ADA University
School of Information Technology and Engineering

Senior Design Project

FINAL REPORT

Project Title: **Speech Recognition in Flight Simulation**

Authors:
1. [CS] Elmin Gasimov
2. [CS] Farid Mammadov
3. [CS] Orkhan Jabrayilov
4. [CS] Ramil Hasanov

Project Advisor: Dr. Samir Rustamov

Baku, May 2018

Table of Contents

Paper Title (use style: *paper title*)

Subtitle as needed *(paper subtitle)*

Elmin Gasimov
School of IT and Engineering
ADA University
Baku, Azerbaijan
egasimov2018@ada.edu.az

Farid Mammadov
School of IT and Engineering
ADA University
Baku, Azerbaijan
fmammadov2018@ada.edu.az

Orkhan Jabrayilov
School of IT and Engineering
ADA University
Baku, Azerbaijan
ojabrayilov2018@ada.edu.az

Ramil Hasanov
School of IT and Engineering
ADA University
Baku, Azerbaijan
rhasanov2018@ada.edu.az

ABSTRACT

Today, Aviation systems, especially flight from one point to another point has increased workloads of flight crew in cockpits which leads to unexpected consequences like decreasing real-time situational awareness. As a developed implementation to manual control of systems, Speech Recognition offers direct access to the most of the cockpit functions. In this research project, we designed and demonstrated a prototype for Speech Recognition in Flight Simulation. We determined command list that are needed and possible to enable recognized by speech, developed software application product, collected and analyzed great amount of data to train the system. Experimented newly developed system. We achieved acceptable recognition performance and identified next steps that need to be done to develop our project in future.

Acknowledgements

We would like to express our sincere gratitude to Dr. Samir Rustamov, our Senior Design Project advisor.

Many thanks to all ADA University, School of Information Technology and Engineering faculty members and staff.

Thanks to ADA University Big Data Center for providing us with necessary equipments required to finish this work done.

Most of all, we would like to thank to everyone who volunteered at providing us with data, as without them completing this project would have been impossible.

I. Introduction

A. Definition

Aviation plays a unique role in connecting business to markets, uniting family and friends, bringing people together to solve problems and develop global insights. In order to provide a better service to customers, and adapt to the environment that develops every day in 21st century, this sector needs some changes that will meet the requirements of users. Since the existence of Information Technologies' applications in every field of our lives is visible, it is significant to have it in aviation sector too. One of the computer technologies that is essential for development of aviation sector is speech recognition. Speech recognition applications spread in aviation sector rapidly and result more widely-spread usage in the future air-dash processes. Speech recognition concept in aviation is not a new idea, and the usage of speech recognition and speech control researches have started decades ago. Having modern computers with much higher level of hardware and software makes speech recognition applications to be used easily and to be applied accordingly in board. The basic principle of these applications is to provide a direct access to the most of the system functions, even when pilot sustains manual control of the aircraft.

B. Purpose

Speech recognition in aviation has been researched and investigated for decades, and has become one of the main concepts in particular area. Speech recognition is the process of entering speech-like information via microphone or or telephone to computer and using computer to convert entered speech signal to text format which explains that speech. The systems that realize these precesses are called automative speech recognition systems.

Different science areas are used for the realization of speech recognition. Physiology, acoustics, signal processing, copy recognition, linguistics, optimization, numerology and etc. are some examples of science areas for creating speech recognition. The difficulties of automative speech recognition result from different aspects of these fields.

One of the primary problems of speech recognition is the difference among the speakers. As example, bustle, accent, emphasis, speaker's age, speaker's gender and etc. can be given. For instance, woman's voice is different that man's, child's voice differ from teenager's. Even the same speech by the same speaker can be seriously changed depending on his/her emotional condition. Another problem of speech recognition is the difference among the environments including the voice of environment, echo, microphones and transmission channels.

Speech recognition systems are the software programs providing recognition of speech. Speech recognition systems are divided into different groups from the first day of their creation according to the application areas of speech recognition. Three types of recognition systems exist according to the dependence of systems on speakers: speaker-dependent system, speaker-independent system, speaker's adaptation system.

