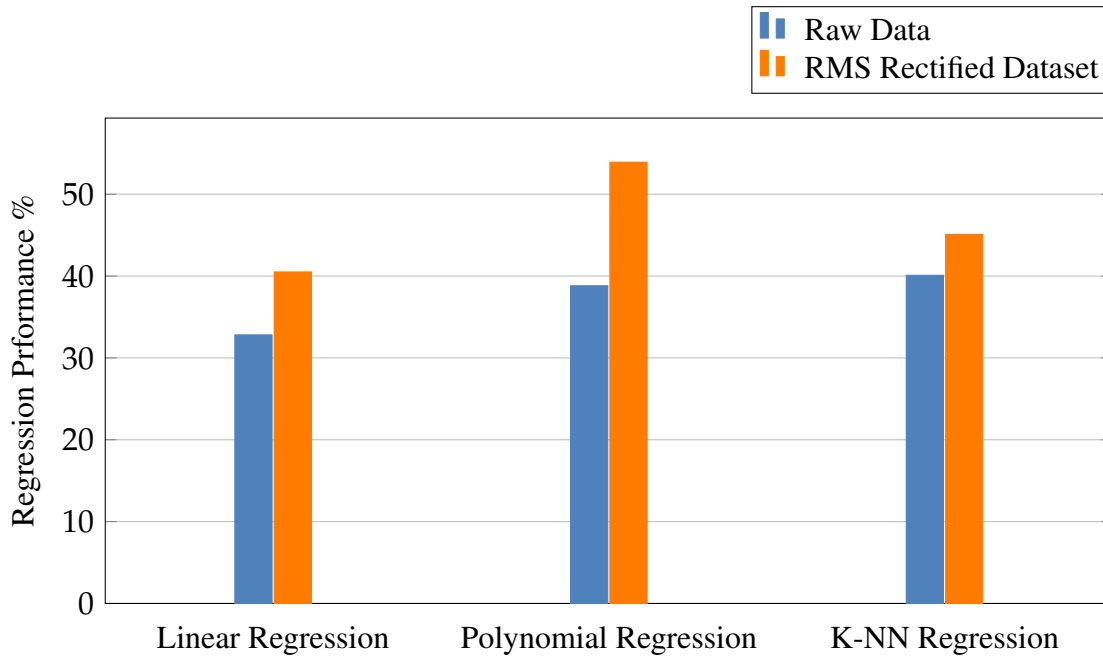


FIGURE 5.7: Regression technique performance for Raw EMG and Rectified Dataset with RMS Filter.

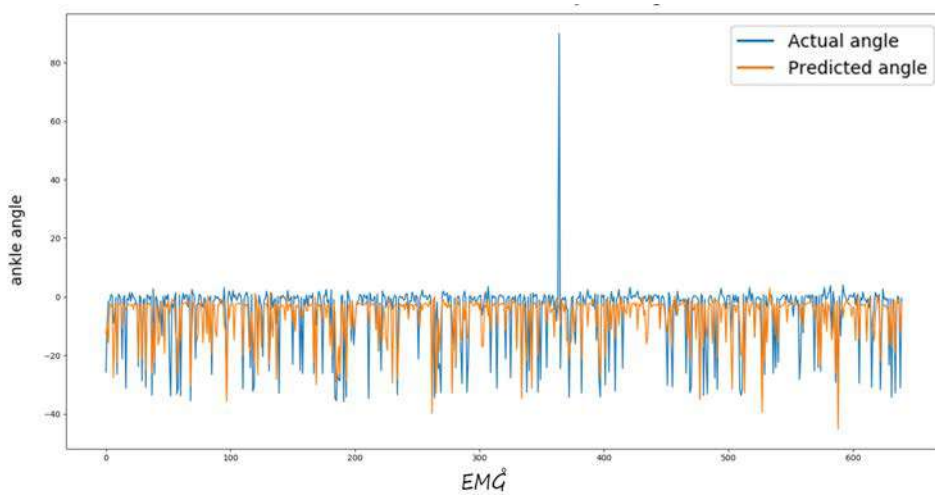


FIGURE 5.8: Actual-Predicted Ankle Joint Angles.

distance from the actual angle. The actual angles have an overshooting for an angle value that is more than 80 deg, which is unreasonable since the movement is a plantarflexion movement and the ankle joint angle should always be negative; a fault may cause this artifact in the data transmission of the sample across the wireless system(WIFI) of the data acquisition system. The row data of the dependent variable is refined furthermore by

removing the overshoots and replace the overshoot sample point with an average value of the neighbours' sample point(a window of 10 sample points is considered)

For better understanding of the signal, it is convenient to convert the signal into the frequency domain. As explained in section 3.10.3, discrete Fourier transformation converts the underlying signal into the frequency domain. The fast Fourier transformation(FFT) is the most efficient algorithm for converting the time domain signal into the frequency domain. A signal is represented in the frequency domain by the frequency spectrum, where its amplitude in the time domain represents each frequency. Figure 5.9 shows the Tibialis Anterior muscle activity both in the time domain and frequency domain. As shown in the time domain, the signal has a clear waveform with small distortions all over the signal, which certainly represents the noise. As shown in the time domain, the noise has a low amplitude with a high fluctuation, which occurs at high frequency.

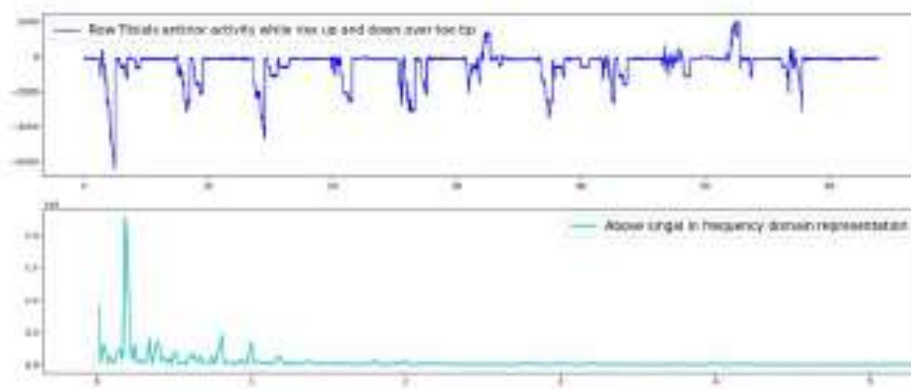


FIGURE 5.9: Tibialis Anterior muscle activity both in the time domain and frequency domain.

Figure 5.9 shows, the noise is occurring within the high frequencies range. Thus it is reasonable to use the lowpass filter. Lowpass Filter(LPF) passes the frequencies lower than a certain cut-off frequency and bands the

rest of the upper frequencies. As explained in section 3.10.6, the parameters that control the Lowpass filter(LPF) are the cut-off frequency and the order that controls the transition at the cut-off frequency, as shown in figure 3.42. Figure 5.9 shows the last peak takes place at frequency 2HZ. Therefore, it is reasonable to choose it as the cut-off frequency. Figure 5.10 shows the application of the lowpass filter(LPF) at a cut-off frequency of 2HZ and having an order of 2 for all recorded muscles.

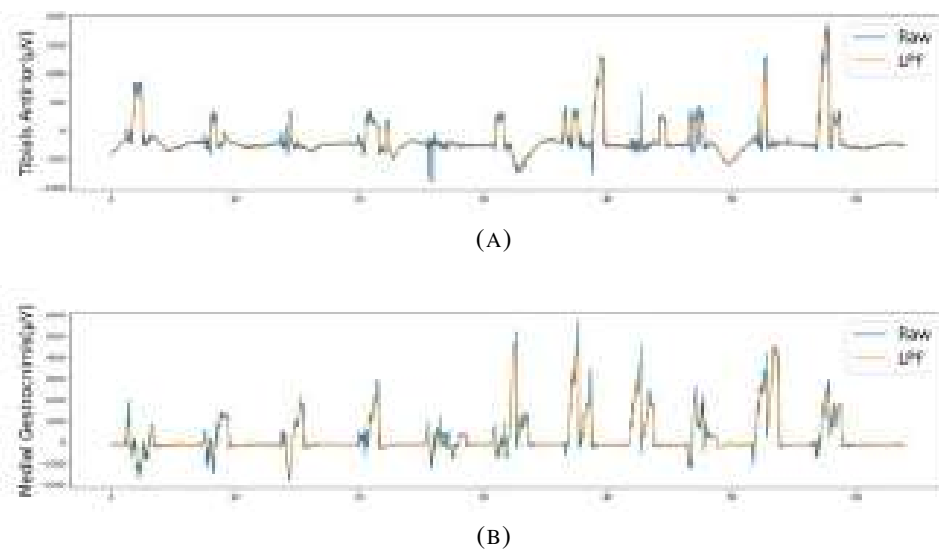


FIGURE 5.10: LPF with $f_c = 2HZ$, and $n = 2$ for all recorded muscles.

Figure 5.10 shows that the LPF removes the distortions in muscles activity signals and produces a more smooth and refined signal. Now it is convenient to apply the RMS rectification into the muscles activity signals in order to remove the negative sample points, as shown in figure 5.11.

Figure 5.12 shows the performance of linear, polynomial, and KNN regression algorithms with all Raw, RMS rectified, LP filtered, and LP filtered, then RMS rectified datasets. Figure 5.12 shows a small enhancement in the performance of all algorithms after the application of the LPF and removing the high-frequency noise as compared with the raw dataset. The LP filtered dataset is still having a lower performance than the

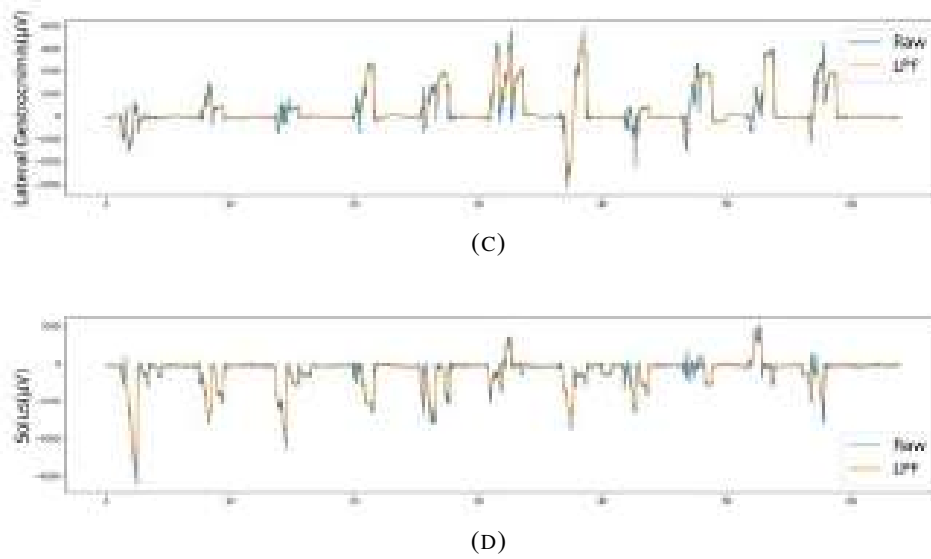


FIGURE 5.10: Application of LPF with $f_c = 2\text{HZ}$, and $n = 2$ for all recorded muscles.

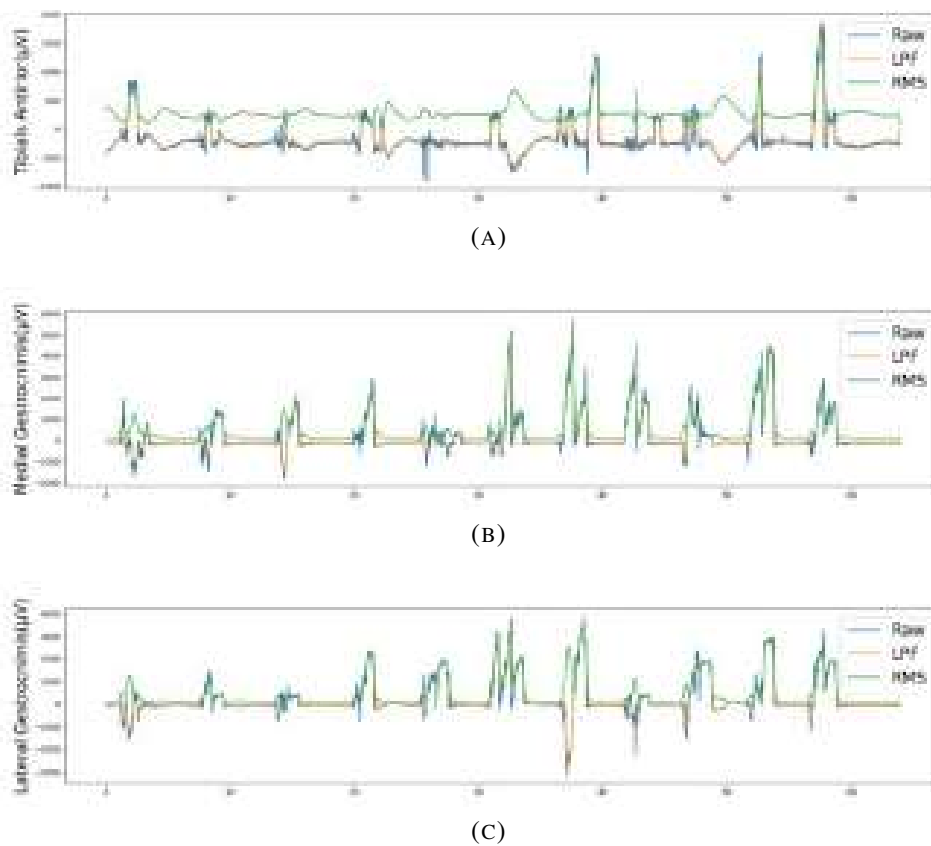


FIGURE 5.11: LPF with $f_c = 2\text{HZ}$, and $n = 2$, with RMS Rectification of 10 Window Size for All Recorded Muscles.

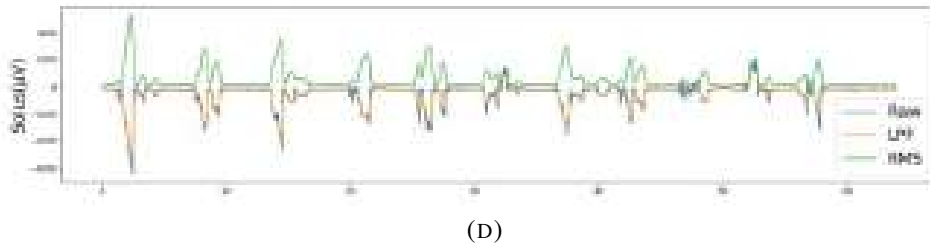


FIGURE 5.11: Application of LPF with $f_c = 2\text{HZ}$, and $n = 2$, with RMS Rectification of 10 Window Size for All Recorded Muscles.

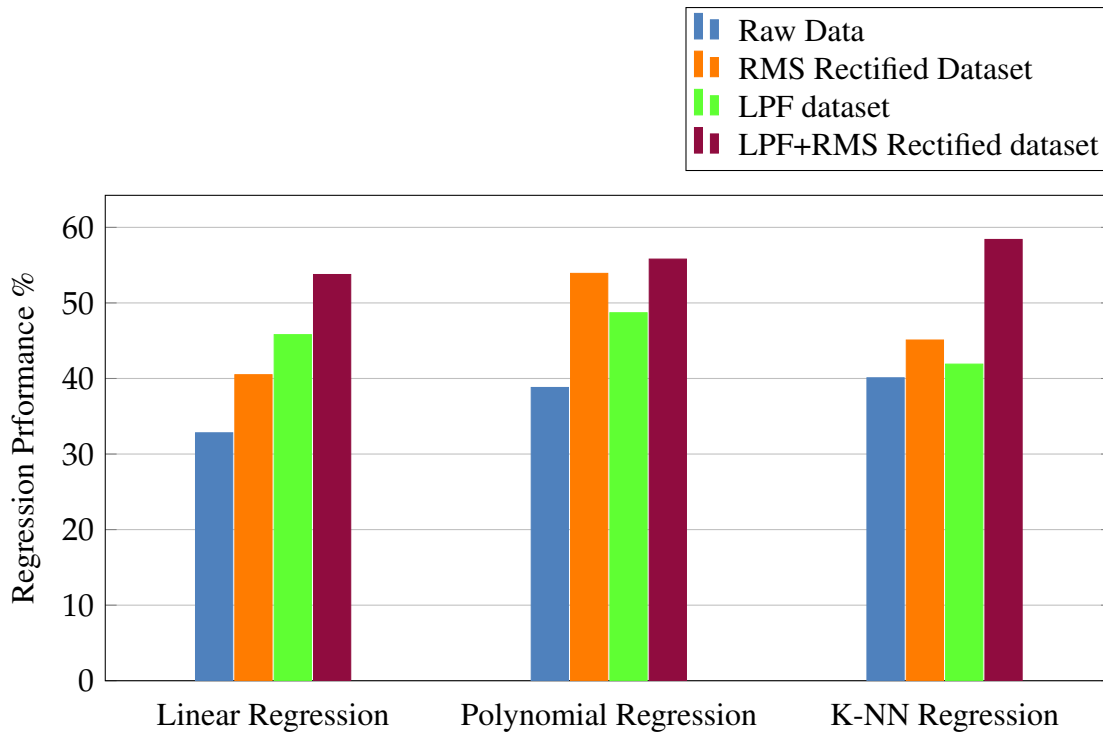


FIGURE 5.12: Regression technique performance for Raw EMG and Rectified Dataset with RMS Filter.

RMS rectified dataset due to the negative side of the signal. After applying the RMS rectification for the LP filtered dataset, a nature enhancement occurs. The performance is increased after applying both LPF and RMS rectification, but it is still around 50%, which means that the prediction has a probability of 50% being false. Therefore, a new methodology must be applied to get a higher performance, resulting in a higher accurate powered ankle-foot prosthesis response.

5.5 Ankle Joint Regression Optimization

As shown in figure 5.5 the high correlation between the ankle joint angle(dependent variable) and the muscles contractions(independent variable) is obvious. Therefore, a high regression performance should be produced. Figure 5.13 shows the muscle contractions after applying LPS and RMS rectification with the ankle angle for a single cycle. Figure 5.13 illustrates that the pattern of the correlation between the muscles contraction and the ankle joint angle is still unclear.

