ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

Project Report
Version 1

CENG 408
Innovative System Design and Development II

Gesture Recognizer Strap using Motion Sensor

Bengi Günay
201217023

Advisor: *Faris Serdar Taşel*

Table of Content

Table of Figures

Table of Tables

Abstract

Today a lot of systems in different areas uses the gesture recognition technology to work. The improvement in the technology make this topic more popular. The purpose of project ‘Gesture Recognizer Strap using Motion Sensor’ is to develop an gesture recognizer software, which could be include an api with using wearable technology, The aim is to develop affordable multipurpose device. To reach the project aims there will be used an accelerometer, gyroscope, compass sensor. These sensors are the one of the most affordable solution for gesture recognition problem. As the wearable device a wristband will be used. They will be on the wristband to track the basic movements of arm of the user like linear movements of arm (such as up, down, left, right) or angular movements of the arm.

Key words:

Gesture Recognition, Gesture Recognizers Software, Wearable Technology, Accelerometer, Gyroscope, Compass Sensor

Özet

Hareket algılama günümüzde farklı alanlarda birçok sistem tarafından kullanılmaktadır. Teknolojinin gelişmesiyle bu konunun önemi daha da artacaktır. Hareket sensörleri ile hareket algılama projesinin hedefi giyilebilir teknoloji kullanarak hareketleri tanımlayan bir yazılım yapmaktır. Ayrıca bu yazılım bir uygulama programlama arayüzüde içerebilir. Projenin bir diğer hedefi ise geliştirilen yazılım ve giyilebilir cihazın ucuz ve çok amaçlı olması yönündedir. Hareket algılamak için ucuz ve etkili çözüm yollarından biri ivme, jiroskop ve manyetik sensör kullanmaktır. Bu projede de bu sensörler kullanılacaktır. Giyilebilir cihaz olarak bir bileklik kullanılacaktır ve bu bahsedilen sensörler bu bilekliğin üzerinde olup bileğin yaptığı basit hareketleri takip edecektir. Bu hareketler bilek ile yapılan üste, alta, sağa, sola yada herhangi bir yönde yapılan düz hareketler veya bilek çevirme hareketleri olabilirler.

Anahtar Kelimeler:

Hareket algılama, Hareket algılama yazılımı, Giyilebilir teknoloji, İvme sensörü, jiroskop, Manyetik sensör

1.INTRODUCTION

1.1 Problem Statement

Gesture recognition is the process of getting data from a sensor and using an algorithm understand which gesture corresponds that data in the database. Gesture recognition is a brand topic in computer science. Today this technology is used in healthcare sector, automotive sector, transit sector, gaming sector, defense sector and electronics sector. There are a lot of studies in literature and some of them will be explained in second part of this document.

1.2 Solution Statement

Our solution for gesture recognition problem is to develop a software with using a wearable technology. Our wearable device will be an wristband consist of accelerometer, gyroscope and compass sensor. Also our software may include an application programming interface beside recognizing gesture. Our solution will be cheap and multipurpose as usage.

1.3 Scope of Document

This document will include related projects from literature, what type of technologies can be used for gesture recognition, which one will be used in the project, the equipment and the algorithm for the project, software requirement specification, general architecture of system and design documentation. Second part of this document consist of literature review. Third part explains software requirement specification. Last part includes software design part.

2. LITERATURE REVİEW

2.1 Introduction

Gestures are crucial part of human communication. Gestures can be defined as meaningful body motions including physical movements of the hands fingers and etc. to make a point or communicate with environment. Since it is a strong communication way to express ourselves, scientists think it to use gestures and postures for human and computer (or robot) interaction. The research about gesture recognition started at earlies of the 1960 with light emitting pen to control sketchpad computer aided design systems. The first research for camera

based computer vision for gesture recognition began at early 1990s at MIT Media Lab and University of Zurich [1].

The problem with the gesture recognition is to make computer understand the meaning of gestures. Today there are different technologies for gesture recognition. These technologies categorized under different sub-title in every paper related with the gesture recognition. Liu and Wand [2] separates these technologies under two sub-categories as Image based, and Non-image based. Image based technologies includes a single camera, a stereo camera and a depth sensor. Non-image based technologies includes wearable materials such as a glove or a band or non-wearable technologies. This paper was written based on this classification which can be seen at figure 1.

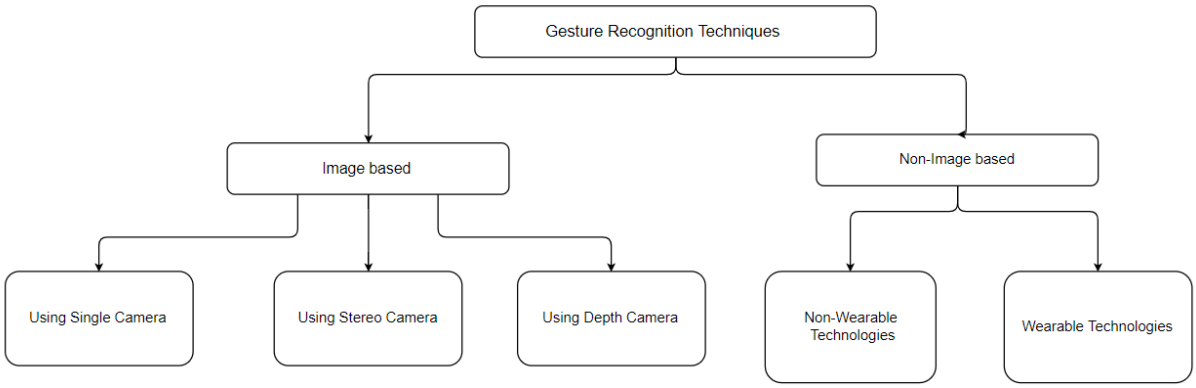

Figure 1:Gesture Recognition Techniques

It is possible to mention also there could be hybrid technology. In some studies image-based and non-image based technology is used together to compensate weak points of each other. However, this hybrid methodology is not common [3].

The remainder of this part is organized as follow: Section 2.2 summarizes the gesture recognition technology. Section 2.3 gives application areas and some example from previous works. Section 2.4 is the overview of section 2.

2.2 Technologies for Gesture Recognition

2.2.1 Image Based Gesture Recognition

The first type technology for gesture recognition is image based tech. In this kind of technology, the movement of the body is recorded by a camera(s) or a depth sensor. If the a single or multiple camera is used, the video coming from camera(s) turns into the frames and these frames are filtered to remove non-essential information and empower the essential part of

it. [4] After recognizing the gesture, the computer has to compare the gesture that is defined before. If a single camera is used there will be a angle restriction for the recognizing gesture but the process will be fast. If stereo camera is used, the 3D environment can be perceived. The disadvantage of stereo camera is to have calibration difficulties and problems such as computational complexity [2]. If the depth sensor is used, the computer has the deep map of the what is seen in the camera in a short range. To using this depth information, it is easy to have an 3d model of the gesture [5].Depth sensor is an easy and cheap solution to recognition problem. Advantages and disadvantages of image based gesture recognition summarized in table 1.