Each system mentioned above has following versions depending on the pronounced speech: isolated speech recognition, continuous speech recognition.

Considering the broad application of this topic, Speech Recognition in Flight Simulation is one of the concepts that need to be researched and developed. This paper is expected to investigate details that need to be considered while implementing Speech Recognition in Flight Simulation and provides the developed prototype software product.

C. Project Objectives, Significance, Novelty

Objectives

The primary objective for the crew of any flight is to get from point A to point B safely. Several technological advances had been developed to improve flight safety. Unfortunately, each new technology intended to assist the pilot adds additional complexity. If there are many various information sources in the pilot cabin, all of them must be visually browsed and their results must be mentally synchronized into the pilots' situational awareness. Pilots' increased physical manipulation, multifunction buttons, menu-type access on displays are required for all of these processes [1]. Briefly, today, pilots spend more time manipulating his flight management system than he does actually manipulating aircraft controls and looking out the window [1]. All of the points mentioned above cause ultimate distraction of pilot from his real-time situational awareness. Application of Speech Recognition in Flight Simulation will decrease the level of working with hundreds of buttons, dials, sswitchess and knoobs in pilot cabin. As speaking is how we primarily communicate with each other, implementation of this project, in short, speaking to the cockpit will help pilots to concreate more on flight and deal with management work in cabin less. Objectives of our project are: Providing speech communication between cockpit and pilot, making pilot deal with less management and manipulation tasks, maintaining safe and efficient flight.

Significance

Even relatively straightforward tasks in general aviation require a number of appropriately-sequenced actions in order to execute them. Speech Recognition in Flight Simulation is actual solution for decreasing the pilots' workload of manipulation and management actions in cockpit, and increase pilots' situational awareness during flight which will maintain more safety. Speech Recognition in Flight Simulation could provide a direct access to the most of the system functions, even when pilot sustains manual control of the aircraft. Although new interfaces that are intuitive and easy to use are produced, shear number of tasks that may be executed by the flight crew still cause cockpit environment to be human factors wise and promote more heads-down activity.

Novelty

Several work has been done to solve this problem. Speech Recognition Enabled Cockpit system which has been developed by Adacel Systems Inc is one of them. A Voice - Activated Cockpit developed by this team has succeeded in following areas. "Data Entry for FMS, Autopilot, Direct Aircraft System Queries, Level and/or Heading Bust Monitoring, Correlation of Unfamiliar Local Data, Glass Cockpit Configuration, Electronic Flight Bag (EFB) Interaction, Radio Frequencies - these are achievements of the project which could be used during flight and maintain flight safety" [1]. However, there are some issues about this product. Excluding lates generation of ASR applications, system has problems such like high noises, a multitude of operator accents, changes in speaker's voice, limited command sets, need to train system to recognize each operator's voice patterns, the difference between the printed word and the ways the word or phrase spoken.

D. Problem Statement

There are many software products in market providing speech recognition services with different features, and all of these products cost too much for customers. Some companies prefer implementation of developed software applications, while other companies even do not plan application of this technology. Considering that the concept of speech recognition in flight simulation is a new concept in Azerbaijan, purchasing and implementation of already developed products would be acceptable, however, taking the monetary and technical sides of the problem into consideration, the optimal solution is developing new speech recognition application for flight simulation.

II. Literature Review

If we look through the history of speech recognition, it is known that the first software product in this area belongs to the USA. In 1971, ARPA (Advanced Research Project Agency) of USA provided the project of software product which was considered to be developed in 5 years. Throughout this project, it was aimed to create a machine that would understand the sentences with 1000 words vocabulary. At the end of 1976, several recognition systems were suggested by research groups, and one of them was HARPY system. This system was able to recognize the sentences by 5 operators with the 95% accuracy. Sentences were derived from vocabulary containing 1011 words and owned serious grammatic limitations. Speech Recognition has been investigated for years to sustain better flight services in aviation sector. It is hypothesized that implementation of speech recognition in flight simulation will display higher levels of services, safety, and quality in aircrafts. In this part of report, the following literature reviews attempt to demonstrate and support this hypothesis.

