DEVELOPMENT OF TURKISH SIGN LANGUAGE RECOGNITION APPLICATION

A THESIS SUBMITTED TO THE GRADUATE

SCHOOL OF APPLIED SCIENCES

OF

NEAR EAST UNIVERSITY

By

BORA OKTEKIN

In Partial Fulfilment of the Requirements for

the degree of Master of Science

in

Computer Information Systems

NICOSIA, 2018

ACKNOWLEDGEMENT

In the finalization of this work, professor who shared valuable information, whenever needed, a lot of effort to be beneficial to me with a great interest, I have been able to go to her side without any hesitation, who fulfils the right of precious consultant teacher status in my opinion. I will always benefit from the valuable information from. Prof. Dr. Nadire Cavus with debt of gratitude.

Again in my study, in terms of subject, source and method, who guided me by constant help and never gave up his help. Prod. Dr. Dogan Ibrahim and Asst. Doc. Dr. Emrah Soyalan. I would like to extend my thanks

During my professional training, I have never been able to support my employees, especially my company, during the last period of my education.

At the same time, Robin Davie, my friends who have never given up their support and have always been with me at every moment

And thanks to my beloved wife and dear redemption at the beginning, and thanks to them always showing their support, and thanks to me for my memories and motivation at every moment, I am thankful to each one individually.

Thank you.

ABSTRACT

Sign language is the movements of the hand, the movements of the fingers, the arms or the movement of the body simultaneously with the face expressions to convey the ideas of the speaker. In recent years, the sign language is in the eyes of all researchers. It is possible to recognize the movements made with the help of sensors. However, it is of great importance to transfer the motion data to computer systems. As a result of the field study, it was determined that the studies conducted in this field were not sufficient at all. It was also found that the studies conducted were mainly in the field of the American Sign Language, the English Sign Language and the Arab Sign Language, and sufficient studies were not done in Turkish Sign Language. In this study, an intelligent system has been developed to facilitate the communication of hearing and speech impaired individuals with other individuals. The work done in this field is thought to help to remove the lack of information in this field. In the intelligent system developed in this Thesis, 33 basic signs in the Turkish Sign Language, which are called as sound informatics are taken as a basis in the study. The developed system uses the Microsoft Kinect v2 sensor to identify the signals. C# programming language and MongoDB are used in the developed system. As a result of the case study, 85% of the 33 basic signs were correctly recognized by the developed system. It is considered that the developed Sign Language recognition system should help both the hearing and speech impaired individuals, and also other individuals, and hopefully solve the problems of communication between these individuals.

Keywords*:* Sign recognition; Turkish Sign Language; passage reading; movement recognition systems; real-time translation; Microsoft Kinect V2

ÖZET

İşaret dili, el hareketlerinin, parmakların, kolların veya vücut hareketinin oryantasyonu ile konuşanın fikirlerini iletmek için yüz ifadeleriyle eş zamanlı olarak yaptıkları hareketlerdir. İşaret dilleri, son yıllarda tüm araştırmacıların gözdesi konumundadır. Yapılan hareketler sensörler yardımı ile tanınabilmektedir. Ancak, hareket verilerinin bilgisayar sistemlerine aktarılması büyük önem taşımaktadır. Alan yazın incelemesi sonucunda bu yönde yapılan çalışmaların yeterli olmadığı belirlenmiştir. Ayrıca, yapılmış çalışmaların daha çok Amerikan İşaret Dili, İngiliz İşaret Dili ve Arab İşaret Dili yönünde olduğu ve Türk İşaret Dili yönünde yapılan çalışmaların yeterli olmadığı tespit edilmiştir. Bu çalışmada, işitme ve konuşma engelli bireylerin diğer bireyler ile iletişimlerini kolaylaştırabilecek akıllı bir sistem geliştirilmiştir. Bu bağlamda yapılan çalışmanın alan yazındaki bu eksikliğin giderilmesine fayda sağlayacağı düşünülmektedir. Çalışma kapsamında geliştirilen akıllı sistemde, Türk İşaret Dili'nde ses bilimi olarak adlandırılan ve işaretlerin de temelini oluşturan 33 tane temel işaret baz alınmıştır. Bu işaretlerin sistem tarafından tanınabilmesi için Microsoft Kinect v2 sensörü kullanılmıştır. Sistemin altyapısında C# programlama dili ile sınıflandırma algoritmalarından Saklı Markov Modeli ve veritabanı olarak da MongoDB kullanılmıştır. Yapılan vaka çalışması sonucunda; 33 temel işaretin %82'inin geliştirilen sistem tarafından doğru bir şekilde tanımlandığı gözlemlenmiştir. Elde edilen doğruluk oranı göz önünde tutularak geliştirilen işaret tanıma sisteminin hem işitme ve konuşma engelli bireylere, hem de diğer bireylere yardımcı olacağı ve aralarındaki iletişim kurma problemini çözeceği düşünülmektedir.

Anahtar Kelimeler: İşaret tanıma; Türk işaret dili; metin okuma; hareket tanıma sistemleri; gerçek zamanlı çeviri; Microsoft Kinect v2

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

3D: Three Dimensional
ANN: Artificial Neural Networks
ARSL: Arabic Sign Language
ASL: American Sign Language
BSL: British Sign Language
HCRF: Hidden Conditional Random Fields
HMM: Hidden Markov Model
IR: Infrared
ISL: Indian Sign Language
K: Cross-Validation K fold
K-NN: k-nearest neighbours algorithm
LED: Light-emitting diode
LMC: Leap Motion Controller
RDF: Random Decision Forest
SDK: Software Development Kit
SDLC: Software Development Life Cycle
SUMI: Software Usability Measurement Inventory
SVM: Support Vector Machines
TSL: Turkish Sign Language
UK: United Kingdom
USA: United States of America
USB: Universal Serial Bus

CHAPTER 1

INTRODUCTION

This section talks about the difficulties of hearing and speech impairment. An overview of the problems identified according to the literature survey, the aim of the study, significanceof the study, the limitations of the study and overview of the studywas explained in detail.

1.1 Background

Sign languages include the use of hand movements along with body movements and the face used as a communication tool in respect of expression and position (Haberdar, 2005). Sign languages have great similarities to oral language. However, grammar and sentence structure are a challenge with sign language, as is fluency (Stokoe, Casterline& Croneberg, 1965). The UK and USA sign languages are different, but they can generally be used in a mutually comprehensible manner (Perlmutter, 2018).

Linguists see both verbal and sign communication as a natural language type. Sign language should not be confused with body language, which is a kind of non-verbal communication. Wherever there are hearing impaired people, sign languages have developed. Although sign is used by hearing-impaired and speech-impaired people, it is also used by people who cannot speak physically, who are disabled or who have problems with spoken language. It is seen that individuals who have hearing problems have difficulties in the field of communication. This affects their lives negatively and brings challenges to their lives because communication is the sharing of feelings and thoughts among people and speech is of great importance in human life.

According to Tüfekçioğlu (1998), when talking about a person able to hear, it is known that if he / she does not consider external factors in understanding speech, hearing is sufficient. The difficulties that individuals with hearing and speech problems experience in the field of communication and the obstacles they encounter in their educational life cause them to lag behind in the field of education compared to others. In order for hearing impaired individuals to understand and learn to speak, family education and supportive training should be given priority. The fact that the training to be done is initiated for hearing-impaired individuals in the childhood phase affects the future achievements of disabled people more positively (Cengiz, Hilal, Ercan &Akgul, 2016).

In recent years research into sign language has become very popular, and has become an opportunity for publication. American Sign Language (ASL), British Sign Language (BSL) and Arabic Sign Language (ARSL) are the most popular stamping grounds for researchers. We can speculate that this is because they are the most widely used and the ones which benefit most from state of the art technology. Turkish sign language has developed a lot, but has still a long way to go. There needs to be more research into Turkish Sign Language, and we hope this work will encourage that.

1.2 Problem

Turkish sign language, though it differs across the country is also being studied with a view to standardizing it nationwide. It can be said that Turkish sign language is the oldest sign language that we know of and it was used in the palaces of the Ottoman period (Demir, 2010). However, in 1953, a law issued by the Ministry of National Education prohibited the use of sign language. It was thought that the impaired could progress better in integrated situations. This ban was rescinded by the Ministry of National Education in 2005.

With an eye to the literature survey, we can say that sign language has been a popular area of research in recent years. It has been observed that the research and studies carried out have been mostly in the direction of American Sign Language (ASL), British Sign Language (BSL) and Arabic Sign Language (ARSL). Turkish sign language is most similar British sign language. Yet it has been pointed out that Turkish Sign Language has not been subject to enough research (present writer's experience). In Turkey, according to the data, there are approximately 1.5 million hearing and speech impaired people (TÜİK, 2011).

There are some professional translators who can serve hearing-impaired people with real-time sign language translations, but the cost is usually high. Moreover, such interpreters are often not available. For this reason, Turkish language supported technological systems are needed as we can see from the above.

Increasing the work of system developers and researchers on the TSL issue is of great importance in terms of support for people living with hearing and speech impairments.

There are serious deficiencies in the study of TSL compared with those into other languages.

1.3 The Aim of the Study

The aim of this study is to develop a Windows based application on the basic signs of the Turkish Sign Language that will help people with hearing and speech impairments solve communication problems.

1.4 Significance of the Study

The developed intelligent system will help the hearing and speech impaired people to communicate more easily with non-impaired people in their daily life. It is important that the Turkish Sign Language has a need to work on the sign recognition systems in the field by developing a windows based system on basic sound informatics signs.

1.5 Limitations of the Study

This thesis has some limitations. These are;

- The study is limited from March, 2018 to September, 2018
- The study is limited Windows Operating System
- The study is limited Microsoft Kinect v2 Sensor
- The Study is Limited HMM and HCRF classification algorithms
- The study is limited Turkish Sign Language phoneme feature figures in Turkish hands in sign language(33 Signs)

1.6 Overview of the Study

The thesis comprises of 6 chapters:

Chapter 1 gives an overview of the introduction, brief history of sign language, communication problem of hearing impaired people, the aim of the study, and importance of the study and the limitation of this work.

Chapter 2 is the related research into sign language where different studies previously published in that subject area, selection and choosing the up-to-date sign language recognition system using technology.

Chapter 3 is the theoretical framework of the study, and gives details of new technology of the recognition sensors and giving a brief explanation of ASL, BSL, ARSL and TSL.

Chapter 4 gives a detailed description of the application and tools and how it works and what the software development life cycle was.

Chapter 5 is the case study of the developed system with screenshots.

Chapter 6 gives the conclusion and recommendations for future research.

CHAPTER 2

RELATED RESEARCH

In this chapter related with sign languages by making different research which different studies publish previously in this subject area.

2.1 Studies in the World

The issue of sign language for the speech and hearing impaired has been under a lot of investigation since the eighteenth century. This is most understandable as the methods involved are the most natural ones for expressing intentions and emotions when unable to do so in the way of most people (Mangera, 2013).

Individuals with hearing and speech impairments who need to use sign language tend to use the sign language of their native country. Therefore, many countries have their particular sign language (Fenlon & Wilkinson, 2015). In 2013, Ethnologue pointed out that 137 of these languages were known to exist. However, Lewis, Simons and Fennig (2013) claim that there may be more. When it comes to computer recognition, all sign languages are at different levels of development. Studies show that American Sign Language subject to the most research globally. If the United States is thought to be one of the most developed countries in the world, this can be no surprise.

With sign language, the hand is the most important tool along with the face and the upper part of the body. Of course, the three-dimensional nature of human-to-computer communication is a challenge for high-tech approaches. These problems can be removed at least with the use of data gloves and cameras with devices visible at two angles (Madabhushi & Aggarwal, 2000) have focused on devices that monitor some parts of the body by working with the “skeleton model”. Mangera (2013) also carried out studies using the skeleton model.

Lei and Dashun (2015) have researched extensively into a data glove and sign language recognition system based on ARM9 and 9-axis IMU flexible sensors. Not only does it measure the degree of flexion of the fingers as well as the movement of the fingers, it also recognizes simple sign language. The use of a serial port or Bluetooth that is in contact with embedded systems makes the device more portable. With real-time data collection and time-domain analysis, the processor matches the incoming data with the intended communication. This system realizes audio and text real-time conversion movements to facilitate communication between a hearing and speech impaired person and the outside world. The system is portable, scalable and highly effective in recognising language.

Data gloves, also called electronic gloves, can be effective in tracking human hand positions, orientations and speeds, but they are expensive. Therefore, contemporary researchers are concentrating on computer recognition done without gloves. In daily life this is a more manageable way. In a study by (Segen & Kumar, 1999), a single camera, a more manageable approach, and a spotlight, the camera worked in dim lighting, night and so on to develop an efficient system. In his work Starner (1998) used a camera and computer training to facilitate data collection and release hands. His work used a fairly cheap and simple coloured glove. Vogler and Metaxas (1988) also used three cameras to acquire three-dimensional (3D) information movements used to train the HMM model.

In his Dong (2015) study, the system used Microsoft Kinect to track movements made by the signaller. In the study they used a segmented hand configuration. This configuration is first achieved using a depth contrast feature based on the per-pixel classification algorithm. Then, a method of finding a hierarchical mode is achieved and applied to locate hand joint positions under kinematic constraints. Finally, a Random Decision Forest (RDF) classifier is developed to recognize ASL markings in relation to joint edges. To attest to the performance of this method, 75,000 samples have been used to accumulate data containing 24 static ASL alphabetic characters. The system achieved an accuracy of around 92%. Dong also cites the use of a public data accrued by the University of Surrey to evaluate the methods they use in their research. As a result, the methods they use have shown that ASL can achieve greater accuracy in comparing alphabetic signs than have been the case with previous methods.

