ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

HEAD GESTURE RECOGNITION FOR A SEMI-AUTONOMOUS POWERCHAIR

M.Sc. THESIS

Ubeyde MAVUŞ

Department of Control and Automation Engineering

Control and Automation Engineering Master's Programme

JUNE 2017

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

HEAD GESTURE RECOGNITION FOR A SEMI-AUTONOMOUS POWERCHAIR

M.Sc. THESIS

Ubeyde MAVUŞ
(504151133)

Department of Control and Automation Engineering

Control and Automation Engineering Master's Programme

Thesis Advisor: Asst. Prof. Dr. Volkan SEZER

JUNE 2017

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

YARI-OTONOM AKÜLÜ TEKERLEKLİ SANDALYE İÇİN KAFA HAREKETLERİ TANIMA

YÜKSEK LİSANS TEZİ

Ubeyde MAVUŞ
(504151133)

Kontrol ve Otomasyon Mühendisliği Anabilim Dalı

Kontrol ve Otomasyon Mühendisliği Yüksek Lisans Programı

Tez Danışmanı: Yrd. Doç. Dr. Volkan SEZER

HAZİRAN 2017

Ubeyde MAVUŞ, a M.Sc. student of ITU Graduate School of Science Engineering and Technology 504151133 successfully defended the thesis entitled "HEAD GESTURE RECOGNITION FOR A SEMI-AUTONOMOUS POWERCHAIR", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Asst. Prof. Dr. Volkan SEZER** Istanbul Technical University	.....
Jury Members :	**Asst. Prof. Dr. Janset Daşdemir** Yıldız Technical University	.....
	Asst. Prof. Dr. Tufan KUMBASAR Istanbul Technical University	.....

Date of Submission : **5 May 2017**
Date of Defense : **9 June 2017**

To my family; my mother, my father and my sisters
without their aid, I could not have finished my study in ITU in time.

FOREWORD

I would like to thank to my advisor Yrd. Doç. Dr. Volkan SEZER who helped me with the difficulties I had faced during my master's study, who also taught me how to prepare and execute scientific studies.

I also would like to thank to the Turkish Scientific and Technological Research Council (TUBITAK) which provided me with a stipend and supported the project I worked on financially under project no 215E140.

I also would like to thank to my friend Mustafa KUTSAL who helped proofreading the conference paper as well as this thesis, and my friend Ruhiyye KHASPOLADLI who helped proofreading the thesis.

Finally, I would like to thank to the 2017 CogSIMA Conference committee which awarded me with a modest amount of student travel grant which lessened my financial burden of presenting my study in the conference.

June 2017

Ubeyde MAVUŞ
(Control and Automation Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

CCS	**:** Cartesian Coordinate System
DFT	**:** Discrete Fourier Transform
DTFT	**:** Discrete-Time Fourier Transform
DTW	**:** Dynamic Time Warping
EMG	**:** Electromyography
FFT	**:** Fast Fourier Transform
HMM	**:** Hidden Markov Model
ICA	**:** Independent Component Analysis
IMU	**:** Inertial Measurement Unit
MEMS	**:** Microelectromechanical Systems
MTD	**:** Minimum Travelled Distance
RP	**:** Random Projection
SGONG	**:** Self-Growing and Self-Organized Neural Gas Network
SVM	**:** Support Vector Machine
TD Learning	**:** Temporal Difference Learning
WDTW	**:** Windowed Dynamic Time Warping
WT	**:** Wavelet Transform

LIST OF TABLES

Page

LIST OF FIGURES

Page

HEAD GESTURE RECOGNITION
FOR A SEMI-AUTONOMOUS POWERCHAIR

SUMMARY

The author had been involved in a project, where a semi-autonomous powerchair which has obstacle avoidance system for the paraplegic who can not use their hands properly, have been developed. In the project, the author was tasked to design a head gesture recognition system which would allow the end user to convey control commands to the chair so that he can have full autonomy. There are many different ways to communicate with a machine from using sound to using flexing of a muscle which is, by the way, this is how Stephen Hawking interacts with his chair. Ultimately, an inertial measurement unit has been chosen for the project. There are two simple reasons for this selection. One of which is that inferring intend of the user from the head orientation is very basic. For example, if the end user is leaning his head forward, that probably means "go forward", on the other hand; if he is leaning his head left, this probably means "turn left". The second reason is that the sensor needed for the task is affordable and easy to find. Design choices such as this, help to make the chair a low cost alternative to commercially available traditional powerchairs which require further engineering that is specific to the person who can not control it manually. For example, Stephen Hawking uses a specific muscle in his face to interact with his chair, which is impractical for others who can use their other limbs.

One may ask that why a semi-autonomous powerchair that uses head orientation as direction command, also needs a gesture recognition system. The answer may not be very obvious. However, there is a reason behind it. As mentioned in the previous paragraph, the end user needs full autonomy. For example; the user may want to cross a road. But the chaotic nature of crossing a road with a group of people, whose behaviour could not be easily predicted, may lead the chair's obstacle avoidance system to behave chaotically too. The chair may avoid humans towards an incoming car. Therefore, the user should be able to suspend the obstacle avoidance system. Or the user may be talking to his friends, and the chair may interpret one of his nods as a "go forward" command which may cause injuries or worse. Therefore, the user should also be able to prevent chair from listening to directional commands. For reasons as such, the semi-autonomous chair needs ways of communicating the intention of the user. This is achieved via head gesture recognition in the project.

There are many methods for gesture recognition in the literature. However, all wrap around several ideas which are data collection, feature extraction and decision making. All of these ideas are detailed in the chapters. However it is worth noting that most of the publications are about feature extraction. The author suspects that one of the reasons for this difference in the number of publications between the three in gesture recognition relate to the recent advancements of the computing technology where signal processing gained relative importance and it advanced relatively more rapidly due to the need created by consumers. Consequently, there exist more mathematical

tools for feature extraction, compared to the number of tools available for the other two.

Three different method have been studied in the thesis. These methods are named as; “Fast Fourier Transform as a Feature Extraction Method”, “Head Gesture Recognition via DTW and Threshold Optimization”, “Gesture Recognition via Neural Network”. The first method failed at recognition task.

The focus of the second method was directed towards decision making of the gesture recognition algorithm, reasons for which are discussed in detail. As a result of our studies, a method has been proposed for recognition algorithms where dynamic time warping is used for signal comparisons, to increase the recognition rate via threshold optimization. Even though the method has been developed to be used with dynamic time warping, the intuitive idea behind the method for threshold optimization which comes from geometry, can be extended to other gesture or pattern recognition problems.

In the proposed method, the optimization algorithm used was genetic algorithm. But anyone who is interested in replicating or implementing the work can use any other meta-heuristic optimization algorithm to optimize the thresholds.

The thresholds found through the method, which are calculated/optimized with very simple geometric shapes, have achieved, on average, success rate of 85% which means that more than eight out of ten repetitions of gestures are recognised successfully. Nonetheless, it is important to understand that this modest success rate can be easily increased towards 100% (if not 100%) by using more complex polygons for optimization.

The third and the last method have achieved around 97% success rate, however this success rate means there is a possibility of switching the mode of the chair from semi-autonomous to manual without the intention of the user.

YARI-OTONOM AKÜLÜ TEKERLEKLİ SANDALYE İÇİN KAFA HAREKETLERİ TANIMA

ÖZET

Bu tezde, ellerini düzgün bir biçimde kullanmakta zorlanan bel altı felçli olan insanlar için yarı otonom akülü tekerlekli sandalye geliştirmeyi hedefleyen bir TÜBİTAK projesinde kafa hareketlerini tanımak için kullanılabilecek mimik tanıma algoritmaları incelenmiştir.

Yer almış olduğum bu projede mimik tanıma algoritması geliştirmekle görevlendirildim. Mimik tanıma algoritması ile son kullanıcının akülü tekerlekli sandalyeye komut verebilmesi hedeflenmektedir, böylelikle son kullanıcının akülü tekerlekli sandalye üzerinde tam otonomi sahibi olması hedeflenmektedir. İnsan ve makina arasında çok çeşitli şekillerde etkileşim sağlanabilmektedir. Bu etkileşim ses, görüntü, kas hareketleri ve bunlar gibi bir çok farklı şekillerde olabilir. Projede bu etkileşimin atalet ölçüm ünitesi vasıtası ile olmasına karar verildi. Bu tercihin yapılmasında iki önemli etken bulunmaktadır.

Bu etkenlerden ilki, atalet ölçüm ünitesi vasıtası ile son kullanıcının niyetinin gayet açık bir biçimde anlaşılabilecek olmasıdır. Örneğin, kullanıcı öne hareket etmek istese, kafasını öne eğmesi bu niyetin bir göstergesi olarak kabul edilebilir. Veya kullanıcı sola dönmek isterse, kafasını sola döndürmesi bu komutun bilgisayar tarafından anlaşılması için yeterlidir. İkinci etken ise, atelet ölçüm ünitelerinin gelişen teknoloji ile ucuzlayıp yaygınlaşmasıdır.

Seçilen sensör tipi gibi tasarım kriterleri halihazırda üretilmekte olan standart akülü tekerlekli sandalyelere alternatif olarak geliştirilmesi planlanan yarı otonom akülü tekerlekli sandalyenin maliyetini azaltacak şekilde seçilmesiyle, elleri yardımı ile standart sandalyeleri kullanamayacak durumdaki ihtiyaç sahiplerine, kişiye özel modifikasyon gerektirdiği için maliyeti yüksek olan akülü tekerlekli sandalyelere daha ucuz bir alternatif getirilmesi hedeflenmektedir.

Kafa oryantasyonunu yarı otonom akülü tekerlekli sandalyeye yön vermek için kullanan bir sistemin neden bir de mimik tanıma sistemine ihtiyacı olduğu merak edilebilir. Bu ihtiyaç, son kullanıcıya tam otonomi sağlanması gerekliliğinden gelmektedir. Örneğin, son kullanıcının yolda karşıdan karşıya geçmesi gerektiğinde yarı otonom sistemin sahip olduğu engelden kaçma algoritması karşıdan karşıya geçmenin kaotik olduğu kavşaklarda sandalyenin kaotik davranmasına, kazalara yahut yaralanmalara neden olabilir. Bu sebeple kullanıcının sandalyenin yarı otonom sistemini geçici olarak durdurabilmesi gerekmektedir. Diğer bir örnek ise kullanıcının arkadaşlarıyla sohbet ettiği esnada sandalyenin doğal kafa hareketlerini yönelim komutu olarak algılamaası problemidir. Böyle bir durumda kullanıcı sandalyesini yanlışlıkla arkadaşının üzerine sürmesi ve kazalara neden olması büyük bir ihtimaldir. Böyle bir durumun önüne geçilebilmesi için kullanıcının sandalyeyi toptan kapatabilmesi gerekmektedir. Bu ve bunlar gibi sebepler nedeniyle, sandalyeye

yön verilmesi için kullanılan komutlar dışında, yarı-otonom sisteme kullanıcının niyetini aktarabileceği komutlara ihtiyaç vardır. Bu ihtiyacın sandalyeye entegre edilecek mimik tanıma algoritması karşılanması amaçlanmıştır.

Literatürde mimik tanıma üzerine çok çeşitli yayınlar ve çok çeşitli yöntemler bulunmaktadır. Bununla beraber, bütün literatür genelde üç fikir etrafında yoğunlaşır, bunlar; veri toplama, öznitelik çıkarma (feature extraction) ve karar verme. Bu aşamalar tezin bölümlerinde ayrıntılı olarak anlatışmıştır. Ancak belirtmekte fayda var ki literatürdeki yayınların geneli öznitelik çıkarmak üzerine yazılmıştır. Bunun sebebinin, tüketiciler tarafından dolaylı olarak oluşturulan bilgi işlem teknolojisinde sinyal işleme ihtiyacının yakın zamanda çip teknolojisinin gelişmesiyle beraber karşılanabilmesi ve sinyal işlemede kullanılan yöntemlerin mimik tanımada yeni ufuklar açması olduğunu düşünmekteyim. Bunun sonucu olarak, öznitelik çıkarmak için geliştirilen matematiksel yöntemlerin sayısının diğer iki aşama için geliştirilen yöntemlerin sayısından daha fazla olduğu söylenebilir.

Sinyal işleme algoritmaları sayesinde var olan sinyalin sahip olduğu özelliklerin incelenmesi kolaylaşmaktadır. Bu kolaylık mimik tanımada öznitelik çıkarma konusunda da kolaylık sağlar. Örnek olarak zaman serilerinin frekans düzleminde incelenmesi, zaman düzleminde incelenmesinden daha kolay olabilir.

Projede mimik tanımak algoritması için üç farklı yöntem denenmiştir. Bu yöntemleri sırası ile, Hızlı Fourier Dönüşümünün öznitelik çıkarmada kullanılması, kafa hareketlerini tanımak için dinamik zaman bükülmesi algoritmasının kullanılması ve bu algoritma için benzerlik eşik değerlerinin optimal olarak belirlenmesi ve yapay sinir ağları kullanarak kafa hareketlerinin tanınması, şeklinde adlandırmak mümkündür.

Mimik tanımada herhangi iki farklı mimiği birbirinden ayırmada kullanılabilecek her özellik, öznitelik olarak kullanılabilir. Bu bağlamda Hızlı Fourier Dönüşümünün mimik tanımada kullanılabileceği düşünülmüştür. Hızlı Fourier Dönüşümü, genellikle zaman serisi verilerinin frekans domaininde incelenmesi için kullanılır. Hızlı Fourier Dönüşümü ile bir zaman serisini oluşturan harmonikler elde edilir. Her farklı mimiğin farklı zaman serilerine sahip olacağı düşünülürse, elde edilecek harmoniklerin de farklı olması beklenilir. Frekans domaininde harmoniklerin karşılaştırılması, zaman domaininde zaman serisine ait örneklerin karşılaştırılmasından cebirsel olarak daha kolay olduğu düşünülürse, Hızlı Fourier Dönüşümü öznitelik çıkarmada ve böylelikle mimik tanıma algoritmasında kullanılabilir sonucuna varılır. Ancak yapılan deneyler sonucunda bu hipotezin tutarlı olmadığı görülmüştür. Buna sebep olarak sonsuz uzunluktaki zaman serilerinden mimiklerin başladığı ve bittiği yerlerin bir kesinlikle bulunması gösterilebilir. Sonsuz uzunluktaki zaman serisinin içerisinde herhangi bir mimiğin başladığı ve bittiği yerin belirlenmesi zor olduğu için, pencereleme yöntemi ile sonsuz uzunluktaki zaman serisinin önbelleğe alınan kısımları üzerinden hızlı fourier dönüşümü yaptık. Ancan hızlı fourier dönüşümü pencerelenen sinyalin periyodik olduğu varsayımı doğru kabul edilirse güvenilir/doğru sonuçlar vermektedir. Bahsi geçen pencerenin sonsuz uzunluktaki zaman serisi üzerinde kaymasından dolayı herhangi bir mimiğin sürekli olarak pencerenin aynı yerinde başlayıp, aynı yerinde bittiği garantilenemez. Dolayısı ile hızlı fourier dönüşümü alınan zaman serisi aynı mimiği içerse bile ilgili pencerede ön belleğe alınan zaman serisi farklı periyotlara sahip olacağı için, elde edilen dönüşüm, mimiğe ait zaman serisi birebir aynı kalsa bile farklı olmaktadır.