In the article - "The Benefits of a Speech Recognition Enabled Cockpit" by Adacel Systems Inc., three main points were addressed for guiding the study. First, the identification and significance of the problem have been investigated. Second, benefits of voice activated cockpit has been derived and examined. Third, author goes further and investigates the issues of Perceived Automated Speech Recognition, and the problematic areas are mentioned clearly. The focus of the investigation was the implementation of Speech Recognition in cockpit, and analyzing all factors in particular study. It is hypothesized that pilots have great amount of workload during flights, and all of these factors add more complexity to pilots' tasks. Under stressful conditions, these additional complex tasks decrease the chance of safe flight, according to author. The results in this study supported the hypothesis that since speaking is how we primarily communicate with each other, speaking to the cockpit as a method of system management can become an effective interaction method. Additionally, study also mentions that many millions of dollars are being and will continue to be spent on this field, that's why talking to the aircraft will ultimately become second nature.

All of these results combined confirm the hypothesis that speech recognition in flight simulation may increase the quality and safety of flights in hard situations decreasing the level of head-down tasks, and increasing the concentration of pilots. The only limitation to the study is that it does not provide and research more optimal financial solution to the problem. The correlations may have been significantly different if mentioned factor would be involved in study.

Next, Voice-Activated Cokpit for General Aviation issue has been investigated and tested. In the research article by Wesson and Pearson (2006), it is hypothesized that some General Aviation cockpit functions can be and should be voice-activated. The research article shows a new BNF grammar crafted for the chosen Voice-Activated Cockpit functionality, and demonstrates and experiments with Voice-Activated Cockpit using a flight simulation. Study achieved acceptable recognition performance and identified the next steps necessary to optimize the developed prototype of product for General Aviation. Article includes work performed to meet the objectives and implementation issues.

The main goal of this study was to prove the feasibility of the VAC concept. Although it was planned to get laboratory tests of a flight simulation, research was extended via a flight-tested prototype as well which makes correlations of study more accurate. Research study achieved expected 98% word accuracy under normal laboratory conditions, and 96% word accuracy under highly unstable (helicopter) noise conditions. VAC System allowed pilots of both fixed wings and helicopters to fly difficult missions solely by voiced commands reducing pilots' workloads. Grammar and Dictionary optimization and other future plans may help this study to achieve better results than current ones.

Next, Automated Speech Recognition in Air Traffic Control Environment has been addressed and studied. In the research article by Cordereo, Dorado, and Pablo (2012), describes the prototype developed to perform Automated Speech Recognition and controllers event detection, as well as the methodology used to reach it. Research investigates the characteristics of ATC Voice communications including system architecture, automation architecture, models, and system training. The focus of the study is investigate how Air Traffic Control systems may be developed by using Speech Recognition. Article provides test cases and calculations to estimate the quality of speech recognition application in ATC environment,

The results obtained from an automated method of controller workload estimation indicate that for accurate analysis of ATC voice communication needed the development of a new system able to recognize, transcribe and understand what is been said. The developed prototype is able to perform according to needs in ATC environment.

Findings of this study suggest that semantic interpretation for controller event operate, great training effort to feed the core of the system and to establish rich HMM logical relationships are essential factors for Automated Speech Recognition in ATC environment. The one limitation to the study is that the future development of the system in complete isolation of the ATC System is not examined clearly. Additionally, conclusion would be significantly different considering the difference in calculations if planned newly designed algorithms for optimal solutions would be tested in this case.

III. Design Concept

A. *Alternative Solutions/Approaches/Technologies*

There are several flight systems using Speech Recognition in market. This section will provide brief information about some of them.

1) Voice Activated Cockpit

VAC is specific product application by Adacel Inc.'s embedded Speech Recognition Service technology.

[1] It uses combination of a Direct Voice Input interaction with text-to-speech response system into a cockpit user interface to enable the pilot and the aircraft to talk to each other.