Chuan, Regina and Guardino (2014) offer the American Sign Language recognition system using a compact and inexpensive 3D motion sensor. The Leap Motion sensor is a much more portable and is a good deal cheaper than the Cyblerglove or Microsoft Kinect. K-NN and SVM algorithms were used to classify the 26 letters of American alphabet letters using sensory data derived features. The experimental result shows that 72.78% and 79.83% of the highest average classification ratios are obtained by KNN and SVM machine learning, respectively.

Different algorithms have been used to identify the alphabet with more than one algorithm in the work that (Souza, Pizzolato & Anjo, 2012) have conducted on the fingerprint alphabet of the Brazilian sign language. These algorithms measure their performance with cross-validation logic and aim to find the best algorithm for recognition. The algorithms used are SVM-HMM, SVM-HCRF, ANN-HMM and ANN-HCRF. The algorithms that show the best recognition performance in 4 combinations are SVM-HCRF and ANN-HCRF algorithms. SVM-HCRF algorithms showed a success rate of 98% while ANN-HCRF algorithms showed a success rate of 99%. However, at the end of the study, the SVM algorism is more understandable and simpler than the ANN algorithm, and at the same time, it has been found that the learning speed of the algorithm is better than the ANN algorithm.

Souza and Pizzolato (2013) have developed sign language recognition system using the depth information in the skeletal model of a person dynamically in Brazil sign language. In this study, SVM and HCRF algorithms are used to successfully detect the sign language. As we did in the previous exercise, we also achieved success with the ANN and HMM algorithms and shared the statistical information that the algorithms were caught using Cohen's kappa statistical method in the study. At the end of the study, the SVM-HCRF algorithm performed 94% of the detection of dynamic motion, while the ANN-HCRF algorithm performed 93% of the time.

Using the Leap Motion sensor, both (Mapari & Kharat, 2015) have developed an Indian Sign Language recognition system that recognizes the ISL. The Leap Motion sensor captures hand movements and gives finger positions in 3D format (X, Y, Z axis values). The positional information of the five fingertips along with the palms of both hands is used to identify the posture of the sign based on Euclidean distance and cosine similarity. In their research, 10 different hearing and speech impaired people and ISL markers were tested to assess the system's testability. The average recognition accuracy of ISL is 88.39% for the Euclidean distance method and 90.32% for the cosines similarity. When pointing, the Leap Motion Camera is tilted about 10 degrees to extract the depth information accurately. Unfortunately, they point out that the Leap Motion sensor in their studies could not capture other body parts and facial expressions, even though both hands were correctly monitored.

2.2 Studies in Turkey

The most recent studies carried out in Turkey concerning Turkish Sign Language are significant. It has been observed that the studies conducted in the new period in the literature search were mostly done with a Leap Motion Controller. Gülağız, Özcan, and Şahin (2017) used a Leap Motion Controller in their study and developed a desktop application to teach sign language to non-impaired individuals. At the end of the study, a questionnaire was used to assess the availability of the application from the non-impaired individuals using the SUMI questionnaire. The results of the survey have been analysed and it has emerged that the application is generally available software. However, it has been pointed out that additional methods have to be used in order for the application to increase the accuracy of the movements of the LMC device.

In another study with LMC by (Demircioğlu, Bülbül, & Köse, 2016), it was decided to introduce to the system the basic Turkish sign language movements with 18 data sets with LMC and to define the movements of the system in real time. In the study, Random Forest and Multi-LayerPeach Ron machine learning was tested and it has been proven that the system's success can be achieved with little data coming from the results gathered. Yalçınkaya, Atvar and Duygulu (2016) show that motion can be recognized by using information obtained from the camera by using the “Movement History Display”, and classification of the motion by means of the K-NN algorithm was performed with eight data sets. Previously obtained information can be found by comparing the meaning of the movement. It was stated that the overall success rate of the system in the classification process was 95%.

2.3 Summary

It has been observed that the systems developed for the Turkish Sign Language and the studies conducted in the field scan are less than the other sign language. It is considered that the developed system in this study will contribute to the system developers and contribute to the field of literature if it is considered that the works done for Turkish sign language are limited in the field.

CHAPTER 3

THEORICAL FRAMEWORK

In this section, the sensors which are used from the signal recognition tools, the most commonly used sign languages in the literature, the software development life cycle, and the agile methodology from the life cycle in the study.

3.1 Sign Recognition Systems

Sign recognition systems used in publications in the literature search are listed as follows:

3.1.1 Microsoft Kinect v2 Sensor

Kinect is motion detection software that is produced by Microsoft for the Xbox game console and can take human natural body movements as an input. It consists of a series of microphones and various sensors detecting colour and infrared (IR). It constructs a depth map that makes motion detection technology possible in 3D by pouring IR lights onto objects and calculating the time taken by the IR receiver of each sensor to "bounce back". Microsoft (2014) has provided a Software Development Kit (SDK) to carry the Kinect sensor forwards from being a video game asset to the fields of human computer interaction, human posture recognition and biomedical engineering.

Kinect v2 is the second manifestation of the Kinect sensor, appearing on the market in 2014. Several improvements can be seen in relation to the previous model. The sensor can process data more accurately with two gigabits per second; there is increased depth and infrared sensor resolution – up from 512x424 - and the colour sensor includes a 1080p resolution video at 30 frames per second (Amon & Fuhrmann, 2014). The sensor can now detect 25 skeletal joints rather than 20. In addition, the number of simultaneous user detections was increased to six from two, upping the field of vision of the camera.

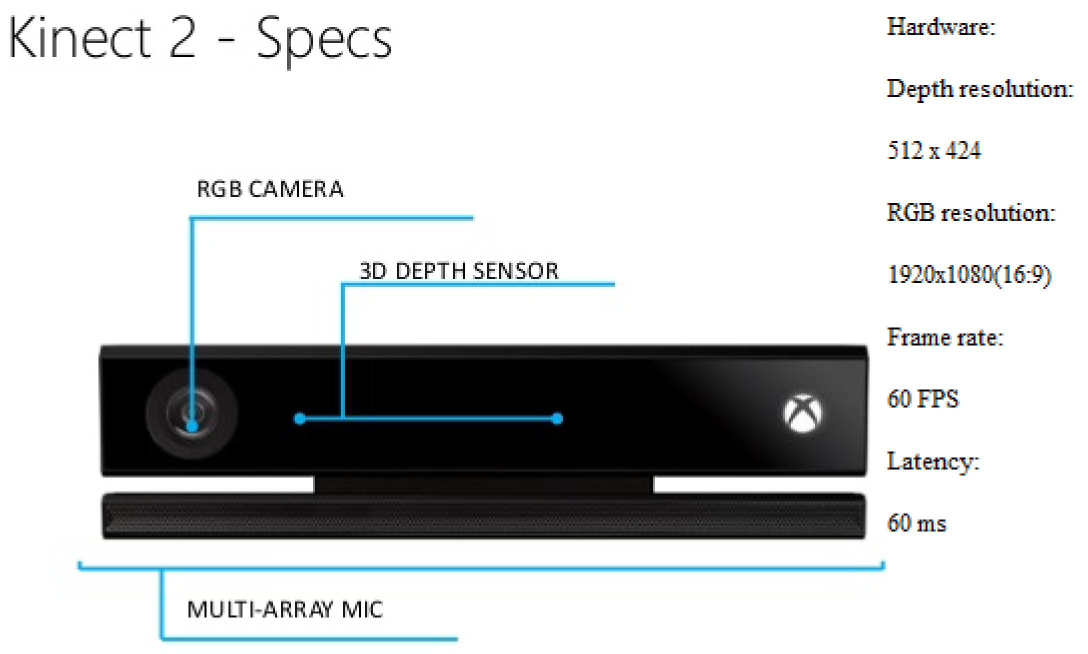


Figure 3.1: Kinect v2 sensor

3.1.2 Leap Motion Controller

The Leap Motion Controls consist of a small USB device which is to be placed upright on a desktop. Using two infrared cameras and three infrared LEDs, the device detects a hemisphere-like area at a distance of around a meter (Mapari & Kharat, 2015). This sensor distinguishes hand movements, finger joints, and monitors their motions (Elons, Ahmed, Shedid & Tolba, 2014).

Figure 3.2: Leap Motion Controller

3.1.3 Data Gloves

The data glove is very much like any other glove and is worn as such. It is fitted with sensors which distinguish the various positions of the hand in 3D. Most of the sign language translation systems in the market use the data glove for this purpose (Akmeliawati, Ooi & Kuang, 2007). The data glove has ten flexible sensors one for each digit (Preetham, Ramakrishnan & Kumar, 2013). These sensors can register the movements of the joints and relay the information to the microcontroller. This includes data from the fingertips to the wrists. Also, a 3-axis accelerometer is employed to increase the accuracy of the readings and to capture the changes in the speed of hand motions (Jingqiu & Ting, 2014). The accelerometer is located at the back of the data glove. Unfortunately, this item is not cheap. A cheaper data glove can be made available, but if the number of sensors is fewer for reasons of economy, this will mean less data collection. Of course, this would lead to less effective communication (Akmeliawati, Ooi & Kuang, 2007).

Figure 3.3: Data gloves

3.2 Methods and Tools

SVM, SVM-HMM or HMM algorithms were used predominantly in the literature. In this study HMM and HCRF algorithms are used.

3.2.1 HMM Algorithm

The hidden Markov model is one of the most important machine learning models in speech and language processing. An HMM is a set of probabilistic sequences: when given a set of units (words, letters, morphs, cues, whatever), they calculate a probability distribution over the possible tag sequences and select the best tag sequence (Jurafsky & Martin, 2017).

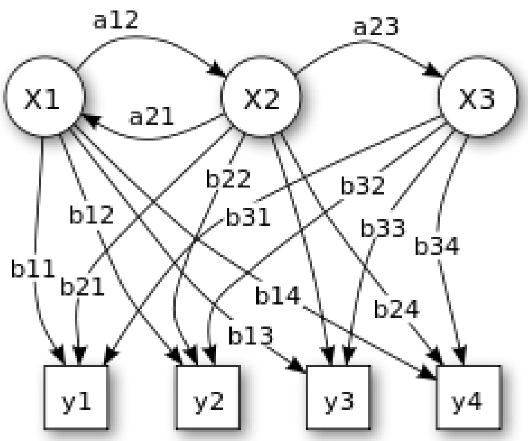


Figure 3.4: Hidden Markov Model

3.2.2 HCRF Algorithm

An HCRF is a randomized Markov on a set of evidentiary variables that some variables cannot be observed during training. The linear-chain HCRF used in speech recognition is a conditional distribution with an ordered structure (Sung & Jurafsky , 2010).

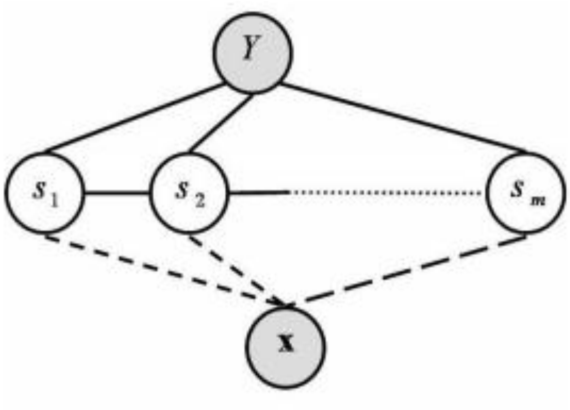


Figure 3.5: Hidden Conditional Random Fields

3.3 Software Development Life Cycle

Each development process, especially the product development process, involves specific stages. Be it noted that software is also a product. Just as with hard manufactured goods, there are stages of development with all forms of software. Broadly speaking, it is feasible to classify these products as system development methodology (Ruparelia, 2010). Different companies have differing methods of product development. The basic steps of the software life cycle are divided into seven phases as Planning, Analysis, Design, Development, Testing, Implementation, and Maintenance. These are;

- ***Planning:***

  It is the most important phase of SDLC. This step is carried out by the customer and the experienced person in the ecosystem and the resulting information is then used to plan the basic project approach and in the process of technical feasibility studies. Planning of the quality assurance conditions of the project will reveal the risks related to the project and the technical approaches with the least risk will be determined to realize the project.

- ***Analysis:***

  After planning the project, the next step has not been approved by the customer although the product requirements have been clearly identified and documented. All product requirements to be developed throughout the project are included in this document

- ***Design:***

  Determine which architecture the software will be developed for, depending on product requirements.

- ***Development:***

  At this stage, developments are made according to the analysis work done.

- ***Testing:***

  Analyse the product and make checks according to the needs of the customer.

- ***Implementation & Maintenance:***

  Once the product is tested and ready for use, it is delivered to the customer. The system can then be turned on to the test environment and the product can be improved according to the feedback.