Dinamik zaman bükülmesi, iki zaman serisinin karşılaştırılmasında kullanılan bir algoritmadır. Bu algoritma giriş olarak iki zaman serisi alır ve iki sinyali birbiri üzerine optimal olarak oturtmaya çalışır. İki sinyalin birbiri üzerine oturtulması işlemi her iki sinyale ait örneklerin bir yakınlık ölçütüne göre birbirleri ile eşleştirilmesiyle yapılır. Bu algoritma çıkış olarak iki sinyalin birbiri üzerine en optimal olarak nasıl oturtulabileceği bilgisini içeren en kısa yol bilgisini, en kısa yola ait maliyet değerini, ve maliyet matrisi verir. En kısa yola ait maliyet değeri iki sinyalin benzerliği azaldıkça artmaktadır. İki sinyal birbiri ile tamamen aynı ise en kısa yola ait maliyet değeri sıfır olur. En kısa yola ait maliyet değeri için bir üst sınır bulunmamaktadır. Aynı kategoriye ait mimiklerin zaman serilerinin benzer olması beklenilir. Bu benzerlik aynı kategoriye ait mimiklerin farklı zaman serileri için farklılık göstermesiyle ile beraber mimiklerin aynı kategoriye ait olduğuna karar verebilmek için bir üst sınır bulunabileceği düşünülürse, dinamik zaman bükülmesi mimik tanıma için uygun bir algoritma olarak görülebilir. Ancak bu noktada, mimik tanıma problemi optimal eşik değer belirleme problemine dönüşmektedir. Bu problemin çözümü için önerdiğimiz metod ile benzerliğe karar vermek için gerekli olan eşik değerlerinin optimal olarak belirlenmesi mümkün olmaktadır. Bu methodu kullanarak yaptığımız deneylerde ortalama olarak %85 başarı oranı elde edilmiştir. Bu başarı oranı, veri kümesinde tanımlı olup on kere tekrar edilen her mimiğin sekizden fazla kez doğru şekilde tanınacağını/kategorilendirileceğini ifade etmektedir. Belirtmekte fayda var ki deneylerde elde edilen yüzde seksen beş başarı oranını methodda seçime bağlı parametre olarak ifade edilen eşik değerlerinin sınırlarını gösteren üç boyutlu cismin geometrik karmaşıklığı arttırılarak kolayca arttırılabilmektedir.

Yapay sinir ağları genel olarak giriş ve çıkış arasındaki ilişkinin kesin olarak ifade edilemediği durumlarda bu ilişkinin modellenebilmesi için kullanılır. Mimik tanımada giriş ve çıkışlar arasındaki ilişki de kesin olarak ifade edilemediğinden, yapay sinir ağları mimik tanımada kullanılmaya uygundur. Bu konuda literatürde çok çeşitli yapılar mevcuttur. Her bir yapay sinir ağı yapısı, zaman serilerinin farklı özelliklerini düşünülerek tasarlanmıştır. Önemli olan soyutlanmaya çalışılan özelliklerin yapay sinir ağının eğitilmesinin ardından ilgili ilişkinin (giriş ve çıkış arasındaki kesin ifade edilemeyen ilişki) genelleşmiş bir model verecek şekilde olmasıdır. Zaman serisinde her elemanın giriş ve çıkış arasındaki kesin olmayan ilişkinin modellenmesinde yararlı olabileceğinden, her noktanın yapay sinir ağına beslenmesi gereklidir. Kişilerin bir bir mimiği bire bir aynı şekilde tekrar etmesi mümkün değildir. Dolayısı ile aynı kişi bişe bir mimiği farklı zamanlarda tamamlayabilir. Bu sebeple zaman serilerinin sahip olduğu eleman sayısı kaydı alınan her mimik örneğinde fakrlı olabilmektedir. Kullanılan yapay sinir ağının giriş sayısı sabit olduğu için ortaya çıkan bu problemin giderilmesi gerekmektedir. Bu problemin giderilebilmesi için ilgili zaman serileri yeniden boyutlandırımalıdır. Bu yeniden boyutlandırma işlemi için zaman serilerinin zamanda sahip olduğu eleman sayısı aynı olacak şekilde zaman düzleminde boyutlandırıması gerekmektedir. Bu boyutlandırma işlemi yapılırken zaman serisinin sahip olduğu harmonik sayısının düşürülmemesi yahut olabildiğince az düşürülmesi gerekmektedir. Örneğin bir zaman serisinin tek bir noktaya indirgenmesi çok yüksek bilgi kaybı anlamına gelmektedir. Bu problemlerin çözümü için bir önişleme yapılması gerekmektedir. Bu önişlemden sonra elde edilen zaman serileri yapay sinir ağının eğitiminde kullanılabilir hale gelmektedir. Yaptığımız deneylerde bu yöntem ile %97 başarı elde edilmiştir. Ancak bu başarı, yapay sinir ağlarının yapısından ve projede gerçeklenme şeklinden mütevellit hatalı olarak mimik

tanınabileceğini ve istemsiz olarak yarı otonom moddan çıkıp manuel moda geçiş yapılabileceğini ifade etmektedir.

1. INTRODUCTION

Gesture recognition has been a popular topic in academia. It has also find its way into industry recently, especially through smart devices. It is generally used to facilitate daily activities such as shuffling an mp3 player, counting the number of steps taken in a day or shaking smart phone to switch on its screen. Besides these simple yet useful applications, it has also been a hot topic due to the fact that gesture recognition may provide more natural ways, as in being more intuitive to humans, for interaction with machines. Reasons as such make this topic not just academically interesting but also an important financial opportunity.

A semi-autonomous chair for paraplegic who can not use their hands properly, has been developed under a Turkish Scientific and Technological Research Council (TUBITAK) project where the chair also required gesture recognition. The chair has an obstacle avoidance algorithm, thus it is semi-autonomous. The purpose of the avoidance algorithm is helping the user with navigating through doors, halls, and rooms with ease. This way, the user does not have to worry about collisions which could occur if the chair had not have any avoidance algorithm.

Gesture recognition provides an intuitive alternative to traditional ways of interaction between humans and machines, which usually takes place in structured environments that requires a learning phase. It is especially important where the traditional ways of interaction with machines is not feasible.

The gesture recognition problem is separate from semi-autonomous navigation problem of the chair. Thus, only the gesture recognition problem on which the author worked, is the focus of this thesis.

1.1 Purpose of the Thesis

The purpose of the thesis is to walk the reader through the development process of the gesture recognition system for the chair, and to provide the proposed method's

formulation and its implementation. The research findings, and the provided method in this thesis has been published in 2017 IEEE CogSIMA Conference [1] under the title "Head Gesture Recognition via Dynamic Time Warping and Threshold Optimization". Those who may be interested can refer to the seven-page conference paper.

1.2 Literature Review

Gesture recognition plays an important role in human computer interaction [2]. This is because as the technology progress, people want more natural means of communication that is not verbal between computers and operators [3], to meet the needs in medicine and military [4], and in the field of robotics to develop more intimate relationship between robot and human [5].

In its simplest description, gesture recognition is estimation of human gestures, or any other living's, through data collected from sensors, via algorithms or mathematical models.

More than thirty different articles, publications or conference papers have been studied and the parts that are related to the research subject is noted. Since there are a lot of different mathematical methods and models to achieve the same goal. Similar methods in literature are grouped into three categories for clarity. There are roughly three types of categories based on sensory input type and methods with which the sensor data is used for gesture recognition.

1.2.1 Vision/Image based methods (Type I)

These methods usually use cameras as in [6] or Microsoft Kinect as in [7] as sensors, and consequently they involve image processing as a fundamental part.

One problem of image based recognition algorithms is that they are computationally heavy. Therefore, other methods are advised for real-time applications [8]. Image based methods usually require 0.5-1.5 seconds [8, 9] before recognition is achieved. This makes image based system inappropriate for real-time applications.

Another aspect worth mentioning is that image based systems can be divided further into two groups: pose detection and path detection [8]. Pose detection refers to the idea of recognition of hand pose such as one finger pointing or "OK" pose of a hand. Path

detection refers to the idea that the hand pose may be irrelevant or change for every person but gestures still can be inferred. An example for such action is waving hands.

In the papers that are related to image based methods, there are commonly 3 steps to achieve gesture recognition.

1.2.1.1 Image preprocessing

Image preprocessing refers to the idea of making the image easier for calculations. For example turning skin color to white and background to black [9] so that it is easier for computer to calculate hand, head or arm region on the image. There are different methods to achieve this goal such as filtering or applying thresholds to colors [3, 6].

1.2.1.2 Feature extraction

Feature extraction refers to the idea of finding properties of images that allows computers to identify and differentiate one gesture from another. This step allows templates which later is compared to the features of input images, for recognition to be small in size and reduces the computation time. (i.e. It is easier to find features of input images, then to compare those features to stored templates of features than comparing whole input image to a stored image.) There are many different methods for defining features. The methods of defining features include, but not limited to, using shape, color or brightness levels [3].

1.2.1.3 Recognition/Classification

Methods that are used to recognize features differ greatly. For instance, while [3] is based on Hidden Markov Models (HMM) and TD learning, in [6] geometric properties are used and; in [9] Self-Growing and Self-Organized Neural Gas Network (SGONG) is used. In others, SVMs (Support Vector Machine) are used for classifications [8]. These methods are important because they have mathematical foundations which can be used for any type of numerical data, even when the sensory input is different than images.

1.2.2 Acceleration/Gyration based methods (Type II)

These methods use electronic units such as MEMS gyroscopes (Micro-electromechanical systems) [10–13] or IMU's (Inertial Measurement

Units) [5, 14–16] as sensors. Raw sensor data is usually noisy, thus filters are employed [10, 12, 13]. These methods can also be divided into several groups based on the way the problem is approached.

One group of methods use similar steps as in vision based systems. They first filter the signal [10, 12, 13] and make a gesture segmentation [12, 15] on windowed infinite input sequence. Then features are extracted [12]. Templates are formed [10, 12]. Then by either Artificial Neural Network [16] or other methods such as Hopfield Networks based on Sign Sequence [12] or Similarity Matching [12, 13] is used for recognition. One major difference from image based system is that these methods use acceleration or position in 3D space of a hand rather than the hand pose itself.

Another group of methods use mappings between input signal and the stored template signals such as Dynamic Time Warping (DTW) [10], Windowed DTW (WDTW) [14], Random Projection (RP) [10], HMM, Wavelet Transform (WT), Discrete Fourier Transform (DFT) and Fourier Descriptors [16]. Even though WT and DFT can be used for feature extraction, it is also possible to use comparison of coefficients of harmonics for recognition. On the other hand, HMM still requires feature extraction first, and then training before it can be used for time series data.

1.2.3 Electromyography (EMG) based methods (Type III)

These methods are very similar to acceleration/gyration based methods in input type, they are both time series signals [14, 17]. Therefore recognition techniques that are used for acceleration/gyration based methods are applicable for EMG based methods. Combination of the two type of sensors is not uncommon in academia [5]. However there are different problems that needs to be addressed in EMG based methods.

Electromyography is a technique for measuring muscle activity from the surface of skin via electrodes [3, 5, 17, 18]. One major problem of the subject is that electrodes catch electrical muscle activity from all neighbouring muscles and not just the target muscle [3, 18]. Thus, to increase accuracy, more than one electrode which make the instrument hard to wear and carry, may be used [5]. There are some methods that address this issue such as ICA algorithms [3] which tries to separate the target source from other sources (other muscles) mathematically. Also noise in measurement is always an issue for which countermeasures are taken such as filters [17, 18].

Our main concern in the project relates to the recognition methods that are used. The methods are similar to other vision based and acceleration/gyration based recognition methods. Artificial neural networks [17,18], HMMs [5], and other signal classification based methods are used for recognition.

1.3 Challenges in Gesture Recognition

In a broad sense, gesture recognition is the mapping between two sets, a meaning/intend set and a data set. Any mapping method can be accepted as a recognition technique regardless of the requirements of an application where it is to be implemented. However, success criteria of the method should depend on the application for practicality. For example, if the relevant set of meanings consists of several elements and the mapping from the data set to the meaning set is randomized, then the recognition algorithm is legitimized yet, it is impractical in day-to-day life. Nonetheless, the recognition algorithm can be accepted as quite successful if the objective is randomized-mapping. Similarly, success of mapping methodology can be quite affected by the number of features the raw data has, if the raw data is a flat line where there is only one feature to be recognized, then 100% success rate can be claimed disregarding its practical use. For reasons as such, the success of a gesture recognition system depends on more parameters (such as representation of gestures in raw data) than just its mapping method. Therefore, standardized benchmarking datasets are irrelevant in determining success of a gesture recognition algorithm where it is developed for a specific application which has additional requirements and restrictions. Because the standardized benchmarking datasets only has limited number of features, a recognition algorithm's success may change drastically with respect to different datasets which may have different features to be recognised. It is also important to note that anything can be defined as a feature as long as it helps making distinction between gestures. This broad definition of features creates a wide spectrum whose limits can not be known. For example, the ratio between the biggest derivative in the first half of a gesture and the biggest derivative in the second half, may help making distinction between gestures if the gestures in a dataset happens to have such features.

The subsections explain general problems and challenges that needs to be overcome in gesture recognition for successful implementation. Typically, one needs to compromise between these challenges while catering a solution for a specific problem. For example, accuracy may suffer if the algorithm works very fast due to loss of spatial information. Or the accuracy may be very high but, the algorithm produces results very slowly due to computational cost created by usage of multiple mathematical algorithms/models.

1.3.1 Computational cost

Computational cost can be defined as the amount of work that needs to be done to achieve recognition. The lesser it is, the more practical the implementation of it becomes. Fast computation is also important for real-time applications. Recognition algorithms that takes two or three seconds are inappropriate for real-time applications where time is of the essence.

1.3.2 Accuracy

Accuracy refers to the rate of confusion between the gestures that the recognition algorithm yields. The algorithm is accepted to be accurate, if the confusion rate is close to zero 0%. It is accepted to be inaccurate if the confusion rate is close to 100%. There is not a formal percentage limit for accuracy. This is a definition which is made by the author to facilitate understanding of the subject.

1.3.3 User independence

User independence refers to the idea that the success of a recognition algorithm should be independent of the individual who is using it. For gesture recognition algorithms to be commercially viable, they need to be user independent.

1.3.4 Complexity of gestures

Complexity of gestures is important in applications where the gestures signal a command. The gestures should be complex to prevent the machine from interpreting gestures wrong, which if not prevented can result in simple accidents at best and injuries or death at worst. Some degree of complexity requirement forces algorithms

to use as much information as possible. This restriction can force engineers to increase computational cost in order to achieve eligible accuracy.

1.4 General Steps for Gesture Recognition

Gesture recognition is usually achieved in six steps that are data collection, preprocessing, feature extraction, comparison to a database and output, namely; decision making. Different methods may leave some steps out, or combine some steps examples of different workflows can be seen in [2, 5, 8, 12, 13, 19, 20]. In the following subsections each step is briefly described and some of the problems relating to each step is given.