Technology	Advantages	Disadvantages
Single Camera	Easy Setup	Low robustness
Stereo Camera	Robust	Computational Complexity, calibration difficulties
Depth Sensor (ToF Camera)	High Frame Rate	Resolution depends on the light power and reflection

Table 1: Advantages &Disadvantages of categories of the image based technology

2.2.2 Non-Image Based Gesture Recognition

The second type of technology is non-image based technology which includes wearable (such as gloves and bands) and non-wearable ones.

2.2.2.1 Wearable Technology

The wearable technology depends on physical interaction with the user. The wearable device could be a band or a glove. Moreover, different kind of sensors are integrated in this wearable material to understand position of the gesture. In the wearable technology generally, accelerometers and gyroscopes are used to get relative orientation information. Also in some studies ultrasonic tracking systems (as microphones) and some magnetic sensors is used separately [3]. The wearable technology generally learns users' gesture data before using the system then recognizes any unknown gestures by comparing the training data [6]. General disadvantages of wearable technologies is the lack of user customization and some calibration problems. To get more accurate result for body segment orientation without external actuators or cameras the internal and magnetic measurement unit is started to be used. It is a cost effective and easy to integrate solution [7].

2.2.2.2 Non-Wearable Technology

Last subject of non-image based technology is non-wearable types. Non-wearable sensors can detect gestures without any interaction with the human body. It uses radio frequency to identify to gestures. Google has a project name Soli to recognizing gestures based on radio frequency. Also, MIT has a study named WiTrack that a user motion can be captured by radio frequency that reflects from human's body in a specific place [2].

2.2.2.3 Comparison of Wearable and Non-Wearable Technology

Comparison between wearable and non-wearable technology is given below at table 2.

Used Technology	Advantages	Disadvantages
Glove (wearable)	Fast response, precise tracking	Cumbersome device with a load of cables
Band (wearable)	Fast response, large sensing area	Contact with human body
Non-wearable	Avoid contact with human body	Low resolution, technology not mature enough

Table 2: Advantages &Disadvantages of Non-image based technology

2.3 The Applications Areas and Previous Works

Gesture recognition can be used in different areas. These areas [5] :

- sign language recognition
- socially assistive robotics
- directional indication though pointing
- control with facial gestures
- alternative computer interfaces
- immersive game technology
- virtual controllers
- affective computing and remote control

2.3.1 Previous Works

There are a lot of studies related gesture recognition. Some of them are given below as example:

A disk jockey support system: This system recognizes the gesture using accelerometers on both hand and helps DJ to configure volume and start/stop music [6].

NaviGaze: NaviGaze is a system that enables the user to control computer without a mouse, instead it uses head movements as mouse movements and eye blinks as clicking option. This system is designed for disabled people who cannot move their body [1].

Qosmio Laptops: In 2008, Toshiba announce that they will produce a laptop that is a gesture aware computer. The windows applications, power point presentations, windows media player can be used with gestures [4].These laptops uses its webcam to recognize 3 different gestures in a specific range.

Sixth Sense Device: A gesture-based wearable computer system is developed at MIT Media Lab. It consist a pocket projector, a mirror, and a camera. These hardware s has a communication with a pendant like device which user wears. The projector gives the visual information about the environment to be used as interfaces while camera tracks the hand gestures of the user. These gestures are turned into instructions for projected application interfaces [5].

Hand Writing on Air: A system is developed that recognizes alphabetic characters written in the air with a cell phone by converting acceleration data in to restrictions [6].

2.3.2 Previous Work Related with Unity 3D

Unity 3D is a multiplatform game development environment which is a fully integrated dedicated game engine. It provides powerful scene simulation and the visualization of the user hands or body that are tracked by the sensors [8]. Because of these reasons it is commonly preferred in the researches related gesture or posture recognition. A few of the researches is given below:

3D Hand Cursors to Explore Medical Volume Datasets: This research aims to develop a touchless interface for medical volume images. The hand and body gestures is used to rotate and position the medical volume images in 3-dimensions, where each hand acts as an interactive 3D cursor. This research uses a depth camera (Kinect One) to track the hand moves to make it "touchless" interface without any physical contact or wearable device. A pc which is connected to depth camera and a second pc is used for the run the application called Voxel Explorer and

display the generated graphical content. This application was developed using Unity 5.1 and Kinect SDK v2 [9].

VR System for Electrician Training: This experiment uses Kinect depth camera and unity 3d. Unity is used for management of 3D system and display platform. The Kinect development can be easily provided in windows 7 environment. A scene and model is created in 3d Max and exported to unity 3d engine. Then user should trigger one of gesture that is thought before. If the gesture is triggered successfully the position of the hand will be in the scenes as an icon [10].

2.3.3 4. Conclusion

Thanks to our developed computer technology computers are able to help people with complex task. To make an easier communication between computers and humans the gesture recognition systems are crucial. This is also one of the aim for the project 'Gesture Recognizer Strap with Motion Sensor'. Other goals of this project produce a cost efficient multipurpose a controller device. There are 2 types of gesture recognition technology introduced in this paper: image based and non-image based. These technologies are mentioned in this paper with their advantages and disadvantages. Moreover, the application areas with some studies are given as examples. A comparison between 2 technologies [3] is given below.

Comp. Factor	Glove	Band	Camera	Depth
Cost	Cheaper	Cheaper	Costly	Cheaper
User Comfort	low	Average	good	good
Computing Power	Low	Low	High	High
Accuracy	High	High	High	High
Noise	Average	Average	Minimal	Minimal

Table 3: Comparison of image based and non-image based technologies

Due to this table the band (Non-image based technology) is decided to use for the 'gesture recognizer strap with motion sensor' project.

3. SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Introduction

3.1.1 Purpose

The purpose of this part is to give a detailed description about the project Gesture Recognizer Strap using Motion Sensor. This project aims to develop low-cost, multi-purpose control device. The software requirements of this project and limitations of this project is specified in this part. This part is written for both stakeholders and system developers.

3.1.2 Product Scope

Intent of this project is to develop a low cost, multipurpose remote controller wearable device and its software for human computer interface and virtual reality systems. A wristband as wearable device for using to identify the human gestures. This wristband has its own accelerometer, gyroscope and compass sensor to determine the orientation of the object. The gyroscope will be used to tract rotation while the accelerometer and compass provide an absolute frame to reference.

This project includes reading data from sensors and extracting features from this data. Considering the training dataset, an algorithm will be applied to detect the gesture. There could be an application programming interface addition to work with Unity 3D.

3.1.3 Glossary

TERM	DEFINITION
User	People who wears the wristband (developer for this project)
Database	Location of all gestures identification data on this system.
System	A system which gets data from sensors on wristband and understands which gesture is made.
Gyroscope Sensor	A sensor which catches the direction of rotation.
Compass sensor	A sensor which is used for navigation and orientation
Accelerometer	A sensor which measures acceleration
API	API means application programing interface
Gesture	A movement of a part of body
Posture	Place(someone) in a particular pose

3.1.4 Overview of Part

The second part of this section describes the overall system with showing a diagram. Moreover, the second part contains the functionalities of the project and user characteristic. Last part gives the detail about requirements of system. It has three subtitles: external interface requirements, functional and non-functional requirements.