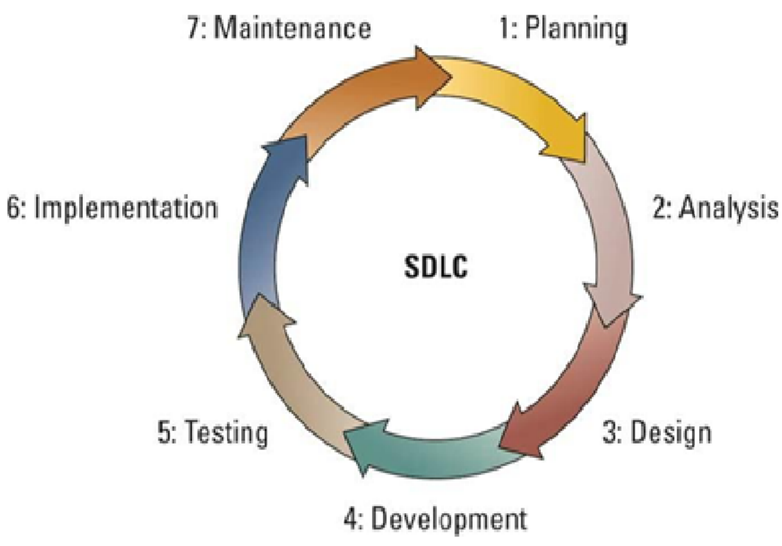


Figure 3.6: Software Development Life Cycle (SDLC)

3.3.1 Software Development Life Cycle Models

Various software development life cycle models that are followed throughout the software development process are defined and designed. Today, many of these companies and software developers have used these models in their projects. In the literature there are various SDLC models in use such as Agile, Lean, Waterfall, Iterative, Spiral, and DevOps. Each one differs from the others in some respects, but the common goal of all is to help software developers to develop high-quality software as quickly and cost-effectively as possible. But, Agile method modelling of enterprise in the front plant is among the most preferred among software development life cycles. The number of companies using the method, showing an increase of 18% in Turkey, is formidable with the overall number of companies using agile methods at 90% (Turkey Annual Agility Report, 2017). It seems that software developers have gained importance in terms of preference in software projects in recent years.

3.3.2 Agile Methodology

Agile processes include modern and bureaucratic software methods developed as an alternative to the existing traditional methods used in the software industry. Seventeen leading software professionals in agile program development in 2001 met in Snowbird, Utah, discussing how to make software faster, simpler, and more human-centred, and described the process with an Agile Manifesto, 2018 that they signed as a result. According to this declaration, agile processes:

1. Contacts and communication come before the process and tools.

2. The program in operation is prioritized from the detailed documentation.

3. Working with the customer is a priority over contracts and agreements.

4. Keeping up with changes is more important than following a plan.

3.3.3 Agile Methodology Models

Agile methodology is divided into 6 within itself. These are as follows;

1. Extreme Programming-XP
2. Agile Unified Process

3. Scrum
4. Test-Driven Development
5. Agile Data Method
6. Feature-Driven Programming

In the agile methodology models, the scrum model adapted to the project was chosen and the success rate of the project increased.

3.3.4 Scrum

(Schwaber & Sutherland, 2017) A framework in which people can address complex adaptive problems and offer products with the highest possible value productively and creatively. Scrum is an agile software development approach where the software process begins directly and the software needs are elaborated in the process. In this model, with the intensive communication between the developer and the customer, the ideal development according to the scrum model takes place in two weeks.

Scrum is team work and the success of the team is largely preliminary to the individual achievements of the team members. In addition, Scrum teams are self-organizing and aim to develop a product after each run. According to the Scrum approach, the product can be agile in the direction of the customer's wishes.

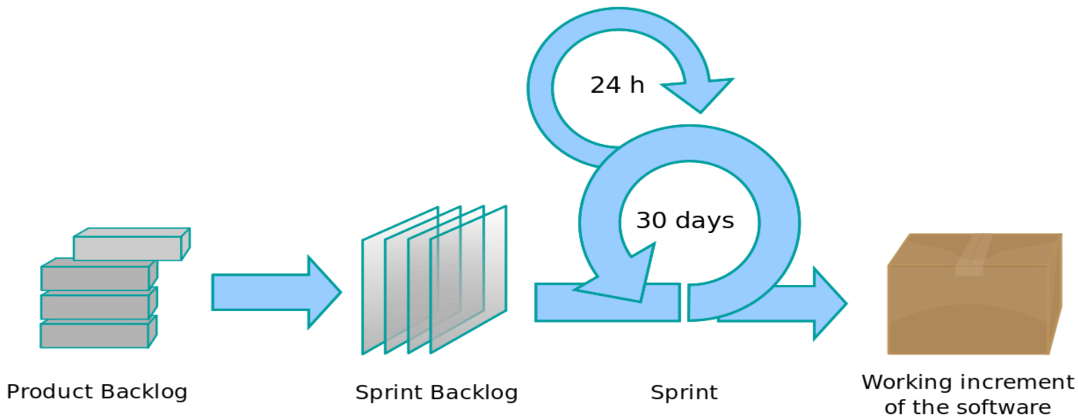


Figure 3.7: Scrum Model

The scrum model is divided into 3 within itself (Schwaber & Sutherland, 2017). These are:

3.3.4.1 Product Backlog

The product backlog list is a list of all the steps in the development of a product, detailing the changes that will be made in future releases. The responsibility of the product owner here is the responsibility of the product request list, including the content of the product, availability and claims. The creation of short user stories that describe requirements content on the product list and the fact that priority information is important for business planning.

3.3.4.2 Sprint Backlog

The document is a step-by-step, detailed, timed content of the team's work throughout the Sprint, and is a useful tool to help the Development Team manage Sprint. It is an output of the Sprint Backlog Development Team, which includes the development team's work in Sprint and how to complete it.

3.3.4.3 Sprint

The Scrum process consists of consecutive Sprints of no more than 1 month in length. Also, Hundermark (2014) underlined that Sprint is the heartbeat of the Scrum cycle and all other events.

3.4 Hearing and Speech Impaired

Those who have partially lost their hearing ability are called *“Hearing impaired”*. According to the information provided by the (World Health Organization, 2018) regarding hearing impairments, it is estimated that the number of individuals with hearing loss in 2018 is 466 million and that this number will be 900 million in 2050. It is generally accepted that there is a kind of speaking problem in the individual if the speaking differs from the boundary that is adopted in any environment to an unreasonable level. Such people are called *“Speech impediments”*. According to the data in (TÜİK, 2011), there are approximately 1.5 million people with hearing and speech disabilities in Turkey, but not officially in the TRNC with the total number of disabled people in the island according to reports in the newspapers there are 5188 people with disabilities. In addition, according to the information received from the Cyprus Hearing and Speech Impaired Foundation, in the statistical information obtained from the Ministry of Labour in 2017, 417 people were registered in the TRNC and two people were registered to the Ministry on average every month. According to the opinion of the Foundation's chairman, this number has increased in recent years and it has been observed that for the hearing and speech impaired individuals who reside in TRNC, their work and awareness should be increased to facilitate their lives.

On the other hand, according to Akmeşe (2016) Turkey in 2005 enacted the law for persons with disabilities, despite the passage of 12 years has not yet reached its rightful place of TSL in the education system. It is also lower than that seen in hearing impaired youth literacy level statistics in Turkey. Therefore, any organization or university studies in the TSL on the subject in Turkey has been intensifying on this issue. It is very important for hearing-impaired children to start TSL at an early age in terms of their language, cognitive and social development. As a result, it is very important to start the bilingual education in which TSL and Turkish are used together in education of hearing-impaired individuals.

3.4.1 Hearing and Speech Impaired Problems

In the study of (Akmeşe, Kayhan, Kirazlı, Öğüt, & Kirazlı, 2018), it is necessary to support the language, speaking and communication skills of the children who have hearing and speech loss that the early clinical and education applications for children who are experiencing hearing and speech loss for a reason born or later require a multidisciplinary study. Findings in the study are thought to contribute to the diagnosis and evaluation processes of children with hearing loss, to cooperation with parents and other specialists, communication approaches, preparation of health and education programs, support education services and monitoring or evaluation of the program. It is also expected that this area will provide a functional contribution to the regulation of undergraduate and graduate education programs during the development of professional qualifications of the personnel to be trained.

3.5 The Most Widely Used Sign Languages

Sign languages widely used by hearing and speech-impaired people in countries around the world are as follows:

3.5.1 American Sign Language

American Sign Language (ASL) is used by the hearing-impaired in the United States and among the English-speaking in Canada. ASL speakers can easily communicate with each other by hand gestures. However, communicating with hearing-impaired people is still a matter of difficulty for those who have not learned the signs (Dong, 2015).

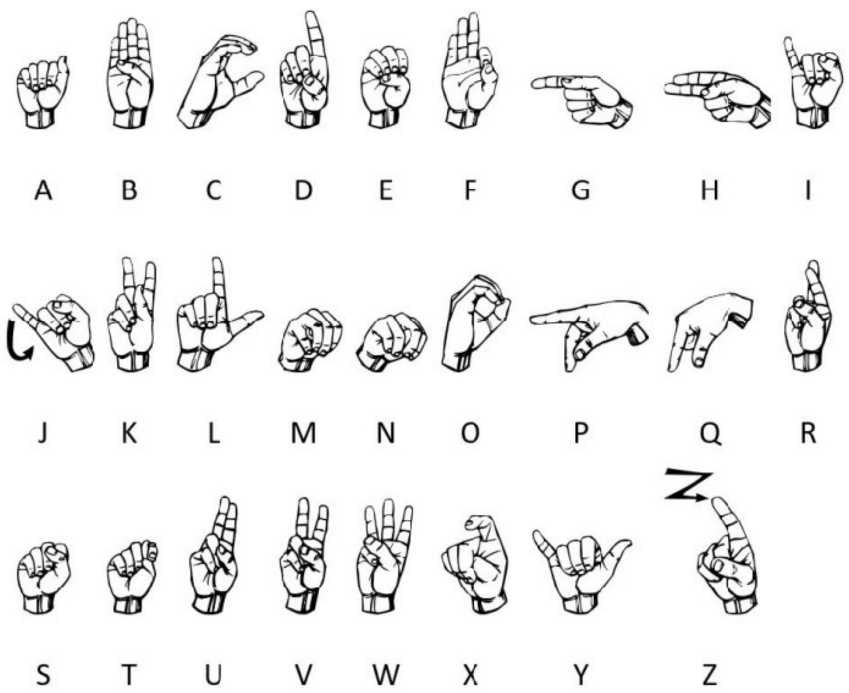


Figure 3.8: ASL – Alphabet

3.5.2 British Sign Language

According to the (BDA, 2011), British sign language users are 127,000 users in the UK, with 73,000 hearing and speech impaired people. In Scotland there are 12,556 users, of whom 7,200 are hearing and speech impaired people. While there are 7,200 users in Wales, 4,000 of them are hearing and speech impaired people. These figures do not include professional BSL users, interpreters, translators, etc.

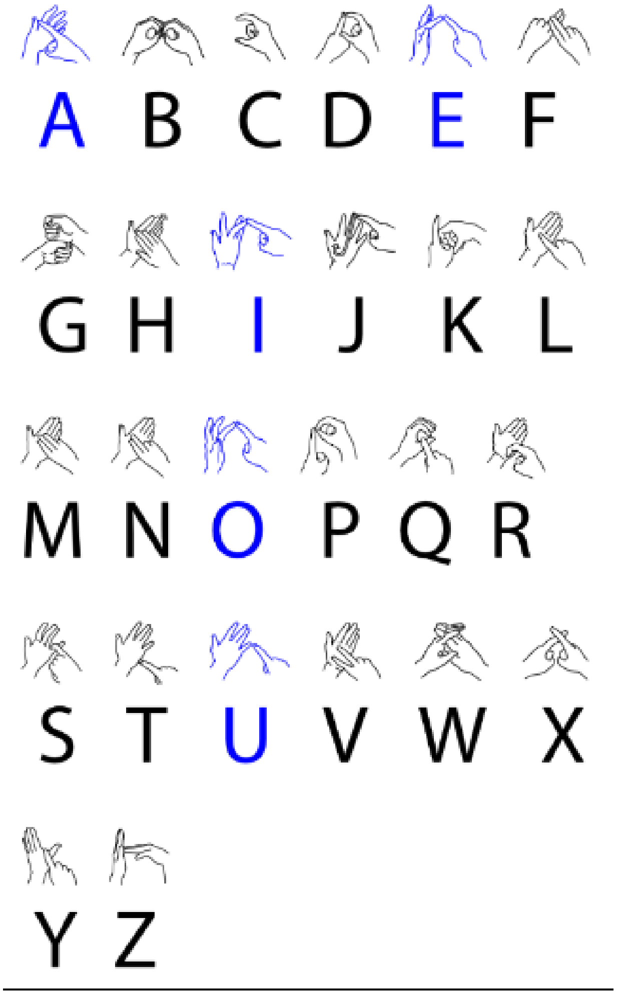


Figure 3.9: BSL- Alphabet

3.5.3 Arabic Sign Language

The sign language of the Arabic World has recently been documented and recognised as standard among all Arabic speakers. Although there are many variations in the Arabic spoken in the many Arab nations, the sign language is the same everywhere.

Figure 3.10: ARSL- Alphabet

3.5.4 Turkish Sign Language

According to the data in (TÜİK, 2011), there are approximately 1.5 million people with hearing and speech disabilities in Turkey, according to the reports in the newspapers, there are 5188 disabled people in the TRNC with a total number of disabled people in the TRNC. Despite the fact that the number is so high, the attention put to the issue is of a lower level than is the case with other high user sign languages. System developers and researchers are working on the TSL issue, and working in this direction is of great importance in terms of support for hearing and speech impaired individuals.