1.4.1 Data collection

Data collection is the same for almost all recognition algorithms, that is gathering of data from sensors. The data collection largely depends on the sensor type that is used. If the sensor is reliable, there is usually no problems to be addressed in this step. Some sensors may have integral filters, preprocessor for ease of use of the raw data. After the data is collected by the sensor, it is then sent to a computer or a computing element such as a microprocessor for further processing. Sending the data is usually done through standard data transmission architectures such as RS-232, CAN and I^2C1001[21].

1.4.2 Preprocessing

The collected data is mostly inappropriate for feature extraction. Or it is not ideal in some sense. It may be noisy, which is almost always the case. it may include information that is not useful for the algorithm. The main purpose of this step is to prepare the data for feature extraction. Noise in the data can be filtered. Unnecessary information in the data is eliminated such as turning colored image to black and white, normalization and resizing.

1.4.3 Feature extraction

Feature extraction refers to the idea that a gesture has some specific properties which make the gesture distinct from other gestures. This step of gesture recognition is much more trickier than other steps due to the fact that anything can be a feature if it fulfils the

requirement of helping making distinctions between gestures. For example, gestures to be recognised can be defined in radically different ways namely, it is easy for humans to notice differences by inspection. This way, it would be obvious what should be a feature or not. But in an example of a very comprehensive gesture recognition task where the gestures look alike, it is not very clear about what properties should be features or count as features.

It is also not clear whether the total number of features exist in a gesture is more important than their combinations or not. As an example; if ten features are identified for a specific gesture. And if the incoming data from the sensor only carries nine out of ten of them. The question of whether the incoming data should be recognised as a gesture or not, has to be answered.

Another important question that should be answered is that, whether one can identify all features or not.

All of the questions above makes the feature extraction step of the gesture recognition problem more interesting than others. This may be one of the reasons why feature extraction is the most popular amongst scholars.

1.4.4 Comparison to database

In this step, features extracted from the defined gestures form a database where important informations about features are stored. This database is used as template for gesture recognition from the incoming data in the experiment phase.

1.4.5 Decision making/Output

This step usually relates to the comparison to database step. Similarity is measured in this step thus, whether a gesture is recognised or ignored is decided here. Depending on the similarity between gestures templates (collection of features) and incoming data, the input is categorised into one of the gesture types or not.

1.5 The Following Chapters

The second chapter gives a problem statement where the general application setup is given, gestures to be recognized are introduced, hardware and software that are

used in the project are detailed. In the following chapters, three different methods used for recognition are explained. The third chapter gives details of our first failed attempt at creating a new feature extraction method to implement it into gesture recognition algorithm. It gives the hypothesis, explains the experiment process, and a conclusion where it is explained that why the hypothesis did not hold. In the fourth chapter, the method that has been presented in the IEEE 2017 CogSIMA conference, which optimizes thresholds for similarity where dynamic time warping is used for comparing two time series, is provided. The chapter gives details of the method, why it is necessary, collection process of the required data for optimization, optimization details, and the results. The fifth chapter explains another method where a simple neural network for gesture recognition is used, finally results are provided. In the last chapter, an overall conclusion is given.

2. PROBLEM STATEMENT

The overall aim of the project had been the development of a semi-autonomous powerchair that is controlled via head commands, which can also avoid obstacles if needed. The chair required gesture recognition, reasons for which given previously, therefore, it is important to understand requirements of such application before creating a recognition algorithm.

The head orientation is selected to be the input for the recognition algorithm. Thus, the recognition algorithm has to process and interpret time series data into categories that indicate the operation mode of the chair. Four different gestures are chosen that correspond to four different switching commands for interaction with the chair. The operation modes of the chair are named as “OPEN”, “CLOSE”, “SUPPORT” and “NO-SUPPORT”, respectively. All of which are explained and defined in the following subsections. From this point on, these capitalized names will refer to both the mode-switching-commands and gesture categories in the way they are described.

The chair is to be used by paraplegic who can not use their hands properly. This is an important starting point to understand the requirements. Since it can not be expected from the user to protect himself by means of his hands if an accident occurs, the recognition algorithm does not have the luxury of interpreting mode-switching-commands as directional commands. It is even better to interpret nothing than interpreting a command wrong. This forced the recognition algorithm to be highly complex with respect to the selection of gestures for mode-switching-commands.

2.1 Gestures

It is important to note that the time series representation of a gesture is independent of its meaning as a command. Because any mapping between commands and gestures can be chosen as desired. Ultimately, time series representations of gestures are design choices. However, due to the requirements of the application, the time series

representations of gestures should be complex enough as a countermeasure for wrong interpretation. In the meantime, they need to be simple enough for ease of use.

Shapes of all of the gestures are decided, after some consideration according to the requirements. There were other shapes that were considered and eliminated. However they are not listed in subsections.

2.1.1 OPEN gesture/command

Gesture OPEN is meant to start the autonomous nature of the chair. It switches the mode of the chair from CLOSE to OPEN. In this mode, the chair moves with respect to head orientation while avoiding obstacles. This mode allows users to navigate through doors, halls, and rooms with ease. The user does not have to worry about crashing into walls or any other dynamic object that may be around of him. Leaning the head forward makes the chair move forward. Leaning the head left makes the chair turn left.

Switching to any other mode is possible in this mode. But the chair does not try to compare sensor input to SUPPORT gesture template because, it would be an unnecessary comparison due to the fact that SUPPORT command starts the autonomous nature of the chair after NO-SUPPORT command has been given. The reason for defining two different gestures for switching to autonomous mode is to make sure the user intended to give the commands. If the chair is in CLOSE mode, then the gesture to be recognised has to be significantly different than any natural movement of the head so that natural head movements are not recognised as a command by mistake.

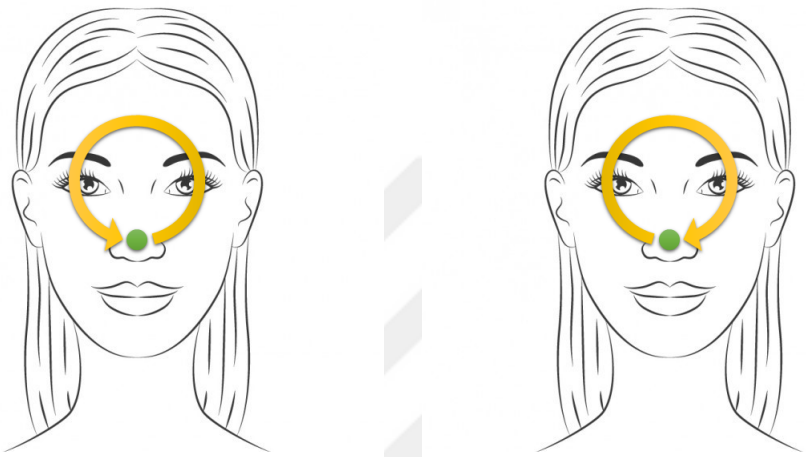
It has been decided that moving the head in clockwise direction once is complex enough and yet, simple enough for ease of repetition. To perform this gesture, one has to draw a circle with their nose starting from the natural position of the head and arriving to it to complete drawing the circle in clockwise direction. One starts to draw the circle from the bottom of it (see Figure 2.1a).

2.1.2 CLOSE gesture/command

Gesture CLOSE is meant to stop the chair from listening to any command including directional commands except OPEN command. It switches the mode of the chair to CLOSE. In this mode, the chair does not move with respect to head orientation. This

mode allows users to be free from worrying whether the chair will interpret one of his head movements as a command or not.

It has been decided that moving the head in counter clockwise direction once is complex enough and yet, simple enough for ease of repetition. To perform this gesture, one has to draw a circle with their nose starting from the natural position of the head and arriving to it to complete drawing the circle in counter clockwise direction. One starts to draw the circle from the bottom of it (see Figure 2.1b).



(a) Gesture OPEN. (b) Gesture CLOSE.

Figure 2.1 : Illustration for gesture OPEN and CLOSE.

2.1.3 SUPPORT gesture/command

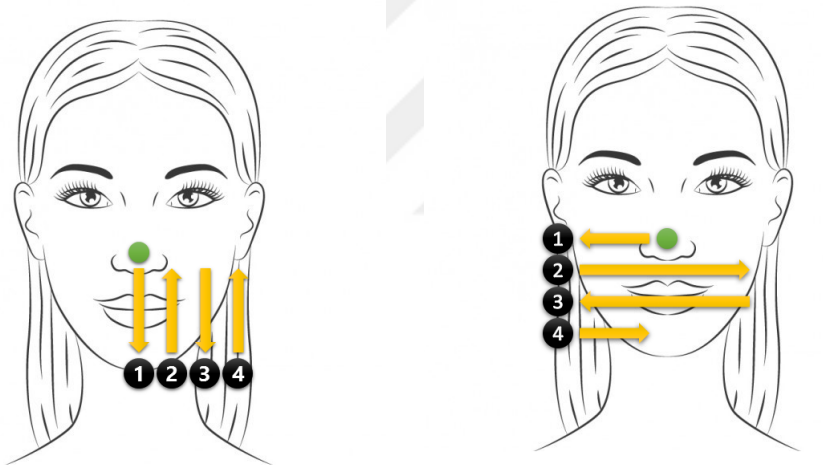
Gesture SUPPORT is meant to start the autonomous nature of the chair after NO-SUPPORT command has been given. It switches the mode of the chair to SUPPORT. In this mode, the chair moves with respect to head orientation while assisting the user in navigating.

It has been decided that moving the head downwards then upwards twice is complex and simple enough for ease of repetition. To perform this gesture, one has to draw a line with their nose starting from the natural position of the head, going downward, then, arriving back to its original position to complete drawing the line. This is done twice to signify a command has been issued. One starts drawing the line from the top of it (see Figure 2.2a).

2.1.4 NO-SUPPORT gesture/command

Gesture NO-SUPPORT is meant to stop the obstacle avoidance algorithm from interfering with directional commands. It switches the mode of the chair to NO-SUPPORT. In this mode, the chair moves with respect to head orientation however, it does not avoid obstacles.

It has been decided that moving the head to right, to then, to left, then to right and finally to original position is complex and simple enough for ease of repetition. To perform this gesture, one has to draw a line with their nose starting from the natural position of the head to complete drawing the line. This is done twice to signify a command has been issued. One starts drawing the line from the middle of it (see Figure 2.2b).



(a) Gesture SUPPORT. (b) Gesture NO-SUPPORT.

Figure 2.2 : Illustration for gesture SUPPORT and NO-SUPPORT.

2.2 Hardware and Software

A commercial powerchair has been modified to implement the obstacle avoidance and the gesture recognition algorithm. Basically, the manual control panel has been removed and a computer has been connected to it. Then sensors are placed into the chair and they are connected to computer where the data coming from them can be processed.

As the software, Robot Operating System [22] is chosen to implement the algorithms on Ubuntu [23]. ROS allows programmers to write individual Python [24] or C++ [25] programs where each of these programs can communicate with each other easily trough ROS's model.

2.2.1 Chair

After the manual control panel is removed, additional softwares had to be implemented so that the chair's drivers can listen to the computer's commands. The software for these drivers has been written in Python programming language by the seller of the chair.

The driver software basically allows our algorithms to send directional commands to the chair's motors as if the computer is the manual control stick of the chair.

The sensors for the avoidance algorithm has been fixed under the seat. The IMU sensor however is not attached to the chair. It is attached the head of the user and it connect to the computer via USB cable. The chair can be seen in Figure 2.3

Figure 2.3 : The modified powerchair.

2.2.2 Sensor

The sensor used for capturing head orientation is an off the shelf IMU, named UM7-LT Orientation Sensor [26] (see Figure 2.5). The sensor contains three-axis accelerometer, rate gyro, magnetometer, and thermometer. It has an integrated extended kalman filter for roll, pitch and yaw data. The sensor uses UART [21] architecture to convey sensor data. In order to connect the sensor to a computer, a serial-to-usb converter is needed. The converter is also an off the shelf circuit that uses FTDI chips [27]. The ROS has a driver for the sensor with which the data can be transferred to the algorithms with ease. The relevant data is sent to computer in packets. The frequency with which the data packets arrive to computer, changes, however not significantly. The frequency is approximately eighty hertz. The sensor does not need an external battery, it can be powered via the USB cable.

The sensor is placed onto a head gear which is worn by the user (see Figure 2.4). Through the USB cable, head orientation data that are roll, pitch and yaw, is sent to the computer. Then, the data is transferred to algorithms where the gesture recognition is achieved. All of this process is automated by the manufacturer of the sensor and the ROS. The orientation data given by the sensor is in radians.

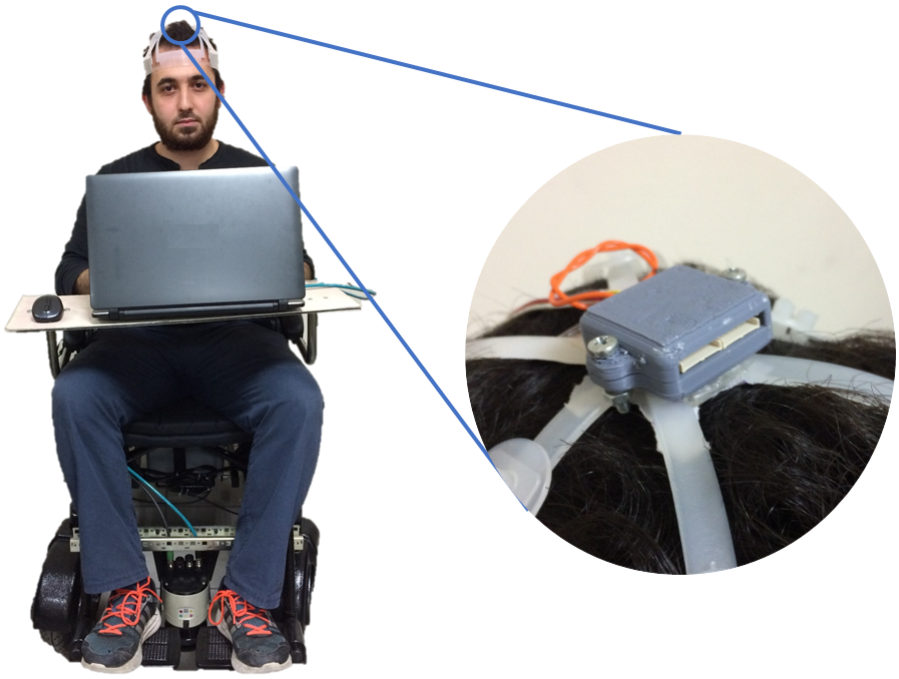


Figure 2.4 : The powerchair with the sensor on top of the head of a person.

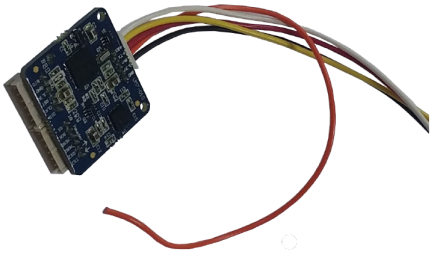


Figure 2.5 : The sensor without its package.