3.2 Overall Description

3.2.1 Product Perspective

Gesture Recognizer Strap using Motion Sensor project recognizes the gestures user makes using a wearable technology which is a wristband. The project is responsible to only recognizes gestures not the postures. The purpose of the project is to developed remote controller device.

There is only one actors in this project: developer. The developer has a wristband on his/her arm. Sensors on the wristband tracks the moves of the arm. The developer has some basic properties. In the section 2.3 the characteristics that the developer have is mentioned.

A use case of the project can be shown in figure 1.

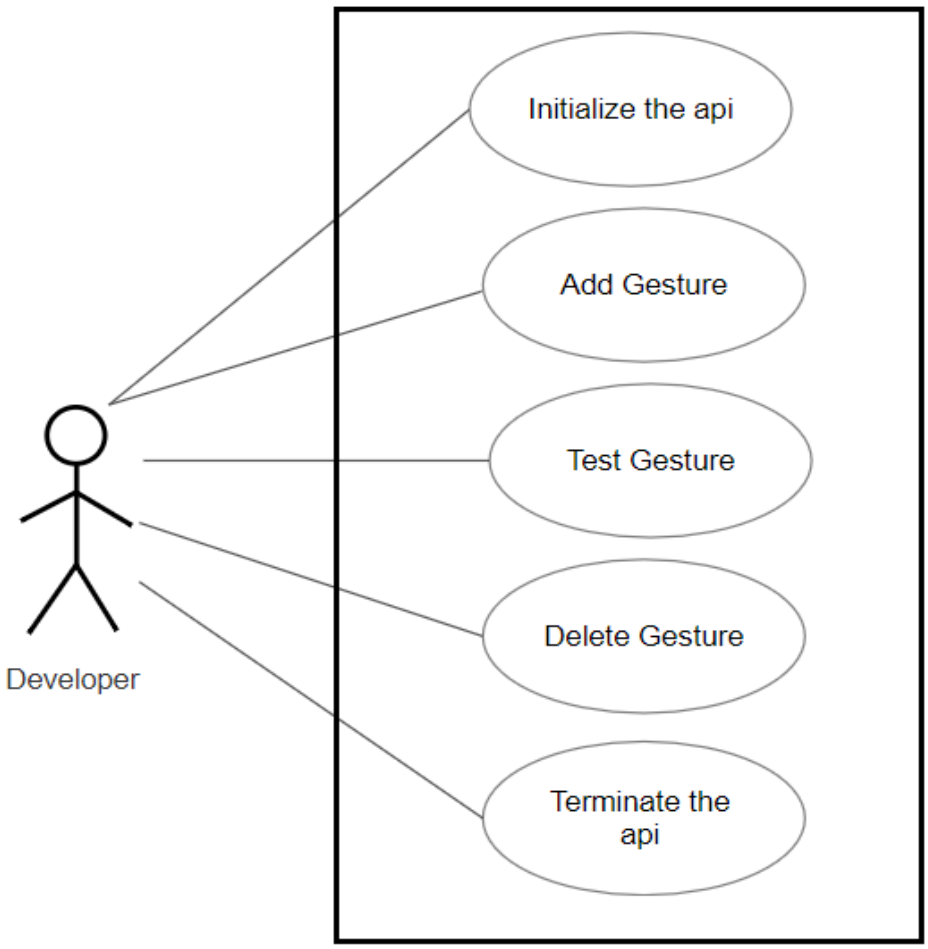

Figure 2: Use case of overall project

3.2.2 Product Functions

The system basically has 2 different modes: a training mode and a recognition mode.

In the training mode system takes raw data from the sensor, then it detects the gestural information using an algorithm. This gestural information is saved to the database. The activity diagram of training mode can be seen at figure 2.

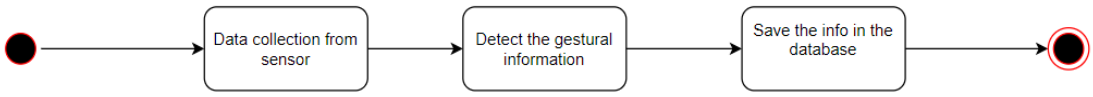

Figure 3: Activity diagram of training mode

In the recognition mode the systems again take raw data from the sensor and detects the gestural information. Compares this information with the predefined gesture information (from training) to see if there is a match. If there is a match system recognizes the gesture. System gives results to developer about if the gesture is recognized or not. The activity diagram of recognition mode can be seen at figure 3.

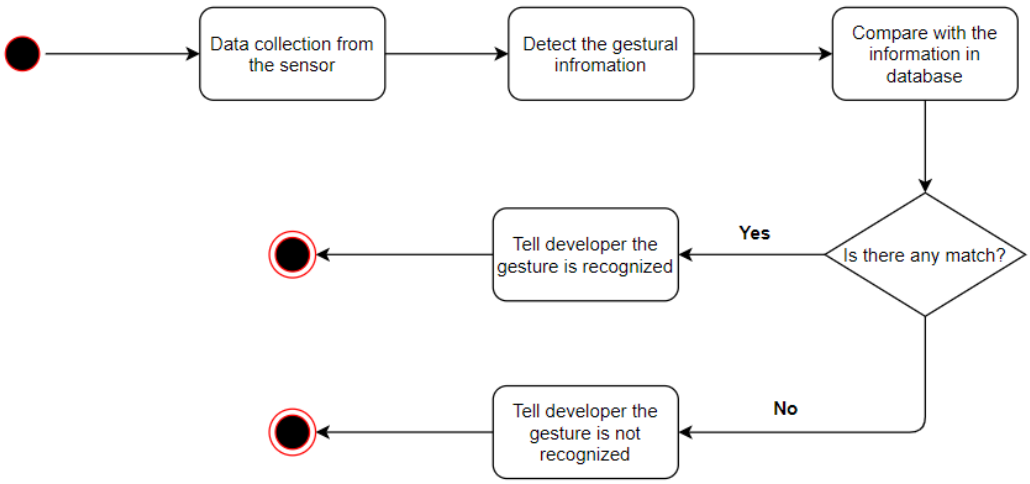

Figure 4: activity diagram of recognition mode

System functions can be summarizing as:

- System shall get the data from sensors.
- System shall detect the gestural information from the incoming data.
- System shall hold the data for each identified gesture in database.
- System shall determine a gesture is in its database or not.
- System shall give a reaction to the developer indicating that if it finds the gesture in its database or not.
- System should provide an API between its own program and unity 3D.

3.2.3 User Characteristics

Developer:

- The people who can wear (at least they have one leg or arm) and do not have an issue about moving leg or arm the wristband can uses this product.
- The people who have the programmer background or the necessary information to work with unity 3d environment.

3.2.4 Operating Environment

The software of the product will be designed for Windows operating system. It could be work at any kind of computer that has Windows as its operating systems.

3.2.5 Design and Implementation Constraints

The product will have only one motion sensor including an accelerometer, gyroscope and compass sensor. Since the wearable device will be the wristband, there will be a hardware limitation at tracking the gestures. The movements of arm or leg can be detected but the hand tracking can not be detected because of that limitation.