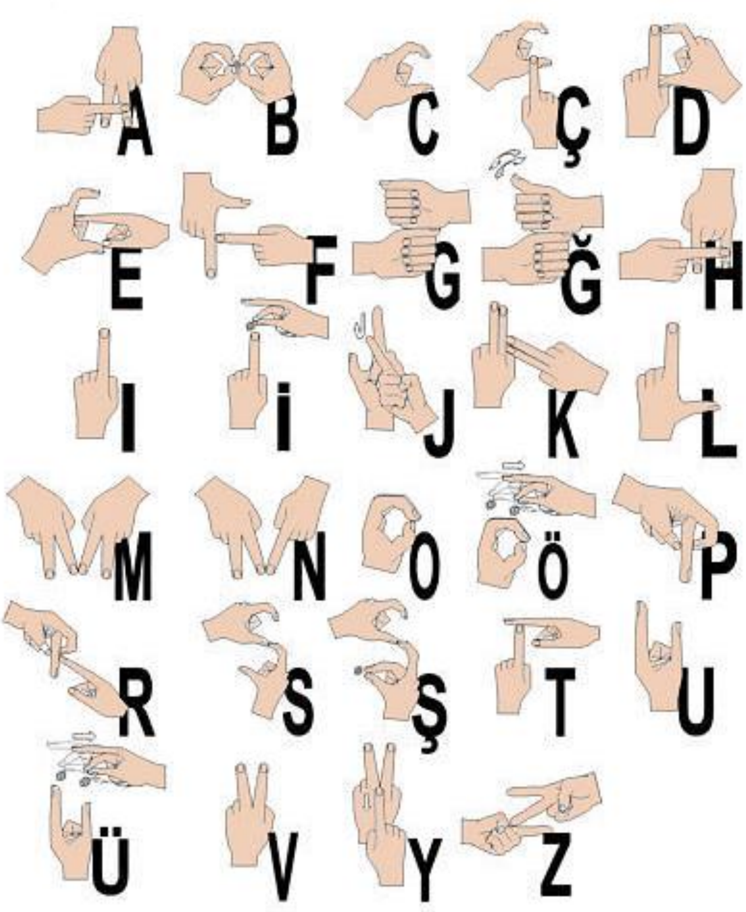


Figure 3.11: TSL- Alphabet

CHAPTER 4

DEVELOPED SYSTEM

In this section, the system that has been developed for hearing and speech impairments will be investigated concerning the design process of the program and where the flows used in the system go.

4.1 Research Schedule

In Scrum stages, the product requirements are created, the intermediate distributions where the software is developed with iterative runs starting with the preparation phase, the characteristics such as architecture, technical details and contracts to be used are determined and the customer product is offered in pieces and the final product is tested and documented and the product project management methodology.

Table 4.1: Schedule of the research

Task Name	Duration(Days)	Start Date	Finish Date
Full thesis Schedule	**412**	**12.07.2017**	**28.08.2018**
Indentifying the research area	15	12.07.2017	27.07.2017
Related research		28.07.2017	28.08.2018
Thesis Proposal	**22**	**28.07.2017**	**19.08.2017**
Writing thesis proposal	16	28.07.2017	13.08.2017
Proposal review	4	14.08.2017	18.08.2017
Thesis approval	2	19.08.2017	21.08.2017
Documentations	**83**	**19.08.2017**	**10.11.2017**
Writing thesis	45	19.08.2017	3.10.2017
Thesis review	26	4.10.2017	30.10.2017
Final thesis draft	12	31.10.2017	12.11.2017
System development	**282**	**3.10.2017**	**12.07.2018**
Product Backlog	15	3.10.2017	18.10.2017
Sprint Backlog	267	19.10.2017	17.08.2018
Sprint(9 Iterations)			
Last revision of the thesis	**10**	**18.08.2018**	**28.08.2018**

In this study, a group was formed as a hearing impaired, one hearing impaired expert educator and a system developer. In order to be able to move forward successfully and faster, the Agile methodology has been used in recent years to develop software that can respond quickly to changes and encourage a recursive approach, considering that it will take a more effective role than traditional methodology used as a life cycle of traditional software development life cycle.

On the developed system, 99 signs were recorded by the developer on the system via the Microsoft Kinect v2 sensor. Then, the system hearing and speech impaired individuals and hearing and speech impaired educator of the individual movements are checked one by one, which movements in the system to create a mismatch and the need for the hand position of the test phase came to a halt. In this way, progress in the work was made according to the steps of the agile method. In short development meetings in the system development, deficiencies in the system were eliminated and the next steps were taken.

4.2 Programmatic View

The system uses the Microsoft Kinect v2 sensor to identify the signs. While the C # programming language is used in the system's infrastructure, MongoDB is also preferred as a database for storing basic signs.

4.3 Sign Used

Kubuş (2008) revealed that 33 different hand shapes are the basis of TSL's movements as phonological features. For this reason, in the developed system 33 basic signs which are called sound informatics in Turkish Sign Language are also the basis of the signs, and they are integrated into the system. They are divided into 5 divisions:

- ***Hand Shape:***
  In particular, studies on ASL define the term handshape as the form in which it takes place during the sign production process and during the hand shaking process
    a. Number of selected and unselected fingers
    b. Node shape
    c. Unspecified hand shapes: index finger or all fingers forming
    d. Specified hand shapes: the shape of selected fingers

- ***Motion:***

  The movement is shown as one of the phonological unit parameters in the sign languages since the sign has a contribution to the formation process and has many important roles

  a. Path and Local movement
  b. Path motion: arc, flat, circle

- ***Place of Articulation:***

  Place of articulation is usually defined as the position above the hand's body or within the sign area in the production of the mark

  a. Head, shoulder, torso and non-dominant hand

- ***Tendency:***

  According to Battison (1978), it can be described as the centre of palms and parks, where it plays an important role in the production of sign language**.**

- ***Hand signs:***

  In sign languages, hand, as well as head, body and facial movements, constitute an important element of language production. These properties are part of all areas of sign language grammar, from lexical to phonetic, from morphology to syntactic. Pfau and Quer (2010) in the sign language made by the classification of the extraterrestrial signs, the group is separated.

  a. Head and body movements:

  When you study the Turkish Sign Language, many words are separated by head and body movements. The characteristic that distinguishes words that have the same hand movements is to play according to the meaning of movement of the head.

  b. Face expressions:

  Facial expressions, which have a distinctive feature in the phonological aspect of the sign language, come into play with the movement of different regions such as eyebrows, lips, cheeks and jaws.

  c. Mouth movements:

  When the signer's daily language use is examined, it is seen that the mouth area is very active. Because of their different functions, these mouths are divided into two types, mouth gestures and mouth movements.

Table 4.2: Phonetic features of the TSL

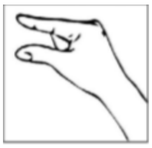
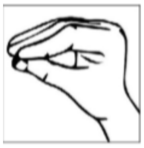
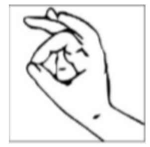
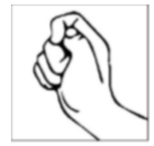
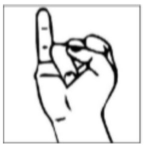
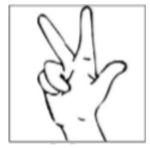
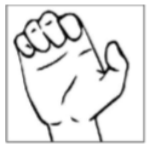
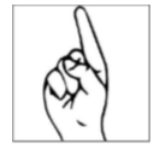
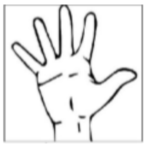
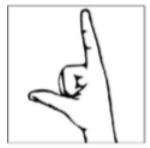
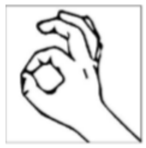
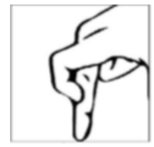
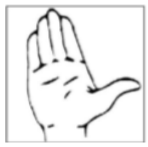
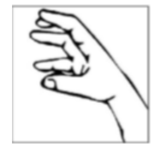
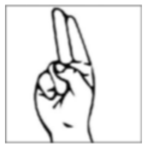
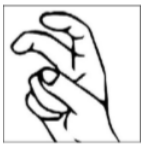
TSL-1	**TSL-2**	**TSL-3**	**TSL-4**	**TSL-5**	**TSL-6**
					
TSL-7	**TSL-8**	**TSL-9**	**TSL-10**	**TSL-11**	**TSL-12**
					
TSL-13	**TSL-14**	**TSL-15**	**TSL-16**	**TSL-17**	**TSL-18**
					
TSL-19	**TSL-20**	**TSL-21**	**TSL-22**	**TSL-23**	**TSL-24**
					
TSL-25	**TSL-26**	**TSL-27**	**TSL-28**	**TSL-29**	**TSL-30**
					
TSL-31	**TSL-32**	**TSL-33**			
					

Table 4.2 shows 33 basic hand signs. Table 4.3 has the word meanings of the signs.

Table 4.3: Hand shape meanings

HAND SHAPES	EXAMPLES
TSL-1	SUPPORT,CHEESE,FUNNY
TSL-2	BUS,GROUP, BINOCULARS
TSL-3	VIDEO,DOCUMENT,SOFT
TSL-4	GRAPE, DOUBT, THIN
TSL-5	GOL, PENALTY, CONSCIENCE
TSL-6	TELESCOPE, LABORATORY, SPINE
TSL-7	CYBRID, EASY, ROPE
TSL-8	CHILDREN, GOOD, FOOD
TSL-9	EJECT,FIRE,DIET
TSL-10	REPORT,KNOW,TWELVE
TSL-11	DIFFICULTY, APPLICATION, PAYMENT
TSL-12	TRAVELING, SAVING, BED
TSL-13	BAD, GUEST, REGIMENT
TSL-14	SAME, AIRCRAFT,FUN
TSL-15	SAD, FORFEITED,ALEVI
TSL-16	FAMILY, PUSH, DEFENSE
TSL-17	FIRST, RED, LUCK
TSL-18	SEE, FASHION, POLICE
TSL-19	WANT,STUPID, SEQUENCE
TSL-20	TURKEY,COFFE,MOON
TSL-21	SHOUT,THURSDAY,VILLAGE
TSL-22	GOLD,ANY,ORGANIZATION
TSL-23	PSYCHOOLOGY, EMPTY, defraud
TSL-24	THROAT,URFA,THICK
TSL-25	BORING,COLLIDE,PRINTING
TSL-26	PRESIDENT,MATCH,ASSIGNMENT
TSL-27	CLOSED,KARATE,MIRROR
TSL-28	FRIEND,ILLITERATE,NOT
TSL-29	OWN,WAIT,MOM
TSL-30	GIRL,WINE,COURSE
TSL-31	LOVE,CHAIR,BLUE
TSL-32	HOT,INVESTIGATION,FIND
TSL-33	FORGET,ESCAPE,FAST