2.2.3 Robot operating system

The Robot Operating System is a framework with which robots can be programmed. It provides tools, libraries and a model (convention) to simplify creating complex robot software across a wide spectrum of robots. It allows modular structures to work in parallel.

The ROS is highly complex framework. The complexity it has, makes it out of the scope of this thesis. The important thing to understand about ROS, is that it is a tool which facilitates development of robots. That is why it has been used in the project. It allowed the focus of the research to be at gesture recognition rather than it being distributed to other related side tasks such as communication of the sensors with the computer, communication between the algorithms, and drivers needed to develop full fledged autonomous chair. Nonetheless, the interested can find complete documentation about it on its website [22].

3. FAST FOURIER TRANSFORM AS A FEATURE EXTRACTION METHOD

The Fast Fourier Transform (FFT) is an algorithm that computes the Discrete-Time Fourier Transform (DTFT) of a time series. How the algorithm computes the DTFT can be a thesis by itself. Thus, detailed explanation of it has not been given in the thesis. However it is important that the reader broadly understands for what the algorithm and the Discrete Fourier Transform (DFT) are generally used.

FFT is used to compute DTFT in applications where time is of essence because it can calculate DFT with a reduced computational complexity.

DFT is a mathematical transformation tool that is a form of Fourier analysis. It calculates the frequency components of a time series signal. Using inverse DFT, it is possible to obtain the original signal from which the DTFT calculated. When the signal is discrete, DFT gives a discrete function of frequency whose output is the coefficient for the input frequency.

Two different hypothesis has been tested with different decision making processes, they are discussed in the following sections named as Hypothesis I and Hypothesis II.

3.1 Hypothesis I

Gesture recognition is done via feature extraction. Since anything that allows to make distinction between gestures can be thought of as a feature, it is possible to treat the DTFT as a feature set. Because every gesture is different in time series representation, it is expected that every gesture also is a different combination of frequency components with different amplitudes. Then, this difference can be used to make distinction between gestures. However, it is obvious that nobody can perform the same gesture in the same way twice. Therefore, it can not be expected to calculate the same combination of frequencies every time, when the sensor data carries a gesture. This understanding forces the recognition algorithm to have some sort of mechanism

where it can measure how the calculated combination of frequencies change when the gesture representation in time series data change ever so slightly.

The mechanism has to be able to help identify gestures whose time series representations are not exactly same, yet similar. To this endeavour, it has been decided that an average can be calculated after collecting gesture samples. After the average change in amplitude of frequencies is found, the hypothesis is that the deviation from the mean can be used as a threshold for recognition.

Implementation of this hypothesis required three steps to achieve gesture recognition namely preprocessing of the raw sensor data, feature extraction, and decision making. Each step is explained under the next three sections.

3.1.1 Preprocessing of the raw sensor data

The orientation signal coming from the sensor is stored if any movement is detected. This detection is based on a simple threshold mechanism on the velocity measurement that is sent from the sensor. Storing process starts, if and only if angular velocity on any axis increases more then 1 radians per second. When the sensor output comes to a rest (meaning there is no increase or decrease on the position or angular velocity coming from the sensor), the stored data is sent to the second stage. The sensor output is thought to be at rest if position and angular velocity does not go above or below a threshold. This threshold was required because of noise in the signal. Then The Fast Fourier Transform (FFT) of the stored signal is taken.

The transform yielded different total number of frequencies for a gesture every time, because the record had a different time window (length). This problem has been overcome by making quantization and interpolation on the frequency axis of the transform output. As a result, an array that has the length of half of the sampling frequency is obtained. This means that every gesture can be represented with the same number of frequencies.

3.1.2 Feature extraction

Before explaining the extraction process, it is important to give a base for the calculations that will be explained here shortly. The desired gestures to be recognized are performed and recorded 30 times for each gesture category so that averages for

frequencies in DTFT can be calculated. The average of each frequency in all samples are taken separately, thus creating an average for each frequencies' amplitude. Then for each frequency, the standard deviation is calculated by using the same samples. The gesture template (with which comparisons are made) formed simply by adding and subtracting one standard deviation to and from the average of each frequency in each gesture. Any input is considered to be an exact match for a gesture template, if and only if the amplitude of all frequencies of the input are in the band of the one standard deviation above or below the average of the frequencies of the gesture. If an arbitrary input is not an exact match for a template, its similarity to the template is checked in all dimensions together. The similarity is calculated by summing the difference from one standard deviation above or below the average of each frequency for each axis of a gesture template. The sum is divided by the sum of the averages of all frequencies in all axes. This calculation results in a number which is used as a similarity measure. The similarity measure is treated as a feature for inputs whose DTFT stayed outside of the one standard deviation band around the average.

3.1.3 Decision making

The decision of an input gesture, whose frequencies do not stay inside the one standard deviation band, to be a match or not is based on empirical data. After experimentation, it has been decided that any arbitrary input gesture is a match if it has a similarity measure that is less then 0.5.

3.1.4 Experiment results and conclusion

The experiments have shown that the similarity measure is not enough to differentiate between different gestures (empirically found success rate 10% to 15%). The reason for it, was thought to be the fact that summing all the error in all dimensions could have resulted in information loss simply because, errors in different dimensions may be very different even though, the total error is the same. The information is lost when summation is applied. And this information may be crucial to decide if an arbitrary input is a match for a particular template or not. Thus, another decision making mechanism is needed. For this reason, Hypothesis II has been proposed.

3.2 Hypothesis II

As it has been mentioned in Hypothesis I, it was desired to separate information in different axes to evaluate similarity based on matching, axis to axis information (total error) so that accuracy is increased. The separation allowed to check similarity in every direction (x,y and z axes) independent from each other.

Feature extraction procedure is the same as it was done in Hypothesis I, the difference is that separation of axes has been done.

The procedure is as follows: The error (as in the distance from the one standard deviation band) for any frequency in one axis is summed. Then the sum is divided by the sum of amplitudes of all frequencies for the same axis. The calculated similarity measure for each axis is output separately as a part of a Python dictionary. The output is a Python dictionary with three values that indicate how much any arbitrary input gesture is similar to the templates, axis-wise. Then, for deciding on whether an input gesture is a match or not, the values in the dictionary are expected to be lower than some experimentally chosen thresholds. To determine what value a threshold should be for a given axis, each gesture is performed 30 times. The average of the outputs are taken. These averages are appointed as the thresholds.

3.2.1 Experiment results

Experiments have shown that the recognition rate is different for different gestures. The found thresholds are given in Table 3.1;

Table 3.1 : Empirically determined thresholds for Hypothesis II.

Gestures	Empirically Determined Thresholds For Axes		
	Axis X	Axis Y	Axis Z
OPEN	0,160390751	0,395692438	1,02E+06
CLOSE	0,226130276	0,343409219	0,281430599
SUPPORT	0,050062286	0,251115359	0,004638223
NO-SUPPORT	0,075792278	0,672857004	0,000460083

3.2.2 Results for OPEN gesture:

In the Figure 3.1, it can be seen that average for all frequencies for a given gesture's DTFT, dashed lines show the one standard deviation band enveloping above or below the average, blue line indicates the input gesture for that instance. And green dot line indicates the average. One successful trial where open gesture has been recognised can be seen in Table 3.2

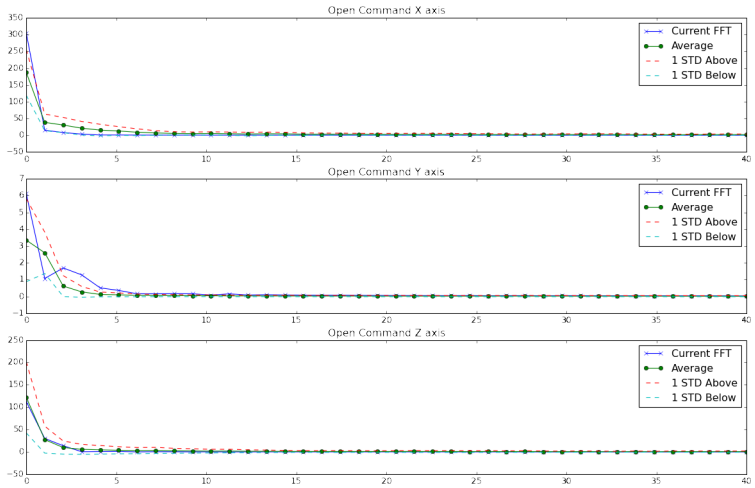


Figure 3.1 : DTFT in an experiment for OPEN Gesture.

Table 3.2 : Measured similarity results for OPEN.

Gestures	Experiment Results (Measured Similarity)		
	Axis X Error	Axis Y Error	Axis Z Error
OPEN	0,122453289	0,321908682	0,00E+00
CLOSE	0,181225452	0,355606413	0,139999434
SUPPORT	0,022809959	0,836353776	0,23738661
NO-SUPPORT	0,022809959	0,836353776	0,23738661

3.2.3 Results for CLOSE gesture:

In the Figure 3.2, it can be seen that average for all frequencies for a given gesture's DTFT, dashed lines show the one standard deviation band enveloping above or below the average, blue line indicates the input gesture for that instance. And green dot line indicates the average. The method used for recognition did not perform well for the CLOSE command gesture. Not one successful trial could be recorded.

3.2.4 Results for SUPPORT gesture:

In the Figure 3.3, it can be seen that average for all frequencies for a given gesture's DTFT, dashed lines show the one standard deviation band enveloping above or below the average, blue line indicates the input gesture for that instance. And green dot line indicates the average. The method used for recognition also did not perform well for the SUPPORT gesture. No recognition could be done.

3.2.5 Results for NO-SUPPORT gesture:

In the Figure 3.4, it can be seen that average for all frequencies for a given gesture's DTFT, dashed lines show the one standard deviation band enveloping above or below the average, blue line indicates the input gesture for that instance. And green dot line indicates the average. The method used for recognition also did not perform well for the NO-SUPPORT gesture. No recognition could be done.

3.3 Conclusion

Experiment results of the hypothesises indicated that either a better similarity measure or a better feature extraction method is needed. Therefore, it has been decided to test the feature extraction method that has been used for hypotheses before proposing another similarity measure.

The test has been done by creating a pseudo-signal and changing the parameters of the signal one parameter at a time. The test proved that change in time windows,

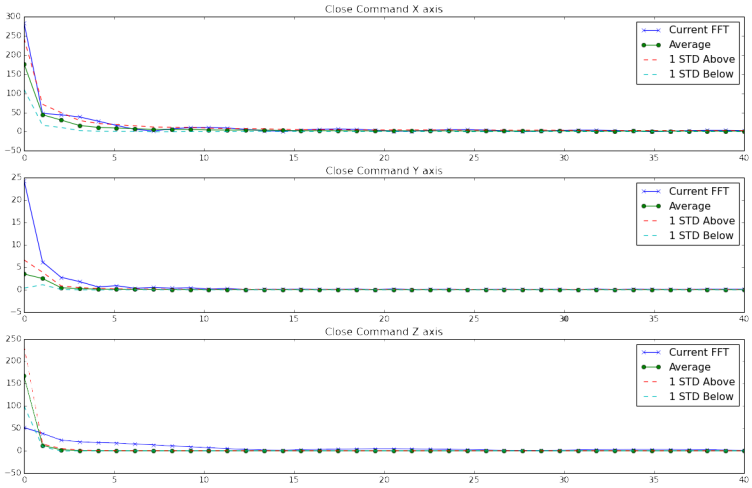


Figure 3.2 : DTFT in an experiment for CLOSE gesture.

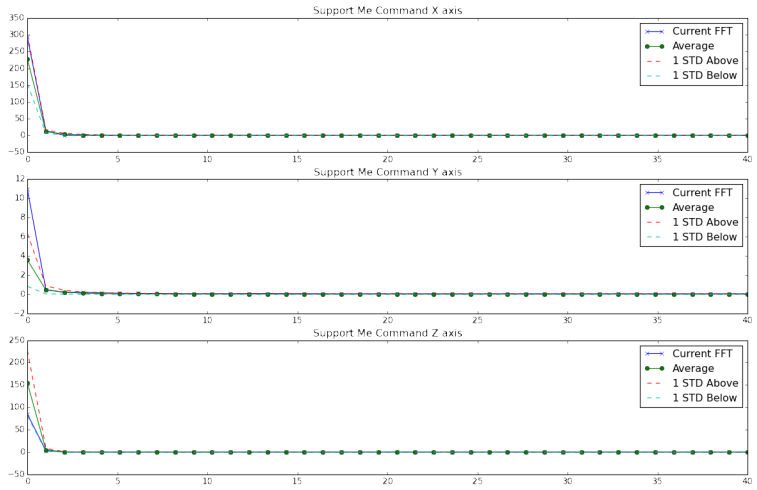


Figure 3.3 : DTFT in an experiment for SUPPORT gesture.

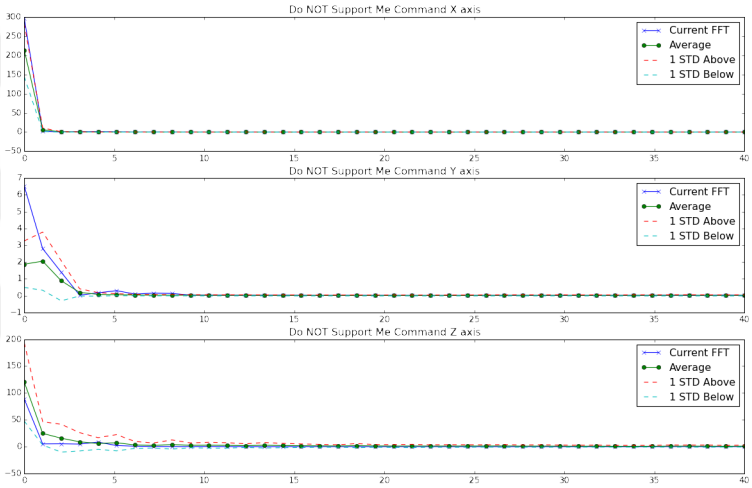


Figure 3.4 : DTFT in an experiment for NO-SUPPORT gesture.

which are selected automatically for taking the DFT of the input data, renders this feature extraction method unfit for the recognition task. The reason is that for DFT to be taken, the signal has to be assumed periodic with its time window. Thus, it is expected that if the time window changes, the DTFT changes too. However, the test have shown that change in DTFT is drastic, which is why using DFT for feature extraction is problematic, even if every parameter is kept the same and, only the time window is changed slightly.

The pseudo signal used in the test is $u(t) + 2 * u(t-1) - 2 * u(t-3)$ where $u(t)$ is the Heaviside step function. In the Figure 3.5, the time window is chosen as 3 seconds. In the Figure 3.6, the time windows is chosen as 2.2 seconds. The windowed signals, and the resulting DTFTs can be seen the figures, respectively.