3.3 Requirement Specification

3.3.1 External Interface Requirements

3.3.1.1 User Interfaces

The user interface will be work on Windows operating system. It will be a command line interface that shows the user if the gesture is recognized or not. If the system recognized the gesture it will return “Recognized!” as a result. Moreover, the name information of the gesture or a number related to recognized gesture will be provided by the system. If the gesture is not recognized, the system cannot recognize the gesture or gesture is not in its database it will return “Not recognized gesture” as the result.

3.3.1.2 Hardware Interfaces

In this project a sensor (myAHRS+) will be used. This sensor has no driver, but it has to be connected by a computer via usb port. So an usb port will be needed.

3.3.1.3 Software Interfaces

In the case of an api between Unity 3d the gesture recognizer program, the unity 3d has to be installed on the computer that will be used.

For the myAHRS+ sensor, its own library will be used.

3.3.1.4 Communications Interfaces

There are no external communications interface requirements.

3.3.2 Functional Requirements

There is only one actor like it is mentioned before: developer.

3.3.2.1 Developer Use Case

Uses Cases:

- Initialize the api
- Test Gestures

- Add Gesture
- Delete Gesture
- Terminate the api

Diagram: The diagram is given in figure 1.

Brief Description: For this project developer basically has 5 capabilities. One of them initializing the api to have the gestural information in the unity 3d platform. The other duty is to test gestures by making a gesture to find out if the gesture is recognized. Moreover, the developer has to train the system by adding and deleting the gestures. Also, developer has to be capable of to terminate the api.

a. Use Case: Initialize the api

Initial Step-by-Step Description:

1. Developer runs the unity 3d
2. Developer initialize the api
3. Developer gets sensor data in unity 3d

Exception Path:

1. Developer runs the unity 3d
2. Developer initializes the api
3. Developer fails to get sensor data in unity 3d

b. Use Case: Terminate the api

Initial Step-by-Step Description:

1. Developer terminates the api.

Exception Path:

2. Developer fails to terminate api.

c. Use Case: Test Gestures

Initial Step-by-Step Description:

1. Developer wears the wristband.
2. Developer runs the application.
3. Developer opens the recognition mode.

4. Developer make gestures.
5. The system gives responses to these by saying to developer if it recognizes the gesture or not.

Exception Path:

1. Developer runs the application.
2. Developer opens the recognition mode.
3. Developer make gestures.
4. The system does not perceive any gestures.
5. The system starts to wait for an input gesture until the application terminated.

d. Use Case: Add Gesture

Initial Step-by-Step Description:

1. Developer wears the wristband.
2. Developer runs the application.
3. Developer opens the training mode.
4. Developer make gestures.
5. The system saves these gestures to its database

Exception Path:

1. Developer wears the wristband.
2. Developer runs the application.
3. Developer opens the training mode.
4. Developer make gestures.
5. The system fails to save this gesture to its database

e. Use Case: Delete Gesture

Initial Step-by-Step Description:

1. Developer wears the wristband.
2. Developer runs the application.
3. Developer opens the delete mode
4. Developer enter the id number of gesture.

5. The system deletes the gesture from its database

Exception Path:

1. Developer wears the wristband.
2. Developer runs the application.
3. Developer opens the testing mode
4. Developer enter the name of gesture or the number of gesture.
5. The system fails to delete the gesture from its database

3.3.3 Non-Functional Requirements

Some important non-functional requirements for the system are listed below.

- **Accessibility:** The gesture information should be kept in json objects.
- **Usability:** The system accepts the movement as gesture if the acceleration is at least 0. 001g in the direction of movement. Moreover, the movements cannot take too short time or too long time to be accepted as a gesture.
- **Performance:** The system should recognize the gesture in at most 5 seconds.
- **Safety:** Sensor on the wristband should be protected from outside factors

4. SOFTWARE DESIGN DOCUMENT

4.1 Introduction

4.1.1 Purpose

The purpose of this part describes the details about how the software system will be structured of the project 'Gesture Recognizer Strap using Motion Sensor' to satisfy its requirements. This project aims to develop low-cost, multi-purpose control device. A wristband will be used as control device. This wristband will have a sensor named myAHRS. This sensor is a combination of a gyroscope sensor, an accelerometer sensor, and a compass sensor.

The audience of this document is developers who wants to develop and use a wristband to catch movements and take the information related with the movements in the other environments such as UNITY 3D.

4.1.2 Scope

The scope of this projects includes the development of an software which has to be capable of:

- Data collection from sensors
- Detection of gestural information
- Saving the gestural information
- Comparison of 2 different gestural information
- Giving response to the user

Additionally, to develop an api for UNITY 3D is desirable but this is not a requirement.

The software has two different modes for the usage. These modes are training mode and recognition mode. In the training modes the movement will be introduced to the software and their information will be kept in the json objects. In recognition mode a movement will be recognized by the software due to sensor data and a comparison will be made with the training data. The software will decide if there is a match between a movement and the training data and give a response to the user.

4.1.3 Glossary

Term	Definition
myAHRS+	A sensor is to get data for calculation of position. It includes a compass sensor, an accelerometer and gyroscope sensor.
Gyroscope Sensor	A sensor which catches the direction of rotation
Compass Sensor	A sensor which is used for navigation and orientation
Accelerometer	A sensor which measures acceleration
Json	(JavaScript Object Notation) is a data-interchange format
CLI	Command line interface

4.1.4 Overview

The second section of the document describes system architecture. It includes the methods and algorithms that will be used to develop this project. Also, it includes class diagram of the project and description of the functions. Third section includes data design and which structure will be used as database. Last chapter describes the user interface of the software.

4.2 System Architecture

4.2.1 Methodology for Gesture Recognition

The template matching method will be used for gesture recognition. This methodology contains two steps. In the first step a template will be created by collecting data values from the sensor. The second step is to take data values sensor and compare them with the templates to find the template most closely matching the current data record.

Strength of this methodology:

- Simple technique
- Accurate for small set of gestures

Weaknesses of this methodology:

- Does not work well with large sets due to overlapping templates
- Harder to implement to gestures than postures

4.2.1.1 Creating a Template

The training modes starts with the recording data when a movement is sensed by the sensor. Sensor values is tracked by the program and when program detects a movement (change in acceleration), it assumes that the gesture is started. The detection of the movement is calculated from this formula given below:

mg=|1-$a_x^2 + a_y^2 + a_z^2$|

where the a_x is the acceleration in x axis in g (acceleration of the gravity)

the $a_y$is the acceleration in y axis given in g

the a_z is the acceleration in the z axis given in g

the mg is the name of these calculation (nothing special)

If the mg>0.25, then the program assumes the gesture has just started. The value of comparison is picked as 0.25 due to experiments.

In same way program tracks the sensor values and when it detects there are no movements for 3 seconds, it assumes that the gesture is finished. A maximum gesture time is specified as one minute. Even there is longer gestures is tried to make by the user, program will record the first 60 seconds of it. The other limitation for recognizing a movement is mentioned in the SRS document of this project.

The template will be created in the training module. To creating the template every gesture must be performed 5 times. The template is hold in an array with the size of 120.Each accelerometer data and angle data take 20 size part of the array. The time value is planned to the hold in the feature vector but due to the dominant acceleration values, it's not needed.