The noise and the unobserved error sources might still affect the original signal and do not produce a clear pattern. The proposed method in section 3.11 is applied, this hypothesis assumes that the sample point for a single ankle joint angle has a Gaussian distribution of the muscles contractions and the distribution expectation is the most accurate record, and the rest points are affected by the noise and the noise increases as the distance between the point and the expectation is increase. The proposed method can be simply imagined by plotting a horizontal line at a certain ankle joint angle and project the intersections with ankle wave onto the muscles wave, then finding Gaussian expectation(mean) at these muscles records, this is illustrated in figure 5.14.

Figure 5.15 shows the application of the proposed method onto the raw EMG dataset of the four recorded muscles activity. Figure 5.16 shows, the muscles activity has a repeated pattern with plantarflexion movements after applying the proposed method.

The proposed method produced a more precise pattern of muscles contractions with the plantarflexion movements. It still, has a more rough signal, but this problem is overcome using traditional filters(discussed in section 5.4). Figures 5.16, and 5.17 show the application of LPF and LPF

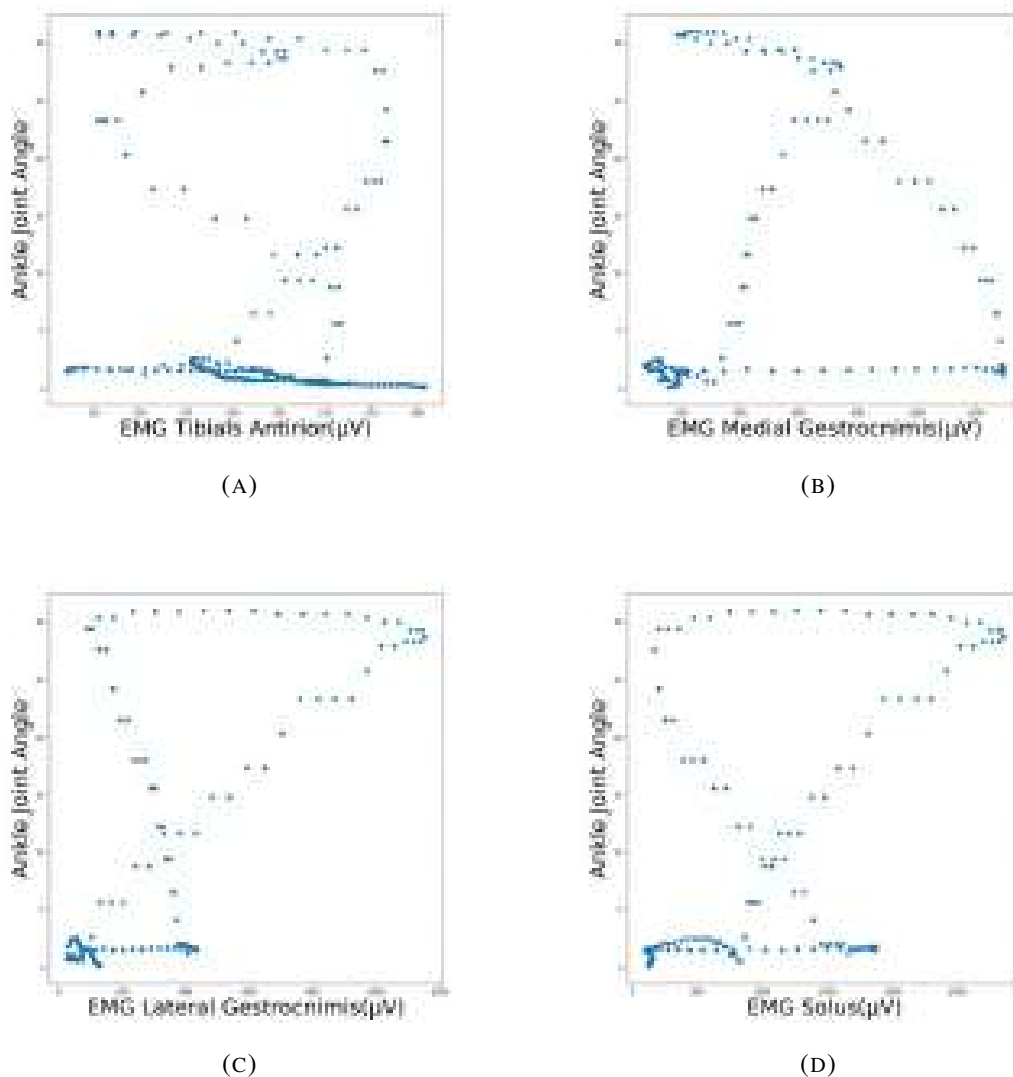


FIGURE 5.13: Recorded Muscles Contractions with Corresponding Ankle Joint Angles.

with RMS rectification with the Gaussian averaged dataset, respectively.

Figure 5.18 shows the performance of the regression above algorithms with different datasets; the proposed method for processing the EMG signals produced the highest performance among all types of processing methods. The performance increased drastically with this method, as shown in figure 5.18. The performance of the polynomial regression for the proposed method raised to 84.157%.

Figure 5.19 shows the polynomial regression fitting for the four recorded

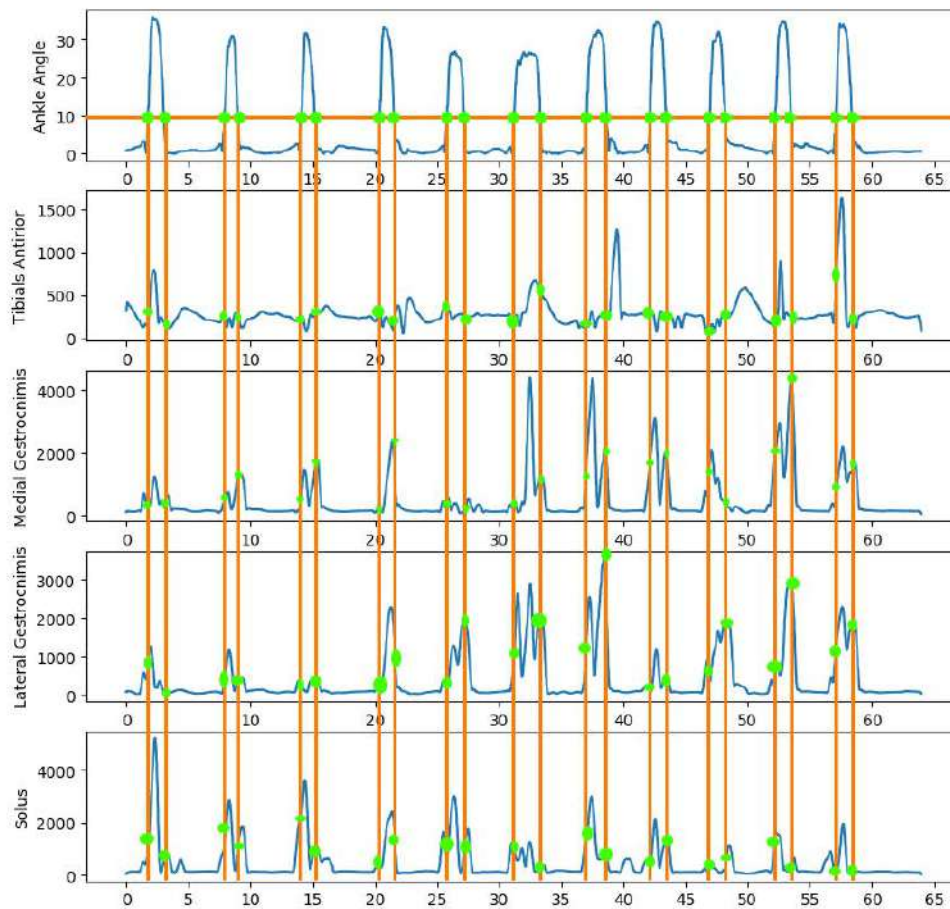
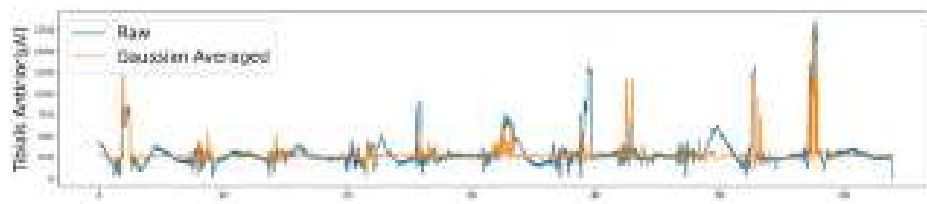


FIGURE 5.14: Proposed Method Illustration.

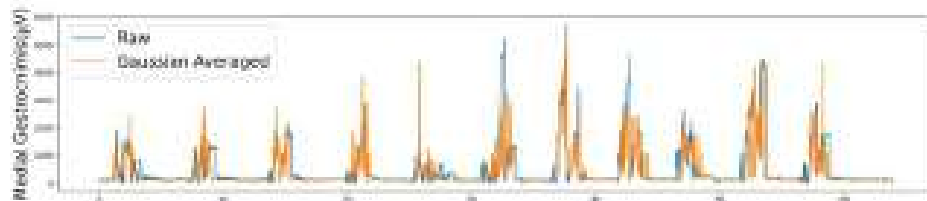
muscles after applying the proposed method. A clear pattern of the muscles contractions and the ankle joint angle correlation could be observed from figure 5.19.

Gaussian smoothing filter is considered as one of the most effecting filters in reducing distortion in the considered signal. It is tried in this work to compare it with LPF, then choose the one with the highest regression performance.

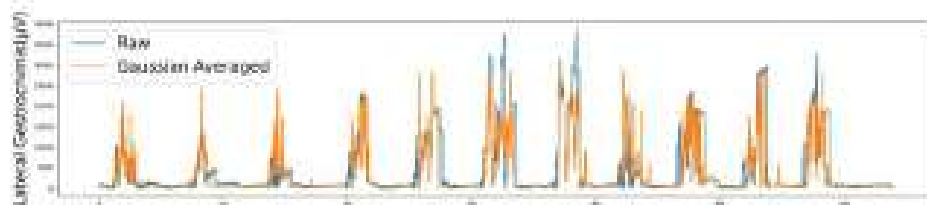
Figures 5.20, and 5.21 show the application of the Gaussian smoothing filter of the standard deviation of 5 with Gaussian averaged dataset then



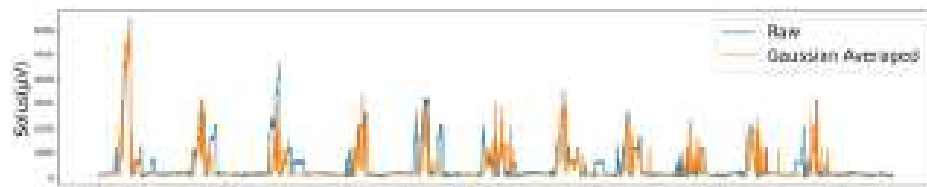
(A)



(B)



(C)



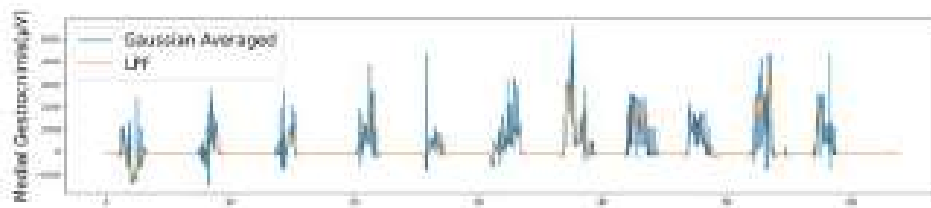
(D)

FIGURE 5.15: Application of The Proposed Method with The Raw Dataset.

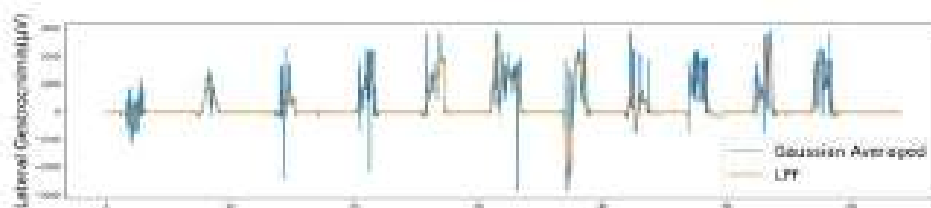


(A)

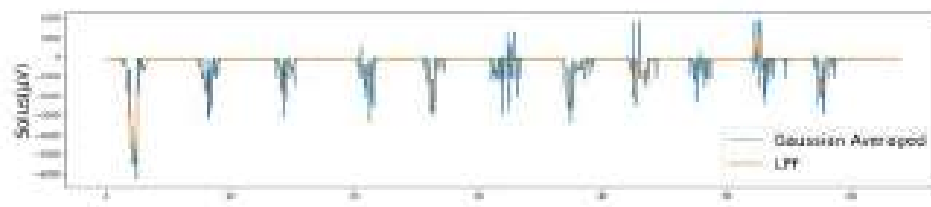
FIGURE 5.16: Application LPF with The Proposed Method.



(B)

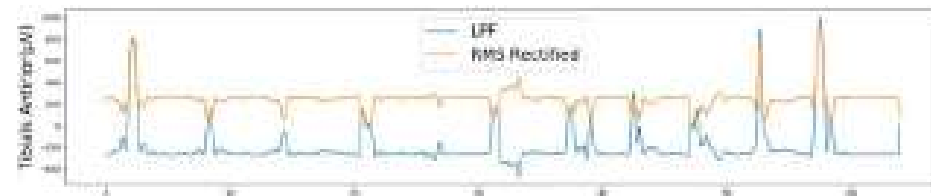


(C)

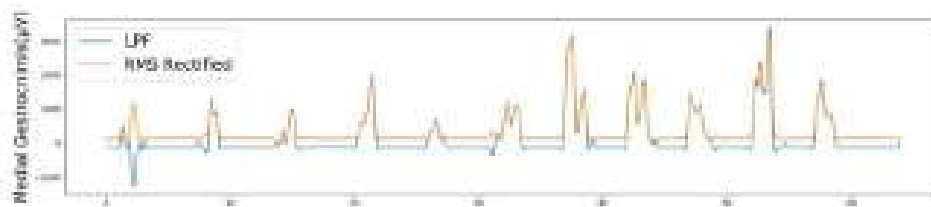


(D)

FIGURE 5.16: Application LPF with The Proposed Method.



(A)



(B)

FIGURE 5.17: Application LPF Then RMS Rectification with The Proposed Method.

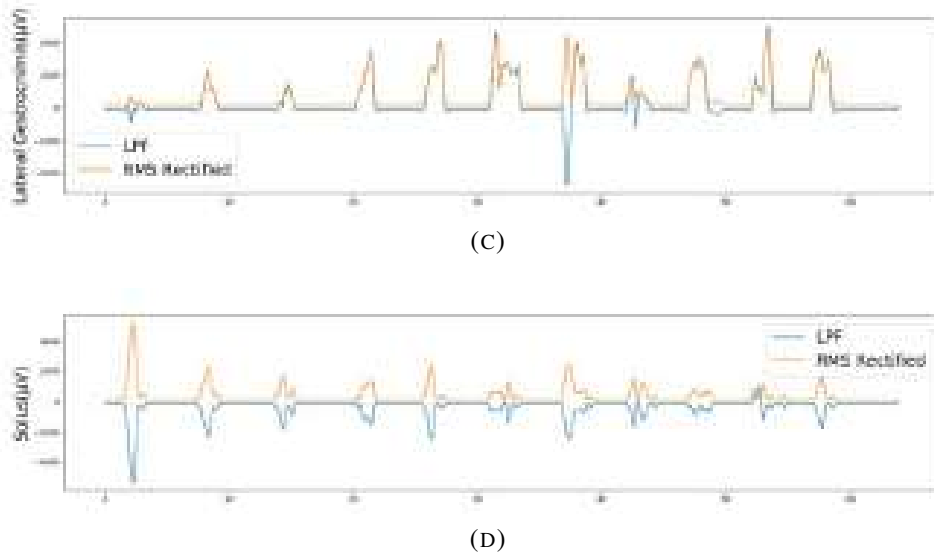


FIGURE 5.17: Application LPF Then RMS Rectification with The Proposed Method.