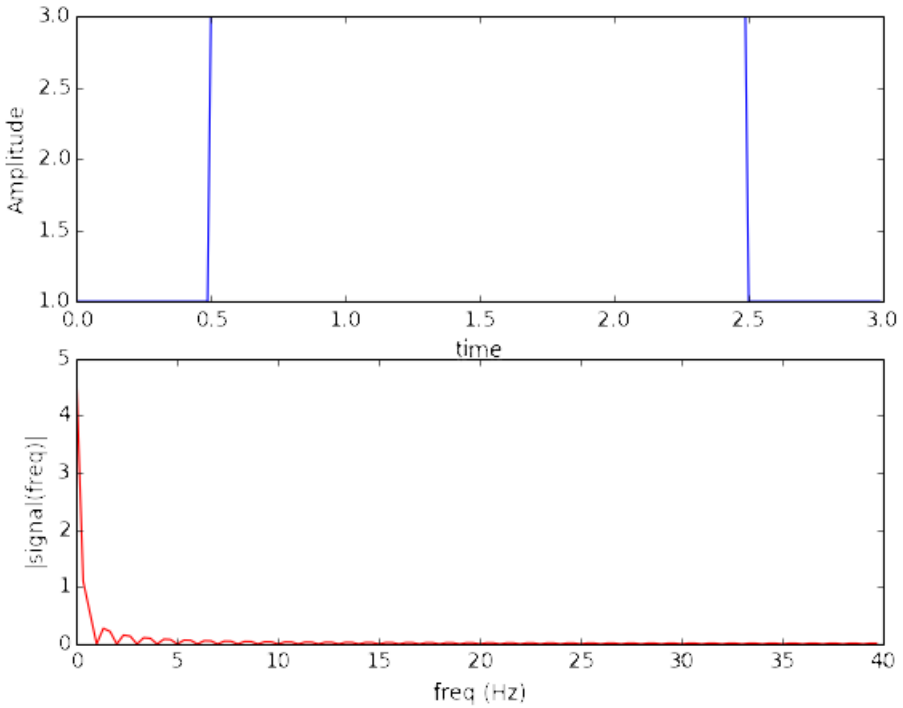


Figure 3.5 : DTFT of 3 seconds time window.

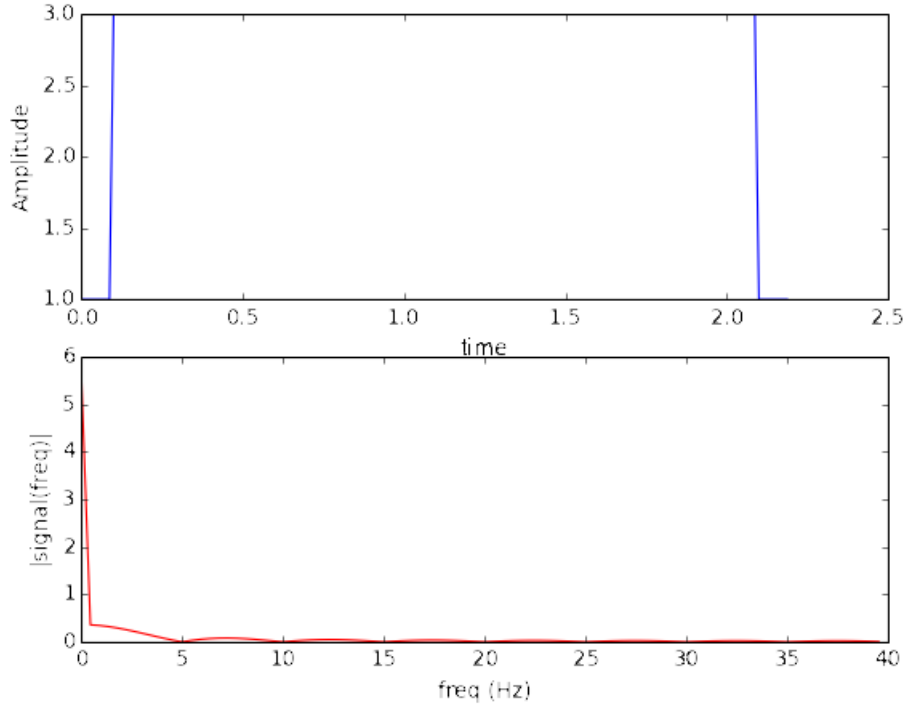
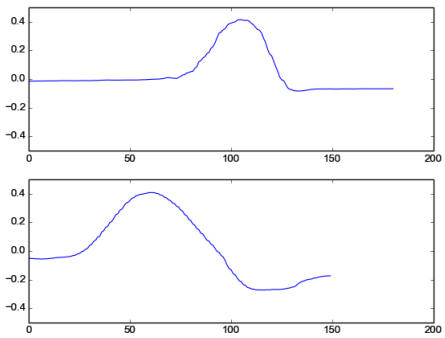


Figure 3.6 : DTFT of 2.2 seconds time window.

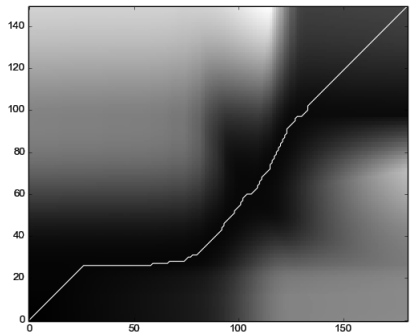
4. HEAD GESTURE RECOGNITION VIA DTW AND THRESHOLD OPTIMIZATION

Dynamic Time Warping (DTW) measures similarity in time domain [28]. DTW is an algorithm that takes two inputs which are time series. It outputs three different entity namely, an accumulated cost matrix, an optimal path and a similarity measure called minimum travelled distance (MTD). Consequently, it is a popular choice in recognition tasks where time series are the primary data source.

DTW makes comparison of the two input signal by creating a cost matrix (see Figure 4.1b) whose elements are the distance between each points of the two time series data (see Figure 4.1a). The black color indicates relatively small distance between the elements. The white color indicates relatively long distance between the elements. This cost matrix is used to find an optimal path which is the mapping between the distance-wise closest points of the two time series data. Finally, minimum travelled distance is calculated by adding each cost (each element of the matrix) on the optimal path (the white line in 4.1b).



(a) Two time series signals.



(b) The cost matrix resulted from comparison of the two time signals.

Figure 4.1 : DTW example. The cost matrix (4.1b). The two signals (4.1a).

MTD is a similarity measure because, it is calculated to be zero when the two inputs are exactly the same. As the two input become more different, MTD value increases. The lower bound for MTD is zero. However, there is no upper bound for it.

Feature extraction is one of the hardest part of gesture recognition. The tools used for feature extraction is usually the tools developed for signal processing where the main problem is extracting the properties of original signal from noisy data that is the result of transferring the original information through physical mediums. These tools, therefore, are developed for extracting features that are already known. But, in gesture recognition, it is impossible to know what features to extract from data. There exist no original information with which a comparison can be made. In other words, features that are the relevant properties of the original signal, can not be expressed with 100% certainty in gesture recognition. This is not the case for signal processing. This difference makes the tools used for signal processing somewhat inappropriate for gesture recognition. Nonetheless, that is not to say these tools are not useful in gesture recognition. Because of the difference, DTW has been chosen for gesture recognition in the project.

DTW is useful for recognition because, it indirectly tries to match features of one time series signal to another one optimally by optimally aligning the two time series data. The features that are required for recognition can not be known with certainty. But, since DTW matches as much feature as possible, it can be used to match features which can not be even imagined that they exist.

4.1 Hypothesis

The minimum travelled distance can be used as a similarity measure. Thus, recognition is possible through determining a threshold for MTD. In other words, two time series data can be said to be similar enough for recognition if the resulting MTD value is below some threshold.

In the project, the gesture data is collected via three-axis accelerometer that measures the head orientation. Therefore, each axis of a gesture template should be compared to their respective counterparts in the data coming from the sensor. This means that DTW has to be used three times for one gesture to be recognised because there exist three different time series data for any given gesture. The resulting three MTD values, then, can be compared to three thresholds each of which determined for one particular axis.

4.2 Proposed Method

The usage of DTW reduced the gesture recognition problem to threshold determination problem. Several attempts have been made to determine the thresholds empirically, however there was no success. One of the reason for not having any success was the user dependence problem.

A gesture recognition algorithm that is to be used in a practical application where commands to a machine is conveyed through gestures would be better, if it allowed user independence since the machine can be used by anybody. Empirically determined thresholds have failed achieving any success due to user dependency issues. DTW does not provide any information about the best time series representation of a gesture. Because of this, it is not possible to determine which template, with which the input data is compared, is the best time series representation of a gesture. Any time series sample of a gesture can be used as a template thus the template, which is sampled from one person, carries the characteristics of one person. Therefore, selection of template affects the empirically determined thresholds in a significant way. This problem renders the empirically determined thresholds inappropriate for user independent recognition. In order to overcome the user dependency issues, a method is required to determine the thresholds in such a way that no matter what the template is, the thresholds allow to achieve high success rate for an arbitrarily large dataset. For this reason, the following method is proposed.

The method, treats the MTD values for each axis as a dimension of a three dimensional point in a three dimensional cartesian coordinate system (CCS). The point is expected to be closer to the origin of the CCS, when the inputs to DTW algorithm are of the same gesture type. The point is expected to be relatively away from the origin of the CCS, when the inputs to DTW algorithm are of different gesture type. The method allows to find the optimal or close to optimal boundary between these two different point types namely, the points resulted from the same gesture comparisons and the points resulted from different gesture type comparisons. It is important to notice that for this distinction between points to be made, the template for any gesture is not important as long as the inputs belong to a gesture category. If the boundary is treated as the

threshold for similarity, it reasonable to expect that the user dependency problem is overcome.

It is not known what the boundary should be. But, it is safe to assume that it will be a three dimensional boundary. In other words, it will look like the surface of a three dimensional object in three dimensional space where it has depth, width, height and a place since the the points are three dimensional. The exact shape of the boundary is not known. However, it is possible to iteratively place an arbitrary three dimensional object to the CCS where the points are, to select the best object that yields the best separation between the different type points.

Iteration without any guidance is impossible because the CCS is a continuous space and the number of combinations for a any given object's depth, height, width and place is unlimited. So, to solve the problem of finding the optimal boundary, in other words, thresholds, it has been decided to convert the problem to an optimization problem by defining a cost function which can guide the iteration. Any meta-heuristic global optimization algorithm can be used. In the project, genetic algorithm [29] has been used.

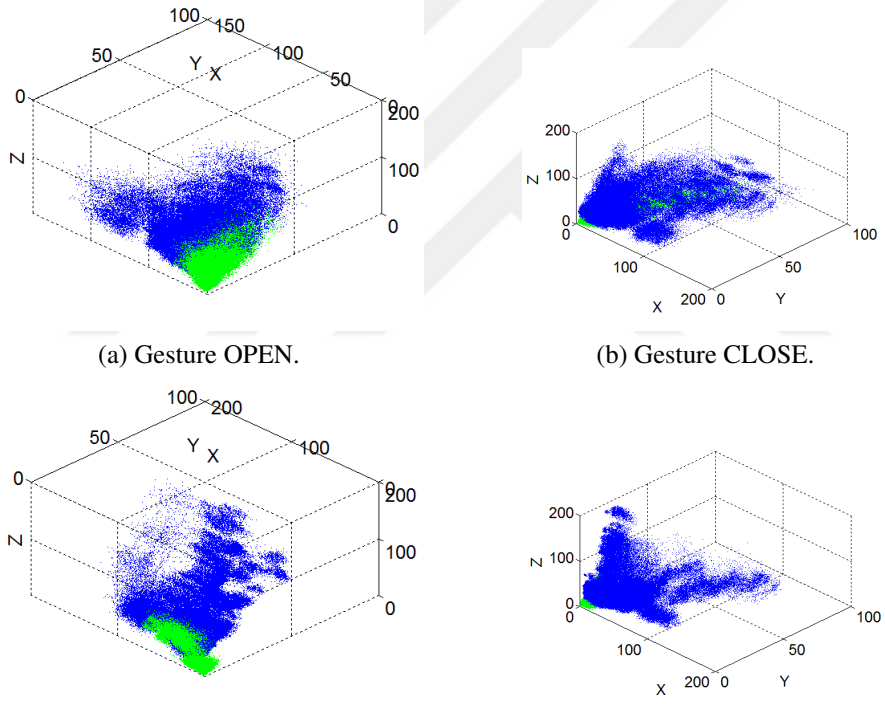
In the following sections, the data collection process, the three dimensional objects, the cost function, the success rate calculation, and the result of optimization has been provided. Finally a conclusion relating the method has been given.

4.3 The Data Collection Process

There are four different gesture types in the project, named as OPEN, CLOSE, SUPPORT and NO-SUPPORT. In order to find the optimal thresholds, a rich dataset is required. Six different people are gathered and instructed to perform each gesture for two minutes while leaving a second break between each execution in order to create the dataset which is composed of time series data of gestures. In the end, 248 OPEN, 268 CLOSE, 272 SUPPORT and 273 NO-SUPPORT gesture samples were collected. The difference in the number of different gesture types caused by people completing different gesture in different time frames.

After completing the dataset of samples for gestures, it was required to create MTD values which are the result of gesture comparisons. To this endeavour, each sample

has been kept as a template while others are used as the second input to DTW algorithm. The resulting MTD values (points) are randomly divided into test set and optimization set. For each gesture type, approximately 250.000 points are obtained. The optimization for thresholds is done for all gesture types by using approximately 125.000 points, the remaining approximately 125.000 points are used in determining the success rate via the optimized thresholds. All points are plotted on three dimensional CCS. The green points in Figure 4.2 are the points resulted from the same gesture comparisons. The blue points in Figure 4.2 are the points resulted from different gesture comparisons. The expected distribution of the points can be seen in the figures. The figures are drawn from different perspectives so that a better understanding of how the points are distributed for each gesture, can be achieved.



(a) Gesture OPEN.

(b) Gesture CLOSE.

(c) Gesture SUPPORT.

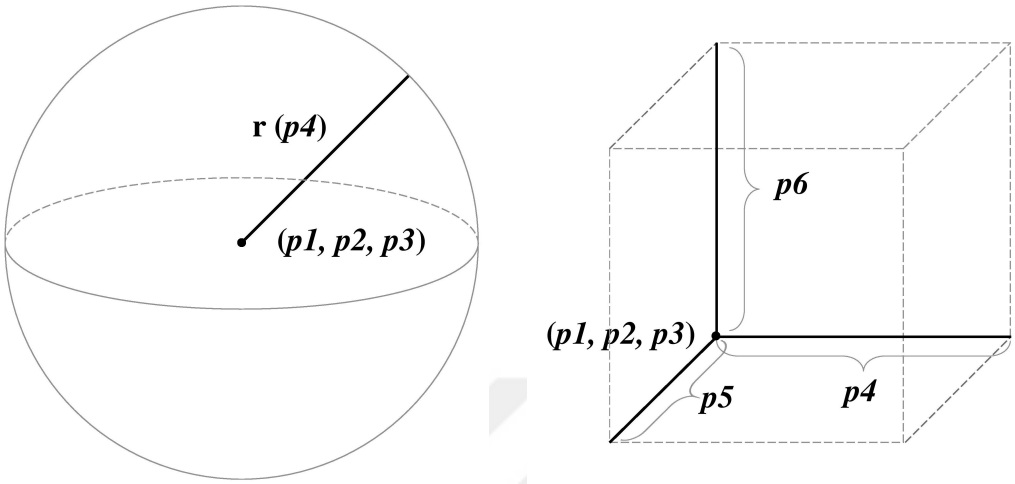
(d) Gesture NO-SUPPORT.