4.4 System Use Case Diagram

The Use-Case diagram of the system developed in Figure 4.1 can be seen

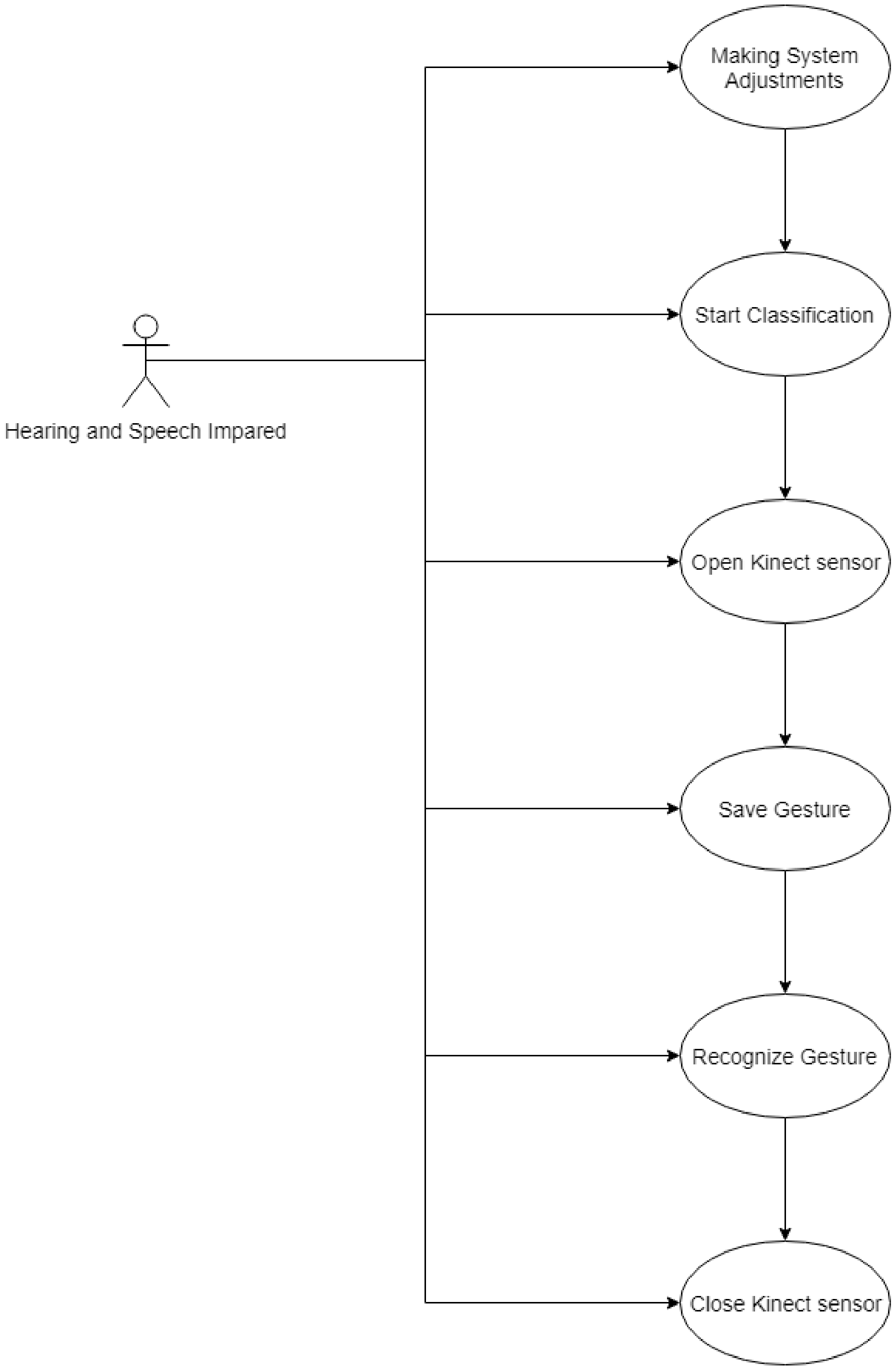


Figure 4.1: Developed system Use-Case diagram

4.5 Flowchart of the Developed System

The flowchart diagram of the developed system in Figure 4.2 can be seen

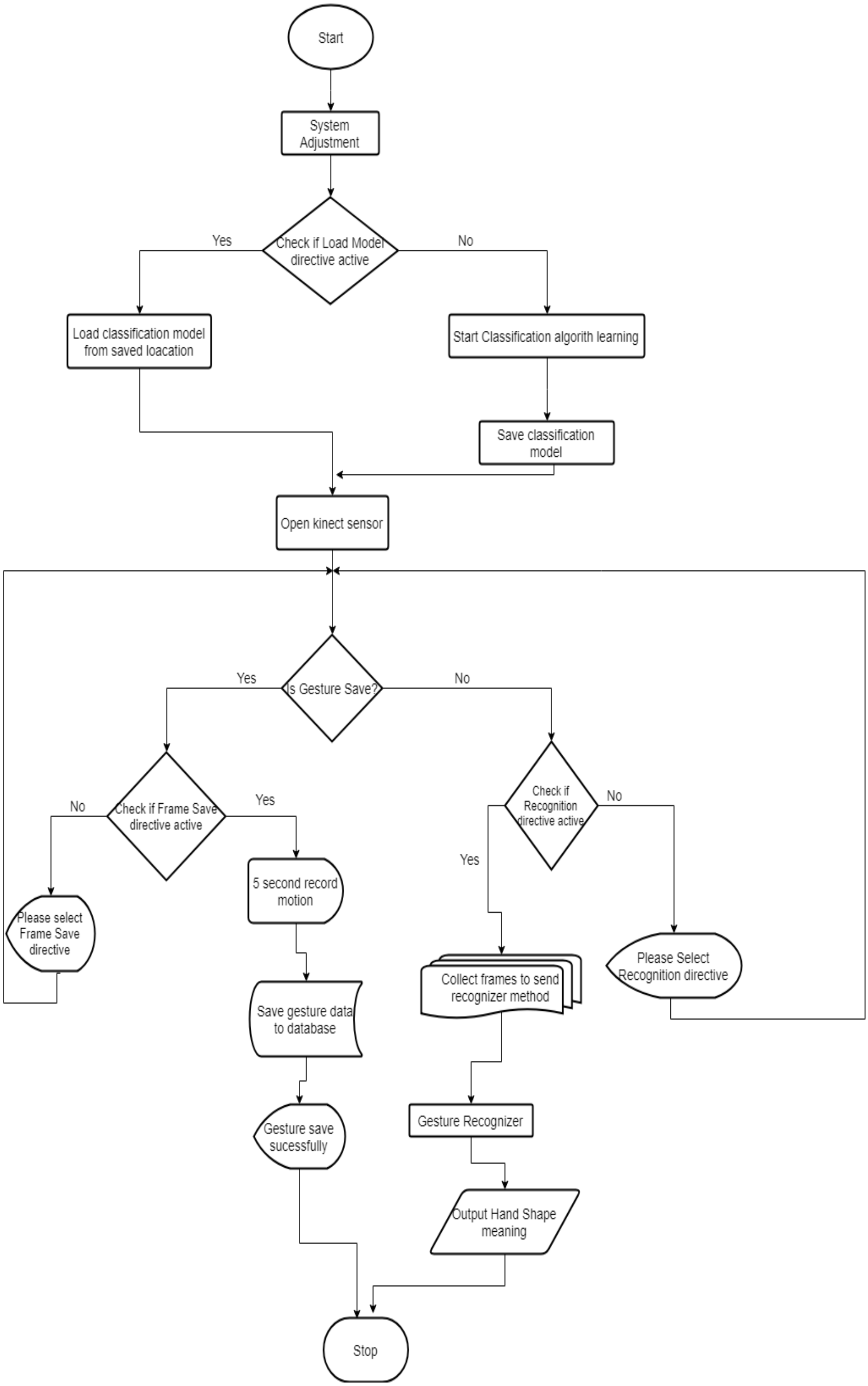


Figure 4.2: Flowchart of the developed system

4.6 Agile Steps during Developing of the Systems

The main steps in developing of Agile are given below:

4.6.1 Product Backlog

In line with the problems experienced in the field of communication of hearing and speech impaired individuals were determined that there was a need for a system that could help them technologically as they experienced difficulties in expressing themselves against the individuals.

4.6.2 Sprint Backlog

During the Sprint, the timetable which includes the detailed and timed schedule content was prepared. Also, the time frame of the system and the time frame of the modules was designed and finished.

4.6.3 Sprint

During 9 weeks, the system was tested on people who have hearing and speech disabilities. In this period, every sprint was done within 2 or 4 weeks which includes process of developing required functions. As a result of the tests, a number of obtained bugs and findings were eliminated and integrated into the system.

4.7 System Operation Logic

All signs were recorded into the system's MongoDB database by using the Microsoft Kinect v2 sensor. Because the system works with intelligent algorithms, we have repeated three times each movement (33 signs) to get a good result and the system is trained in this way. A total of 99 signs are recorded in the database, with an average of 150 frames on each sign. So this corresponds to a total of 14,850 frames.

Cross-validation is used to measure the performance of algorithms. In cross-validation, the value of **K** is taken as five and the division of the data used in the training and testing of the algorithms into five equal parts is ensured. This has helped us to measure the performance of algorithms in training.

In this work, which we can call the basic sign recognition system, classification algorithms have been used in order that the motion can be perceived in real time by the system. In this study however, a data set is formed from the movements of the person by taking the data of the skeleton data. Generated data sets are sent to the sign recognition method. In this method HMM-HCRF algorithms are used.

The system carries out the sign recognition process by taking the position and frequency data of the right hand only of the signer in the training and test phases. The intelligent system processes the HMM and HCRF classification algorithms in the system according to the individual position and skeletal data, and decides which class the incoming data belongs to. The data obtained after the decision is displayed on the computer screen to inform the user of the class value. Thus, the information that the hearing and speech impaired person wants to tell is easily understood by other people. The operating logic of the system is described in Figure 4.2.

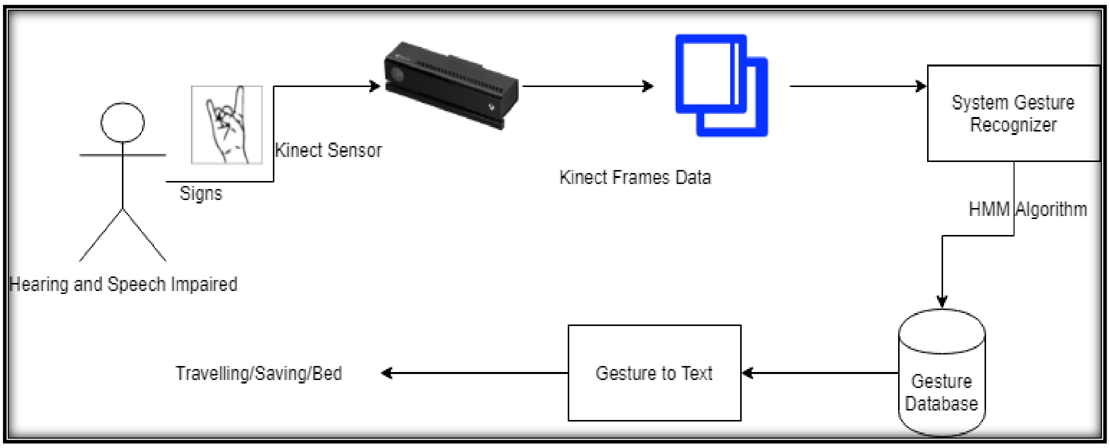


Figure 4.3: System operation logic

While the infrastructure of the system is being prepared according to the phonetic features of Turkish sign language, there are 33 different signs in the phonetic alphabet according to the Turkish Sign Language book (Dikyuva, Makaroğlu, & Arik, 2015). These signs constitute the linguistic structure of Turkish Sign Language. All the signs in the sign language derive from the sign table in Table 4.2.

CHAPTER 5

IMPLEMENTATION OF THE DEVELOPED SYSTEM

This chapter is the system implementation section which contains full description of how the system works to recognition Turkish Sign Language.

5.1 Main Screen of the Developed System

The screen capture of each step from the developed system in Windows Application view in Figure 5.1

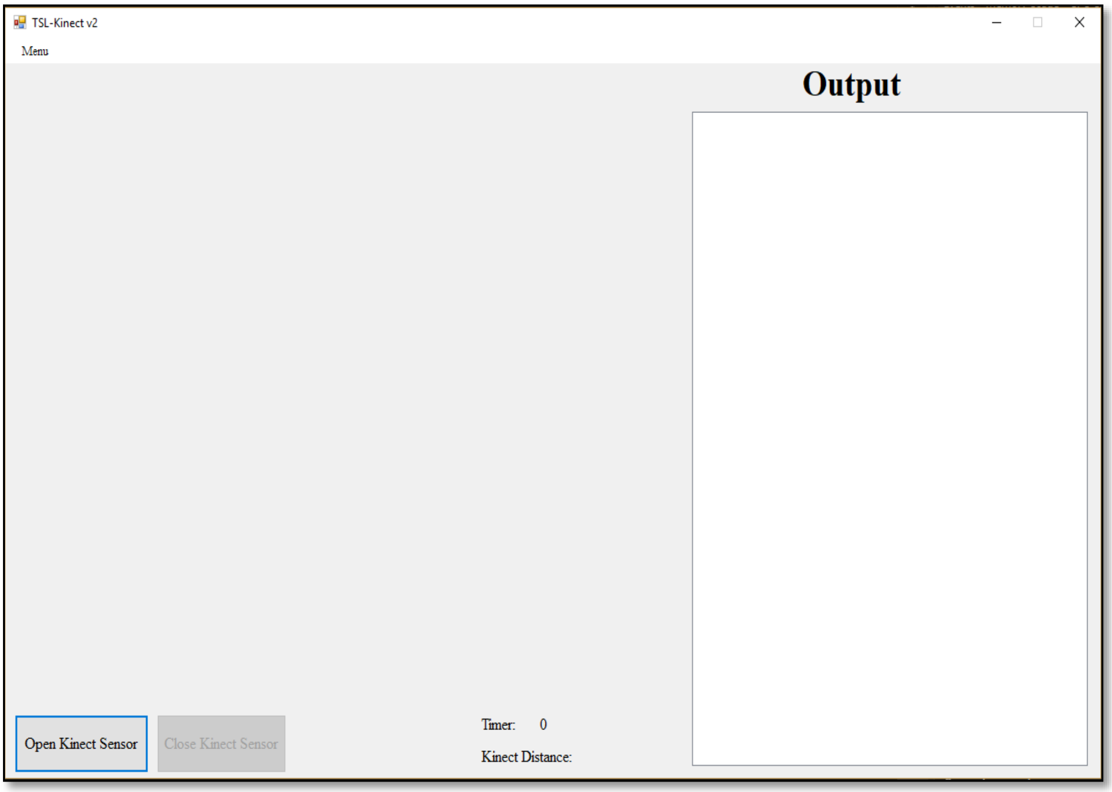


Figure 5.1: Snapshot of the *"Home page"*

Step 1: Making System Adjustments

The system directives in this section describe the operating conditions of the system. In order to be able to go to the system directives as shown in Figure 5.2, access should be made from the menu and *"System Adjustment"* should be selected.