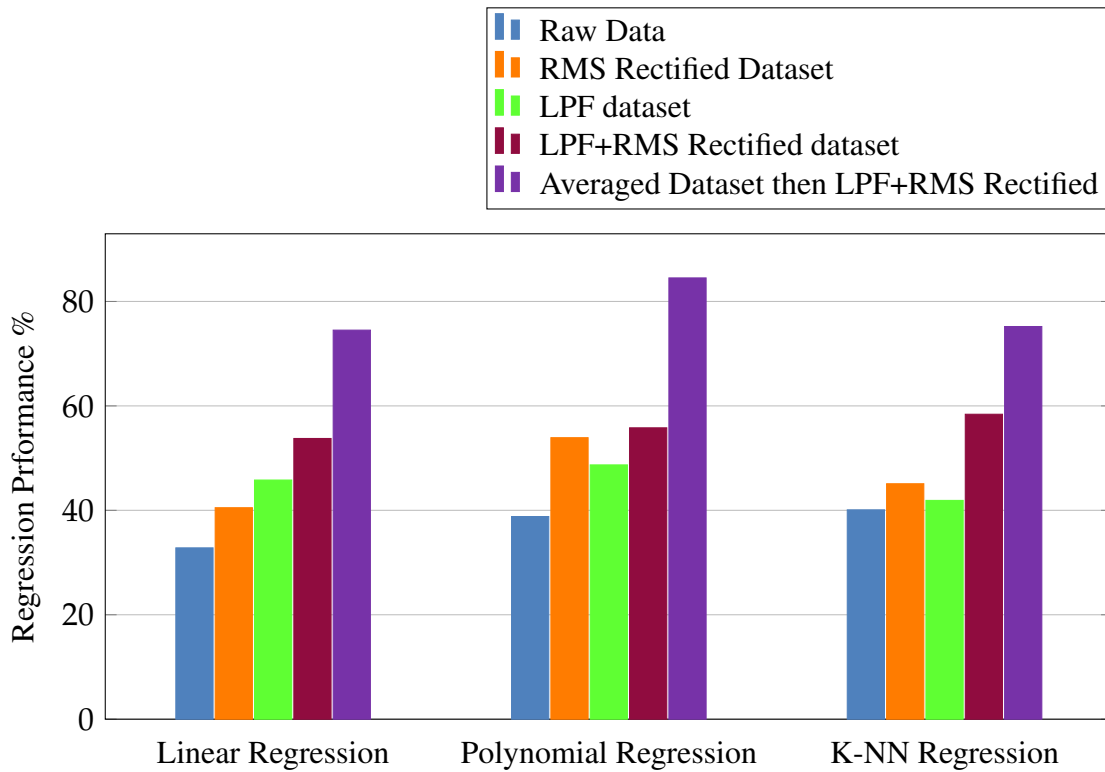


FIGURE 5.18: Regression technique performance for Raw EMG and Rectified Dataset with RMS Filter.

application of RMS rectification, respectively. Figure 5.21 shows that the RMS rectified dataset after the application of the Gaussian smoothing filter is

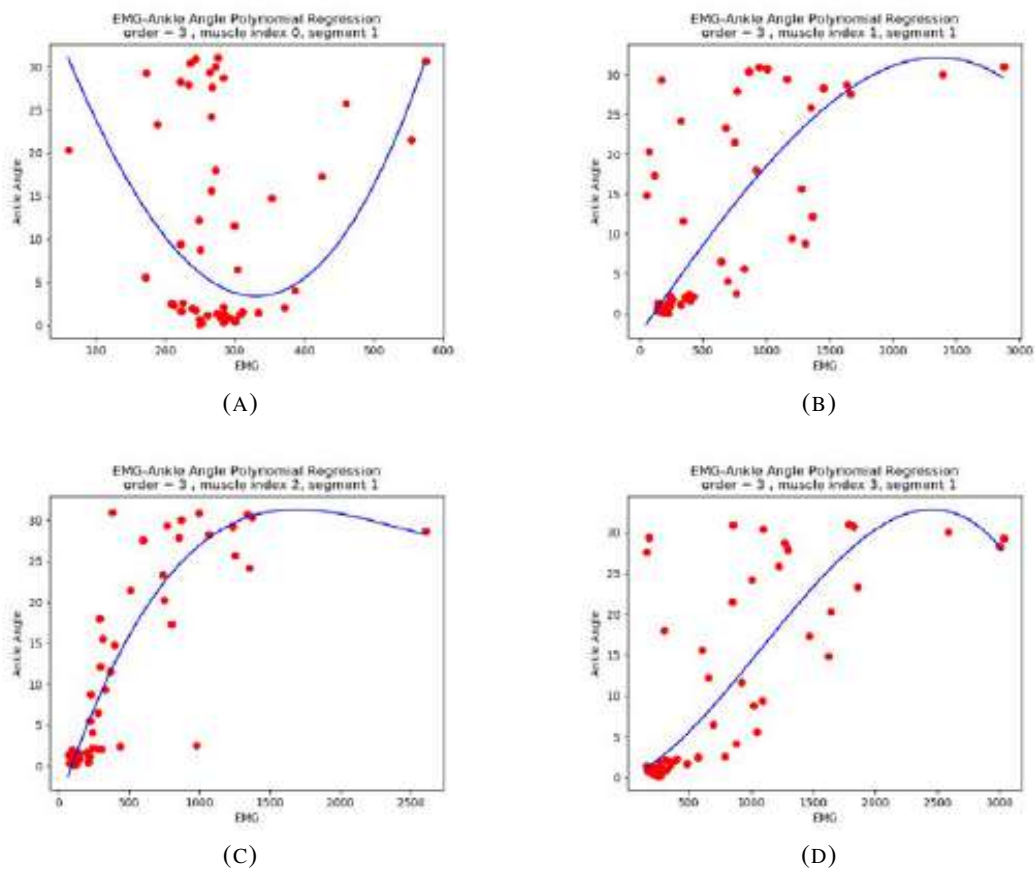


FIGURE 5.19: Recorded Muscles Contractions with Its corresponding Ankle Joint Angles.

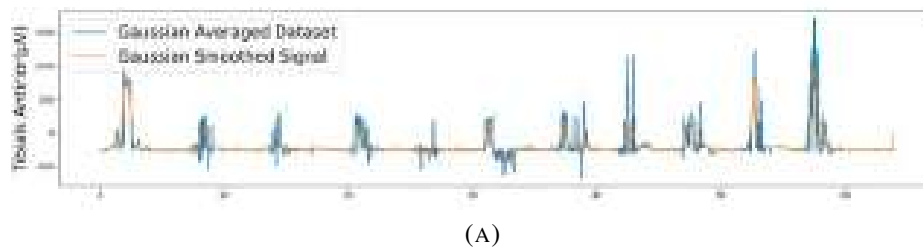


FIGURE 5.20: Gaussian smoothing($\sigma = 5$) Filter Application with Gaussian Averaged Dataset.

identical with the dataset that is only processed using a gaussian smoothing filter. GSF is smoothing the underlying signal, which produces a similar signal after applying the RMS rectification. GSF is still producing a negative part of the signal, which does not have any meaning in the EMG signal, and here is the benefit of the RMS rectifier, which removes the negative parts of

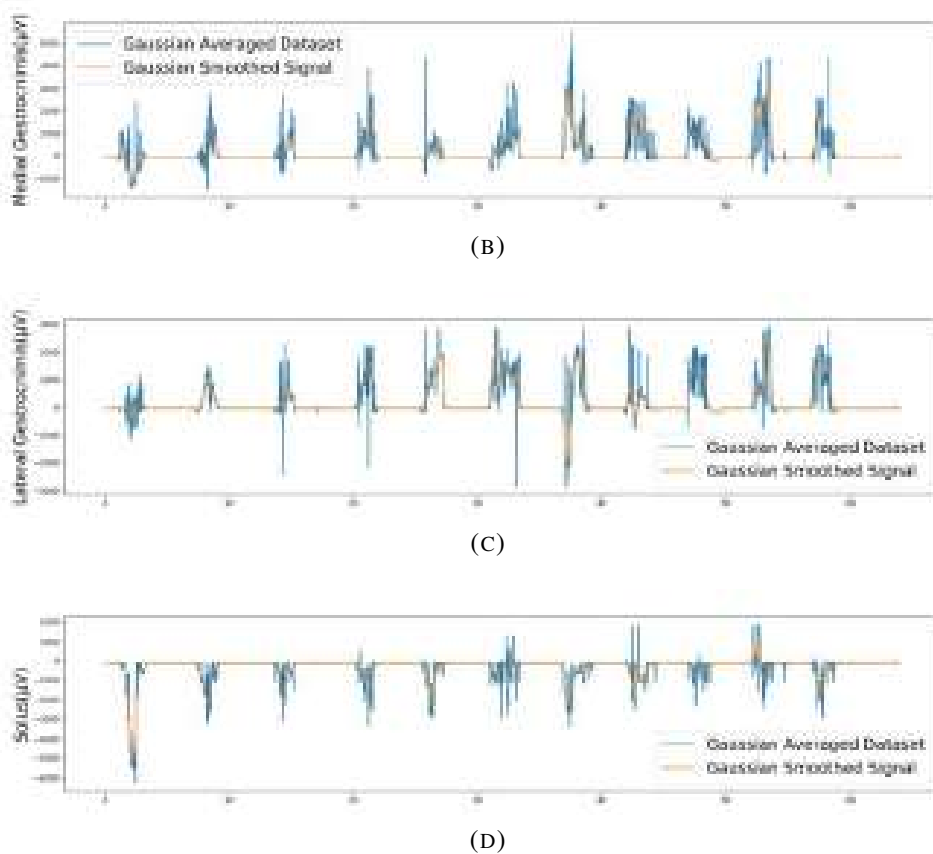


FIGURE 5.20: Gaussian Smoothing($\sigma = 5$) Filter Application with Gaussian Averaged Dataset.

the signal, as shown in figure 5.21.

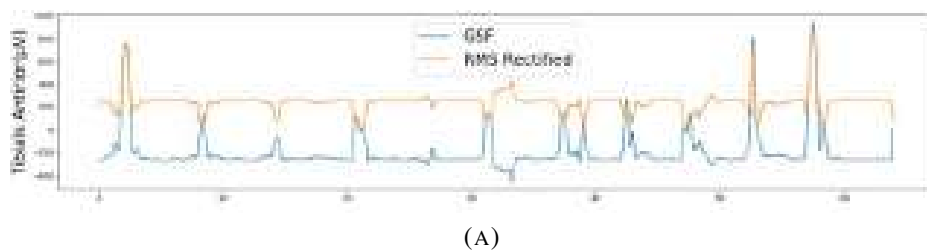


FIGURE 5.21: The Gaussian Smoothing($\sigma = 5$) Filtered Dataset the RMS Rectified.

Figure 5.22 shows the regression algorithms performance for the Gaussian averaged dataset processed with the Gaussian smoothing filter(GSF). GSF produces a higher performance for all regression algorithms and even higher than the dataset processed with both LPF and

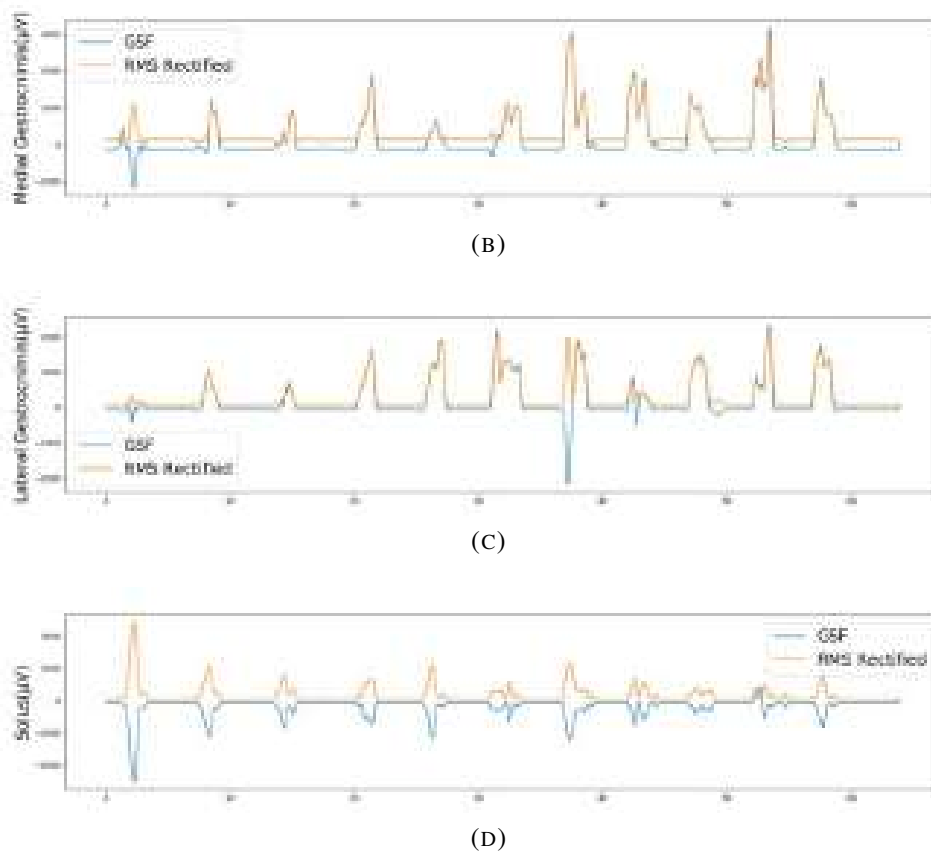


FIGURE 5.21: The Gaussian Smoothing($\sigma = 5$) Filtered Dataset the RMS Rectified.

RMS rectifier. GSF shows an equal performance in KNN with the averaged dataset processed with LPF and RMS rectifier; this means that GSF could be a reliable process that gives higher performance than both LPF and RMS rectification in a compilation.

Figure 5.23 shows the regression algorithms performance for the gaussian averaged dataset processed with both GSF then RMS rectifier, which removes the negative side of the EMG signal. Figure 5.23 shows the last configuration of the process(Gaussian average then GSF, then RMS rectification) gives the highest performance for all algorithms. KNN shows the highest performance, which reached 94.88%. The highest performance for KNN was observed at $k=80$, which was 95.51%; for Polynomial regression, the highest regression is 87.92% at a degree of 4.

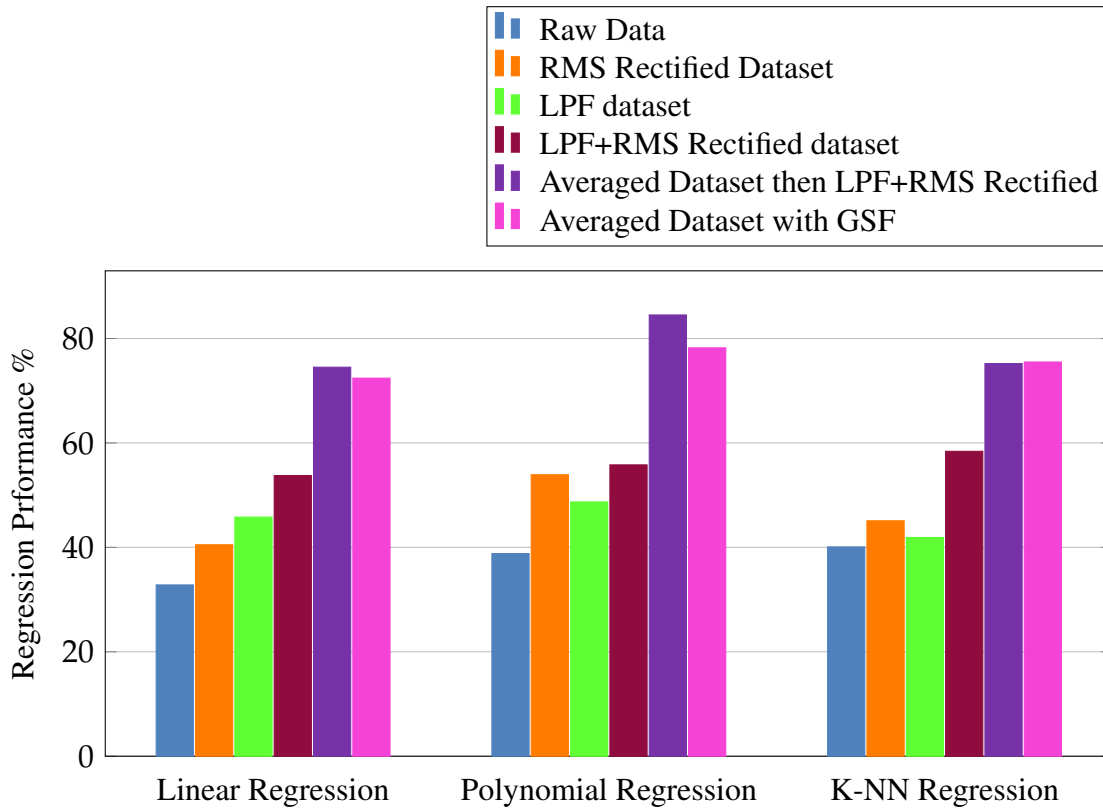


FIGURE 5.22: Regression technique performance for GSF of The Averaged Dataset.

The high regression performance is sensitive in predicting the response of the powered prosthesis for the given signals. The accuracy of the proposed method can be increased if reinforcement learning is used where the training data will be increased for longer gait cycle trails, which may produce a more accurate Gaussian mean.

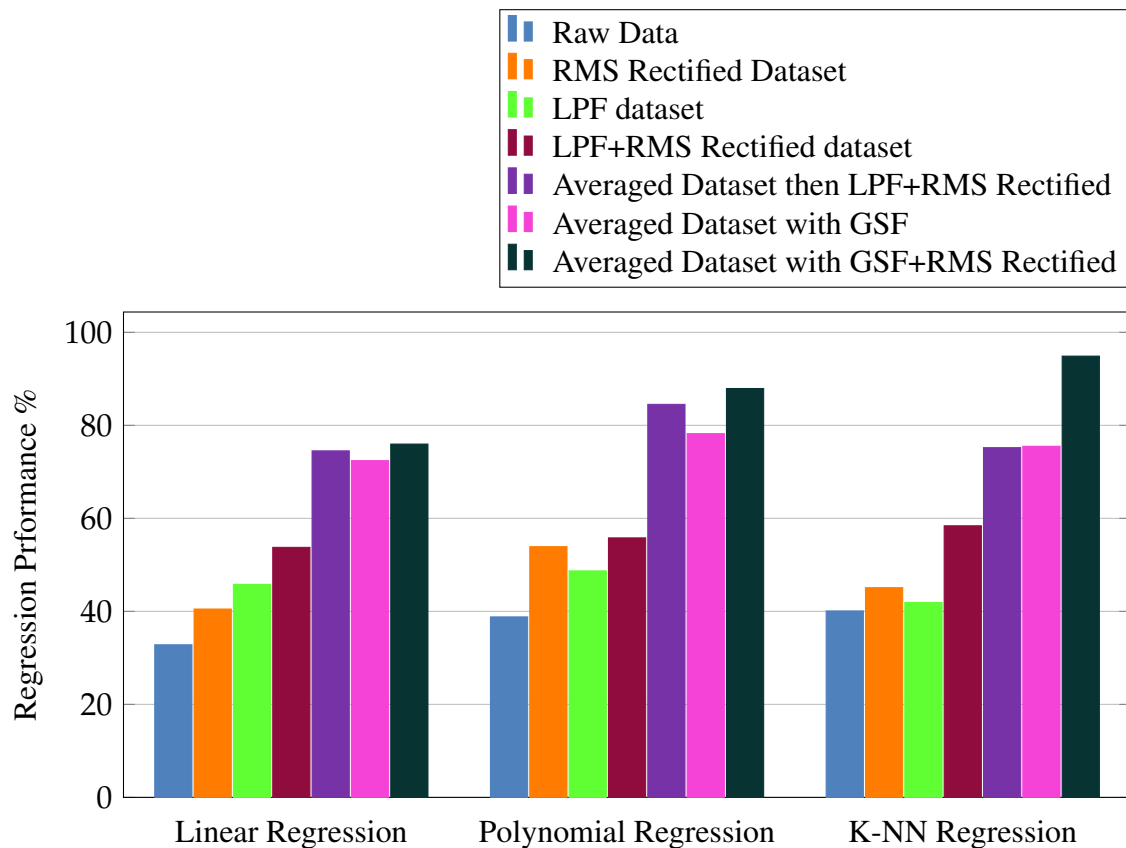


FIGURE 5.23: Regression technique performance for GSF and RMS Rectification of The Averaged Dataset.

5.6 Ankle Joint Regression Feature Selection

As explained in section 3.7, feature selection reduces the number of redundant features and gets the most compelling feature on the dependent variable. The linear correlation coefficient is used in this work to gain the most effective muscle on the regression of the ankle joint in terms of plantarflexion and dorsiflexion movements. Figure 5.24 shows the correlation coefficient of the four recorded muscles with the ankle joint angle regression. The symbols shown in figure 5.24 are interpreted in table 5.2.