Figure 4.2 : Illustration of how the points are distributed in CCS for each gesture type.

4.4 The Three Dimensional Objects Used In Optimization

The exact shape of the object is not known. The object may be a highly complex polygon. Therefore, to provide the proof of concept, two simple objects are used in the project. These objects are a sphere and a quadrangular, which can be seen in Figure 4.3. The sphere has four parameters (see Figure 4.3a). Three of which are the coordinate

for the center point of the sphere, the remaining one is the radius. The quadrangular has six parameters (see Figure 4.3b). Three of which are the coordinate for one edge of the quadrangular, the remaining three correspond to the length of each fundamental axis in the CCS.



(a) The Shpere. (b) The Quadrangular.

Figure 4.3 : Three dimensional objects used for optimization.

4.5 The Cost Function

A cost function is required for optimization. The cost function can be defined as desired as long as it maximises for high recognition rate. Therefore, there exists a set of plausible cost functions. Equation 4.1 is used in the project as the cost function for optimization.

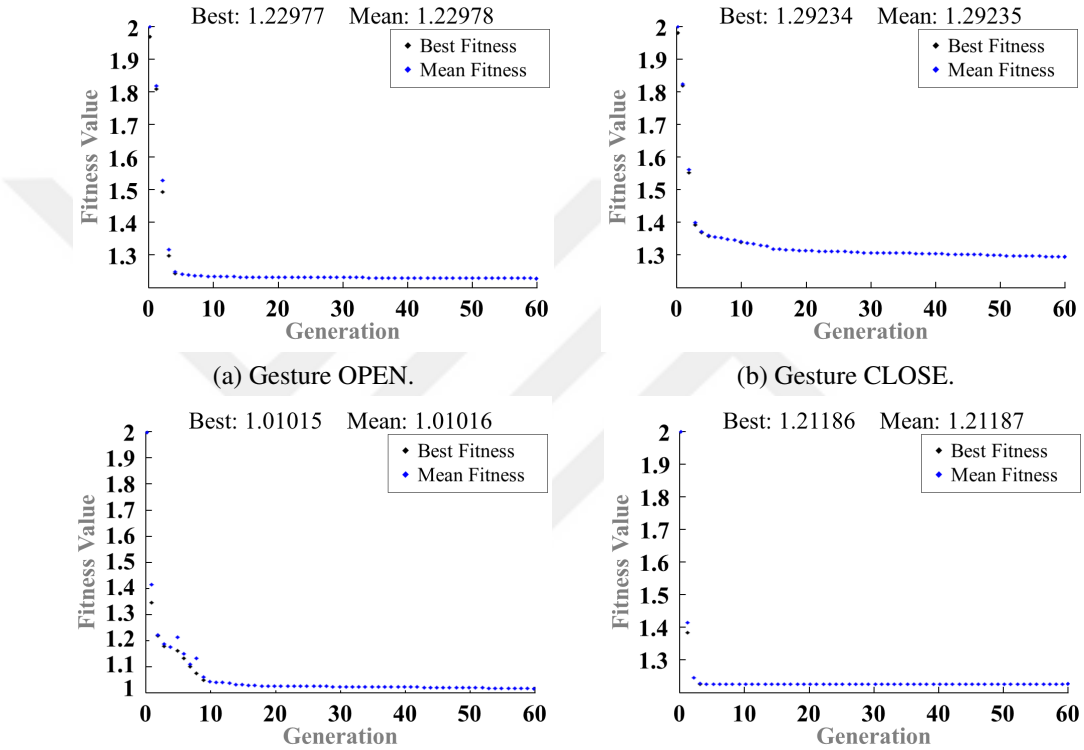
$$f_{cost}(\alpha, A, \beta, b, \upsilon, \Upsilon) = -\frac{\alpha}{A} + \frac{\beta}{B} + \frac{\upsilon}{\Upsilon} \tag{4.1}$$

Where α corresponds to the green points inside the object, β corresponds to the blue points inside the object, υ corresponds to the volume of the object, A corresponds to the total number of green points, B corresponds to the total number of blue points and Υ corresponds to the smallest possible volume of the object that contains all points.

Equation 4.1 is a plausible cost function because, as the volume of the object increases, the total number of blue points inside the object and the green points inside the object increases. However, since the coefficient of α is -1; α, β and Υ are competing variables. This competition allows this function to be used as a cost function for the problem at hand.

As the optimization algorithm, genetic algorithm [29] is used. The genetic algorithm minimizes the cost function to maximise the success rate.

In Figure 4.4, evaluation of the cost function can be seen for each gesture type when the object that is used for optimization is prism. In Figure 4.5, evaluation of the cost function can be seen for each gesture type when the object that is used for optimization is prism. It can be deduced from the figures that around iteration number twenty, genetic algorithm approaches a minimum very closely for each optimization.



(a) Gesture OPEN.

(b) Gesture CLOSE.

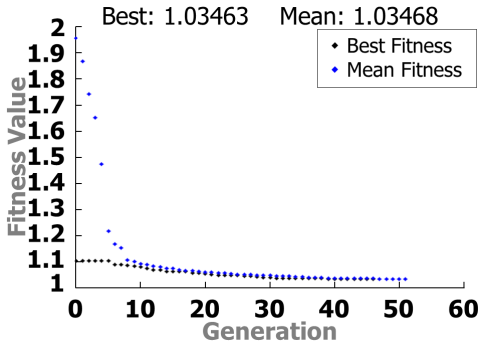
(c) Gesture SUPPORT.

(d) Gesture NO-SUPPORT.

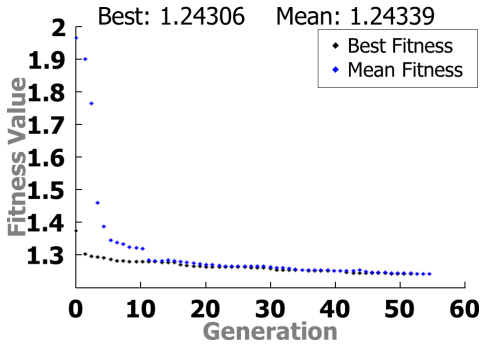
Figure 4.4 : Evaluation of the cost function with each iteration for the quadrangular object.

4.6 The Success Rate

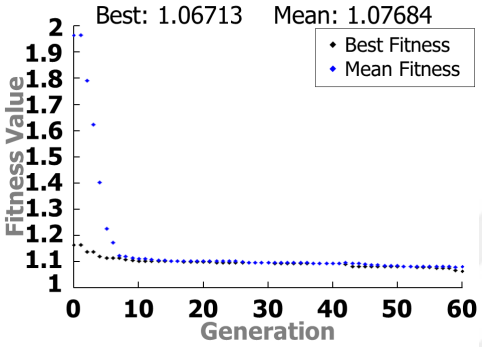
After the optimization process and determining the optimal thresholds, the success rate is needed to be calculated. For that purpose, equation 4.2 is used. The determined thresholds for the two different object, are visualised and can be seen in Figure 4.6 which is the visualization of the quadrangular thresholds, and in Figure 4.7 which is the visualization of the sphere thresholds. The red circles indicate the shapes of the objects. The figures show the thresholds from different perspectives for ease of understanding, because three dimensional space can not be perfectly drawn into two



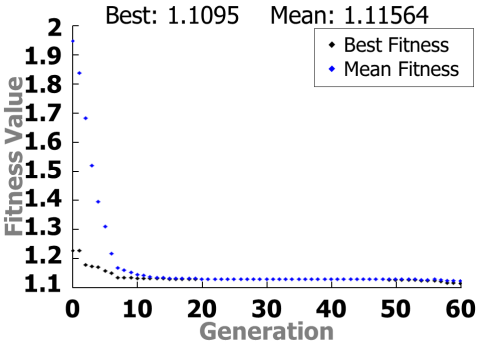
(a) Gesture OPEN.



(b) Gesture CLOSE.



(c) Gesture SUPPORT.



(d) Gesture NO-SUPPORT.

Figure 4.5 : Evaluation of the cost function with each iteration for the sphere object.

dimensional paper.

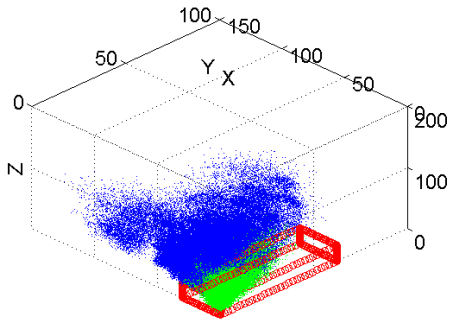
$$f_{successrate}(\theta,\phi,\varepsilon) = \frac{\theta}{\theta+\phi+\varepsilon} * 100 \tag{4.2}$$

Where θ corresponds to the total number of green points inside the object, ϕ corresponds to the total number of blue points inside the object and ε corresponds to the total number of green points outside of the object.

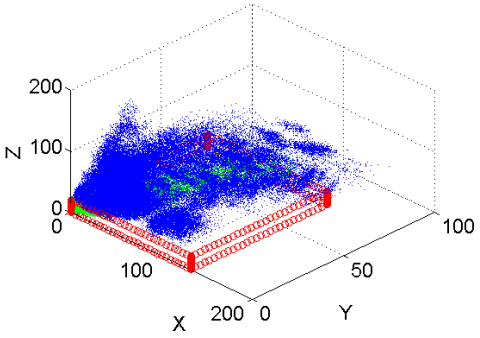
4.7 The Results and The Confusion Table

The optimized thresholds with one half of the collected MTD values are tested with the other half to calculate the success rate. The success rate is calculated as in equation 4.2. The resulting success rates for gestures, optimized via the quadrangular object and via the sphere object are given in the tables 4.2 for the quadrangular object and 4.1 for the sphere object. There is also one more important side that needs to be explained that when a point ends up in two different shapes, the category of the input gesture is chosen based on the proximity of the point to the origin in the objects.

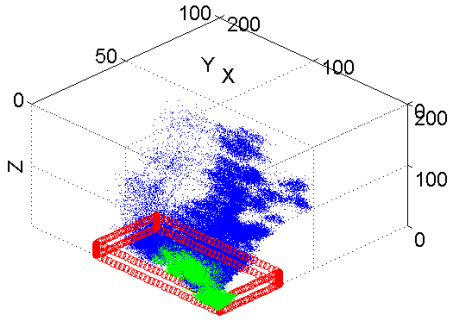
The average achieved success rate for the sphere object is $81,54\%$. The average achieved success rate for the quadrangular object is 83.14%. Table 4.1 shows better



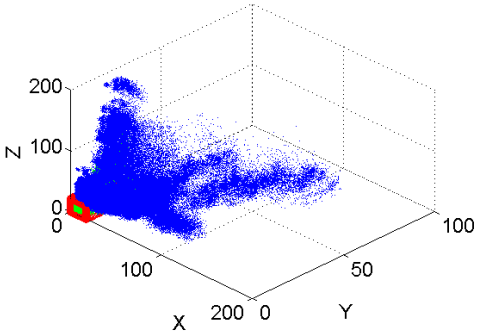
(a) Quadrangular for OPEN.



(b) Quadrangular for CLOSE.

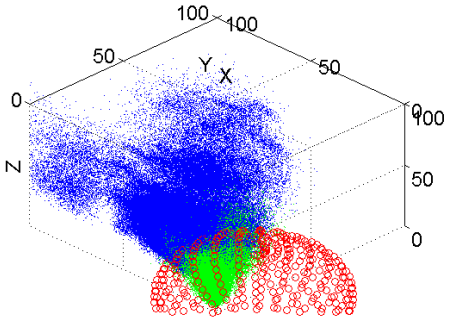


(c) Quadrangular for SUPPORT.

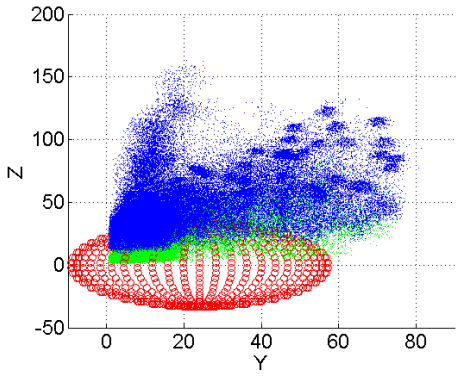


(d) Quadrangular for NO-SUPPORT.

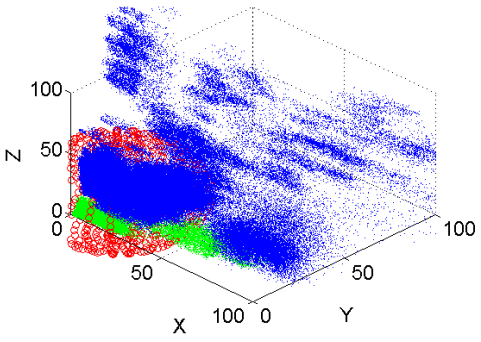
Figure 4.6 : Visualization of the quadrangular thresholds for gesture types from different perspectives.



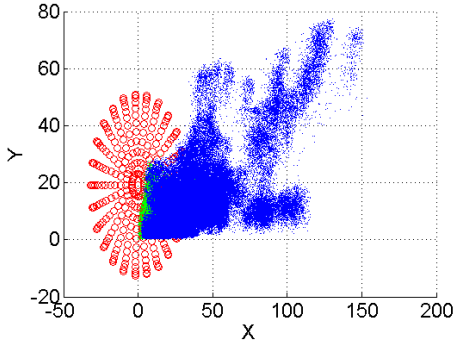
(a) Sphere for OPEN.



(b) Sphere for CLOSE.



(c) Sphere for SUPPORT.



(d) Sphere for NO-SUPPORT.

Figure 4.7 : Visualization of the sphere thresholds for gesture types from different perspectives.

Table 4.1 : Confusion matrix for sphere fitting.

SPHERE		PREDICTED GESTURE					SUCCESS RATE
	Gestures	OPEN	CLOSE	SUPPORT	NOSUPPORT	IGNORED	
INPUT GESTURE	OPEN	12737	94	68	68	1364	88,87%
	CLOSE	34	11898	51	32	4550	71,82%
	SUPPORT	0	0	14047	0	2918	82,79%
	NO-SUPPORT	0	0	0	14127	2968	82,63%

Table 4.2 : Confusion matrix for quadrangular fitting.

QUADRAN.		PREDICTED GESTURE					SUCCESS RATE
	Gestures	OPEN	CLOSE	SUPPORT	NOSUPPORT	IGNORED	
INPUT GESTURE	OPEN	11673	20	20	29	2428	82,37%
	CLOSE	211	12550	241	238	3898	73,22%
	SUPPORT	0	0	16620	0	345	97,96%
	NO-SUPPORT	0	0	0	13504	3591	78,99%

success rate for gestures OPEN and NO-SUPPORT. On the other hand, Table 4.2 shows better success rate for gestures CLOSE and SUPPORT. When the thresholds for gestures are chosen according to the best success rate yielding object, namely using sphere as thresholds for gestures OPEN and NO-SUPPORT and using quadrangular as thresholds for gestures CLOSE and SUPPORT, the resulting average success becomes $85,68\%$.