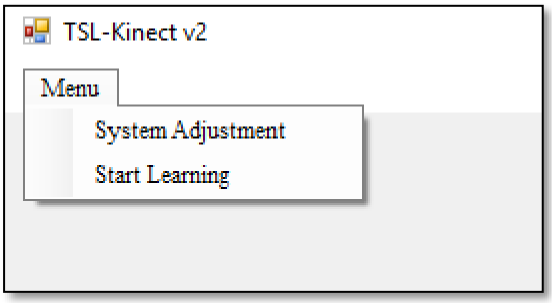


Figure 5.2: Snapshot of the *"Menu"*

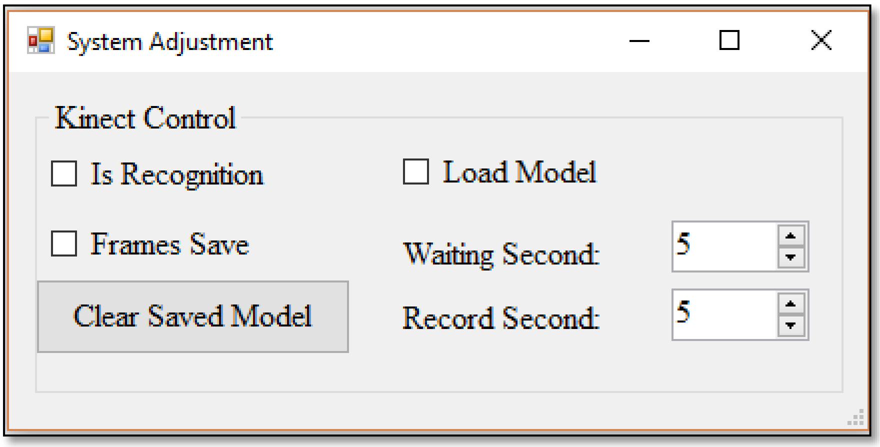


Figure 5.3: Snapshot of the *"System Adjustment"*

Figure 5.3 shows the directive groups within the panel as they are understood in the screenshot. These;

Is Recognition:

This directive needs to be activated if the user wants to detect the sign via the system after the learning phase of the classification algorithms.

Frames Save:

If a sign data that has not been previously saved on the system is to be saved, this directive must be activated. Our registration processes are time-based over 5 seconds. The movement to be performed is recorded for 5 seconds and the very set obtained in the time-out is written in the collection table on the MongoDb.

Load Model:

Since the learning periods of classification algorithms are long, the classification models are recorded in the system after the learning process is finished so that the system can work more effectively and quickly. If this directive is switched to passive, the recorded model will repeat its learning without being restored to the system in the learning phase.

Kinect Distance:

The hearing and speech impaired individual measures the distance of the sensor in real time while the sensor is in the open state.

Clear Saved Model:

Since the learning process of the classification algorithms is long, our models were recorded in the system. If the *"Clear Saved Model"* key is pressed as shown in Figure 5.4, all saved operations will be permanently deleted and the learning process will be repeated.

Figure 5.4: Snapshot of the *"Clear Saved Model"*

Step 2: Start Learning

The system works with machine learning. As can be seen in Figure 5.5, in order to start the learning process, it is necessary to click the "Start Learning" menu in figure 5.2. Then press the "*Start Classification"* button.

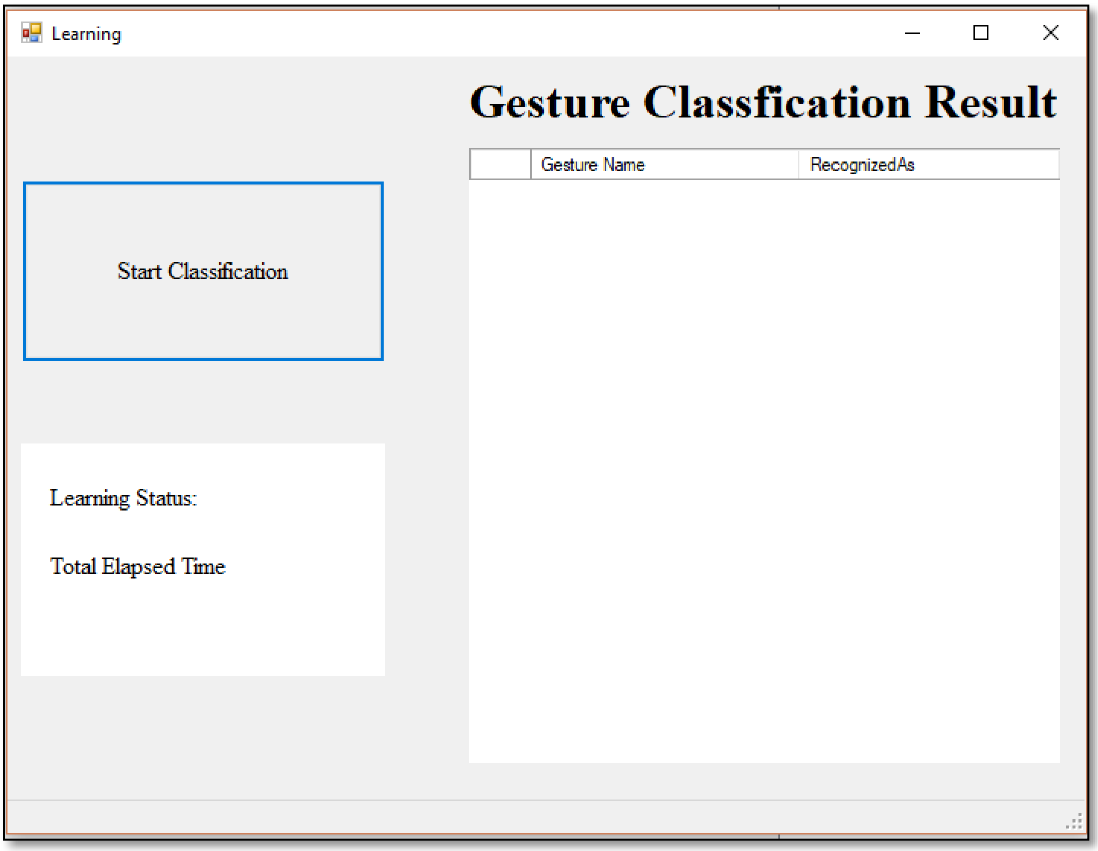


Figure 5.5: Snapshot of the *"Classification"*

The classification algorithms used in the system are HMM and HCRF. With these algorithms, the information that the system developer has recorded is trained by algorithms so that the system can make real-time decisions and notify the user how long the system has learned the data in the database. Information on the duration of training and success rate is shown in Figure 5.6.

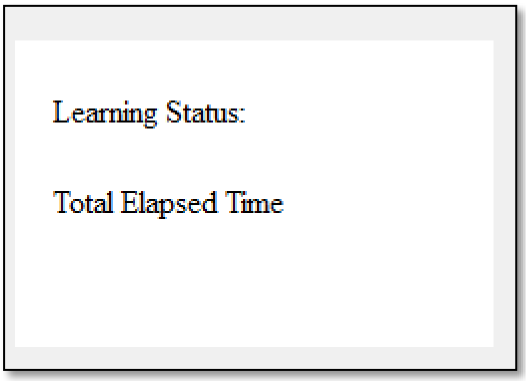
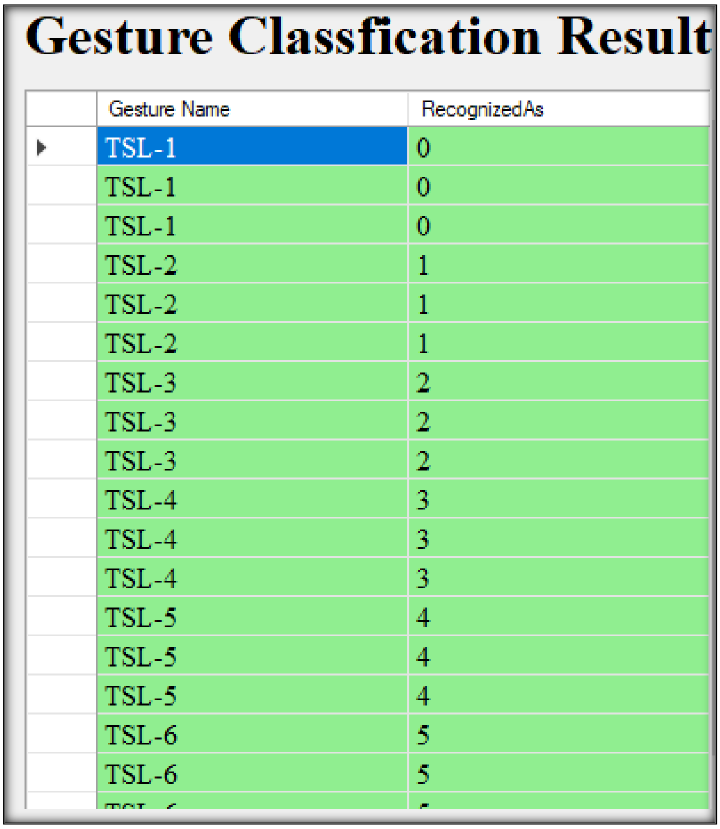


Figure 5.6: Snapshot of the *"Learning Status"*

Gesture classification result is an information section that shows which signs have been included in the classification and which signs have been correctly learned since system training was completed. If the system has learned the signs correctly, the line in which the sign is turning to green colour.



Gesture Classfication Result

Gesture Name	RecognizedAs
TSL-1	0
TSL-1	0
TSL-1	0
TSL-2	1
TSL-2	1
TSL-2	1
TSL-3	2
TSL-3	2
TSL-3	2
TSL-4	3
TSL-4	3
TSL-4	3
TSL-5	4
TSL-5	4
TSL-5	4
TSL-6	5
TSL-6	5

Figure 5.7: Snapshot of the *"Gesture Classification Result"*

Step 3: Open Kinect Sensor

In order to activate the sensor, the Open Kinect button on the system must be pressed.

Figure 5.8: Snapshot of the *"Open Kinect sensor"*

Step 4: Recognize Gesture

Sign recognition is done by classification algorithms in system infrastructure. Recognition processes are performed together with the time rule. By default, the recognition time is set to 5 seconds. This time can be changed through system settings.

Step 5: Save Gesture

This step is used to save a previously unsaved sign. The Frame Save directive must be selected from the system settings for recording. The generated sign dataset is used for 5 seconds in order to be saved in the database. When registration ends, registration is terminated by notifying the user.

Step 7: System Testing

At this stage, if the user brings the instructions on the system to the necessary conditions, the system is ready to be recognized as time rule. When the "Open Kinect" key is pressed, the sensor will be activated and time rule recognition will start. The next screen image after the recognition process is as follows:

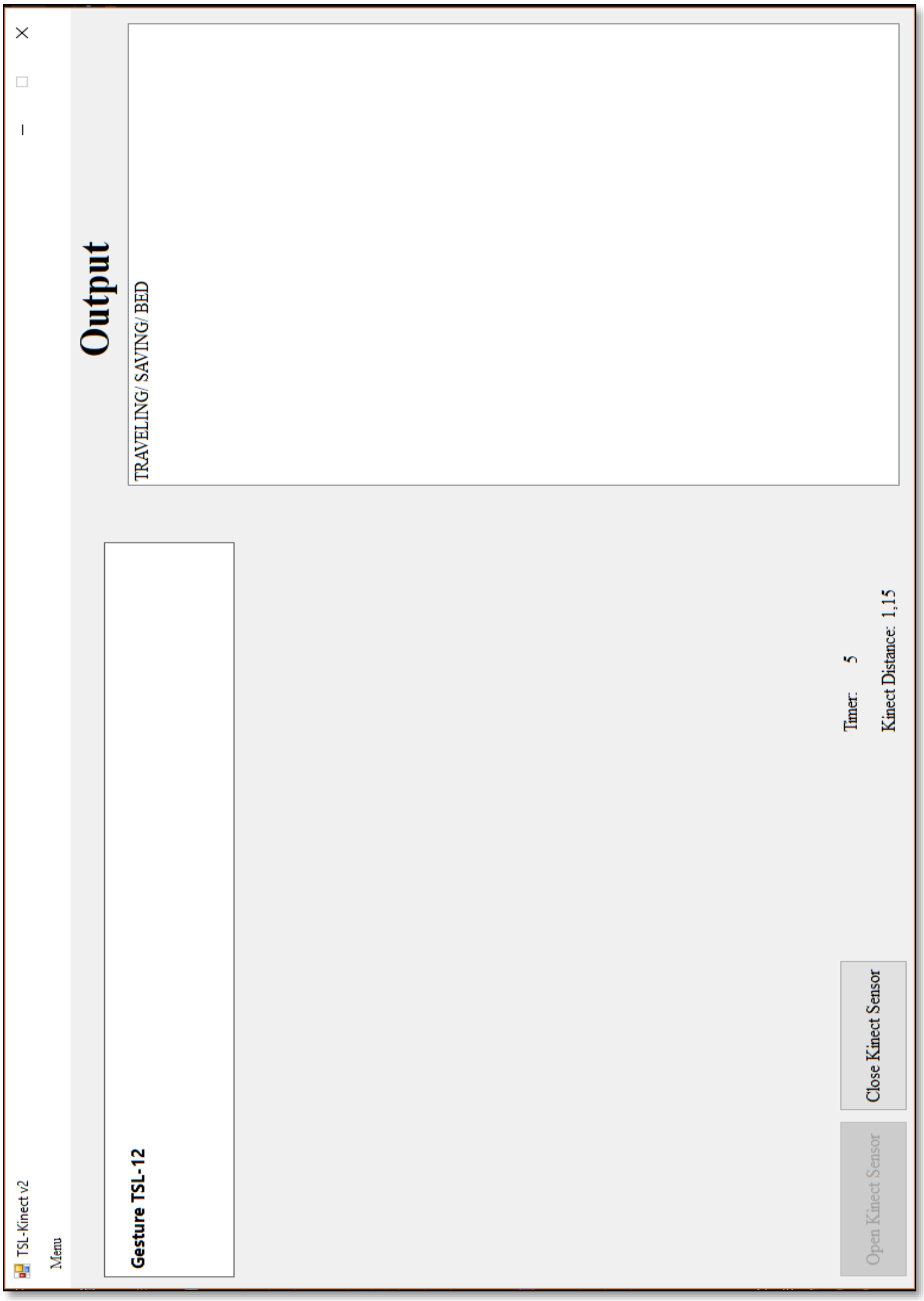


Figure 5.9: Snapshot of the *"Gesture Recognition Result"*

After the sign made by the system is recognized, the "Output" section on the right side of the screen indicates the sign. The sign is made of TSL-12, which is in the phonology of Turkish Sign Language and is written as TRAVELING / SAVING / BED on the right side. The result we achieved in performance revealed an 82% success rate. This suggests that the system correctly recognizes 27 of the 33 basic signs.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

This chapter gives conclusion and recommendation for the future work.