Both the Lateral Gastrocnemius muscle and Medial Gastrocnemius muscle showed the highest correlation with the ankle joint angle. Medial and Lateral Gastrocnemius muscles are the main planter-flexors muscles.

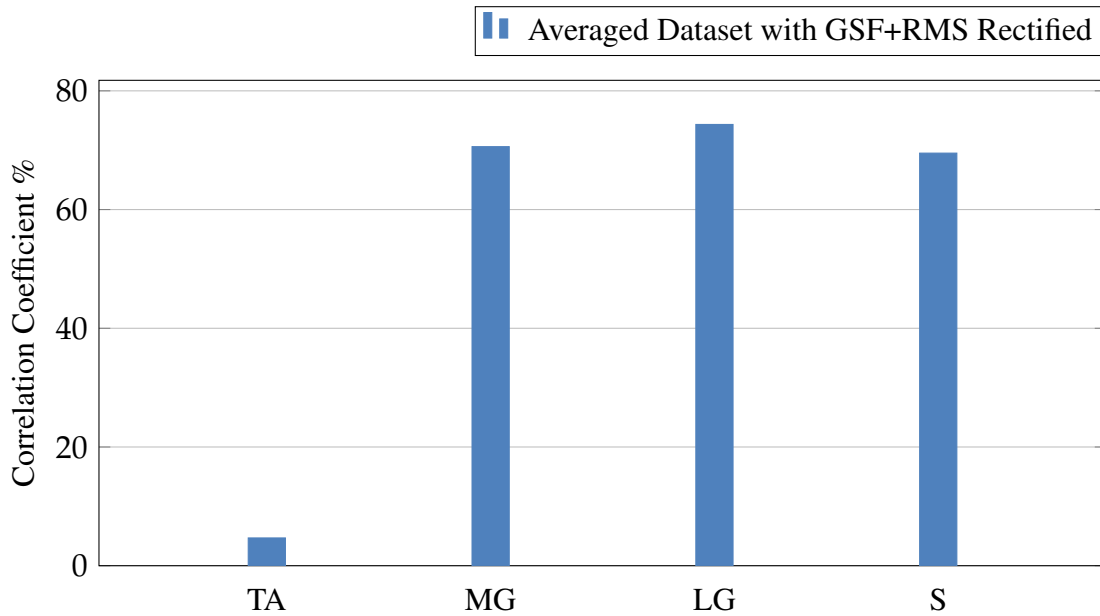


FIGURE 5.24: Correlation Coefficient of Four Muscles Contractions With Ankle Joint Angle.

TABLE 5.2: Muscles Symbols Interpretations

Symbol	Full muscle name
S	Soleus muscle
MG	Medial Gastrocnimius
LG	Lateral Gastrocnimi
TA	Tibials Antirior

The plantarflexion movement is used to lift the whole body. Therefore, these muscles exert the highest activity force, as shown in figure 5.25.

Figure 5.26 shows the enhancement in the regression performance by the use of the Lateral Gastrocnemius muscle and compares it with the performance done by four muscles. Feature selection increases the performance and will direct us to few muscles that can operate the powered prosthesis instead of recording the leg muscles. A consequence of recoding fewer muscles is the need for less complicated-expensive data acquisition devices, such as single or double EMG sensor channels instead of four-sixteen channels.

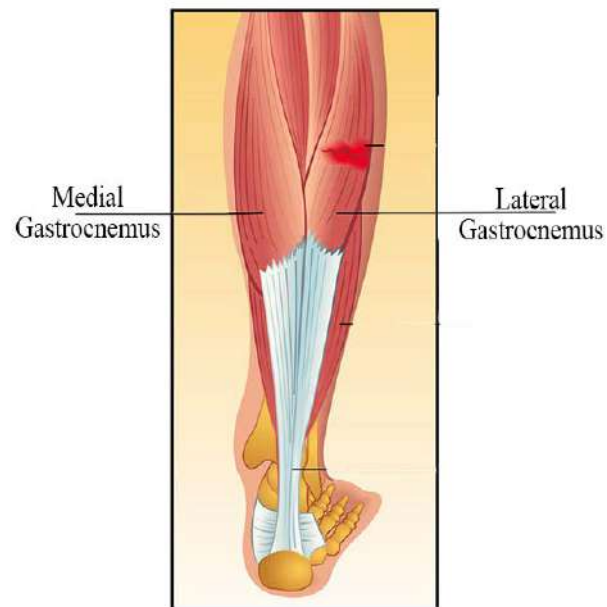


FIGURE 5.25: Gastrocnemius Muscle.

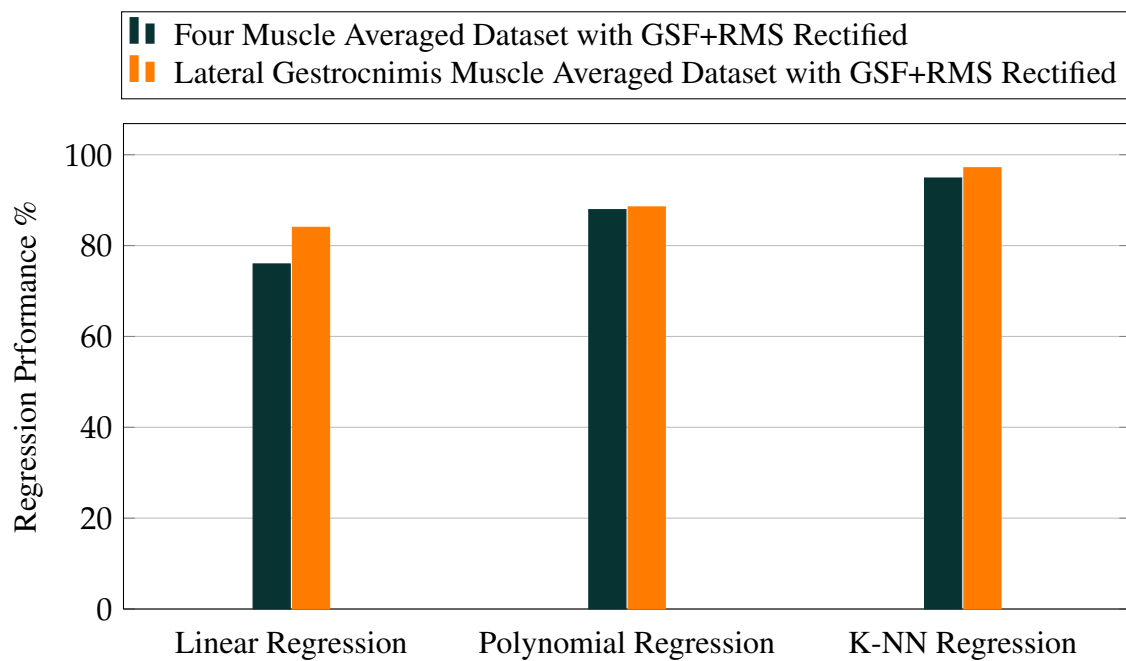


FIGURE 5.26: Regression performance for GSF and RMS Rectification of The Averaged Dataset of Lateral Gastrocnemius Muscle and The Four Muscles.

Chapter 6

Conclusions and Recommendations

6.1 Conclusions

This work aims to investigate, design, and build a suitably powered ankle-foot prosthesis. The following points are concluded from this work:

1. The linear actuator should exert a force almost equal or higher to the amputee's weight to lift and push his body forward.
2. Complementary filters produce the most smooth and accurate records using data acquisition devices of two sensors.
3. The timestamp is considered an essential variable in synchronising two datasets with different sampling frequencies.
4. The median filter produced a higher gait cycle classification based on EMG signals than the RMS filter. The median filter is known for removing the outlier produced by the noises, which could be why its high ability in noise filtering.
5. K-Nearest Neighbors(KNN) classification algorithm produced the highest performance, and this was caused due to the increased separation between the dataset for each case.

6. The Rectus Femoris muscle showed the highest separation between gait cycle phases(stance and swing phases). Therefore, it is considered the best parameter to distinguish between the phases. The Rectus Femoris muscle functions during the knee flexion movement. Consequently, it has produced the highest activity during the swing phase.
7. RMS filter shows a decent improvement in the regression performance due to its ability to remove the negative signals side, which does not have any meaning in EMG singles, and they could be considered noises.
8. EMG is mainly corrupted with noise falls in the high-frequency band. Therefore, a low pass filter will be optimal to filter this noise.
9. A hypothesis proposed in this work stated that a correlated signal corrupted with noise should have a Gaussian distribution for each dependent variable point and the most accurate value located at the Gaussian distribution expectation. This process produced a noticeable and drastic improvement in the regression performance. This process also produced a more precise pattern(i.e the performance rise to 80-95% according to the modelling algorithm).
10. Gaussian smoothing filter(GSF) showed a maximum effect over the low pass filter regarding noise movement and the smoothing of the EMG signals.
11. Both Lateral and Medial Gastrocnemius muscles produce the highest correlation coefficient with the ankle joint angle(70%, 74%), and indeed, the regression for this muscle will deliver higher performance. Regression feature selection will improve the regression performance,

but it will require less complicated and less expensive EMG sensors for operating the powered prosthesis.

6.2 Recommendations for Future Work

1. Classification in the frequency domain to find the noise band.
2. Design the optimal control system that operates the powered ankle-foot prosthesis.
3. Ankle joint regression based on above-knee muscles contractions for above-knee amputees.
4. Ankle joint angle regression deviation between the intact and amputated limbs
5. The mechanical analysis includes finite element analysis and material selection for the proposed powered ankle-foot prosthetic CAD mode.

References

- [1] C. L. McDonald et al. “Global prevalence of traumatic non-fatal limb amputation”. In: *Prosthetics and orthotics international* (2021), p. 0309364620972258.
- [2] K. Ziegler-Graham et al. “Estimating the prevalence of limb loss in the United States: 2005 to 2050”. In: *Archives of physical medicine and rehabilitation* 89.3 (2008), pp. 422–429.
- [3] K. Ziegler-Graham et al. “Estimating the prevalence of limb loss in the United States: 2005 to 2050”. In: *Archives of physical medicine and rehabilitation* 89.3 (2008), pp. 422–429.
- [4] *Mines in Iraq - A Silent Enemy, Deferred Death*. URL: https://www.dw.com/ar/%D8%A7%D9%84%D8%A3%D9%84%D8%BA%D8%A7%D9%85-%D9%81%D9%8A-%D8%A7%D9%84%D8%B9%D8%B1%D8%A7%D9%82-%D9%80-%D8%B9%D8%AF%D9%88-%D8%B5%D8%A7%D9%85%D8%AA-%D9%88%D9%85%D9%88%D8%AA-%D9%85%D8%A4%D8%AC%D9%84/a-14816385#:~:text=%D9%88%D8%A7%D9%84%D9%86%D8%AA%D9%8A%D8%AC%D8%A9%20%D8%A7%D9%84%D8%A5%D8%AC%D9%85%D8%A7%D9%84%D9%8A%D8%A9%D8%8C%20%D8%A7%D9%84%D8%AA%D9%8A%20%D8%AA%D9%88%D8%B5%D9%84%20%D8%A5%D9%84%D9%8A%D9%87%D8%A7,%D8%A8%D9%8A%D9%86%

- 2080%20%D9%88100%20%D8%A3%D9%84%D9%81%20%D8%B4%D8%AE%D8%B5..
- [5] U. nations. “4,800 amputation cases in Mosul conflict”. In: *BMC health services research* (). URL: middleeastmonitor.com/20180404-un-4800-amputation-cases-in-mosul-conflict/.
- [6] M. R. Pitkin. *Biomechanics of lower limb prosthetics*. Springer, 2009.
- [7] W. Pirker and R. Katzenschlager. “Gait disorders in adults and the elderly: A clinical guide”. In: *Wiener klinische Wochenschrift* 129 (Oct. 2016). DOI: [10.1007/s00508-016-1096-4](https://doi.org/10.1007/s00508-016-1096-4).
- [8] A. Bertomeu-Motos. “Biomechanics of human walking and stability descriptive parameters”. In: *Revista Doctorado UMH* 2 (Mar. 2016), p. 4. DOI: [10.21134/doctumh.v1i1.880](https://doi.org/10.21134/doctumh.v1i1.880).
- [9] O. Jones. *Anatomical Terms of Movement*. May 2020. URL: <https://teachmeanatomy.info/the-basics/anatomical-terminology/terms-of-movement/>.
- [10] O. Jones. *Muscles in the Anterior Compartment of the Leg*. Apr. 2019. URL: <https://teachmeanatomy.info/lower-limb/muscles/leg/anterior-compartment/>.
- [11] M. W. Whittle. *Gait analysis: an introduction*. Butterworth-Heinemann, 2014.

- [12] A. health direct. *Prostheses*. URL: <https://www.healthdirect.gov.au/prostheses#:~:text=Prostheses%20are%20artificial%20parts%20that,through%20the%20public%20health%20system..>
- [13] P. S. Selvam et al. “Prosthetics for Lower Limb Amputation”. In: *Prosthetics and Orthotics*. IntechOpen, 2021.
- [14] B. Majdic et al. “Redistributing the pressure of prosthetic systems”. In: Apr. 2017, pp. 294–299. DOI: [10.1109/SIEDS.2017.7937734](https://doi.org/10.1109/SIEDS.2017.7937734).
- [15] J. Andrysek. “Lower-limb prosthetic technologies in the developing world: A review of literature from 1994–2010”. In: *Prosthetics and orthotics international* 34.4 (2010), pp. 378–398.
- [16] L. D. Weiss, J. M. Weiss, and J. K. Silver. *Easy EMG e-book: A guide to performing nerve conduction studies and electromyography*. Elsevier Health Sciences, 2015.
- [17] P. Konrad. *The ABC of EMG: A practical introduction to kinesiological electromyography*. 2005.
- [18] T. A. Kuiken, M. Lowery, and N. Stoykov. “The effect of subcutaneous fat on myoelectric signal amplitude and cross-talk”. In: *Prosthetics and orthotics international* 27.1 (2003), pp. 48–54.
- [19] A. L. Hof. “EMG and muscle force: an introduction”. In: *Human Movement Science* 3.1-2 (1984), pp. 119–153.

- [20] Udacity. *Youtube Video : Deduction, Induction, Abduction - Georgia Tech*. Feb. 2015. URL: <https://www.youtube.com/watch?app=desktop&v=-nn3XMoPC7s/>.
- [21] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.
- [22] D. A. Winter and S. E. Sienko. “Biomechanics of below-knee amputee gait”. In: *Journal of biomechanics* 21.5 (1988), pp. 361–367.
- [23] H. Bateni and S. J. Olney. “Kinematic and kinetic variations of below-knee amputee gait”. In: *JPO: Journal of Prosthetics and Orthotics* 14.1 (2002), pp. 2–10.
- [24] A. H. Hansen et al. “The human ankle during walking: implications for design of biomimetic ankle prostheses”. In: *Journal of biomechanics* 37.10 (2004), pp. 1467–1474.
- [25] J. Gardiner et al. “Transtibial amputee gait efficiency: Energy storage and return versus solid ankle cushioned heel prosthetic feet”. In: *Journal of rehabilitation research and development* 53.6 (2016), pp. 1133–1138.
- [26] B. G. Lambrecht and H. Kazerooni. “Design of a semi-active knee prosthesis”. In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pp. 639–645.
- [27] C. A. Blatchford and S. Ltd. *Chas A Blatchford and Sons Ltd*. URL: <https://www.blatchford.co.uk/products/category/prosthetics/>.

- [28] *Triton smart ankle*. URL: <https://www.ottobock.com.tr/en/prosthetics/lower-limb/solution-overview/triton-smart-ankle/>.
- [29] *riton-smart-ankle brochure*. *Ottobock*. 2017. URL: https://media.ottobock.com/_website/prosthetics/lower-limb/triton-smart-ankle/files/646d880-en-02-1603w.pdf.
- [30] Össur. *PROPRIO FOOT Technical Manual*. URL: <https://www.ossur.com/en-us/prosthetics/feet/proprio-foot>.
- [31] MIT. *Biomechatronics – MIT Media Lab*. URL: <https://biomech.media.mit.edu/>.
- [32] S. K. Au and H. M. Herr. “Powered ankle-foot prosthesis”. In: *IEEE Robotics & Automation Magazine* 15.3 (2008), pp. 52–59.
- [33] S. K. Au, J. Weber, and H. Herr. “Powered ankle-foot prosthesis improves walking metabolic economy”. In: *IEEE Transactions on robotics* 25.1 (2009), pp. 51–66.
- [34] A. E. Ferris et al. “Evaluation of a powered ankle-foot prosthetic system during walking”. In: *Archives of physical medicine and rehabilitation* 93.11 (2012), pp. 1911–1918.
- [35] E. J. Rouse et al. “Design and testing of a bionic dancing prosthesis”. In: *PloS one* 10.8 (2015), e0135148.

- [36] C. D. Joshi, U. Lahiri, and N. V. Thakor. “Classification of gait phases from lower limb EMG: Application to exoskeleton orthosis”. In: *2013 IEEE Point-of-Care Healthcare Technologies (PHT)*. IEEE. 2013, pp. 228–231.
- [37] J. Ziegler, H. Gattringer, and A. Mueller. “Classification of gait phases based on bilateral emg data using support vector machines”. In: *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*. IEEE. 2018, pp. 978–983.
- [38] M. Derlatka and M. Bogdan. “Ensemble kNN classifiers for human gait recognition based on ground reaction forces”. In: *2015 8th International Conference on Human System Interaction (HSI)*. IEEE. 2015, pp. 88–93.
- [39] H. Huang, T. A. Kuiken, R. D. Lipschutz, et al. “A strategy for identifying locomotion modes using surface electromyography”. In: *IEEE Transactions on Biomedical Engineering* 56.1 (2008), pp. 65–73.
- [40] S. Dey et al. “A support vector regression approach for continuous prediction of ankle angle and moment during walking: An implication for developing a control strategy for active ankle prostheses”. In: *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*. IEEE. 2019, pp. 727–733.
- [41] A. Mucherino, P. J. Papajorgji, and P. M. Pardalos. “K-nearest neighbor classification”. In: *Data mining in agriculture*. Springer, 2009, pp. 83–106.