The structure of the template can be seen in the figure given below.

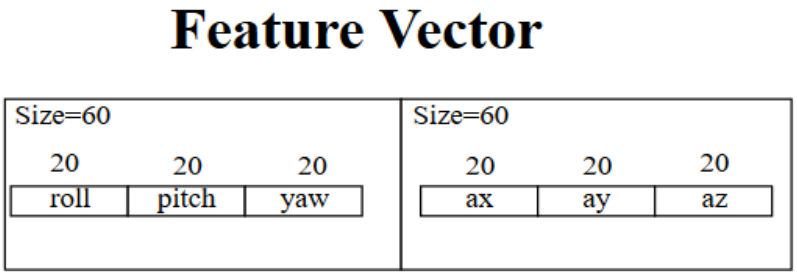

Figure 5 Structure of the feature vector

Each gesture can take different time interval when every time they are performed. This situation makes harder to create a template, since similar sensor values performed at different times. The taking directly the average of the sensor values and store as template would be misleading. Because of this operation the template could be irrelevant from the its gesture. So, in the recognition mode (testing) when the developer performs the gesture, it can not be recognized by comparing the template. To solve this problem, a linear extrapolation method is used in this project. The extrapolation is a way to make an estimation based on extending a known sequence of values beyond the area of certainly known. In a general sense, to extrapolate is to infer something that is not explicitly stated from existing information. As an example, assuming a gesture is performed three different times and we have three different duration values as 0.7s, 0.6s and 0.45s. Each vector of this gesture will be extrapolated to 1 second and the template will be calculated (getting average of all) from these extrapolated vectors.

a. Problems with Raw Data

The template matching methodology stores the raw sensor values in the templates. The raw sensor values are Euler angles and the acceleration values. If you make your arm parallel to the ground and turn around yourself, the sensor values will be different for every direction, but the posture is not changed during this movement. Same thing is valid for the gestures. If you move your arm from left to the right the sensor values will be different according to your arms direction during in this time interval. This means after the training, you must find your exact training position including the direction to make your gesture recognize. One way to solve this problem is to store the angle differences not the angles (not the position.). Therefore, looking at the differences the movements is same or not same can be specified. Another way to solve this problem is to make a calibration for one time, accepting there as zero point, and assuming like every gesture started from the calibration point. The system knows the angle difference between the data in the training mode and the data coming from the testing (input data) and the system can rotate the input data (testing data) and make a comparison between templates. Since the taking angle differences didn't give the expected good results, the calibration methodology is used in the project.

4.2.1.2 Recognition

In the training, a set of templates will be stored for the gestures. In the recognition mode these templates will be compared with the data belongs the input gesture to be performed for testing. The closest data will be counted as a match. If the gesture data is not close the templates, then there will be no match and program returns negative output to the developer.

There are some different recognition techniques. One of them will be used.

a. Comparison

A Boolean function returns the result of explicit comparison between each sensor value in the input data record and corresponding value of templates. The comparisons are often made with a range of values to improve the accuracy. This method has problems for too detailed measurement because of the range of the values. Since in this project there is no detailed movements, there wont be any problem to use this method.

b. Distance Measurements

Another method for template comparing method is the distance measurement. In this method the distance between the input test data and the template data will be calculated. The

lowest distance will be candidate for the recognized gesture. There must be a threshold value to avoid positive false recognition. There are 2 different ways to calculate distance first is to consider the sum of the absolute difference and second is to consider the sum of the squares.

c. Dynamic Time Warping

Dynamic time warping algorithm is an optimal alignment algorithm for 2 sequences. It creates a cumulative distance matrix that warps the sequences in a non-linear way to match with each other. This algorithm has planned to use for this project at first but after the selection of template matching method with extrapolation, there is no need to use this algorithm. In literature there are some studies that the template matching method and dynamic time warping is used together for gesture recognition. Therefore, this algorithm kept as plan b and will be used in the case of comparison and distance measuring fails.

4.2.2 Class Diagram

The class diagram of this project can be seen at figure 1. Basically, there are 3 classes for this project. One is the gesture object. It has its own data coming from sensor which are Euler angles and accelerometer values for each axis. Since a gesture contains more than one posture, these variables will be an array to hold all different position of the arm in a gesture. The template of the gesture will be hold at featureVec variable. Also, Gesture has its own unique id to define itself. Lastly making a gesture takes time. Therefore, gesture will have a time interval field which named as duration. Other Class is the sensor class which is used by the gesture class for the calibration and taking sensor values correctly. Last class is named as User Interface. The user of this project is developer. The capabilities of developer are given in the user interface class. The relationship between the classes are the association.

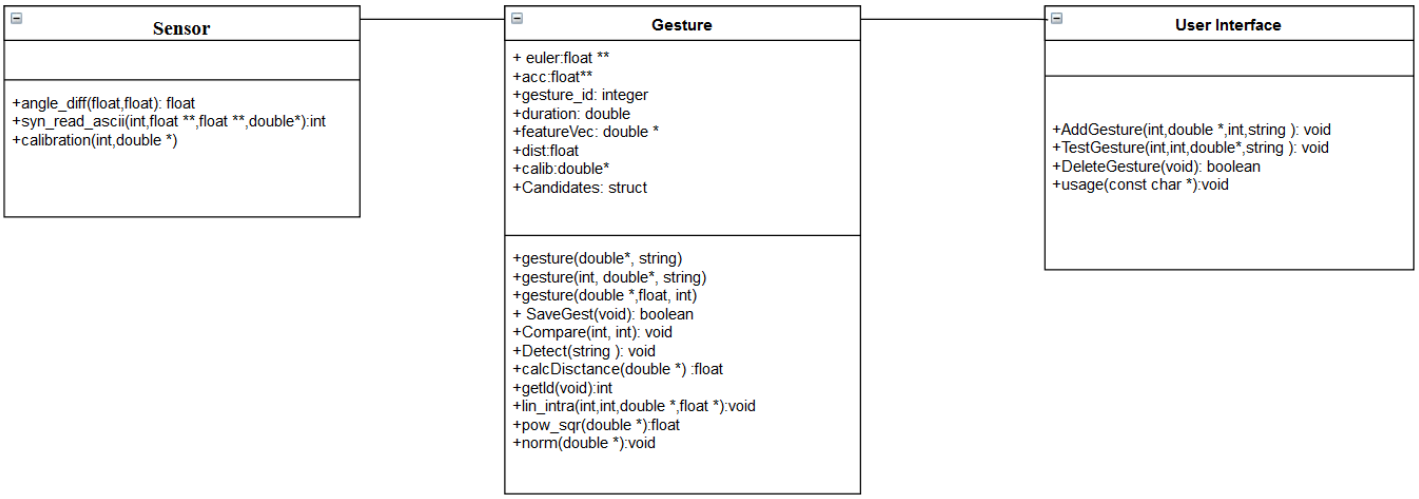

Figure 6: Class Diagram

4.2.3 Decomposition Description

4.2.3.1 Gesture Class

The function of the gesture class is mentioned below

- ***Detect:*** This function is responsible for creating template.
- ***Compare:*** This function is responsible to classify the gestures by making a comparison somehow between the input data and the templates.
- ***SaveGest:*** This function is responsible to store the data related with the gesture. It returns a Boolean to show the operation is successful or not.
- ***calcDistance:*** According to the gesture samples from training calculates a distance threshold value to the feature vector. It takes the gesture sample as input and returns the calculated distance as float.
- ***getId:*** It returns the id of the gesture
- ***Lin_intra:*** This function is responsible from the linear interpolation. It takes a sample gestures and fit it to array with 60 sizes.
- ***Pow_sqr:*** Takes an array. Takes all the elements square of the array and add them. Lastly it takes the square root of the total value and return it.
- ***Norm:*** This function is responsible for the normalization.