6.1 Conclusion

This research concerning Turkish Sign Language as is used in the TRNC and Turkey looks into ways in which the hearing and speech impaired can live richer lives through communication facilitated by technology. We hope that this work will contribute to the as yet inadequate corpus of literature on this important subject and that it will encourage further research into enhancing the lives of the impaired people concerned. This is with particular reference to system developers who can work on ever better technological solutions for the impaired.

6.2 Recommendations

We propose the use of intelligent algorithms with a view to integrating them into the kinds of systems we have been looking at so that the limitations currently experienced can be addressed particularly in perceiving letters and signs. The long term aim is to have the technology in place whereby a non-impaired individual can be readily understood by the impaired. The text-to-speech systems that are already in place can be upgraded with multi-media support. We need to consult the impaired in order to ascertain what they feel they need in the field of communication so that we can provide it. In real life this means that those with communication disabilities can far better interact with others and be more active and empowered members of their communities.

REFERENCES

Abdel-Fattah, M. (2005). Arabic sign language. *The Journal of Deaf Studies and Deaf Education*, *10*(2), (pp. 212–221).

Agile Manifesto. (2018, Aughust 28). *Agile Manifesto*. Retrieved Aughust 27, 2018, from Agile Manifesto: http://agilemanifesto.org

Akmeliawati, R., Ooi, M. P.-L., & Kuang, Y. (2007). Real-Time malaysian sign language translation using colour segmentation and neural network. *Proceedings of the IEEE Instrumentation and Measurement Technology Conference IMTC*, (pp. 1-6). Warsaw.

Akmeşe, P. P. (2016). Türk İşaret Dili (TİD) eğitimi ve yüksek öğretim programlarında işaret dili dersi. *Proceedings of the Ahi Evran Üniversitesi Kırşehir Eğitim Fakültesi Dergisi*, (pp. 341-361).

Akmeşe, P. P., Kayhan, N., Kirazlı, G., Öğüt, F., & Kirazlı, T. (2018). Odyoloji ve konuşma bozuklukları alanında lisansüstü eğitim öğrencilerinin işitme kayıplı çocukların dil, konuşma ve iletişim becerilerinin desteklenmesi ve eğitimleri hakkındaki görüşleri. *Proceedings of the Türkiye Sağlık Bilimleri ve Araştırmaları Dergisi, 1*(1), (pp. 13-23).

Amon, C., & Fuhrmann, F. (2014). Evaluation of the spatial resolution accuracy of the face tracking system for Kinect for windows v1 and v2. *Proceedings of the 6th Congress of Alps-Adria Acoustics Assosiation.* Graz, Austria. Retrieved June 30, 2018, from https://pdfs.semanticscholar.org/b4d0/8a2ceb8083f097271cbbf38d39c086c4708f.pdf

Anant, A., & Manish, K. (2013). Sign language recognition using Microsoft Kinect. *Proceedings of the IEEE International Conference on Contemporary Computing*, (pp. 181-185).

BDA, B. (2011). *British Deaf Association*. Retrieved August 27, 2018, from British Deaf Association: https://bda.org.uk/help-resources/#statistics

Cengiz, D. U., Hilal, E., Ercan, M. K., & Akgül, F. (2016). İşitme engelli çocuklarda dil kazanımı ve konuşma eğitimi. *Proceedings of the 1. International Congress on Woman and Chıld Health and Training*.

Chuan, C.-H., Regina, E., & Guardino, C. (2014). American sign language recognition using leap motion sensor. *Proceedings of the 13th International Conference on Machine Learning and Applications,* (pp. 541-544). Detroit.

Demircioğlu, B., Bülbül, G., & Köse, H. (2016). Turkish sign language recognition with leap motion. *Proceedings of the 24th Signal Processing and Communication Application Conference (SIU),* (pp. 589-592).

Dikyuva, H., Makaroğlu, B., & Arik, E. (2015). *Türk İşaret Dili Dilbilgisi Kitabı.* Ankara: Aile ve Sosyal Politikalar Bakanlığı.

Dong, C. (2015). *American Sign Language alphabet recognition using Microsoft Kinect.* Masters Theses 7392. Retrieved June 29, 2018, from http://scholarsmine.mst.edu/masters_theses/7392

Elons, A. S., Ahmed, M., Shedid, H., & Tolba , M. F. (2014). Arabic sign language recognition using leap motion sensor. *Proceedings of the 9th International Conference on Computer Engineering & Systems (ICCES),* (pp. 368-373). Cairo.

Fatmi, R., Rashad, S., Integlia, R., & Hutchison, G. (2017). American sign language recognition using hidden markov models and wearable motion sensors. *Proceedings of the Transactions on Machine Learning and Data Mining, 10*(2), (pp. 41-55).

Fenlon, J., & Wilkinson, E. (2015). Sign languages of the world. *Proceedings of the Sociolingustics and Deaf Communities,* (pp. 5-28).

Gülağız, F. K., Özcan, H., & Şahin, S. (2017). An Interactive turkish sign language learning application using leap motion controller. *Proceedings of the International Conference on Advanced Technology and Sciences 2017*.

Haberdar, H. (2005). *Saklı Markov Modeli Kullanılarak Görüntüden Gerçek Zamanlı Türk İşaret Dili Tanıma Sistemi.* İstanbul.

Hundermark, P. (2014). *Do Better Scrum.*

Jingqiu, W., & Ting, Z. (2014). An ARM-based embedded gesture recognition system using a data glove. *Proceedings of the 26th Chinese Control and Decision Conference (2014 CCDC),* (pp. 1580-1584). Changsha.

Jurafsky , D., & Martin, J. H. (2017). *Speech and Language Processing.*

Kalin S, & Jonas, B. (2013). A Kinect corpus of Swedish sign language signs. *Proceedings of the workshop on multimodal corpora: beyond audio and video.*

Kubus, O. (2008). *An analysis of Turkish Sign Language (T.ID) phonology and morphology*.

Lei, L., & Dashun, Q. (2015). Design of data-glove and Chinese sign language recognition system based on ARM9. *Proceedings of the 12th IEEE International Conference on Electronic Measurement and Instruments (ICEMI)*, (pp. 1130-1134). Qingdao.

Lewis, M. P., Simons, G. F., & Fennig, C. D. (2013). Deaf sign language, Ethnologue: Languages of the World (17th ed.). *Proceedings of the SIL International*.

Madabhushi, A., & Aggarwal, J. K. (2000). Using head movement to recognize activity. *Proceedings of the 15th International Conference on Pattern Recognition*, (pp. 698-701).

Mangera, R. (2013). Static gesture recognition using features extracted from skeletal data.

Mapari, R. B., & Kharat, G. (2015). Real time human pose recognition using leap motion sensor,. *Proceedings of the IEEE International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, (pp. 323-328). Kolkata.

Microsoft. (2014). *Microsoft Download Center*. Retrieved June 30, 2018, from Microsoft Download Center: https://www.microsoft.com/en-us/download/details.aspx?id=44561

Perlmutter, & David M. (2018). *What İs Sign Language?* Retrieved June 29, 2018, from Linguistic Society of America: https://www.linguisticsociety.org/content/what-sign-language

Preetham, C., Ramakrishnan, G., & Kumar, S. (2013). Hand Talk-Implementation of a gesture recognizing glove. *Proceedings of the Texas Instruments India Educators' Conference*, (pp. 328-331). Bangalore.

Pterneas, V. (2016, 05 30). *Github*. Retrieved from Github: https://github.com/LightBuzz/Kinect-Finger-Tracking

Ruparelia, N. (2010). Software development lifecycle models. (pp. 8-13). ACM SIGSOFT Software Engineering Notes.

Schwaber, K., & Sutherland, J. (2017, August 28). *The Definitive Guide to Scrum: The Rules of the Game.* The Scrum Guide. Retrieved from https://www.scrumguides.org: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf

Segen, J., & Kumar, S. (1999). Shadow gestures: 3D hand pose estimation using a single camera. *Proceedings of the IEEE international conference on computer vision and pattern recognition*, (pp. 1479–1485).

Souza , C. R., & Pizzolato, E. B. (2013). Sign language recognition with support vector machines and hidden conditional random fields. *Proceedings of the Machine Learning and Data Mining in Pattern Recognition*, (pp. 84-98).

Souza, C. R. (2014, 12). *Accord.NET Framework.* Retrieved from Accord.NET Framework: http://accord-framework.net

Souza, C. R., Pizzolato, E. B., & Anjo, M. d. (2012). Fingerspelling recognition with support vector machines and hidden conditional random fields. *Proceedings of the Advances in Artificial Intelligence – IBERAMIA 2012*, (pp. 561-570).

Starner, W. (1998). A Real-time american sign language recognition using desk and wearable computer based video. *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(12), (pp. 1371-1375).

Stokoe, W. C. (1960,2013). Sign Language Structure: An Outline of the Visual Communication Systems of the American DeafArchived 2013-12-02 at the Wayback Machine., Studies in linguistics: Occasional papers (No. 8). Buffalo: Dept. of Anthropology and Linguis.

Stokoe, W. C., Casterline, D. C., & Croneberg, C. G. (1965). A dictionary of American sign language on linguistic principles. *Gallaudet College Press*.

Sung, Y.-H., & Jurafsky , D. (2010). Hidden Conditional Random Fields for Phone.

Tazhigaliyeva , N., GermanI, P., Yerniyaz, N., & Sandygulova, A. (2016). *SignLanguage Interpreting System for Human-Robot Interaction.*

Turkey Annual Agility Report. (2017). *6 th Annual Agility Report Turkey*. Retrieved August 29, 2018, from https://drive.google.com/file/d/0B0V39X4X2Q0mbXpsUnB4NzV4Z0E/view

Tüfekçioğlu, U. (1998). *İşitme Engelliler.* Eskişehir: Anadolu Üniversitesi Açıköğretim Yayınları. Retrieved June 24, 2018, from İşitme Engelliler: http://content.lms.sabis.sakarya.edu.tr/Uploads/79408/49984/unite08_i%C5%9Fitme_engelliler.pdf

TÜİK. (2011). *Nüfus ve Konut Araştırması*. Retrieved May 03, 2018, from tuik.gov.tr: http://www.tuik.gov.tr/PreHaberBultenleri.do?id=15843

Vogler C, M. D. (1988). ASL recognition based on a coupling between HMMs and 3D motion analysis. *Proceedings of the IEEE international conference on computer vision*, (pp. 363–369).

World Health Organization. (2018). *World Hearing Day 3 March 2018 Report of activities.* Retrieved September 01, 2018, from http://www.who.int/deafness/world-hearing-day/World_Hearing_Day_2018_activity_report_rev1.pdf?ua=1

Yalçınkaya, Ö., Atvar, A., & Duygulu, P. (2016). Turkish sign language recognition application using motion history image. *Proceedings of the 24th Signal Processing and Communication Application Conference, SIU 2016*, (pp. 801-804).