4.8 Conclusion

The proposed method achieved success rate of $85,68\%$ even for simple objects like a sphere and a quadrangular. Therefore, it is plausible to say that the method can be used to determine thresholds optimally where MTD value is used as the similarity measure. It is safe to say that more complex objects (polygons) that have more free parameters can be used to increase the success rate for the application as long as the distribution of the points does not change as the number of gesture samples goes to infinity. It is worth noting that the distribution of the points that are the result of the same gesture comparisons, can fit into a simpler object better than a much more complex object. Therefore, a simpler object may yield better success rate compared a more complex object. However, it is safe to assume that as the number of free parameters increases, the success rate increases.

Another way to increase success rate may be expanding the idea of placing an object on to three dimensional CCS. For example, MTD values for angular velocity comparisons

can be used with MTD values for position comparisons to form a six dimensional point. A six dimensional object can be iteratively placed to maximise success rate.

The method also may be used in other applications with little modification, where separation of high dimensional feature vectors is needed.

5. GESTURE RECOGNITION VIA NEURAL NETWORK

Using machine learning for gesture recognition is popular among scholars [4, 9, 17, 19, 20]. The structure of such algorithms depends on the type of the problem. They are generally used for feature extraction in gesture recognition tasks. This is due to the fact that machine learning algorithms allow abstraction, and modelling at a higher level than conventional programming does. Conventional programming becomes impractical when the relationship of a function between its output and its input can not be expressed precisely. In other words, when the relationship can not be hard coded into the algorithm, higher level abstraction is required. Higher level abstraction allows such relationships between the output and the input to be modelled in most of the cases. There are numerous techniques for machine learning.

In the project, neural networks have been used as the machine learning method for recognition because the implementation of neural networks have become easier by the reason of the recent advancements in computing technology.

5.1 Hypothesis

The main problem in gesture recognition is feature extraction as it has been explained previously. Neural networks can be used for gesture recognition because it allows to model the unknown relationship between an input set and an output set. This understanding makes neural networks appropriate for gesture recognition because the relationship between the input sensor data and the gestures to be recognised can not be expressed precisely. The author's hypothesis is that gesture recognition is possible through the direct use of time series data.

The following sections give the structure of the neural network that has been used, the preprocessing step, the training process of the network with only gesture data, the training process of the network with samples gathered while driving the chair, the obtained results and a conclusion.

5.2 The Structure of The Neural Network

Each sample in time series data of a gesture can not be given to a neural network one by one as the samples arrive, since neural networks compute an output for each input. In other words, neural networks have no context of time. The structure in Figure 5.1 is used to overcome this problem. This structure can be used coupled with first in, first out data buffering in order to keep the number of inputs the same. First in, first out data buffering refers to the idea that when a new sample that is collected from the sensor enqueued to the buffer, the oldest sample in the buffer is dequeued. This way a sliding window in time is created.

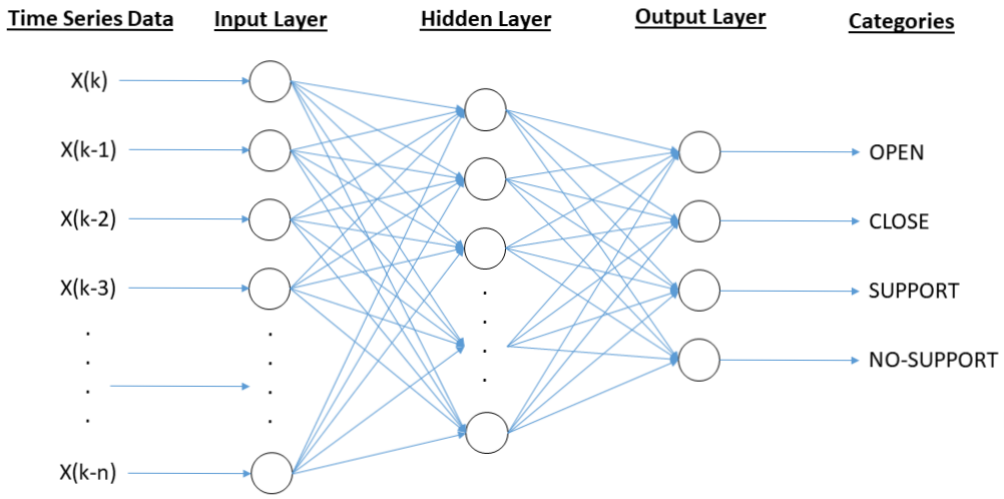


Figure 5.1 : Selected neural network structure.

5.3 The Preprocessing Steps

Neural networks require the same number of inputs, once it is created. Training of neural networks are usually done offline. Thus a dataset is required for training. The collected samples of gestures in the dataset that is used in section 4 vary in length, therefore, a preprocessing of the collected samples is required to scale the gesture samples into a fixed number of points. For example, Figure 5.2 shows examples of original gesture sample and Figure 5.3 show the scaled versions of those original gesture samples.

In the real application, the scaling is also needed because the buffer can store 200 points. It may be questioned why all gestures are scaled to 124 points. This is primarily a design choice, however 124 has been chosen in the project due to the fact that the

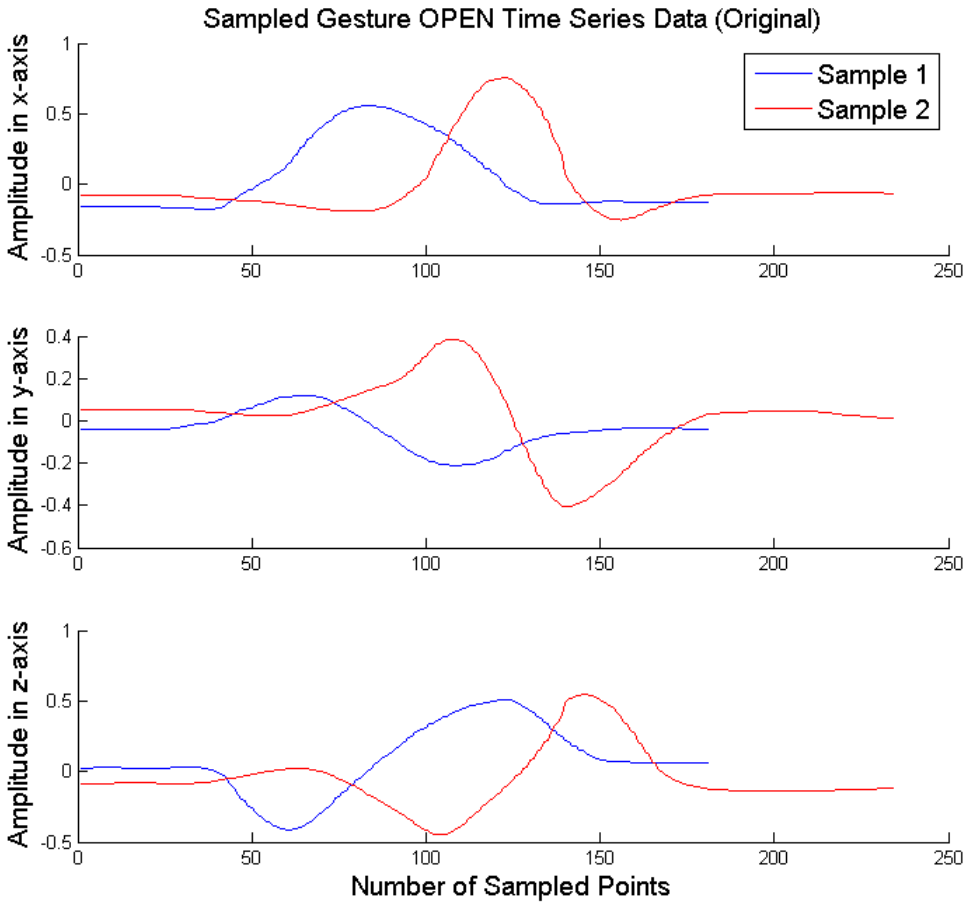


Figure 5.2 : Samples of gesture OPEN before scaling.

smallest length among all the samples in the gesture dataset is 124, thus they are scaled to 124 points. The scaling is done through interpolating the length of the original signal to the least common multiplier of 124 and the length of the original signal. The interpolated gesture data is then down sampled to only have 124 points in total.

After scaling, each time series data for each axes for each gesture sample is connected back to back for form a vector. First the time series data is for x axis, the second time series data is for y axis and the last time series data is for z axis. When time series data for each axis combined to create a vector, the total number of inputs to the neural network amounted to be $3 * 124 = 372$.

5.4 The Training of The Network With Only Gesture Data

The network is ready to be trained after preprocessing step. There are numerous frameworks for training neural networks. All of these frameworks aims to solve one problem or another of training neural networks. However the simple neural network

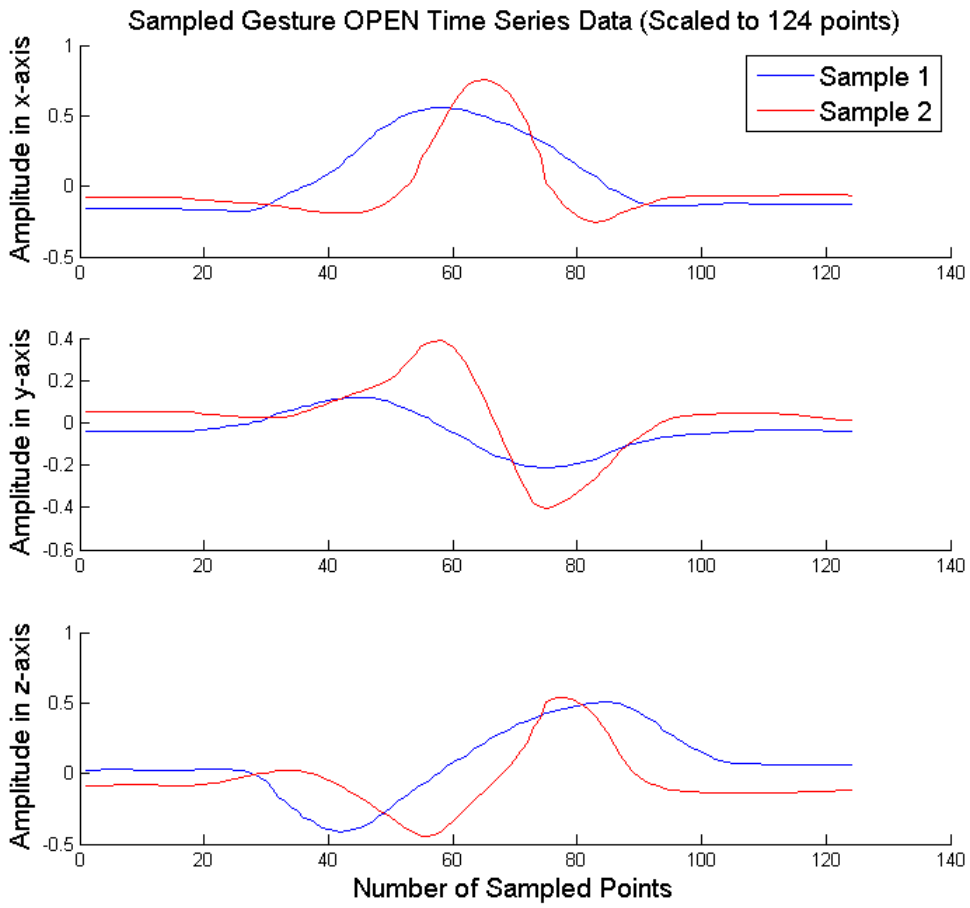


Figure 5.3 : Samples of gesture OPEN after scaling.

that is proposed in section 5.2 can be trained with all most all of available frameworks. However, Matlab R2013a has been used for training in the project because it has an interface that facilitates training and testing of neural networks.

There are design choices such as the activation functions for neurons (nodes), the total number of hidden layers and hidden nodes. As the activation function sigmoid has been chosen mostly because it is generally used in pattern recognition tasks. The sigmoid function also allows higher abstraction because it can be scaled unlike Heaviside step function or signum function. The total number of hidden layers is chosen to be one. The reason is that gradients can vanish due to back propagation, which is how the neural networks are trained, as the number of hidden layers increase. There is no need for increasing the number of hidden layers as long as the network can meet some criteria according to the problem that is being solved. The total number of hidden nodes is chosen as 200. This choice is arbitrary, there exist no rule of thumb for choosing the total number of hidden nodes.

The dataset (see Table 5.1) is divided into 3 parts for training, validation and test. 70% of the data is used for training the neural network, 15% of the data is used for validation checks and the last 15% of the data is used for testing. The resulting confusion table is shown in Figure 5.4

Table 5.1 : Number of samples in the dataset for section 5.4.

Gestures	Number Of Samples
OPEN	248
CLOSE	268
SUPPORT	272
NO-SUPPORT	273

Class numbers 1, 2, 3 and 4 correspond to OPEN, CLOSE, SUPPORT and NO-SUPPORT respectively. The test confusion matrix show 100% average success rate. This high success rate may indicate that the neural network has memorized the data set. It is also important to understand any arbitrary time series data fed to the neural network will be categorised into one of the four categories. This means that gesture detection can occur while driving the chair. To solve this problem, a new neural network has been created with five output classes four of which is the same as the neural network, the added new fifth class is labelled as “DRIVING”.

5.5 The Training of The Network With Driving Data

The neural network that has been trained categorises any arbitrary time series data, this is a problem due to the fact that gestures has be recognised from infinite time series sequence in the real application. This means, the gesture recognition algorithm can recognise a gesture even though the user is actually driving the chair. This is more of a dangerous situation than an inconvenience. In order to overcome this problem, it has been decided to categorise the time series data while driving the chair into a new category. The hypothesis is that the time series data coming from the sensor while driving will be categorised into the new “DRIVING” class by the neural network, thus eliminating the problem of detecting gestures while driving.