4.2.3.2 User Interface Class

The function of the user interface class is mentioned below.

- **AddGesture:** This function is responsible to add gesture in the training. Therefore, it takes an gesture as a parameter and returns Boolean.0 shows the operation is not successful and 1 shows the operation is successful.
- **TestGesture:** This function is responsible to recognition of the gesture. Takes an input gesture and find outs if one the templates match with it. It returns a true (1) if there is a match.
- **DeleteGesture:** This function is responsible to delete gestures. Therefore, it takes an gesture as a parameter and returns Boolean.0 shows the operation is not successful and 1 shows the operation is successful.

4.2.3.3 Sensor Class

The function of the gesture class is mentioned below:

- ***Anglediff:*** This function is responsible to take the angle difference to record values. The angle differences are taken so all gestures can be assumed at the started from the calibration point. This function takes 2 float number and return the difference of those 2 numbers as float.
- ***Syn_read_Ascii*:** This function is responsible to recording acceleration and Euler angle data for the gestures at most 60 seconds. It returns the count number of the recorded data.
- ***Calibration:*** This function is responsible to taking calibration values.

4.3 Data Design

4.3.1 Data Description

All gestural information including raw data coming from the sensors and template will be held into the gesture object. In the training, some of the gestures is recorded to the system. They should be stored as permanently. Therefore, there is a database to hold gesture objects in the training. On the other hand, the gestures for using testing is not stored in the database.

Since there is no complicated structure that need to be store, there won't be needed to use a real database like sql. It will be a proper to use JSON objects as simpler solution. JSON is a lightweight data-interchange format. It is easy to read and write. The text format of the JSON is independent of the language that will be used.

The JSON object includes a set of name/value pairs can be seen in figure 2 given down below. The name value is the Gesture_ idNumber and the value will be a gesture object for this project. They are holded like this in the JSON object:

{Gesture_1: Gesture_object1, Gesture_2: Gesture_object2, …..}

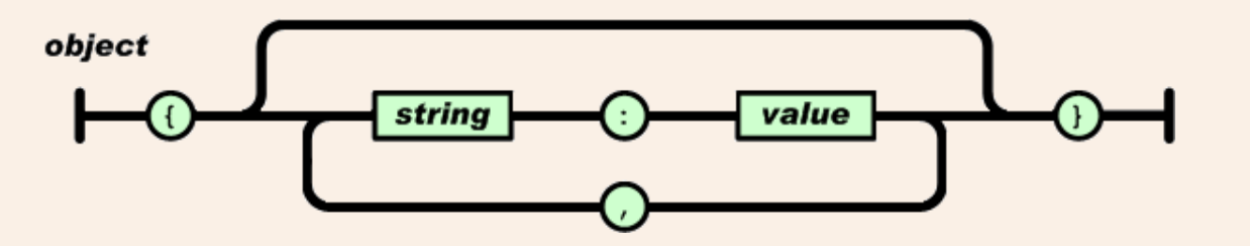

Figure 7: JSON object holds name/value pairs

Also calibration values, count of the current gesture in json object and deleted gesture count is holded in some other json file to detect the available first id number.

There are a lot of C/C++ library to use for reading from/write to the JSON objects. The nlohmann's json library is picked and used for the project.

4.4 Human Interface Design

4.4.1 Overview of User Interface (User Manual)

This application uses command line interface. To run this application, the user should open the command line interface. Move to the directory that includes the application or user can enter the directory as path.

Application takes tree parameter:

- Path
- Method name
- The Serial Port of the sensor

Path can be the full directory or the name of application if the current directory is the one that has the application.

Method name can be train , test and delete.

The serial port name will be something like COM3, COM4, COM5.User should check the serial port every time he/she connects sensor to the computer. It can be checked from Computer Management >> Device Manager >> Ports >> Usb Serial Device(Port name is written here) in Windows 10.

An example usage given below:

```
usage : C:\Users\bengi\OneDrive\Belgeler\GitHub\ceng-407-408-project-gesture-recognizer\ConsoleApplication4-v19\x64\Debug\ConsoleApplication4.exe "Mode name"  "serial port list "
        ex) $ C:\Users\bengi\OneDrive\Belgeler\GitHub\ceng-407-408-project-gesture-recognizer\ConsoleApplication4-v19\x64\Debug\ConsoleApplication4.exe train COM3

       Mode name can be "train" or "test" or "delete"
```

Figure 8 CLI- Example Execution

According to the mode that user enter before running the application, application will make different operations. If the application is run for the first time, in the train mode it will make an calibration and user will record train gestures. In test mode user made a gesture and system will compare if its exist between the recorded gesture. Delete mode requires the id number of the gesture which is wanted to be deleted. If user enters and invalid number, program gives error. If the gesture with the id number exist, it will be deleted.

5. TEST PLAN, TEST DESIGN SPECIFICATIONS AND TEST CASES

5.1 Introduction

5.1.1 Version Control

Version No	Description of Changes	Date
1.0	First Version	March 13, 2018

5.1.2 Overview

All of the requirements which was specified in the srs document [1] will be tested.

5.1.3 Scope

This document includes the test plan of use cases, test design specification and test cases corresponding to the test plan.

5.1.4 Terminology

Acronym	Definition
API	Application programming interface
JSON	(JavaScript Object Notation) is a data-interchange format
CLI	Command Line Interface
TM	Training Mode
GTM	Gesture Testing Mode

5.2 Features To Be Tested

This section lists and gives a brief description of all the major features to be tested. For each major feature there will be a Test Design Specification added at the end of this document.

5.2.1 Training Mode (TM)

In the training, main purpose is to take the raw values from sensor, then extracting features and lastly saving features to a JSON object. The capability of the meeting with these requirements will be tested.

5.2.2 Gesture Testing Mode (GTM)

In the testing mode, main purpose is to create a new gesture by taking input from sensor and comparing it with the ones in training mode, to detect if this gesture is done before or not. So basically the comparison will be tested between gestures.

5.2.3 API

Since there will be a api between the gesture recognition program and Unity 3d, the api should be tested.

5.2.4 CLI

This project requires a simple command line interface which takes 2 arguments from user when the program is run. Also, CLI should return required output for action of the developer. The cli will be tested to be sure it works right.

5.3 Features Not to be Tested

The input which is taken from the sensor won't be tested. Although these raw data will be used and has big importance for the project, the testing the sensor values is not in the projects scopes.

5.4 Item Pass/Fail Criteria

This section describes the general rule to use to decide when a test case passes and when it fails.