- [42] É. O. Rodrigues. “Combining Minkowski and Chebyshev: New distance proposal and survey of distance metrics using k-nearest neighbours classifier”. In: *Pattern Recognition Letters* 110 (2018), pp. 66–71.
- [43] S. Dreiseitl and L. Ohno-Machado. “Logistic regression and artificial neural network classification models: a methodology review”. In: *Journal of biomedical informatics* 35.5-6 (2002), pp. 352–359.
- [44] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [45] R. H. Myers and R. H. Myers. *Classical and modern regression with applications*. Vol. 2. Duxbury press Belmont, CA, 1990.
- [46] O. Kramer. “Unsupervised K-nearest neighbor regression”. In: *arXiv preprint arXiv:1107.3600* (2011).
- [47] A. Chugh. “MAE, MSE, RMSE, Coefficient of Determination, Adjusted R Squared — Which Metric is Better?” In: *medium* (2002).
- [48] M. A. Hall and L. A. Smith. “Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper.” In: *FLAIRS conference*. Vol. 1999. 1999, pp. 235–239.
- [49] M. Sebbna. “On Feature Selection: A New Filter Model.” In: *FLAIRS Conference*. 1999, pp. 230–234.

- [50] L.-Y. Chuang, C.-H. Ke, and C.-H. Yang. “A hybrid both filter and wrapper feature selection method for microarray classification”. In: *arXiv preprint arXiv:1612.08669* (2016).
- [51] A. G. Asuero, A. Sayago, and A. Gonzalez. “The correlation coefficient: An overview”. In: *Critical reviews in analytical chemistry* 36.1 (2006), pp. 41–59.
- [52] M. Ghalyan, M. Alher, and M. Jweeg. “Human Gait Cycle Classification Improvements Using Median and Root Mean Square Filters Based on EMG Signals”. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 1067. 1. IOP Publishing, 2021, p. 012146.
- [53] A. D. Poularikas. *Understanding Digital Signal Processing with MATLAB® and Solutions*. CRC Press, 2017.
- [54] D. Sundararajan. *The discrete Fourier transform: theory, algorithms and applications*. World Scientific, 2001.
- [55] E. Onajite. “Chapter 2 - Understanding Seismic Wave Propagation”. In: *Seismic Data Analysis Techniques in Hydrocarbon Exploration*. Ed. by E. Onajite. Oxford: Elsevier, 2014, pp. 17–32. ISBN: 978-0-12-420023-4. DOI: <https://doi.org/10.1016/B978-0-12-420023-4.00002-2>. URL: <https://www.sciencedirect.com/science/article/pii/B9780124200234000022>.
- [56] L. Tan and J. Jiang. “Chapter 2 - Signal Sampling and Quantization”. In: *Digital Signal Processing (Third Edition)*. Ed. by

- L. Tan and J. Jiang. Third Edition. Academic Press, 2019, pp. 13–58. ISBN: 978-0-12-815071-9. DOI: <https://doi.org/10.1016/B978-0-12-815071-9.00002-6>. URL: <https://www.sciencedirect.com/science/article/pii/B9780128150719000026>.
- [57] A. Saidi. “Decimation-in-time-frequency FFT algorithm”. In: *Proceedings of ICASSP’94. IEEE International Conference on Acoustics, Speech and Signal Processing*. Vol. 3. IEEE. 1994, pp. III–453.
- [58] W. T. Cochran et al. “What is the fast Fourier transform?” In: *Proceedings of the IEEE* 55.10 (1967), pp. 1664–1674.
- [59] A. I. Zayed. “A convolution and product theorem for the fractional Fourier transform”. In: *IEEE Signal processing letters* 5.4 (1998), pp. 101–103.
- [60] V. Zschorlich. “Digital filtering of EMG-signals”. In: *Electromyogr Clin Neurophysiol* 29.2 (1989), pp. 81–6.
- [61] I. Ghalyan, Z. Abouelenin, and V Kapila. “Gaussian filtering of EMG signals for improved hand gesture classification”. In: *2018 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*. IEEE. 2018, pp. 1–6.
- [62] *NodeMCU ESP-32S*. URL: https://docs.zerynth.com/latest/reference/boards/nodemcu_esp32/docs/.

- [63] *IMU (Inertial Measurement Unit)*. URL: https://product.tdk.com/info/en/products/sensor/motion-inertial/imu/technote/pov_imu.html.
- [64] *Introduction to MPU6050*. URL: <https://www.theengineeringprojects.com/2019/02/introduction-to-mpu6050.html>.
- [65] R. J. Fetick. *MPU6050 light library documentation*. Jan. 2021. URL: https://github.com/rfetick/MPU6050_light/blob/master/documentation_MPU6050_light.pdf..
- [66] *Unity3D*. URL: <https://unity.com/>.
- [67] M. Fahad. *Unity Regression Experiments Application*. 2021. URL: <https://github.com/icheetahgames/AnkleJointRegressionExperiments>.
- [68] R. Horne and K. K. Hausman. *3D printing for dummies*. John Wiley & Sons, 2017.
- [69] E. Criswell. *Cram's introduction to surface electromyography*. Jones & Bartlett Publishers, 2010.
- [70] M. Fahad. *Experiments Sample Video*. 2021. URL: <https://youtu.be/k6tCBXJcBlI>.
- [71] M. Fahad. *Raw and Combined Datasets*. 2021. URL: <https://github.com/icheetahgames/AnkleJointRegressionExperiments/commit/31702079e884d51816379e6112ecb9b744a20f9f>.

-
- [72] M. Fahad. *Powered Ankle-Foot Prosthesis (Solidworks Design)*. 2021. URL: <https://youtu.be/U8P3CwF-Ulk>.
- [73] M. Fahad. *Powered Prosthesis SolidWorks Design*. 2021. URL: <https://drive.google.com/file/d/1rhgxHiEQrooZ86hIna4nJHpbkJ-bipX5/view?usp=sharing>.
- [74] J. Wojtuszczyk and O. von Stryk. “Humod-a versatile and open database for the investigation, modeling and simulation of human motion dynamics on actuation level”. In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2015, pp. 74–79.

Appendix A

Matlab and Python Code for Theoretical Work

A.1 Linear Actuator Displacement Against Ankle Joint Angle Matlab Plot Code

```

1 clear; clc;
2 l = 22; j = 11, k = 80;
3 eta = 90 + acosd(l/k);
4 beta = atand(l/j);
5 n = (l^2 + j^2)^(0.5);
6 x = [0 : 0.5 : 90];
7
8 d = sqrt(n^2 + k^2 - 2* n * cosd(eta - beta - x));
9
10 plot(x, d);
11 title('Linear Actuator Displacement Against Ankle Joint Angle')
12
13 xlabel('Ankle Joint Angle')
14 ylabel('Linear Actuator Displacement')

```

A.2 Ankle Joint Vs Motor Joint angle variation Matlab Plot Code

```

1 clear; clc;

```

```

2 clear; clc;
3 l = 22; j = 11, k = 80;
4 eta = 90 + acosd(l/k);
5 beta = atand(l/j);
6 n = (l^2 + j^2)^(0.5);
7 x = [0 : 0.5 : 90];
8
9 d = sqrt(n^2 + k^2 - 2* n * cosd(eta - beta - x));
10
11 for i = 1:90;
12     lamda = asind((n/d(i)) * sind(90 + acosd(l/k) - asind(l/n) - x)) - asind(l/k);
13 end
14
15 plot(x, lamda)
16 title('Ankle Joint Vs Motor Joint angle variation')
17
18 xlabel('Ankle Joint Angle')
19 ylabel('Motor Joint angle')

```

A.3 Ankle Joint Vs Motor Joint angle variation Matlab Plot Code(Constant Crank Length Approach)

```

1 % This program used to make a relation between ankle angle(x) and motor
2 % angle(lamda)
3 clear; clc;
4 close all
5 l = 68; q = 262; j = 83;
6 % l is the crank length
7 % q is vertical distance between ankle joint to motor joint
8 % j vertical distance between ankle joint to crank joint
9
10 omega = atand(l/q);
11 beta = atand(l/j); gama = atand(l/q);
12 n = sqrt(l^2 + j^2); k = sqrt(l^2 + q^2);
13 x = [0 : 90];
14 d = sqrt(k^2 + n^2 - 2* n * k * cosd(180 - gama - beta - x));

```

```
15
16 for i=1:91;
17     lambda = asind((j * sind(x) + l* cosd(x) - 1)/d(i));
18 end
19 plot(x, lambda)
20 title('Ankle Joint Vs Motor Joint angle variation')
21
22 xlabel('Ankle Joint Angle')
23 ylabel('Motor Joint angle')
```

A.4 the natural range of Ankle Joint Vs Motor Joint angle variation Matlab Plot Code.

```
1 %Applying real Ankle Joint Angles
2 % This program used to make a relation between ankle angle(x) and motor
3 % angle(lamda)
4 clear; clc;
5 close all
6 l = 68; q = 262; j = 83;
7 % l is the crank length
8 % q is vertical distance between ankle joint to motor joint
9 % j vertical distance between ankle joint to crank joint
10
11 omega = atand(l/q);
12 beta = atand(l/j); gama = atand(l/q);
13 n = sqrt(1^2 + j^2); k = sqrt(1^2 + q^2);
14 x = [-27 : 10];
15 d = sqrt(k^2 + n^2 - 2* n * k * cosd(180 - gama - beta - x));
16
17 for i=1:38;
18     lambda = asind((j * sind(x) + l* cosd(x) - 1)/d(i));
19 end
20 plot(x, lambda)
21 title('Ankle Joint Vs Motor Joint angle variation')
22
23 xlabel('Ankle Joint Angle')
```

```
24 ylabel('Motor Joint angle')
```

A.5 the maximum force exerted by the linear actuator for the natural range of ankle joint angles. Matlab Plot Code.

```
1 %Anke angle VS Max force
2 % DrSalah Analysis
3 % Applying read ankle angles
4 % This program used to make a relation between ankle angle(x) and motor
5 % angle(lamda)
6 clear
7 clc
8 close all
9 l = 68; q = 262; j = 83;
10 % l is the crack length
11 % q is vertical distance between ankle joint to motor joint
12 % j vertical distance between ankle joint to crank joint
13
14 omega = atand(l/q);
15
16 beta = atand(l/j); gama = atand(l/q);
17 n = sqrt(l^2 + j^2); k = sqrt(l^2 + q^2);
18 x = [-27 : 10];
19 d = sqrt(k^2 + n^2 - 2 * n * k * cosd(180 - gama - beta - x));
20
21 for i = 1:38
22     lamda = asind((j*sind(x) + l*cosd(x)-l)/d(i));
23 end
24
25 maximumMomentInNmPerKg = 1.5*1000;
26 mass = 75;
27 maximumMoment = maximumMomentInNmPerKg * mass;
28 for i = 1:38
29     maxForce(i) = maximumMoment/(q*sin(lamda(i) + l*cos(lamda(i))));
```

```
30 end
31 plot(x,maxForce)
32 title('maxForce vs ankle angle for mass of 75kg')
33 xlabel('Ankle angle(deg)')
34 ylabel('max force(N)')
35 grid on
36 m = max(maxForce)
```

A.6 Lead screw length variation with the change of Z dimension(vertical distance from ankle joint to motor joint) given $J = 36.65$, $l = 60.14$. python Plot Code.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # this program is used to find the variation of Q(i.e. vertical distance
5 # between motor joint to ankle joint) with vairation of dt(lead screw movement)
6
7 # q variable
8 l = 60.14
9 j = 36.65
10 q = np.arange(1, 1000)
11 gama = np.zeros(999)
12 beta = np.zeros(999)
13 k = np.zeros(999)
14 n = np.zeros(999)
15 dtq = np.zeros(999)
16 # l is the crank length
17 # q is vertical distance between ankel joint to motor joint
18 # j is vertical distance between ankle joint to crank joint
19 x = np.arange(-27, 10) # ankle joint angle
20 for i in range(1, 1000):
21     beta[i-1] = np.degrees(np.arctan(l/j))
```

```

22     gama[i-1] = np.degrees(np.arctan(l/i))
23
24     n[i-1] = np.sqrt(np.square(j) + np.square(l))
25     k[i-1] = np.sqrt(np.square(i) + np.square(l))
26
27 for i in range(0, 999):
28     d = np.sqrt(np.square(k[i]) + np.square(n[i]) - 2*n[i]*k[i]*np.cos(np.radians(180-
29     dtq[i] = d[0] - d[len(d)-1]
30
31 plt.plot(q, dtq, color = 'blue', label = 'Ankle angles Vs Lead screw linear motion, ma
32 plt.grid()
33 plt.title('Lead screw length variation with the change of Z\n dimension( vertical disto
34 plt.xlabel('Z dimension( vertical distance from ankle joint to motor joint in mm)')
35 plt.ylabel('lead screw length in mm')
36 plt.show()

```

A.7 RMS python function..

```

1 1. def RMS(signal, windowsize):
2 2.     import math
3 3.     q = windowsize
4 4.     a = lst
5 5.     c = [None] * len(lst)
6 6.     for j in range(0, len(lst)):
7 7.         print(j)
8 8.         b = 0
9 9.         for i in range(j - int(q/2), j + int(q/2) + 1):
10 10.             if(i < 0 or i > len(lst)):
11 11.                 a[i] = 0
12 12.             if(i >= len(lst)):
13 13.                 a = np.append(a, 0)
14 14.                 b = b + (a[i])**2
15 15.
16 16.         b = b/(q+1)
17 17.         b = math.sqrt(b)
18 18.         c[j] = b
19 19.     return c

```

A.8 Low Pass Filter with Butterworth's order variation to plot figure 3.42

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Jun  8 13:51:10 2021
4
5 @author: AG
6
7 this code is used to text butterworth low pass filter equatoin
8 """
9
10
11 #importing the libraries
12 import numpy as np
13 import matplotlib.pyplot as plt
14 import pandas as pd
15 import time
16 import datetime
17 from scipy import signal
18 import math
19
20 w = np.arange(start=0, stop=1000, step=1)
21 vector = np.vectorize(np.int)
22 w = vector(w)
23 h = np.zeros(1000)
24
25 w_o = 500
26 a_o = 1
27
28 aa = w+1
29 h[0] = 1
30
31 n = 1
32
33 def plotLPF(nn):
```

```

34     for i in range(len(w)):
35         h[i] = math.sqrt((a_o)/(1+math.pow(w[i]/w_o, 2*nn)))
36
37     plt.plot(w,h, label = "n = "+ str(nn))
38     plt.title('Low pass Filter')
39     plt.xlabel('Frequency(Hz)')
40     plt.ylabel('H(w)')
41     plt.legend()
42     plt.show()
43
44
45 plotLPF(1)
46 plotLPF(5)
47 plotLPF(20)
48 plotLPF(100)

```

A.9 Data Matching Python Code

```

1
2 """
3 Date : 2021/04/20
4 This code is a trial to synchronized the Ankle joint angle data with EMG singal data
5 I'll used a linear interpolation for upsampling the ankle joint angle dataset
6 F:\Prosthetic\my work\footages\7.Regression Experiments(2021.4.2)\2021.4.17 me\data\exp
7 """
8 #importing the ibraries
9 import numpy as np
10 import matplotlib.pyplot as plt
11 import pandas as pd
12 import time
13 import datetime
14 from scipy import signal
15
16 # Importing the data set
17 df_angle = pd.read_csv('ESP32 2021-04-17-16-33-13.csv', delimiter = ",")
18 df_EMG = pd.read_csv('OpenBCI-RAW-2021-04-17_16-33-24.csv', delimiter = ";")
19

```



```
20 #for downsampled emg data
21 emg = df_EMG.iloc[:, [1, 2, 3, 4, 5]].values
22 # for non downsampled emg data
23 emg = df_EMG.iloc[4:., [1, 2, 3, 4, 13]].values
24
25 # Convert string numpy array to float
26 emg = emg.astype(np.float)
27
28 angle = df_angle.iloc[:, [3, 4, 5]].values
29
30
31 start_time = datetime.datetime(2021, 4, 17, 16, 33, 33)
32
33 end_time = datetime.datetime(2021, 4, 17, 16, 34, 37)
34
35 emg = TrimEMG(emg, start_time, end_time)
36 angle = TrimAngle(angle, start_time, end_time)
37
38 # delete all data before the start time of recording
39
40
41 emg[:, [0, 1, 2, 3, 4]] = emg[:, [4, 0, 1, 2, 3]] # put timestampe at first colomn
42
43 dataset = np.zeros([angle[:, 0].size, emg[0, :].size + angle[0, :].size + 3]);
44
45 for i in range(angle[:, 0].size - 1):
46     ind = closest(emg[:, 0], angle[i, 0])
47     print('i : ', i)
48     print('angle : ' , angle[i, 0])
49     print('ind : ', ind)
50     dataset[i, 0] = emg[ind, 0]
51     dataset[i, 1] = angle[i, 0]
52     dataset[i, 2] = emg[ind, 1]
53     dataset[i, 3] = emg[ind, 2]
54     dataset[i, 4] = emg[ind, 3]
55     dataset[i, 5] = emg[ind, 4]
56     dataset[i, 6] = angle[i, 1]
```

```

57     dataset[i, 7] = angle[i, 2]
58     emg = np.delete(emg, ind, 0)
59
60 dataset[:, 8] = dataset[:, 7] - 90
61 dataset[:, 9] = dataset[:, 6] - dataset[:, 8]
62 dataset[:, 10] = np.arange(start=0, stop=dataset[:, 0].size, step=1)
63
64 df = pd.DataFrame(data=dataset, columns=["emg timestamp", "esp timestamp", "ch_1", "ch_2"])
65 df.to_csv("Prosthetic/my work/EMG Regression/refined data/1.standing planterflexion(mol))")
66
67 def closest(lst, K):
68     lst = np.asarray(lst)
69     idx = (np.abs(lst - K)).argmin()
70     return idx
71
72 def TrimEMG(emg, start_time, end_time):
73     start_time_unix = time.mktime(start_time.timetuple())
74     end_time_unix = time.mktime(end_time.timetuple())
75
76     index_1 = np.where(emg[:, 4] >= start_time_unix)[0]
77     index_1 = index_1[0]
78
79     index_2 = np.where(emg[:, 4] >= end_time_unix)[0]
80     index_2 = index_2[0]
81     emg = emg[index_1:index_2, :]
82     return emg
83
84 def TrimAngle(angle, start_time, end_time):
85     start_time_unix = time.mktime(start_time.timetuple())
86     end_time_unix = time.mktime(end_time.timetuple())
87
88     index_1 = np.where(angle[:, 0] >= start_time_unix)[0]
89     index_1 = index_1[0]
90
91     index_2 = np.where(angle[:, 0] >= end_time_unix)[0]
92     index_2 = index_2[0]
93     angle = angle[index_1:index_2, :]