The same neural network structure is used with one addition of an output neuron in the output layer. A new time series dataset is collected while driving the chair and added into the already existing dataset of gestures. In total, 250 time series of driving data is

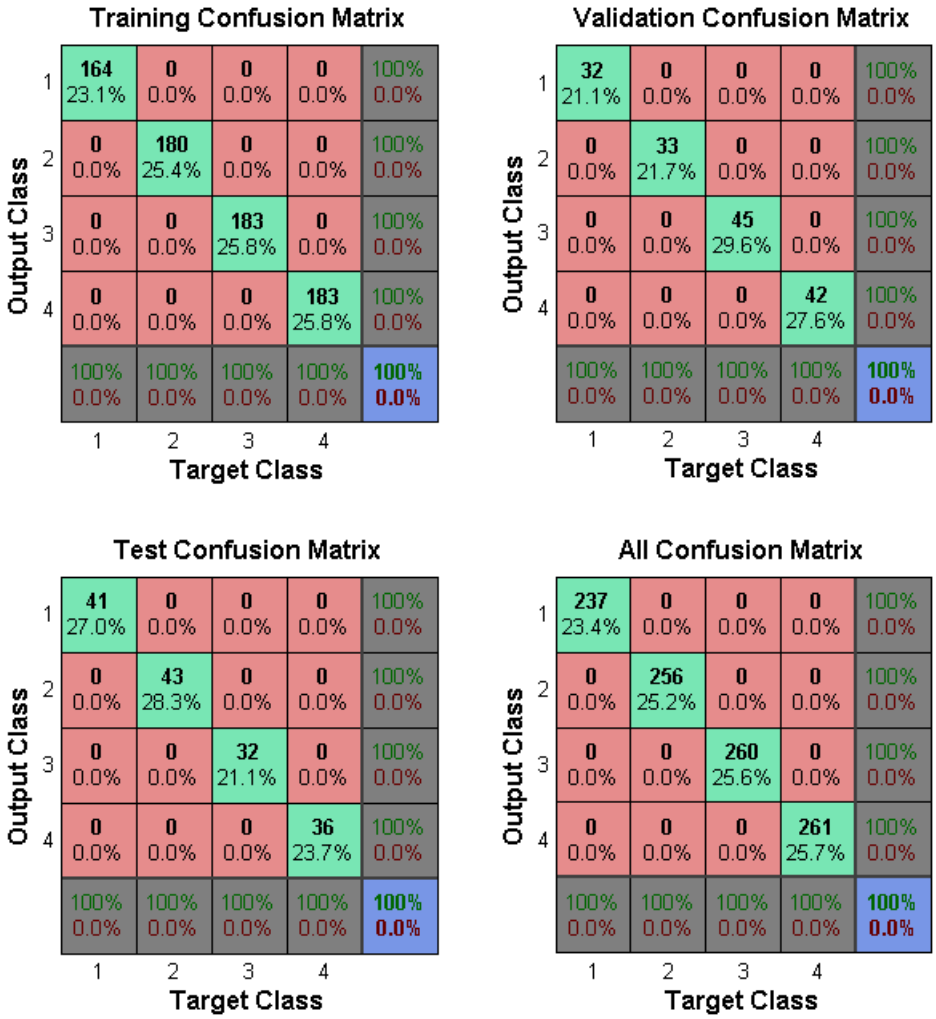


Figure 5.4 : Confusion Tables (Only gesture data).

added to the dataset, since the existing dataset has around 250 samples for each gesture (see Table 5.2) . The new dataset is divided into 3 parts for training, validation and test. 70% of the data is used for training the new neural network, 15% of the data is used for validation checks and the last 15% of the data is used for testing. The resulting confusion table is shown in Figure 5.5

Class numbers 1, 2, 3,, 4 and 5 correspond to OPEN, CLOSE, SUPPORT, NO-SUPPORT, DRIVING respectively. The test confusion matrix show 97.9% average success rate. This high success rate may indicate that memorization of dataset issue may not be solved. In order to claim that the neural network will continue achieve the same success rate, more data is required especially for testing.

Table 5.2 : Number of samples in the dataset for section 5.5.

Gestures	Number Of Samples
OPEN	248
CLOSE	268
SUPPORT	272
NO-SUPPORT	273
DRIVING	250

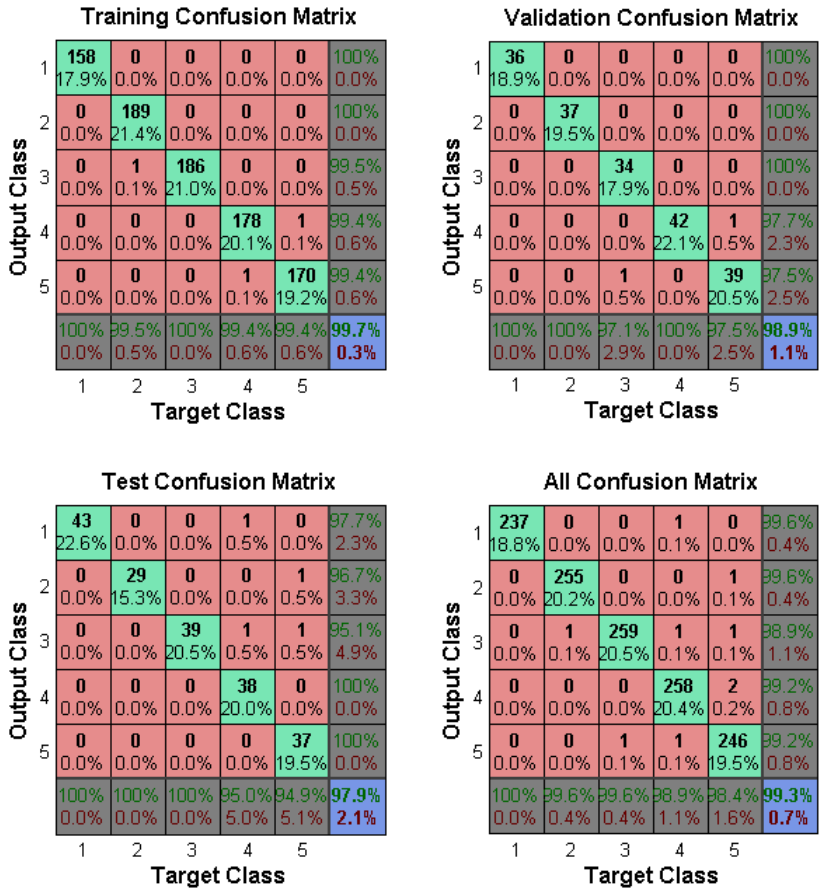


Figure 5.5 : Confusion Tables (Driving class added).

5.6 Conclusion

The neural network has achieved 97.9% success rate for the testing data. This is a very promising success rate. However, it is not know what the neural network learns from the data. Therefore, more testing is required with more data before implementing the neural network into the chair's gesture recognition algorithm. This success rate, if it stays the same after testing with more data, means that out of every 1000 evaluation of time series data that is buffered, 979 of them will be recognised correctly. Since

the neural network is fed the time series data that is in the buffer, after every 10 new enqueued sample in the recognition algorithm of the project, what this recognition rate means for the chair is that in every two minutes, there is a very good chance that the gesture recognition algorithm will recognise a gesture mistakenly even though the user does not intend to do so. If the neural network for gesture recognition is implemented in the chair with this success rate, it can potentially cause very dangerous situations to emerge very frequently. Therefore, either better recognition rate is required, if the neural network approach is to be implemented as the gesture recognition algorithm in the chair or a countermeasure needs to be taken against wrong recognition while driving the chair. The countermeasure that is to be implemented into the chairs gesture recognition algorithms is decided to be preventing the chair from listening for commands while driving. This means, the mode of the chair will only be changed if a gesture is recognised while the chair is stationary otherwise the mode will not change in accordance with the gesture type. This countermeasure will get rid of problems and dangerous situations that may occur if the countermeasure is not taken.

6. CONCLUSION

In this thesis, several possible methods of head gesture recognition for a semi-autonomous powerchair has been studied. Head orientation information is collected from an IMU sensor that is fixed on a head gear which is worn by the end user of the chair. The time series data coming from the sensor is used as the primary source for gesture recognition. One of the reason for this is that it is easy to infer the intent of the user from head orientation. Leaning head forward means go forward, turning head left means turn left and so on. Another reason is that IMU sensors are getting cheaper as the markets for electronic devices expands. Three different ways of gesture recognition methods have been studied, namely; Fast Fourier Transform as a Feature Extraction method in chapter 3, Gesture Recognition via DTW and Threshold Optimization in chapter 4 and Gesture Recognition via Neural Network in chapter 5. From the studies of this thesis, a paper named "Head Gesture Recognition via Dynamic Time Warping and Threshold Optimization" has been presented in IEEE 2017 CogSIMA conference. No gestures could be recognised with the method explained in chapter 3. 85.68% success rate is achieved with the method proposed in chapter 4. This success rate mean that more than eight out of ten gestures are recognised successfully. 97.9% success rate is achieved with the method explained in chapter 5. However due to the structure of implementation of the neural network and the fact that it is not know what neural networks learn, this success rate means that in every two minutes, the neural network may recognise a gesture even though the end user is just driving the chair. Therefore, testing with more data and either higher success rate or a countermeasure for possible problems is required if the neural network approach is to be taken in gesture recognition. It is also important to remind the reader, the success rate given in chapter 4 is achieved via fitting simple objects (a sphere and a quadrangular). This success rate can be increased easily by fitting more complex objects namely a polygon. If the method proposed in chapter 4 is used in the implementation phase of developing the chair, it is important to test the optimized thresholds against data collected while driving to have broader perspective

on the success rate. If desired success rate is not achieved while testing the thresholds against collected driving data, the complexity of the object that is used for fitting should be increased so that better success rate can be achieved.

REFERENCES

[1] **2017 IEEE Conference on Cognitive and Computational Aspects of Situation Management**, `http://cogsima2017.ieee-cogsima.org/`, date retrieved : 26.04.2017.

[2] **Gupta, S.**, **Jaafar, J. and Ahmad, W.F.W.** (2012). Hybrid algorithm for hand gesture recognition, *2012 International Conference on Computer Information Science (ICCIS)*, volume 1, pp.538–542.

[3] **Ho, M.A.T.**, **Yamada, Y. and Umetani, Y.** (2005). An adaptive visual attentive tracker for human communicational behaviors using HMM-based TD learning with new State distinction capability, *IEEE Transactions on Robotics*, *21*(3), 497–504.

[4] **Naik, G.R.**, **Kumar, D.K. and Weghorn, H.** (2007). Performance comparison of ICA algorithms for Isometric Hand gesture identification using Surface EMG, *2007 3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, pp.613–618.

[5] **Shin, S.O.**, **Kim, D. and Seo, Y.H.** (2014). Controlling Mobile Robot Using IMU and EMG Sensor-Based Gesture Recognition, *2014 Ninth International Conference on Broadband and Wireless Computing, Communication and Applications*, pp.554–557.

[6] **Kang, S.P.**, **Rodnay, G.**, **Tordon, M. and Katupitiya, J.** (2003). A hand gesture based virtual interface for wheelchair control, *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, volume 2, pp.778–783 vol.2.

[7] **Pisharady, P.K. and Saerbeck, M.** (2013). Robust gesture detection and recognition using dynamic time warping and multi-class probability estimates, *2013 IEEE Symposium on Computational Intelligence for Multimedia, Signal and Vision Processing (CIMSIVP)*, pp.30–36.

[8] **Yin, Y. and Davis, R.** (2014). Real-time continuous gesture recognition for natural human-computer interaction, *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pp.113–120.

[9] **Stergiopoulou, E. and Papamarkos, N.** (2006). A New Technique for Hand Gesture Recognition, *2006 International Conference on Image Processing*, pp.2657–2660.

[10] **Akl, A.**, **Feng, C. and Valaee, S.** (2011). A Novel Accelerometer-Based Gesture Recognition System, *IEEE Transactions on Signal Processing*, *59*(12), 6197–6205.

[11] **q. Kong, J.**, **Wang, H. and q. Zhang, G.** (2009). Gesture recognition model based on 3D accelerations, *2009 4th International Conference on Computer Science Education*, pp.66–70.

[12] **Xu, R.**, **Zhou, S. and Li, W.J.** (2012). MEMS Accelerometer Based Nonspecific-User Hand Gesture Recognition, *IEEE Sensors Journal*, *12*(5), 1166–1173.

[13] **Xie, R.**, **Sun, X.**, **Xia, X. and Cao, J.** (2015). Similarity Matching-Based Extensible Hand Gesture Recognition, *IEEE Sensors Journal*, *15*(6), 3475–3483.

[14] **Cheng, H.**, **Luo, J. and Chen, X.** (2014). A windowed dynamic time warping approach for 3D continuous hand gesture recognition, *2014 IEEE International Conference on Multimedia and Expo (ICME)*, pp.1–6.

[15] **Lementec, J.C. and Bajcsy, P.** (2004). Recognition of arm gestures using multiple orientation sensors: gesture classification, *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No.04TH8749)*, pp.965–970.

[16] **Harding, P.R.G. and Ellis, T.** (2004). Recognizing hand gesture using Fourier descriptors, *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pp.286–289 Vol.3.

[17] **Ahsan, M.R.**, **Ibrahimy, M.I. and Khalifa, O.O.** (2011). Electromygraphy (EMG) signal based hand gesture recognition using artificial neural network (ANN), *2011 4th International Conference on Mechatronics (ICOM)*, pp.1–6.

[18] **Rajesh, V.**, **kumar, P.R. and Reddy, D.V.R.K.** (2009). SEMG based human machine interface for controlling wheel chair by using ANN, *2009 International Conference on Control, Automation, Communication and Energy Conservation*, pp.1–6.

[19] **Hashiyama, T.**, **Sada, K.**, **Iwata, M. and Tano, S.** (2006). Controlling an Entertainment Robot through Intuitive Gestures, *2006 IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pp.1909–1914.

[20] **Nguyen, T.N.**, **Vo, D.H.**, **Huynh, H.H. and Meunier, J.** (2014). Geometry-based static hand gesture recognition using support vector machine, *Control Automation Robotics Vision (ICARCV), 2014 13th International Conference on*, pp.769–774.

[21] **Wikipedia**, `https://en.wikipedia.org/wiki/Serial_communication`, date retrieved : 26.04.2017.

[22] **Open Source Robotics Foundation**, `http://www.ros.org/about-ros`, date retrieved : 23.11.2016.

[23] **Open Source Linux Operating System**, `https://www.ubuntu.com/`, date retrieved : 23.11.2016.

[24] **Python Programming Language**, `https://www.python.org/`, date retrieved : 23.11.2016.

[25] **C++ Programming Language**, `http://www.cplusplus.com/`, date retrieved : 23.11.2016.

[26] **UM7-LT Orientation Sensor**, `http://www.chrobotics.com/shop/um7-lt-orientation-sensor`, date retrieved : 22.11.2016.

[27] **FTDI chip website**, `http://www.ftdichip.com/`, date retrieved : 26.04.2017.

[28] **Akl, A. and Valaee, S.** (2010). Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, x00026; compressive sensing, *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp.2270–2273.

[29] **Wan, H.** (1993). Applying The Genetic Algorithm To Optimization Problems, *WIT Transections on Information and Communication Technologies*, *2*(12), 451–463.

CURRICULUM VITAE

Name Surname: Ubeyde Mavuş

Place and Date of Birth: 13.03.1992 ISTANBUL/TURKEY

E-Mail: umavus@gmail.com

EDUCATION:

- **B.Sc.:** 2015, Istanbul Technical University, Electric-Electronics, Control and Automation Engineering
- **M.Sc.:** 2017, Istanbul Technical University, Electric-Electronics, Control and Automation Engineering

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Mavuş U., Sezer V.** 2017. Head Gesture Recognition via Dynamic Time Warping and Threshold Optimization. *IEEE Conference on Cognitive and Computational Aspects of Situation Management - CogSIMA*, March 27-31, 2017 Savannah, Georgia, United States of America.