5.4.1 Exit Criteria

The product will be considered as successful under these conditions:

- For training mode and CLI features:
  - 100%of the test cases are executed.

- 80%of the test cases passed.

- For testing mode and API features:
  - 100% of the test cases are executed.
  - 80% of the test cases passed.
  - All high and medium priority test cases passed.

5.5 Test Design Specifications

5.5.1 Training Mode

5.5.1.1 Sub features to be tested

a. Create a Gesture(TM.CG)

User should be able to create a gesture by entering the train mode. The program should be able to detect start and end of the gesture correctly.

b. Extracting Feature Vector(TM.EFV)

Program should be able to take sensor values 5 times at the training and creating a feature vector from them.

c. Save Gesture(TM.SG)

After a gesture is created by the user, program saves the gesture in to a JSON object automatically.

d. Delete Gesture(TM.DG)

User should be able to delete a gesture that is created before.

5.5.1.2 Test Cases

Here list all the related test cases for this feature

TC ID	Requirements	Priority	Scenario Description
TM.CG.01	*3.2.1.4*	H	Create a Gesture
TM.EFV.01	*3.2.1.4*	H	Extracting Feature Vector
TM.SG.01	*3.2.1.4*	H	Save Gesture
TM.SG.02	*3.2.1.4*	H	Save Gesture Unsuccesfully
TM.DG.01	*3.2.1.5*	L	Delete Gesture

5.5.2 Gesture Testing Mode

5.5.2.1 Subfeatures to be tested

a. Create a Gesture(GTM.CG)

For testing user should make a gesture. These gestures and the other gesture created by different constructor (with different parameters) so creation of gesture should be tested in also testing mode.

For unsuccessful creation case:

If the user makes a gesture which has duration more than 1 minute (current value on the code) , the all of the data cannot be recorded. Therefore, the program should recognize this and give a warning about it.

b. Compare Gestures (GTM.CoG)

Program should be able to compare with the gesture which is newly created with the ones saved in JSON object.

c. Show the Results (GTM.SR)

User should be able to see the result of testing after enters a gesture

d. **No Training Data Case (GTM.NTD**)

If there is no gesture has created at the training mode before and user pick the testing mode immediately, since there is no gesture recorded in JSON object, program should give an error about it.

5.5.2.2 Test Cases

Here list all the related test cases for this feature

TC ID	Requirements	Priority	Scenario Description
GTM.CG.01	*3.2.1.3*	H	Create a Gesture
GTM.CG.02	*3.2.1.3*	H	Create a Gesture Unsuccessfully
GTM.CoG.01	*3.2.1.3*	H	Compare Gestures
GTM.SR.01	*3.2.1.3*	H	Show Successful results
GTM.SR.02	*3.2.1.3*	H	Show Unsuccessful results
GTM.NTD.01	*3.2.1.3*	H	No training Data Case

5.5.3 API

5.5.3.1 Subfeatures to be tested

a. Initialize the API (I)

User should be able to initialize api between unity 3d and the program by entering the init parameter at the start of the program.

b. Terminate the API (T)

User should be able to terminate api between unit 3d and program, after initializing an api by entering the terminate command.

5.5.3.2 Test Cases

Here list all the related test cases for this feature

TC ID	Requirements	Priority	Scenario Description
API.I.01	*3.2.1.1*	H	Initialize the Api.
API.I.02	*3.2.1.1*	H	Initialize the Api Unsuccesfully
API.T.01	*3.2.1.2*	L	Terminate the Api
API.T.02	*3.2.1.2*	L	Terminate the Api Unsuccessfully.

5.5.4 CLI

5.5.4.1 Subfeatures to be tested

a. Execute the Program (CLI.EP)

User opens the clı. User writes the place of the program at the computer and should add at least 2 valid parameters. One is the port that sensor is connected by to the computer and the other must be a mode name.

For failure case:

If user write the right place of the program (means that computer finds the program and try to execute it) but enters less than 2 parameters ,program should give an error about it.

b. Test Cases

Here list all the related test cases for this feature:

TC ID	Requirements	Priority	Scenario Description
CLI.EP.01	*3.1.1*	H	Execute the Program
CLI.EP.02	*3.1.1*	H	Execute the Program Unsuccesfully
CLI.EP.03	*3.1.1*	H	Execute the Program Unsuccesfully-2

5.6 Detailed Test Cases

5.6.1 TM.CG.01

TC_ID	TM.CG.01
Purpose	Create a gesture
Requirements	3.2.1.4
Priority	High.
Estimated Time Needed	10 Minutes
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in training mode.
	[A02] User makes a gesture more than one times
	[V01] User sees the output 'Gesture is created and saved successfully'
	-

5.6.2 TM.EFV.01

TC_ID	TM.EFV.01
Purpose	Extracting Feature Vector
Requirements	3.2.1.4
Priority	High.
Estimated Time Needed	10 Minutes
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in training mode.
	[A02] Program waits to sense a movement to start the gesture
	[A03] User makes a gesture
	[A04] Program wait a second to record the new gesture.
	[A05] Program and user repeats 5 times between [A02]-[A04].
	[V01] Program creates the feature vector successfully.

5.6.3 TM.SG.01

TC_ID	TM.SG.01
Purpose	Save Gesture
Requirements	3.2.1.4
Priority	High.
Estimated Time Needed	Less then 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in training mode.
	[A02] User makes a gesture
	[A03] Program creates the gesture
	[A05] Program saves the gesture into a JSON object
	[V01] User sees "Gesture is created and saved successfully" message on the screen

5.6.4 TM.SG.02

TC_ID	TM.SG.02
Purpose	Save Gesture Unsuccesfully
Requirements	3.2.1.4
Priority	High.
Estimated Time Needed	Less then 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in training mode.
	[A02] User makes a gesture
	[A03] Program creates the gesture
	[A05] Program tries to save the gesture into a JSON object
	[V01] User sees "An error happened when the gesture was saved" message on the screen

5.6.5 TM.DG.01

TC_ID	TM.DG.02
Purpose	Delete Gesture
Requirements	3.2.1.5
Priority	Low
Estimated Time Needed	Less then 1 Minute
Dependency	There should be at least gesture in the JSON object.
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in training mode.
	[A02] User enters delete command
	[A03] Program load the list of gesture
	[A05] User picks the gesture and writes its id number
	[A06] Program deletes the gesture
	[A07] Program saves other gestures to the JSON object.
	[V01] Program gives "procedure is completed" message to the user

5.6.6 GTM.CG.01

TC_ID	GTM.CG.01
Purpose	Create a Gesture
Requirements	3.2.1.3
Priority	High.
Estimated Time Needed	About 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in testing mode.
	[A02] User makes a gesture
	[V01] Program creates the gesture successfully

5.6.7 GTM.CG.02

TC_ID	GTM.CG.02
Purpose	Create a Gesture Unsuccesfully
Requirements	3.2.1.3
Priority	High.
Estimated Time Needed	More than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in testing mode.
	[A02] User makes a gesture which takes more than 1 minute.
	[V01] Program creates the gesture unsuccessfully