```

```
94     return angle
```

A.10 Data Trimming

```
1 def TrimEMG(emg, start_time, end_time):
2     start_time_unix = time.mktime(start_time.timetuple())
3     end_time_unix = time.mktime(end_time.timetuple())
4
5     index_1 = np.where(emg[:, 4] >= start_time_unix)[0]
6     index_1 = index_1[0]
7
8     index_2 = np.where(emg[:, 4] >= end_time_unix)[0]
9     index_2 = index_2[0]
10    emg = emg[index_1:index_2, :]
11    return emg
12
13 def TrimAngle(angle, start_time, end_time):
14     start_time_unix = time.mktime(start_time.timetuple())
15     end_time_unix = time.mktime(end_time.timetuple())
16
17     index_1 = np.where(angle[:, 0] >= start_time_unix)[0]
18     index_1 = index_1[0]
19
20     index_2 = np.where(angle[:, 0] >= end_time_unix)[0]
21     index_2 = index_2[0]
22     angle = angle[index_1:index_2, :]
23     return angle
```

A.11 Data Visualization Linear Regression and Feature Scaling

```
1
2
3 #2021.4.20 Visualizing data over time
4 plt.plot(dataset[:, 10], dataset[:, 2], label = "EMG Tibials Anterior")
5 plt.plot(dataset[:, 10], dataset[:, 3], label = "EMG Medial Gastrocnemius")
```

```
6 plt.plot(dataset[:, 10],dataset[:, 4], label = "EMG Lateral Gastrocnimis")
7 plt.plot(dataset[:, 10],dataset[:, 5], label = "EMG Soleus")
8 plt.plot(dataset[:, 10],dataset[:, 9], label = "Ankle Joint angle")
9 plt.title('Planter-Flexion While Standing')
10 plt.xlabel('Unix timestampe')
11 plt.ylabel('EMG, Ankle angle')
12 plt.legend()
13 plt.show()
14
15 #2021.4.20 Visualizing data over ankle angle
16 plt.scatter(dataset[:, 9],dataset[:, 2], label = "EMG Tibials Antirior")
17 plt.scatter(dataset[:, 9],dataset[:, 3], label = "EMG Medial Gastrocnimis")
18 plt.scatter(dataset[:, 9],dataset[:, 4], label = "EMG Lateral Gastrocnimis")
19 plt.scatter(dataset[:, 9],dataset[:, 5], label = "EMG Soleus")
20 plt.title('Planter-Flexion While Standing')
21 plt.xlabel('Ankle joint angle')
22 plt.ylabel('EMG')
23 plt.legend()
24 plt.show()
25
26
27
28 #2021.4.20 Trying regression for the first time
29 x = dataset[:, [2, 3, 4, 5]]
30 y = dataset[:, 9]
31
32 # Splitting the dataset into the Training set and Test set
33 from sklearn.model_selection import train_test_split
34 X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state=42)
35
36 # Fitting Multiple Linear Regression to the Training set
37 from sklearn.linear_model import LinearRegression
38 regressor = LinearRegression()
39 regressor.fit(X_train, y_train)
40
41 # Predicting the Test set results
42 y_pred = regressor.predict(X_test)
```

```

43
44 # Building the optimal model using Backward Elimination
45 import statsmodels.formula.api as sm
46
47 # Testing the regressor performance
48 from sklearn.metrics import mean_squared_error
49 test_set_rmse_Row = np.sqrt(mean_squared_error(y_test, y_pred))
50 from sklearn.metrics import r2_score
51 test_set_r2_Row = r2_score(y_test, y_pred)
52
53 # plotting y_test, and y_pred
54 yy =np.zeros([y_pred.size, 3])
55 yy[:, 0] = np.arange(start=0, stop=yy[:, 0].size, step=1)
56 yy[:, 1] = y_test
57 yy[:, 2] = y_pred
58 plt.plot(yy[:, 0], yy[:, 1], label = "Actual angle")
59 plt.plot(yy[:, 0], yy[:, 2], label = "Predicted angle")
60 plt.title('Actual-Predicted ankle joint angle')
61 plt.xlabel('o')
62 plt.ylabel('ankle angle')
63 plt.legend()
64 plt.show()
65 #-----
66 #2021.4.22 Feature scaling
67 from sklearn.preprocessing import StandardScaler
68 sc_X = StandardScaler()
69 dataset[:, [2, 3, 4, 5]] = sc_X.fit_transform(dataset[:, [2, 3, 4, 5]])
70 # Feature Scalling did not increase the regressor performance

```

A.12 Plotting EMG data in Frequency Domian Using FFT

```

1 #2021.6.20 Plot FFT
2 dataset = dataset[:-1, :] # remove last zero row
3 dt = (dataset[-1, 1] - dataset[0, 1])/(len(dataset[:, 0]) - 1) # Sampling time (time
4 t = np.arange(0, dataset[-1, 1] - dataset[0, 1] + dt, dt) # time

```

```

5
6 plt.plot(t, dataset[:, 2], LineWidth = 1.5, label = "Row Tibials antirior activity while
7 plt.xlim(t[0], t[-1])
8 plt.legend()
9
10 n = len(t)
11 fhat = np.fft.fft(dataset[:, 2], n) #Compute the FFT
12 PSD = fhat *np.conj(fhat) / n      #Power Spectrum (Power per frequency)
13 freq = (1/(dt*n)) * np.arange(n)  #Create x-axis frequencies
14 L = np.arange(1, np.floor(n/2), dtype='int')    #Only plot the first half
15
16 fig,axs = plt.subplots(2)
17
18
19 axs[0].plot(t, dataset[:, 2], color='b', LineWidth = 1.5, label = "Row Tibials antirior
20 #axs[0].xlim(t[0], t[-1])
21 axs[0].legend()
22
23 axs[1].plot(freq[L], PSD[L], color='c', LineWidth = 2, Label = 'Above singal in frequen
24 #axs[1].xlim(freq[L[0]], freq[L[-1]])
25 axs[1].legend()

```

A.13 LowPass Filter

```

1 def butter_lowpass_filter(data, cutoff, fs, order):
2     normal_cutoff = cutoff / nyq
3     # Get the filter coefficients
4     b, a = butter(order, normal_cutoff, btype='low', analog=False)
5     y = filtfilt(b, a, data)
6     return y

```

A.14 Feature Selection Using Correlation Matrix with Heatmap and Information gain - mutual information

```
1 #2021.7.1 Correlation Matrix with Heatmap Feature selection
2
3 import seaborn as sns
4 #get correlations of each features in dataset
5 corrmatrix = dataset.corr()
6 top_corr_features = corrmatrix.index
7 plt.figure(figsize=(20,20))
8 #plot heat map
9 g=sns.heatmap(dataset[top_corr_features].corr(),annot=True,cmap="RdYlGn")
10
11 #-----
12 # Results: I got channel channel_4 as the highest impact on ankle angle
13 #-----
14 #-----
15 #2021.7.3 Feature Selection-Information gain - mutual information In Regression
16 dataset.info()
17 dataset = dataset.drop(["Unnamed: 0", "emg timestamp", "esp timestamp", "foot angle"])
18
19
20 ### It is always a good practice to split train and test data to avoid
21 #overfitting
22 from sklearn.model_selection import train_test_split
23 X_train,X_test,y_train,y_test=train_test_split(dataset.drop(labels=['ankle ankle'],
24         dataset['ankle ankle'],
25         test_size=0.3,
26         random_state=0)
27
28
29 from sklearn.feature_selection import mutual_info_regression
30 # determine the mutual information
31 mutual_info = mutual_info_regression(X_train.fillna(0), y_train)
32 mutual_info
33
34 mutual_info = pd.Series(mutual_info)
35 mutual_info.index = X_train.columns
36 mutual_info.sort_values(ascending=False)
37
```

```

38 mutual_info.sort_values(ascending=False).plot.bar(figsize=(15,5))
39 #-----
40 # Results: I got channel channel_3 is the most matual data for perfoming regression
41 #-----

```

A.15 Gaussian Distribution Mean of a Set of Data

```

1 def Averaging(X, y):
2     ave = np.zeros([len(X[:, 0]), len(X[0, :])])
3     for j in range(len(X[0, :])):
4         for i in np.arange(min(y), max(y)+1, 0.01):
5             lst = np.where(y == round(i, 3))
6             ave[lst, j] = np.average(X[lst, j])
7     return ave

```

A.16 Optimized Data Regressions

```

1 def Medianing(X, y):
2     ave = np.zeros([len(X[:, 0]), len(X[0, :])])
3     for j in range(len(X[0, :])):
4         for i in np.arange(min(y), max(y)+1, 0.01):
5             lst = np.where(y == round(i, 3))
6             ave[lst, j] = np.median(X[lst, j])
7     return ave
8
9 def Moding(X, y):
10    from scipy import stats
11    ave = np.zeros([len(X[:, 0]), len(X[0, :])])
12    for j in range(len(X[0, :])):
13        for i in np.arange(min(y), max(y)+1, 0.01):
14            lst = np.where(y == round(i, 3))
15            ave[lst, j] = stats.mode(X[lst, j])[0]
16    return ave
17
18 def GaussianMean(X, y):
19    from sklearn.mixture import GaussianMixture
20    from scipy import stats

```



```

21     ave = np.zeros([len(X[:, 0]), len(X[0, :])])
22     for j in range(len(X[0, :])):
23         for i in np.arange(min(y), max(y)+1, 0.01):
24             lst = np.where(y == round(i, 3))
25             if(np.size(X[lst, j]) > 0):
26                 gm = GaussianMixture(n_components=1, random_state=0).fit(X[lst, j].r
27                 ave[lst, j] = gm.means_[0][0]
28     return ave
29
30 #2021.7.5 Finding Mean EMG at certain angle
31
32 emg = dataset.iloc[:, 3:7].values
33 emg = np.absolute(emg)
34 angle = dataset.iloc[:, 10].values
35 angle = np.absolute(angle)
36 angle = np.around(angle, decimals=2)
37 time = dataset.iloc[:, 1].values
38 time = time[:] - time[0]
39
40 emgAve = np.zeros([len(emg[:, 3]), 4])
41
42 for j in range(4):
43     for i in np.arange(0, 40, 0.01):
44         lst = np.where(angle == round(i, 3))
45
46
47
48
49
50 fig, axs = plt.subplots(5)
51 plt.xticks(np.arange(0, 70, 1))
52 fig.suptitle("All Row Data Plots with Averaged emg values at each angle x axis in se
53 axs[0].plot(time, angle)
54 axs[0].set_ylabel('Ankle Angle')
55 axs[0].set_xticks(np.arange(0, 70, 1))
56 axs[1].plot(time, emg[:, 0])
57 axs[1].plot(time, emgAve[:, 0])

```

```

58 axs[1].set_ylabel('Tibials Antirior')
59 axs[2].plot(time, emg[:, 1])
60 axs[2].plot(time, emgAve[:, 1])
61 axs[2].set_ylabel('Medial Gestrocnimis')
62 axs[3].plot(time, emg[:, 2])
63 axs[3].plot(time, emgAve[:, 2])
64 axs[3].set_ylabel('Lateral Gestrocnimis')
65 axs[4].plot(time, emg[:, 3])
66 axs[4].plot(time, emgAve[:, 3])
67 axs[4].set_ylabel('Solus')
68
69 # Cross Validation with Linear Regression
70 from sklearn.model_selection import cross_val_score
71 from sklearn.linear_model import LinearRegression
72 from sklearn.metrics import r2_score
73 regressor = LinearRegression()
74 r2_CV_Scores = cross_val_score(regressor, emgAve, angle, cv = 11, scoring = "r2")
75
76 # Predicting the Test set results
77 y_pred = regressor.predict(X_test)
78
79 from sklearn.metrics import r2_score
80 test_set_r2 = r2_score(y_test, y_pred)
81
82
83 # 2021.7.6 Ave data with cross validation and polynomial regression
84
85
86 from sklearn.preprocessing import PolynomialFeatures
87 poly_features = PolynomialFeatures(degree=4)
88 X_poly = poly_features.fit_transform(butter_lowpass_filter(emgAve[:, 2], cutoff, fs, order, nyq))
89 poly = LinearRegression()
90 r2_CV_Scores_poly = cross_val_score(poly, X_poly, angle, cv=11, scoring = "r2")
91
92 x_grid = np.arange(min(butter_lowpass_filter(emgAve[:, 2], cutoff, fs, order, nyq)), max(butter_lowpass_filter(emgAve[:, 2], cutoff, fs, order, nyq)),
93 x_grid = x_grid.reshape(len(x_grid), 1)
94 plt.scatter(butter_lowpass_filter(emgAve[:, 2], cutoff, fs, order, nyq), angle, color =

```

```
95 plt.plot(x_grid, LinearRegression.predict(poly_features.fit_transform(x_grid)), color='red')
96 plt.title('LPF of Averaged data')
97 plt.xlabel('Lateral Gastrocnemius')
98 plt.ylabel('Ankle Angle(deg)')
99 plt.show()
100
101 #-----
102 #Result : I got a greate r2 accurcy from averaged data, LPF and polynomial regression
103 #-----
104
105 #2021.7.7 Segmentation of the expermint
106
107 segments = [0, 6, 11, 19, 24, 29, 35, 40, 46, 51, 56, 61]
108
109 # 2021.7.10 Plotting All muscles with segments to get the best figure for illustration
110 def plot(muscleInd, segment, degree):
111     start = np.where(np.trunc(time) == segments[segment])[0][0]
112     end = np.where(np.trunc(time) == segments[segment+1])[0][0]
113
114     angle1 = angle[start:end]
115     emg1 = emgAve[start:end, muscleInd]
116
117     # Fitting Linear Regression to the dataset
118     from sklearn.linear_model import LinearRegression
119     # Fitting Polynomial Regression to the dataset
120     x = emg1.reshape(-1, 1)
121     y = angle1
122     from sklearn.preprocessing import PolynomialFeatures
123     poly_reg = PolynomialFeatures(degree=degree)
124     X_poly = poly_reg.fit_transform(x)
125     pol_reg = LinearRegression()
126     pol_reg.fit(X_poly, y)
127
128     step = (max(x) - min(x))/len(x)
129     xx = np.arange(min(x), max(x), step).reshape(-1, 1)
130
131     # Visualizing the Polyomial Regression results
```

```

132 plt.scatter(x, y, color='red')
133 plt.plot(xx, pol_reg.predict(poly_reg.fit_transform(xx)), color='blue')
134 plt.title('EMG-Ankle Angle Polynomial Regression\n order = ' + str(degree) + ', n
135 plt.xlabel('EMG')
136 plt.ylabel('Ankle Angle')
137 #plt.show()
138 dirName = 'C:/Users/AG/Desktop/regression/'
139 name = dirName + 'muscle ' + str(muscleInd) + ' segment ' + str(segment) + '.png'
140 plt.savefig(name)
141 plt.clf()
142 print('muscle ' + str(muscleInd) + ' segment ' + str(segment))
143
144
145
146 for i in range(0, 4):
147     for j in range(0, 11):
148         plot(muscleInd = i, segment = j, degree = 3)
149
150 #-----
151 #Result : I got the best illustration in muscle order 2 segment 1
152 #-----
153
154
155 #2021.7.11 Trying KNN regression
156
157
158 ave = Averaging(emg, angle)
159
160 # 2021.7.1 Ave data with cross validation and KNN regression
161 from sklearn import neighbors
162 from sklearn.model_selection import cross_val_score
163 from sklearn.metrics import r2_score
164 for k in range(10, 200, 10):
165     model = neighbors.KNeighborsRegressor(n_neighbors = k)
166     model.fit(butter_lowpass_filter(ave[:, 2],cutoff, fs, order, nyq).reshape(-1, 1), a
167     r2_CV_Scores_poly = cross_val_score(model, butter_lowpass_filter(ave[:, 2],cutoff,
168     print (k)

```

```

169     print(r2_CV_Scores_poly[1])
170     print('-----')
171
172 #-----
173 # Result : I got r2 score = .89 at k = 70
174 #-----
175
176 # Random Forest Regression
177 from sklearn.ensemble import RandomForestRegressor
178 regressor = RandomForestRegressor(n_estimators = 10, random_state = 0)
179 regressor.fit(butter_lowpass_filter(ave[:, 2], cutoff, fs, order, nyq).reshape(-1, 1))
180 r2_CV_Scores_poly = cross_val_score(regressor, butter_lowpass_filter(ave[:, 2], cutoff, fs, order, nyq).reshape(-1, 1), angle, cv=11, scoring = "r2")
181 #-----
182 # Result : I got r2 score = .78
183 #-----
184
185 #-----
186 #2021.7.11 Trying medianing
187 med = Medianing(emg, angle)
188
189 #Polynomial
190 from sklearn.preprocessing import PolynomialFeatures
191 from sklearn.model_selection import cross_val_score
192 from sklearn.linear_model import LinearRegression
193 from sklearn.metrics import r2_score
194 poly_features = PolynomialFeatures(degree=4)
195 X_poly = poly_features.fit_transform(butter_lowpass_filter(med[:, 2], cutoff, fs, order, nyq).reshape(-1, 1))
196 poly = LinearRegression()
197 r2_CV_Scores_poly = cross_val_score(poly, X_poly, angle, cv=11, scoring = "r2")
198
199 #KNN
200 from sklearn import neighbors
201 from sklearn.model_selection import cross_val_score
202 from sklearn.metrics import r2_score
203 for k in range(10, 200, 10):
204     model = neighbors.KNeighborsRegressor(n_neighbors = k)
205     model.fit(butter_lowpass_filter(med[:, 2], cutoff, fs, order, nyq).reshape(-1, 1))

```

```

206     r2_CV_Scores_poly = cross_val_score(model, butter_lowpass_filter(ave[:, 2], cutoff,
207     print (k)
208     print (max(r2_CV_Scores_poly))
209     print ('-----')
210
211
212 #-----
213 #2021.7.11 Trying mode
214
215 mode = Moding(emg, angle)
216 #Polynomial
217 from sklearn.preprocessing import PolynomialFeatures
218 from sklearn.model_selection import cross_val_score
219 from sklearn.linear_model import LinearRegression
220 from sklearn.metrics import r2_score
221 poly_features = PolynomialFeatures(degree=4)
222 X_poly = poly_features.fit_transform(butter_lowpass_filter(mode[:, 2], cutoff, fs, order, nyq).reshape(-1, 1),
223 poly = LinearRegression()
224 r2_CV_Scores_poly = cross_val_score(poly, X_poly, angle, cv=11, scoring = "r2")
225 print (max(r2_CV_Scores_poly))
226 #KNN
227 from sklearn import neighbors
228 from sklearn.model_selection import cross_val_score
229 from sklearn.metrics import r2_score
230 for k in range(10, 200, 10):
231     model = neighbors.KNeighborsRegressor(n_neighbors = k)
232     model.fit(butter_lowpass_filter(med[:, 2], cutoff, fs, order, nyq).reshape(-1, 1),
233     r2_CV_Scores_poly = cross_val_score(model, butter_lowpass_filter(ave[:, 2], cutoff,
234     print (k)
235     print (max(r2_CV_Scores_poly))
236     print ('-----')
237
238 #-----
239
240 #2021.7.12 Trying Gaussian smoothing fitler
241 from scipy.ndimage import gaussian_filter1d
242 g = gaussian_filter1d(emg[:, 2], 1)