APPENDIX A

SOME PART OF THE SOURCE CODE OF THE DEVELOPED SYSTEM

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Globalization;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Timers;
using System.Windows.Forms;
using LightBuzz.Vitruvius.FingerTracking;
using Metrilus.Aiolos.Core;
using Metrilus.Aiolos.Kinect;
using Microsoft.Kinect;
using MongoDB.Bson;
using MongoDB.Driver;
using SignLanguage.Library.DbContext;
using SignLanguage.Library.Gesture;
using SignLanguage.Library.Helpers;
using SignLanguage.Library.Kinect;
using SignLanguage.Library.MachineLearning;
using Timer = System.Windows.Forms.Timer;

namespace SignLanguage.WinForms
{
    public partial class KinectHmmSvm : Form
    {
```

```
private readonly KinectSensor _kinectSensor;
private readonly CoordinateMapper _coordinateMapper;
private readonly DepthFrameReader _depthReader = null;
private readonly BodyFrameReader _bodyReader = null;
private Body[] _bodies;
private Body _body;
private readonly HandsController _handsController;
private bool _isRecord;
private bool _isRecognition;
private GestureRecorder _recorder;
private int _counter;
private bool _dataReceived;
private bool _bodyTracked;
private int _bodyIndex;
private bool _isTimerStart = false;
public int CorrectGesture = 0;
public int WrongGesture = 0;
private readonly SynchronizationContext _synchronizationContext;
private ushort[] depthFrameData;
private ushort[] irFrameData;
private int _waitingSecond = 0;

public KinectHmmSvm()
{

    InitializeComponent();
    _kinectSensor = KinectSensor.GetDefault();
    if (_kinectSensor != null)
    {
        _coordinateMapper = _kinectSensor.CoordinateMapper;
        _depthReader = _kinectSensor.DepthFrameSource.OpenReader();
        _depthReader.FrameArrived += DepthReader_FrameArrived;
        _bodyReader = _kinectSensor.BodyFrameSource.OpenReader();
        _bodyReader.FrameArrived += BodyReader_FrameArrived;
```

```
            _bodies = new Body[_kinectSensor.BodyFrameSource.BodyCount];
            _isRecord = false;
            _isRecognition = false;

            _handsController = new HandsController();
            _recorder = new GestureRecorder();
            //_kinectHiddenMarkovModel = new KinectHiddenMarkovModel(MongoDbContext);
            //_kinectHiddenMarkovModel2 = new KinectHiddenMarkovModel2(MongoDbContext);
            //_kinectDynamicWrapingTime = new KinectDynamicWrapingTime(MongoDbContext);

        }
        _synchronizationContext = SynchronizationContext.Current;



    }

    private void BodyReader_FrameArrived(object sender, BodyFrameArrivedEventArgs e)
    {
        using (var bodyFrame = e.FrameReference.AcquireFrame())
        {
            if (bodyFrame != null)
            {
                bodyFrame.GetAndRefreshBodyData(_bodies);
                if (_bodies == null)
                {
                    _bodies = new Body[bodyFrame.BodyCount];
                }
                _dataReceived = true;
```

```
        //_body = _bodies.FirstOrDefault(b => b.IsTracked);
        //_bodies = _bodies.Where(s => s.IsTracked.Equals(true)).ToList();

        //if (_bodies.Count > 0)
        //{
        //    string json = _bodies.Serialize(_coordinateMapper, Mode.Color);

        //}
    }
}

if (_dataReceived)
{

    if (_bodyTracked)
    {
        if (_bodies[_bodyIndex].IsTracked)
        {
            _body = _bodies[_bodyIndex];
        }
        else
        {
            _bodyTracked = false;
        }
    }
}

if (!_bodyTracked)
{
    for (int i = 0; i < _bodies.Length; ++i)
    {
        if (_bodies[i].IsTracked)
        {
```

```
                    _bodyIndex = i;
                    _bodyTracked = true;
                    break;
                }
            }
        }
    }

    private async void DepthReader_FrameArrived(object sender, DepthFrameArrivedEventArgs e)
    {

        await Task.Run(() =>
        {
            using (DepthFrame frame = e.FrameReference.AcquireFrame())
            {
                if (frame != null)
                {
                    // 2) Update the HandsController using the array (or pointer) of the depth depth data, and the tracked body.
                    using (KinectBuffer buffer = frame.LockImageBuffer())
                    {
                        if (_body != null && _bodyTracked && _body.IsTracked && buffer != null)
                        {

                            _handsController.Update(buffer.UnderlyingBuffer, _body);
                            var handsData = _handsController.HandsColletionData;
                            //depthFrameData = new ushort[frame.FrameDescription.Width * frame.FrameDescription.Height];
                            //irFrameData = new ushort[frame.FrameDescription.Width * frame.FrameDescription.Height];
                            //frame.CopyFrameDataToArray(depthFrameData);
                            //frame.CopyFrameDataToArray(irFrameData);
```

```
                // KinectHand[] hands = engine.DetectFingerJoints(frame.FrameDescription.Width, frame.FrameDescription.Height, irFrameData, depthFrameData, _bodies, _coordinateMapper);

                if (handsData != null)
                {
                    if (_isTimerStart)
                    {
                        StartTimer();
                        _isTimerStart = false;
                    }

                    var spineBaseDistance = _body.Joints[JointType.SpineBase];
                    var distanceResult = MathKinect.GetDistanceFromSensor(spineBaseDistance);
                    var disRoundedResult = Math.Round(distanceResult, 2)
                        .ToString(CultureInfo.CurrentCulture);
                    UIUpdateKinectStatus(disRoundedResult);
                    //lblKinectStatus.Text = ;

                    if (_isRecord)
                    {
                        _recorder?.RecordFrame(_body, handsData);

                        if (_isRecognition && _isRecord)
                        {
                            //RecognizeGesture(body);
                            if (_recorder != null)
                            {
                                _kinectSensor.Close();
                                try
                                {
                                    ClearFistTwentyFiveFramesAndLastFive();
```

```
                    var gestureName =
GlobalVariables.KinectHiddenMarkovModel2.RecognationGestureByHcrf(_record
er.Frames);
                    _isRecognition = false;

                    UIUpdateGestureName(gestureName,
GestureRecognized.GestureMeaning);

                  }
                  catch (Exception ex)
                  {
                    // ignored
                  }
                }
              }
            }
          }
        }
      }
    }
  }
});


}

private void UIUpdateKinectStatus(string disResult)
{
  _synchronizationContext.Post(new SendOrPostCallback(o => {
lblKinectStatus.Text = disResult; }), disResult);
}
private void UIUpdateGestureName(string gestureName,string
gestureMeaning)
{
```

```
_synchronizationContext.Post(new SendOrPostCallback(o =>
{
   if (!string.IsNullOrEmpty(gestureName))
   {
      if (gestureName != "NOT FOUND")
      {
         //System.Console.WriteLine($"Gesture Found: {gestureName}");
         //System.Console.WriteLine("Gesture   Correct?   1:   Yes,2:   No,3:
Forget it ");
         //var ans = System.Console.ReadLine();

         //lbHaveYouDrawn.Text = $"Have Guest Gesture {gestureName} ?";
         lbHaveYouDrawn.Text = $"Gesture {gestureName}";
         panelClassification.Visible = true;
         lstOutput.Items.Add(gestureMeaning);
         _recorder.Frames = new List<Frame>();

      }
      else
      {
         lbHaveYouDrawn.Text = $"{gestureName} ?";
         panelClassification.Visible = true;
         _recorder.Frames = new List<Frame>();
      }
   }
   else
   {
      lbHaveYouDrawn.Text = $"{gestureName} ?";
      panelClassification.Visible = true;
      _recorder.Frames = new List<Frame>();
      //_kinectSensor.Open();
      //StartTimer();
```

```
            }
        }), gestureName);



    }


    private void btnopenKinect_Click(object sender, EventArgs e)
    {
        //if (GlobalVariables.IsRecognition == false || chcContinous.Checked == false)
        //{
        //    _isTimerStart = true;
        //}
        //else
        //{
        //    //_kinectHiddenMarkovModel2.LearnHcrf();
        //}


        if (GlobalVariables.FrameSave == false && GlobalVariables.IsRecognition == false)
        {
            MessageBox.Show("Please make the system adjustment!", "System Adjustment", MessageBoxButtons.OK,
                MessageBoxIcon.Warning);
            return;
        }
```

```
            if (GlobalVariables.FrameSave &&
(string.IsNullOrEmpty(GestureRecognized.GestureMeaning) ||
string.IsNullOrEmpty(GestureRecognized.GestureName)))
            {
                GestureRegister gestureRegister = new GestureRegister();
                gestureRegister.ShowDialog();
                return;
            }
            BeReady();
            btnCloseKinect.Enabled = true;

        }

        public void BeReady()
        {
            _waitingSecond = GlobalVariables.WaitingSecond;
            beReady = new Timer();
            beReady.Interval = 1;
            beReady.Tick += beReady_tick;
            beReady.Enabled = true;
            btnopenKinect.Enabled = false;
            panel1.Visible = true;
            _counter = GlobalVariables.RecordSecond;
        }

        private async void beReady_tick(object sender, EventArgs e)
        {
            _waitingSecond--;

            if (_waitingSecond == 0)
            {
                beReady.Stop();
                panel1.Visible = false;
                await Task.Run(() =>
```

```
            {
                _kinectSensor.Open();
                //FrameDescription frameDescription =
_kinectSensor.DepthFrameSource.FrameDescription;

                //width = frameDescription.Width;
                //height = frameDescription.Height;

                _isRecord = true;
                _isTimerStart = true;
            });
        }

        lblWaitingStatus.Text = $@"Be Ready:{_waitingSecond}";
        Thread.Sleep(1000);
    }

    private void UIUpdateTimer(string counter)
    {
        _synchronizationContext.Post(o =>
        {
            tmrCounter.Text = _counter.ToString();

        }, counter);

    }
    private void UIUpdateTimer2(string counter)
    {
        _synchronizationContext.Post(o =>
        {
            tmrCounter.Text = _counter.ToString();

        }, counter);
```

```
        }
        private void timer1_Tick(object sender, EventArgs e)
        {
            _counter--;

            //if (_counter == 1 && chcContinous.Checked == false)
            //{
            //    _isRecognition = true;
            //}

            if (_counter == 0)
            {
                timer1.Stop();
                //isRecognition = true;
                _isRecord = false;
                _kinectSensor.Close();

                if (GlobalVariables.FrameSave)
                {
                    try
                    {
                        _counter = GlobalVariables.RecordSecond;
                        // tmrCounter.Text = _counter.ToString();
                        UIUpdateTimer(_counter.ToString());
                        _isTimerStart = false;
                        ClearFistTwentyFiveFramesAndLastFive();
                        var gesture = _recorder.GetRecordedGesture();
                        //gesture.Name = txtGestureName.Text;
                        GlobalVariables.MongoDbContext.Gesture.Save(gesture);
                        //MessageBox.Show("Recording was successfully!", "Create
Operation", MessageBoxButtons.OK,
                        //    MessageBoxIcon.Information);
                        _recorder = new GestureRecorder();
                    }
```

```
                catch (Exception exception)
                {
                    //MessageBox.Show(exception.Message, "DB Kayıt Sorunu",
MessageBoxButtons.OK,
                    //    MessageBoxIcon.Error);
                }
            }

        }

        UIUpdateTimer(_counter.ToString());



        //tmrCounter.Text = _counter.ToString();
    }
    public async void StartTimer()
    {
        await Task.Run(() =>
        {
            timer = new System.Timers.Timer(GlobalVariables.RecordSecond *
1000);

            timer.Elapsed += OnTimedEvent;
            timer.SynchronizingObject = this;
            timer.AutoReset = false;
            timer.Enabled = true;

            //_counter = 5;
            ////_isRecognition =  chckIsRecognition.Checked;
            //_isRecognition = chcContinous.Checked &&
chckIsRecognition.Checked;
            //timer1 = new Timer();
            //timer1.Tick += new EventHandler(timer1_Tick);
            //timer1.Interval = 1000; // 1 second
```

```
            //timer1.Start();
            UIUpdateTimer2(_counter.ToString());

        });



    }
    private void OnTimedEvent(object source, ElapsedEventArgs e)
    {



        if (GlobalVariables.FrameSave)
        {
            _isRecord = false;
            _kinectSensor.Close();
            try
            {
                _counter = GlobalVariables.RecordSecond;
                // tmrCounter.Text = _counter.ToString();
                UIUpdateTimer(_counter.ToString());
                _isTimerStart = false;
                ClearFistTwentyFiveFramesAndLastFive();
                var gesture = _recorder.GetRecordedGesture();
                gesture.Name = GestureRecognized.GestureName;
                gesture.GestureHandName = GestureRecognized.GestureMeaning;
                GlobalVariables.MongoDbContext.Gesture.Save(gesture);
                MessageBox.Show("Recording was successfully!", "Create Operation",
MessageBoxButtons.OK,
                    MessageBoxIcon.Information);
                _recorder = new GestureRecorder();
            }
            catch (Exception exception)
            {
```

```
                MessageBox.Show(exception.Message,      "DB      Kayıt      Sorunu",
MessageBoxButtons.OK,
                    MessageBoxIcon.Error);
                }
            }
            else if (GlobalVariables.IsRecognition)
            {
                //_isRecord = false;

                _isRecognition = true;
            }
        }

        public System.Timers.Timer timer { get; set; }

        private void btnCloseKinect_Click(object sender, EventArgs e)
        {
            try
            {
                _kinectSensor.Close();
                _recorder.Frames = new List<Frame>();
                btnCloseKinect.Enabled = false;
                btnopenKinect.Enabled = true;
            }
            catch (Exception exception)
            {
                MessageBox.Show(exception.Message,    "",    MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }
        }

        public void ClearFistTwentyFiveFramesAndLastFive()
        {
            var frames = _recorder.Frames;
```

```
            if (frames.Count > 30)
            {
                for (int i = 0; i < 25; i++)
                {
                    frames.RemoveAt(i);
                }

                for (int i = 0; i < 5; i++)
                {
                    frames.RemoveAt(frames.Count - 1);
                }
            }
            _recorder.Frames = frames;
        }

        private void KinectHmmSvm_Load(object sender, EventArgs e)
        {
        }

        private void backgroundWorker1_DoWork(object sender, DoWorkEventArgs
e)
        {

        }

        private void btnYes_Click(object sender, EventArgs e)
        {
            CorrectGesture = CorrectGesture + 1;
            var accuracyRate = (double)CorrectGesture /
GlobalVariables.GestureList.Count;
            lstOutput.Items.Add(GestureRecognized.GestureMeaning);
            //lblAccuracyRate.Text = $"Accuracy Rate complete: {accuracyRate:P2}";
        }
```

```
            private void btnNo_Click(object sender, EventArgs e)
            {
                WrongGesture = WrongGesture + 1;
            }

            private void systemAdjustmentToolStripMenuItem_Click(object sender,
        EventArgs e)
            {

            }

            private void systemAdjustmentToolStripMenuItem1_Click(object sender,
        EventArgs e)
            {
                SystemAdjustment systemAdjustment = new SystemAdjustment();
                systemAdjustment.ShowDialog();
            }

            private void startLearningToolStripMenuItem_Click(object sender, EventArgs
        e)
            {
                Learning learning = new Learning();
                learning.ShowDialog();
            }
        }
}
```