5.6.8 GTM.CoG.01

TC_ID	GTM.CoG.01
Purpose	Compare Gestures
Requirements	3.2.1.3
Priority	High.
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in testing mode.
	[A02] User makes a gesture.
	[A03] Program creates the gesture
	[A04] Program loads a gesture list from JSON object
	[V01] Program compares the gesture and the gesture list

5.6.9 GTM.SR.01

TC_ID	GTM.SR.01
Purpose	Show Result Succesfully
Requirements	3.2.1.3
Priority	High.
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in testing mode.
	[A02] User makes a gesture.
	[A03] Program creates the gesture
	[A04] Program loads a gesture list from JSON object
	[A05] Program compares the gesture and the gesture list
	[A06] Program finds a match for the gesture
	[V01] User sees “Gesture is recognized” message on the screen

5.6.10 GTM.SR.02

TC_ID	GTM.SR.02
Purpose	Show Result Unsuccesfully
Requirements	3.2.1.3
Priority	High.
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in testing mode.
	[A02] User makes a gesture.
	[A03] Program creates the gesture
	[A04] Program loads a gesture list from JSON object
	[A05] Program compares the gesture and the gesture list
	[A06] Program cannot find a match for the gesture
	[V01] User sees “Gesture is not recognized” message on the screen

5.6.11 GTM.NTD.02

TC_ID	GTM.NDT.02
Purpose	No Training Data
Requirements	3.2.1.3
Priority	High.
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in testing mode.
	[A02] User makes a gesture.
	[A03] Program creates the gesture
	[A04] Program cannot find a gesture list from JSON object
	[A05] Program cannot compare the gesture and the gesture list
	[V01] User sees an error message on the screen

5.6.12 API.I.01

TC_ID	API.I.01
Purpose	Initialize the API
Requirements	3.2.1.1
Priority	High.
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in init mode.
	[V01] Program initialize the api.

5.6.13 API.I.02

TC_ID	API.I.02
Purpose	Initialize the API Unsuccessfully
Requirements	3.2.1.1
Priority	High.
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in init mode.
	[A02] Program cannot initialize the api.
	[V01] Program gives an error.

5.6.14 API.T.01

TC_ID	API.T.01
Purpose	Terminate the API
Requirements	3.2.1.2
Priority	Low
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in init mode.
	[A02] Program initialize the api.
	[A03] User enters the terminate command.
	[V01] Program terminates the api.

5.6.15 API.T.02

TC_ID	API.T.02
Purpose	Terminate the API Unsuccessfully
Requirements	3.2.1.2
Priority	Low
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User executes the program in init mode.
	[A02] Program initialize the api.
	[A03] User enters the terminate command.
	[A04] Program cannot terminate the api.
	[A05] Program gives an error about it.

5.6.16 CLI.EP.01

TC_ID	CLI.EP.01
Purpose	Execute the program
Requirements	3.1.1
Priority	High
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User opens the command line interface.
	[A02] User writes the directory of the program first.
	[A03] Then writes the program mode and the port of sensor (2 valid parameter)
	[V01] Program is executed successfully

5.6.17 CLI.EP.02

TC_ID	CLI.EP.02
Purpose	Execute the program Unsuccesfully
Requirements	3.1.1
Priority	High
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User opens the command line interface.
	[A02] User writes the directory of the program first.
	[A03] Then writes an invalid parameter for one of the two parameters.
	[V01] Program gives an error and does not work.

5.6.18 CLI.EP.03

TC_ID	CLI.EP.03
Purpose	Execute the program Unsuccesfully
Requirements	3.1.1
Priority	High
Estimated Time Needed	Less than 1 Minute
Dependency	No dependency
Setup	Sensor has to be connected to the computer.
Procedure	[A01] User opens the command line interface.
	[A02] User writes the directory of the program first.
	[A03] Then writes only one parameter
	[V01] Program gives an error and does not work.

6. CONCLUSION

This document includes literature review, software requirement specification and software design document of Gesture Recognition Strap with Motion Sensor. The aim of this project to develop an affordable and multipurpose wristband which has accelerometer, gyroscope and compass sensor. A program is needed to get this data from these sensors and compare the gestures in database and finally return a information about if there is a match or not. The project includes to develop this software. Moreover, an api can be develop as addition to this software.

One of the purpose of the project develop something cheap. Therefore, the cost of the sensor that will use must be inexpensive. Since the accelerometer, gyroscope and compass sensor sensors (wearable technology) are cost-effective and easy solution to gesture recognition problem. They will be used in this project. The sensor myAHRS+ which includes all of these sensors will be used. This sensor has another advantage: usability. This sensor can connect with directly to the computer although other sensors which have the same capability with the myAHRS+ can not directly connect to the computer. They can generally be a part of an circuit which controlled by a microprocessor.

Feature of the gestures will be hold in a template as raw data or will be hold in a quaternion. Once a feature extracted in training it will be stored in a Json file. If a feature extracted in a testing, then software will compare the training data and testing data using one of the algorithm mentioned in 4.2.1.2 section. After this process complete software will return an answer as positive or negative the result of comparison. Addition of this recognition software an api between unity 3d could be added. The example cli interface is showed in 4.4.2 but there is a big chance to change this interface in real application.

Acknowledgement

I would like to thank Faris Serdar Taşel, advisor of the project, for providing opinion and expertise to help in this project.

Compilation / Installation Guide

To run the application required libraries are given below:

- json.hpp (from https://github.com/nlohmann/json)
- myahrs_pluss.hpp (from https://github.com/withrobot/myAHRS_plus)

This application can run on only Windows.

To work with this application, the user must have myAHRS Plus sensor.

References

[1] D. Geer, "Will gesture recognition technology point the way?," *Computer,* pp. 20-23, 2004.

[2] W. L. Liu H., «Gesture recognition for human-robot collaboration: A review,» *Industrial Ergonomics,* 2017.

[3] J. Joseph J. LaViola, «A Survey of Hand Posture and Gesture Recognition Techniques and Technology,» Brown University Providence, , RI,USA, 1999.

[4] N. A. S. S. Prapatş Garg, «Vision Based Hand Gesture Recognition,» *International Journal of Computer and Information Engineering,* 2009.

[5] R. Vadehra, «Gesture Recognition Technology».

[6] K. M. T. T. Ryo Izuta, «Early Gesture Recognition method with an accelerometer,» *International Journal of Pervasive Computing and Communications,* pp. 270-287, 2015.

[7] B. F. e. al., «3D Human Gesture Capturing and Recognition by the IMMU-based data glove,» *Neurocomputing,* 2017.

[8] M. Covarrubias, M. Bordegoni ve U. Cugini, «A hand gestural interaction system for handling a desktop haptic strip for shape rendering,» *Sensors and Actuators A: Physical,* pp. 500-511, 2015.

[9] D. Lopes, V. Nunes, P. Parreira, S. Paulo, P. Rego, M. Neves, P. Redrigues ve J. Jorge, «On the utility of 3d hand cursors to explore the medical volume datasets with a touchless interface,» *Journal of Biomedical Informatics,* pp. 140-149, 2017.

[10] H. Liao ve ZheQu, «Virtual Experiment System for Electrician Training,» %1 içinde *International Conference on Mechatronic Sciences, Electric Engineering and Computer*, China, 2013.

[11] «JSON,» . Available: http://www.json.org.