```

```
243
244 sigma = 5
245
246 fig, axs = plt.subplots(5)
247 plt.xticks(np.arange(0, 70, 1))
248 fig.suptitle()
249 axs[0].plot(time, angle)
250 axs[0].set_ylabel('Ankle Angle')
251 axs[0].set_xticks(np.arange(0, 70, 1))
252
253 axs[1].plot(time, emg[:, 0])
254 axs[1].plot(time, gaussian_filter1d(emg[:, 0], sigma))
255 axs[1].set_ylabel('Tibials Antirior')
256 axs[2].plot(time, emg[:, 1])
257 axs[2].plot(time, gaussian_filter1d(emg[:, 1], sigma))
258 axs[2].set_ylabel('Medial Gestrocnimis')
259 axs[3].plot(time, emg[:, 2])
260 axs[3].plot(time, gaussian_filter1d(emg[:, 2], sigma))
261 axs[3].set_ylabel('Lateral Gestrocnimis')
262 axs[4].plot(time, emg[:, 3])
263 axs[4].plot(time, gaussian_filter1d(emg[:, 3], sigma))
264 axs[4].set_ylabel('Solus')
265
266
267 for sigma in range(1, 50):
268     #Polynomial
269     from sklearn.preprocessing import PolynomialFeatures
270     from sklearn.model_selection import cross_val_score
271     from sklearn.linear_model import LinearRegression
272     from sklearn.metrics import r2_score
273     poly_features = PolynomialFeatures(degree=4)
274     X_poly = poly_features.fit_transform(gaussian_filter1d(ave[:, 2], sigma).reshape(
275     poly = LinearRegression()
276     r2_CV_Scores_poly = cross_val_score(poly, X_poly, angle, cv=11, scoring = "r2")
277     print(max(r2_CV_Scores_poly))
278     print(sigma)
279     print('-----')
```

```

280
281 #KNN
282 from sklearn import neighbors
283 from sklearn.model_selection import cross_val_score
284 from sklearn.metrics import r2_score
285 for k in range(10, 200, 10):
286     model = neighbors.KNeighborsRegressor(n_neighbors = k)
287     model.fit(gaussian_filter1d(ave[:, 2], sigma).reshape(-1, 1).reshape(-1, 1), angle)
288     r2_CV_Scores_poly = cross_val_score(model, gaussian_filter1d(ave[:, 2], sigma).resl
289     print (k)
290     print (max(r2_CV_Scores_poly))
291     print ('-----')
292
293 #-----
294 # Using the mean of a single gaussian mixer for each emg sample at the same angle
295 Gmean = GaussianMean(emg, angle)
296
297 def GaussianMean(X, y):
298     from sklearn.mixture import GaussianMixture
299     from scipy import stats
300     ave = np.zeros([len(X[:, 0]), len(X[0, :])])
301     for j in range(len(X[0, :])):
302         for i in np.arange(min(y), max(y)+1, 0.01):
303             lst = np.where(y == round(i, 3))
304             if(np.size(X[lst, j]) > 1):
305                 gm = GaussianMixture(n_components=1, random_state=0).fit(X[lst, j].resl
306                 ave[lst, j] = gm.means_[0][0]
307     return ave
308
309 fig, axs = plt.subplots(5)
310 plt.xticks(np.arange(0, 70, 1))
311 fig.suptitle()
312 axs[0].plot(time, angle)
313 axs[0].set_ylabel('Ankle Angle')
314 axs[0].set_xticks(np.arange(0, 70, 1))
315
316 axs[1].plot(time, emg[:, 0])

```



```
317 axs[1].plot(time, Gmean[:, 0])
318 axs[1].set_ylabel('Tibials Antirior')
319 axs[2].plot(time, emg[:, 1])
320 axs[2].plot(time, Gmean[:, 1])
321 axs[2].set_ylabel('Medial Gestrocnimis')
322 axs[3].plot(time, emg[:, 2])
323 axs[3].plot(time, Gmean[:, 2])
324 axs[3].set_ylabel('Lateral Gestrocnimis')
325 axs[4].plot(time, emg[:, 3])
326 axs[4].plot(time, Gmean[:, 3])
327 axs[4].set_ylabel('Solus')
328
329 from sklearn.model_selection import cross_val_score
330 from sklearn.linear_model import LinearRegression
331 from sklearn.metrics import r2_score
332 regressor = LinearRegression()
333 from sklearn.preprocessing import PolynomialFeatures
334 poly_features = PolynomialFeatures(degree=4)
335 X_poly = poly_features.fit_transform(butter_lowpass_filter(Gmean[:, 2], cutoff, fs, o
336 poly = LinearRegression()
337 r2_CV_Scores_poly = cross_val_score(poly, X_poly, angle, cv=11, scoring = "r2")
338
339 #-----
340 # Result : I got the same results of using the average method
341 #-----
342
343 # 2021.7.15 Tring to get the highest accuracy using median value of all emg data at
344 # ankle angle and using the Gaussian smoothing fitler
345 med = Medianing(emg, angle)
346
347 #Polynomial
348 from sklearn.preprocessing import PolynomialFeatures
349 from sklearn.model_selection import cross_val_score
350 from sklearn.linear_model import LinearRegression
351 from sklearn.metrics import r2_score
352 from scipy.ndimage import gaussian_filter1d
353 poly_features = PolynomialFeatures(degree=4)
```

```
354 X_poly = poly_features.fit_transform(gaussian_filter1d(med[:, 2], 3).reshape(-1, 1))
355 poly = LinearRegression()
356 r2_CV_Scores_poly = cross_val_score(poly, X_poly, angle, cv=11, scoring = "r2")
357 print(max(r2_CV_Scores_poly))
```

Appendix B

Arduino for Experimental Work

B.1 Programming NodeMcu-32s to be Server, Access Point with OTA

```
1 //#include "WiFi.h"
2 #include <WiFi.h>
3 #include <WiFiClient.h>
4 #include <WiFiAP.h>
5 #include <ESPmDNS.h>
6 #include <WiFiUdp.h>
7 #include <ArduinoOTA.h>
8
9 const char* ssid = "ESP";
10
11 WiFiServer wifiServer(80);
12
13 void setup() {
14
15     Serial.begin(115200);
16     Serial.println("-----");
17     delay(1000);
18
19     Serial.println("Configuring access point...");
20
21     // You can remove the password parameter if you want the AP to be open.
22     WiFi.softAP(ssid);
```

```
23  IPAddress myIP = WiFi.softAPIP();
24  Serial.print("AP IP address: ");
25  Serial.println(myIP);
26  // Start OTA
27
28  ArduinoOTA
29    .onStart([]() {
30      String type;
31      if (ArduinoOTA.getCommand() == U_FLASH)
32        type = "sketch";
33      else // U_SPIFFS
34        type = "filesystem";
35
36      // NOTE: if updating SPIFFS this would be the place to unmount SPIFFS using SPIFFS
37      Serial.println("Start updating " + type);
38    })
39    .onEnd([]() {
40      Serial.println("\nEnd");
41    })
42    .onProgress([](unsigned int progress, unsigned int total) {
43      Serial.printf("Progress: %u%%\r", (progress / (total / 100)));
44    })
45    .onError([](ota_error_t error) {
46      Serial.printf("Error[%u]: ", error);
47      if (error == OTA_AUTH_ERROR) Serial.println("Auth Failed");
48      else if (error == OTA_BEGIN_ERROR) Serial.println("Begin Failed");
49      else if (error == OTA_CONNECT_ERROR) Serial.println("Connect Failed");
50      else if (error == OTA_RECEIVE_ERROR) Serial.println("Receive Failed");
51      else if (error == OTA_END_ERROR) Serial.println("End Failed");
52    });
53
54  ArduinoOTA.begin();
55  // OTA End
56  wifiServer.begin();
57 }
58
59 void loop() {
```

```
60  ArduinoOTA.handle();
61  WiFiClient client = wifiServer.available();
62
63  if (client) {
64      Serial.println("Client Connected ... 1");
65      while (client.connected()) {
66          while (client.available()>0) {
67              char c = client.read();
68              Serial.write(c);
69          }
70
71          while(Serial.available()>0){
72              char c = Serial.read();
73              Serial.print(c);
74              client.print(c);
75          }
76          client.println("Hello World");
77          delay(500);
78      }
79
80
81      client.stop();
82      Serial.println("Client disconnected");
83
84  }
85
86 }
```

B.2 Arduion Program for Collecting a Single MPU6050 Orientation While enabling Access Point, Local Server and OTA

```
1  // #include "WiFi.h"
2  #include <WiFi.h>
3  #include <WiFiClient.h>
```

```
4 #include <WiFiAP.h>
5 #include <ESPmDNS.h>
6 #include <WiFiUdp.h>
7 #include <ArduinoOTA.h>
8
9 #include "Wire.h"
10 #include <MPU6050_light.h>
11
12 MPU6050 mpu(Wire);
13 unsigned long timer = 0;
14
15 const char* ssid = "ESP";
16
17 WiFiServer wifiServer(80);
18
19 void setup() {
20
21   Serial.begin(115200);
22   // MPU6050 start
23   Wire.begin();
24
25   byte status = mpu.begin();
26   Serial.print(F("MPU6050 status: "));
27   Serial.println(status);
28   while(status!=0){ } // stop everything if could not connect to MPU6050
29
30   Serial.println(F("Calculating offsets, do not move MPU6050"));
31   mpu.calcOffsets(); // gyro and accelero
32   Serial.println("Done!\n");
33   // MPU6050 end
34   Serial.println("-----");
35
36
37   Serial.println("Configuring access point...");
38
39   // You can remove the password parameter if you want the AP to be open.
40   WiFi.softAP(ssid);
```

```
41  IPAddress myIP = WiFi.softAPIP();
42  Serial.print("AP IP address: ");
43  Serial.println(myIP);
44  // Start OTA
45
46  ArduinoOTA
47    .onStart([]() {
48      String type;
49      if (ArduinoOTA.getCommand() == U_FLASH)
50        type = "sketch";
51      else // U_SPIFFS
52        type = "filesystem";
53
54      // NOTE: if updating SPIFFS this would be the place to unmount SPIFFS using SP
55      Serial.println("Start updating " + type);
56    })
57    .onEnd([]() {
58      Serial.println("\nEnd");
59    })
60    .onProgress([](unsigned int progress, unsigned int total) {
61      Serial.printf("Progress: %u%%\r", (progress / (total / 100)));
62    })
63    .onError([](ota_error_t error) {
64      Serial.printf("Error[%u]: ", error);
65      if (error == OTA_AUTH_ERROR) Serial.println("Auth Failed");
66      else if (error == OTA_BEGIN_ERROR) Serial.println("Begin Failed");
67      else if (error == OTA_CONNECT_ERROR) Serial.println("Connect Failed");
68      else if (error == OTA_RECEIVE_ERROR) Serial.println("Receive Failed");
69      else if (error == OTA_END_ERROR) Serial.println("End Failed");
70    });
71
72  ArduinoOTA.begin();
73  // OTA End
74  wifiServer.begin();
75 }
76
77 void loop() {
```

```
78 ArduinoOTA.handle();
79 WiFiClient client = wifiServer.available();
80 if (client) {
81     Serial.println("Client Connected ... 1");
82     while (client.connected()) {
83         mpu.update();
84         if((millis()-timer)>10){ // print data every 10ms
85             client.print(mpu.getAngleX());
86             client.print("/");
87             client.print(mpu.getAngleY());
88             client.print("/");
89             client.println(mpu.getAngleZ());
90             timer = millis();
91         }
92     }
93     client.stop();
94     Serial.println("Client disconnected");
95 }
96 }
97 }
```

B.3 Arduion Program for Collecting a Multiple MPU6050 Orientation While enabling Access Point, Local Server and OTA Using Single Line Connection

```
1 //#include "WiFi.h"
2 #include <WiFi.h>
3 #include <WiFiClient.h>
4 #include <WiFiAP.h>
5 #include <ESPmDNS.h>
6 #include <WiFiUdp.h>
7 #include <ArduinoOTA.h>
8
```



```
9 #include "Wire.h"
10 #include <MPU6050_light.h>
11 MPU6050 mpu(Wire);
12
13 #include <MMPU.h>
14 MMPU mmpu(Wire);
15
16 unsigned long timer = 0;
17
18 const char* ssid = "ESP";
19
20 WiFiServer wifiServer(80);
21
22 void setup() {
23
24     Serial.begin(115200);
25     // MPU6050_1 start
26     Wire.begin();
27
28     byte status = mpu.begin();
29     Serial.print(F("MPU6050 status: "));
30     Serial.println(status);
31     while(status!=0){ } // stop everything if could not connect to MPU6050
32
33     Serial.println(F("Calculating offsets, do not move MPU6050"));
34     mpu.calcOffsets(); // gyro and accelero
35     Serial.println("Done!\n");
36     // MPU6050_1 end
37
38     Serial.println("-----");
39
40     // MPU6050_2 start
41     byte status_1 = mmpu.begin();
42     Serial.print(F("Second MPU6050 status: "));
43     Serial.println(status_1);
44     while(status_1!=0){ } // stop everything if could not connect to MPU6050
45
```

```
46 Serial.println(F("Calculating offsets, do not move Second MPU6050"));
47 delay(1000);
48 mmpu.calcOffsets(); // gyro and accelero
49 Serial.println("Done!\n");
50 // MPU6050_2 end
51 Serial.println("-----");
52 Serial.println("Configuring access point...");
53
54 // You can remove the password parameter if you want the AP to be open.
55 WiFi.softAP(ssid);
56 IPAddress myIP = WiFi.softAPIP();
57 Serial.print("AP IP address: ");
58 Serial.println(myIP);
59 // Start OTA
60
61 ArduinoOTA
62   .onStart([]() {
63     String type;
64     if (ArduinoOTA.getCommand() == U_FLASH)
65       type = "sketch";
66     else // U_SPIFFS
67       type = "filesystem";
68
69     // NOTE: if updating SPIFFS this would be the place to unmount SPIFFS using SPIFFS
70     Serial.println("Start updating " + type);
71   })
72   .onEnd([]() {
73     Serial.println("\nEnd");
74   })
75   .onProgress([](unsigned int progress, unsigned int total) {
76     Serial.printf("Progress: %u%%\r", (progress / (total / 100)));
77   })
78   .onError([](ota_error_t error) {
79     Serial.printf("Error[%u]: ", error);
80     if (error == OTA_AUTH_ERROR) Serial.println("Auth Failed");
81     else if (error == OTA_BEGIN_ERROR) Serial.println("Begin Failed");
82     else if (error == OTA_CONNECT_ERROR) Serial.println("Connect Failed");
```

```
83     else if (error == OTA_RECEIVE_ERROR) Serial.println("Receive Failed");
84     else if (error == OTA_END_ERROR) Serial.println("End Failed");
85   });
86
87   ArduinoOTA.begin();
88   // OTA End
89   wifiServer.begin();
90 }
91
92 void loop() {
93   ArduinoOTA.handle();
94   WiFiClient client = wifiServer.available();
95   if (client) {
96     Serial.println("Client Connected ... 1");
97     while (client.connected()) {
98       mpu.update();
99       mmpu.update();
100      if((millis()-timer)>10){ // print data every 10ms
101        client.print(mpu.getAngleX());
102        client.print("/");
103        client.print(mpu.getAngleY());
104        client.print("/");
105        client.print(mpu.getAngleZ());
106        client.print("|");
107        client.print(mmpu.getAngleX());
108        client.print("/");
109        client.print(mmpu.getAngleY());
110        client.print("/");
111        client.println(mmpu.getAngleZ());
112        timer = millis();
113      }
114    }
115
116
117    client.stop();
118    Serial.println("Client disconnected");
119  }
```

```
120 }
121 }
```

B.4 Unity3D Input Controller Script

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using System.Net.Sockets;
5 using System.IO;
6 using System.Runtime.InteropServices;
7 using System.Text;
8 using System.Threading;
9
10 public class InputController_Double : MonoBehaviour
11 {
12     public float x_1, y_1, z_1, x_2, y_2, z_2, Arangle;
13     public bool StateClient;
14
15     public void Begin(string ipAddress, int port)
16     {
17         //Give the network stuff its own special thread
18         var thread = new Thread(() =>
19         {
20             //This class makes it super easy to do network stuff
21             var client = new TcpClient();
22             //Change this to your real device address
23             client.Connect(ipAddress, port);
24             var stream = new StreamReader(client.GetStream());
25             //We'll read values and buffer them up in here
26             var buffer = new List<byte>();
27             StateClient = client.Connected;
28             print("StateClient : " + StateClient);
29             while (client.Connected)
30             {
31                 //Read the next byte
32                 var read = stream.Read();
```

```
33         //We split readings with a carriage return, so check for it
34         if (read == 13)
35         {
36             //Once we have a reading, convert our buffer to a string, since
37             var str = Encoding.ASCII.GetString(buffer.ToArray());
38             string[] MPUs = str.Split(',');
39             // print("MPUs.Length : " + MPUs.Length);
40             string[] MPU_1 = MPUs[0].Split('/');
41             //print("MPU_1.length : " + MPU_1.Length);
42             // string[] MPU_2 = MPUs[1].Split('/');
43             //print("MPU_2.Length : " + MPU_2.Length);
44             //x_1     = float.Parse(MPU_1[0]);
45             y_1     = float.Parse(MPUs[1]);
46             //z_1     = float.Parse(MPU_1[2]);
47
48             //x_2     = float.Parse(MPU_2[0]);
49             y_2     = float.Parse(MPUs[2]);
50             //z_2     = float.Parse(MPU_2[2]);
51             Arangle = float.Parse(MPUs[3]);
52
53
54             //Clear the buffer ready for another reading
55             buffer.Clear();
56         }
57         else
58             //if this was not the end of a reading, then just add this new b
59             buffer.Add((byte) read);
60     }
61     print("DisConnected");
62 });
63
64     thread.Start();
65 }
66 }
```

B.5 Unity3D Manager Script

```
1 using System;
2 using System.Collections;
3 using System.Collections.Generic;
4 using UnityEngine;
5 using UnityEngine.UI;
6
7 public class Manager_Double : MonoBehaviour
8 {
9     InputController_Double inputController;
10    [SerializeField] private Text MPU_1_text;
11    [SerializeField] private Text MPU_2_text;
12    [SerializeField] private Text Connection;
13    [SerializeField] private Text angle;
14    [SerializeField] private Text angleA;
15    [SerializeField] private GameObject MPU_1;
16    [SerializeField] private GameObject MPU_2;
17
18    private float x_1, y_1, z_1, x_2, y_2, z_2, Arangle;
19    void Start()
20    {
21        //This will do the network stuff
22        inputController = new InputController_Double();
23        inputController.Begin("192.168.4.1", 80);
24        Connection.text = "Connection : " + inputController.StateClient;
25    }
26
27    void Update()
28    {
29        x_1 = (float)Math.Round(inputController.x_1, 1);
30        y_1 = (float)Math.Round(inputController.y_1, 1);
31        z_1 = (float)Math.Round(inputController.z_1, 1);
32
33        x_2 = (float)Math.Round(inputController.x_2, 1);
34        y_2 = (float)Math.Round(inputController.y_2, 1);
35        z_2 = (float)Math.Round(inputController.z_2, 1);
36        Arangle = (float)Math.Round(inputController.Arangle, 1);
37
```

```
38 //     print("Foot Anlge : " + x_1);
39 //     print("Shank Anlge : " + x_2);
40     MPU_1_text.text = "MPU_1 // "+" x : " + x_1.ToString() + "\ty : " + y_1.ToSt
41     //MPU_1.transform.rotation = Quaternion.Euler(y_1, 0, x_1); // Activate thi
42     MPU_1.transform.rotation = Quaternion.Euler(0, 0, y_1);
43     MPU_2_text.text = "MPU_2 // "+" x : " + x_2.ToString() + "\ty : " + y_2.ToSt
44     //MPU_2.transform.rotation = Quaternion.Euler(y_2, 0, x_2); // Activate
45     MPU_2.transform.rotation = Quaternion.Euler(0, 0, 90 + y_2 );
46     angle.text = "Angle : " + (y_2 - y_1).ToString();
47     angleA.text = "Angle A : " + Arangle;
48 }
49
50
51 }
```


Appendix C

Extra

C.1 Surface EMG Appropriate Installation Position for Both Sagittal and Frontal Plans

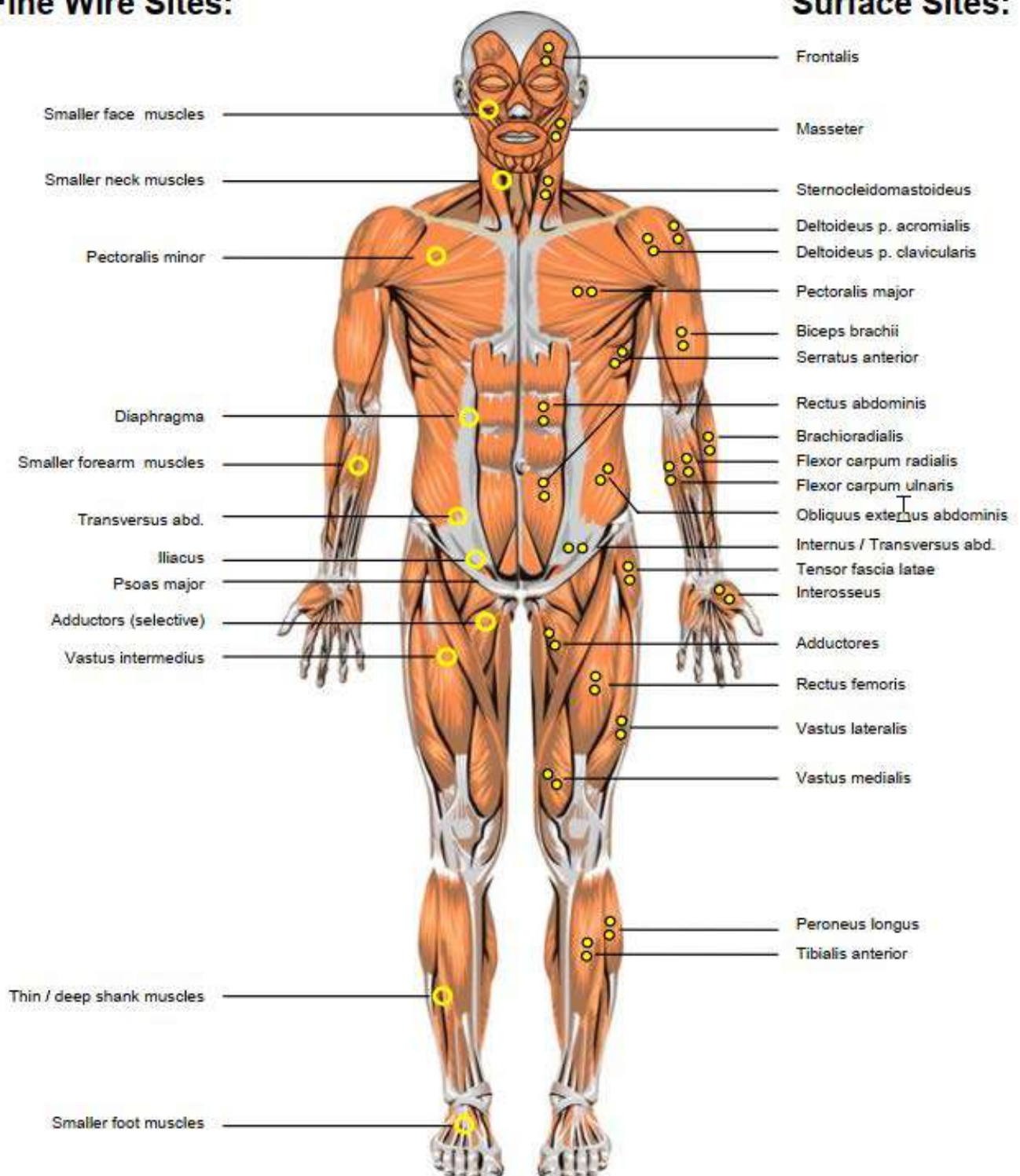
Fine Wire Sites:**Surface Sites:**

FIGURE C.1: Surface EMG Appropriate Installation Position for The Frontal Plan [17].

C.1. Surface EMG Appropriate Installation Position for Both Sagittal and Frontal Planes

Fine Wire Sites:

Surface Sites:

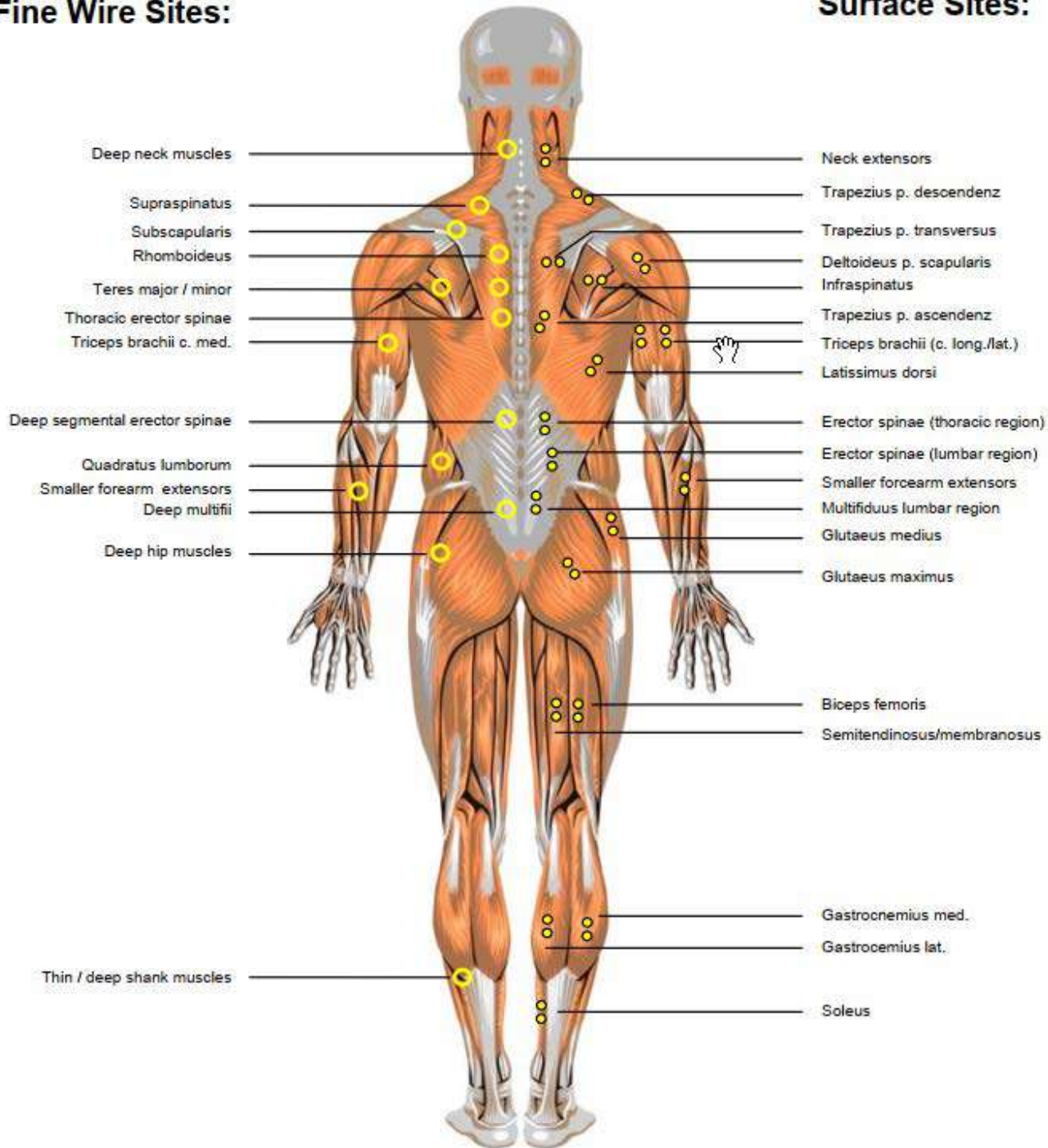


FIGURE C.2: Surface EMG Appropriate Installation Position for The Sagittal Plan [17].

C.2 Surface EMG Appropriate Installation Position for Both Sagittal and Frontal Plans

The maximum likelihood of the normal distribution is used to find the optimal value of μ and σ to give some data x . The peak in the likelihood graphs can be determined where the slope of the curve is equaled to zero. As known that the likelihood of the vector of variables is equal to the multiplication of each likelihood, as shown in equation C.1.

$$L(\mu, \sigma | x, \dots, x_n) = L(\mu, \sigma | x_1) \times \dots \times L(\mu, \sigma | x_n) \quad (C.1)$$

$$L(\mu, \sigma | x, \dots, x_n) = \frac{e^{-\frac{(x_1 - \mu)^2}{2\sigma^2}}}{(2\pi\sigma^2)^{\frac{1}{2}}} \times \dots \times \frac{e^{-\frac{(x_n - \mu)^2}{2\sigma^2}}}{(2\pi\sigma^2)^{\frac{1}{2}}} \quad (C.2)$$

To find the maximum likelihood, we should differentiate L two times, first with respect to μ and considering σ as Constant, then with respect to σ and considering μ as constant and as shown in the following derivation (equation X2 - Xn)

$$\ln[L(\mu, \sigma | x, \dots, x_n)] = \ln[L(\mu, \sigma | x_1) \times \dots \times L(\mu, \sigma | x_n)] \quad (C.3)$$

$$\ln L(\mu, \sigma | x_1) = \ln \frac{e^{-\frac{(x_1 - \mu)^2}{2\sigma^2}}}{(2\pi\sigma^2)^{\frac{1}{2}}} = \ln \frac{1}{\sqrt{2\pi\sigma^2}} + \ln e^{-\frac{(x_1 - \mu)^2}{2\sigma^2}} \quad (C.4)$$

$$\ln L(\mu, \sigma | x_1) = \ln(2\pi\sigma^2)^{-\frac{1}{2}} - \frac{(x_1 - \mu)^2}{2\sigma^2} \quad (C.5)$$

$$\ln L(\mu, \sigma | x_1) = \frac{-1}{2} \times \ln(2\pi\sigma^2) - \frac{(x_1 - \mu)^2}{2\sigma^2} \quad (\text{C.6})$$

$$\ln L(\mu, \sigma | x_1) = \frac{-1}{2} \ln(2\pi) - \frac{-1}{2} \ln(\sigma^2) - \frac{(x_1 - \mu)^2}{2\sigma^2} \quad (\text{C.7})$$

$$\ln L(\mu, \sigma | x_1) = \frac{-1}{2} \ln(2\pi) - \ln(\sigma) - \frac{(x_1 - \mu)^2}{2\sigma^2} \quad (\text{C.8})$$

$$\ln L(\mu, \sigma | x_n) = \frac{-1}{2} \ln(2\pi) - \ln(\sigma) - \frac{(x_n - \mu)^2}{2\sigma^2} \quad (\text{C.9})$$

$$\begin{aligned} \ln[L(\mu, \sigma | x, \dots, x_n)] &= \frac{-1}{2} \ln(2\pi) - \ln(\sigma) - \frac{(x_1 - \mu)^2}{2\sigma^2} + \dots \\ &\quad + \frac{-1}{2} \ln(2\pi) - \ln(\sigma) - \frac{(x_n - \mu)^2}{2\sigma^2} \end{aligned} \quad (\text{C.10})$$

$$\frac{\partial \ln(L)}{\partial \mu} = \frac{1}{2\sigma^2}(x_1 - \mu) + \dots + \frac{1}{2\sigma^2}(x_n - \mu) \quad (\text{C.11})$$

$$\frac{\partial \ln(L)}{\partial \mu} = \frac{1}{2\sigma^2}(x_1 - \mu) + \dots + \frac{1}{2\sigma^2}(x_n - \mu) \quad (\text{C.12})$$

$$\frac{\partial \ln(L)}{\partial \mu} = \frac{1}{2\sigma^2}(\sum_1^n x - n\mu) \quad (\text{C.13})$$

$$\frac{\partial \ln(L)}{\partial \mu} = 0 \longrightarrow \boxed{\mu = \frac{\sum_1^n x}{n}} \quad (\text{C.14})$$

$$\frac{\partial}{\partial \sigma}(\ln(L)) = \frac{-n}{\sigma} + \frac{(x_1 - \mu)^2}{\sigma^3} + \dots + \frac{(x_n - \mu)^2}{\sigma^3} \quad (\text{C.15})$$

$$\frac{\partial}{\partial \sigma}(\ln(L)) = \frac{-n}{\sigma} + \frac{1}{\sigma^3}[(x_1 - \mu)^2 + \cdots + (x_n - \mu)^2] \quad (\text{C.16})$$

$$\frac{\partial}{\partial \sigma}(\ln(L)) = 0 \iff n = \frac{1}{\sigma^2}[(x_1 - \mu)^2 + \cdots + (x_n - \mu)^2] \quad (\text{C.17})$$

$$\boxed{\sigma = \sqrt{\frac{\sum_1^n (x - \mu)^2}{n}}} \quad (\text{C.18})$$

الخلاصة

تعد الأطراف الأصطناعية من الأجهزة الضرورية في حياة المبتورين. بسبب ازدياد الحروب والكوارث و الأمراض زاد الطلب على الأطراف الأصطناعية بشكل ملحوظ. تجاريا تعد الأطراف غير المحركة من أكثر الأنواع أنتشارا، ولكن من مشاكلها انها لا تضيف طاقة أثناء الحركة كما هو في الأطراف السليمة التي تضيف طاقة نتيجة تقلصات العضلات. من مشاكل الأطراف غير المتحركة حدوث مشاكل على مستوى العمود القري مثل الأنزلاق و عدم أنسيابية المشي. في هذا العمل تم البحث عن النموذج الرياضي الأمثل و أعلى قوة واجب تنفيذها من قبل المحرك الخطي لأستدامة الحركة في مفصل الكاحل.

تم الأعتداع على موجات تقلصات العضلات للسيطرة على الطرف الأصطناعي. كما هو معروف أن الموجات العضلية غالبا ما تكون مصحوبة بتشوشات، لذلك تم أستعمال نموذج كاوسين لأجل تحسين أداء هذه الموجات. تم أستخدام نموذج أحصائي مستمر لأجل التنبؤ بالأستجابة المناسبة للطرف الأصطناعي. في هذا البحث تم التحقيق في أيجاد أفضل خوارزمية و أفضل مرشح موجات رياضي قادر على نمذجة موجات تقلصات العضلات، للحصول على أفضل أستجابة للطرف الاصطناعي. تمكن نموذج كاوسين من رفع أداء النموذج الأحصائي المستمر من ٥٥% الى ٨٢% مع أستعمال كل من مرشح الموجات الممر للترددات الواطئى و المعدل التربيعي للجذر مع خوارزمية الذكاء الأصطناعي متعددة الحدود.

فلسفة العمل هو في حال أيجاد النودج الأحصائي الأمثل لأستجابة الطرف الأصطناعي من خلال تدريب النموذج على أشخاص سليمين، هل من الوارد نقل نفس النموذج لذوي البثور و ماهي نسبة التعديل.

تم أستخدام معامل الأرتباط الخطي لأيجاد العضلة الأكثر تأثيرا في حركتي الأنتشاء الخمصي و العطف الظهري في مفصل الكاحل. تم التوصل الى أن عضلتي الكاستروكنيميس الوسطى و الجانبية هي أكثر العظلات تأثيرا على المفصل



جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة كربلاء- كلية الهندسة
قسم الهندسة الميكانيكية

تصميم و تحليل مفصل الكاحل الأسطناعي الذكي

رسالة مقدمة الى كلية الهندسة - جامعة كربلاء كجزء من متطلبات نيل درجة ماجستير
علوم في الهندسة الميكانيكية - ميكانيك تطبيقي

من قبل

محمد فهد جاسم كاظم الشمري

بكالوريوس ٢٠١٧

بإشراف

الأستاذ الدكتور محسن جبر جويج
الأستاذ المساعد الدكتور مرتضى عبدالمعین الهر