[2] The Adacel VAC system reduces pilot workload and cockpit distractions providing natural user interface with control input for modern aircraft.

[3] Many tasks that would require multi-step, manual inputs that diverts pilots attention from other critical tasks, are done by single command enabled by advanced speech recognition. Speaker independent recognition with continuous technology is used by VAC.

[4] Voice commands can be issued in a natural manner without pauses between words and without any requirement for the pilot to carry an acoustic training card.

[5] Adacel's VAC is an effective voice user interface performing above a 98% word accuracy.

2) Air Traffic Management (Aurora by Adacel Inc.)

Aurora ATM automation systems manage all types of airspace and domains from oceanic, en route, terminal and approach sectors to control tower units.

- All available surveillance data such as ADS-B, Multi-Lateration, Radar, ADS-C and pilot position reports are integrated by Aurora with advanced flight data processing capabilities.
- Aurora's automation capabilities enable higher standards of air navigation services and airspace efficiency in all regions that it is currently in use.
- Trajectory based operations in oceanic , en route and terminal airspaces are done with the help of safety nets and highly accurate four dimensional profiles coupled with sophisticated conflict detection.

3) ATCiB by Adacel Inc.

ATCiB is implementation of Simulated ATC Environment for Flight Simulation Training Devices.

- ATCiB establishes and maintains a realistic air traffic and radio communications environment that can operate autonomously within the flight simulation exercise without any required intervention by instructors.
- ATCiB helps to improve the pilots' overall development of concepts such like communications, situational awareness, workload management, and critical decision-making.

- The crew communicates with the simulated controllers to receive ATC clearances and instructions and is able to hear contextually correct communication with other traffic.
- Phraseologies and Grammars basedd on ICAO and FAA standards are flexible to be modified for user specific requirements such as military tactical phraseology.

B. Research Methodology and Techniques

The first thing that had to be done for this project was determining the functions that are needed to implement Speech Recognition. Some functions are impossible to perform via voice, while most of the functions are technically feasible. Within that technically feasible category, some functions are more appropriate to voice activate, while others are not.

Once the specific functions that Speech Recognition will be applied have been identified, the second task to be done was to design BNF grammar to enable recognition. A set of high-level natural language commands that make sense are included in this grammar.

Next step is to develop software product and train system with chosen grammar. For this step, data collecting and analysis have been realized to get as much data as possible.

Another task to be done as a major component of this system was to demonstrate and experiment newly developed product to prove system's ability to recognize spoken commands and response based on input. The initial plan was to maintain laboratory-level experiment.

C. Detailed description of Solutions/Approaches/Technologies of choice

Computers' speech interface should be created firstly for the application of speech in computer technology. Let's look through the creation of speech interface before starting speech recognition.

Following subjects should be completed for the creation of speech interface.

First part explains that computer recognizes (understands) what human says to it, meaning, computer can derive information from human speech according to requirements of application areas. Converting speech to text, recognizing and running of different commands and etc. can be some application areas. At that time, keyboard is replaced with microphone. Another application area can be getting any characteristics from speech (for example: speaker's identification, his/her emotional condition, gender, age identification and etc.).

Second part is about requirement of recognition of speech information by computer. For now, software programs that understand collection of short command-like words have been developed, and their realization is not difficult. Nevertheless, this kind of approach is more difficult than using keyboard and mouse, because clicking on icons with mouse is more comfortable that pronouncing speech correctly (it also disturbs surroundings). For example, "Turn off the program", "Start", "Stop", and etc.

Third part is involved with computer's converting information which it derived to the human-like information.

The factors proved above could also be named speech recognition, speech understanding, and speech synthesis factors respectively. Out of these three factors, only speech synthesis problem had been solved enough. Second factor strongly depends on the solution of first factor. We should also mention that the solution of both factors is realized with the help of Artificial Intelligence systems.

In general, a pilot has three bidirectional channels for information flow, and these are visual, manual, and auditory. Pilot usually, receives cockpit-generated information visually and responds or commands manually. Auditory channel of the pilot is usually reserved for communications with ATC or copilot and passengers. Pilot's visual channel is maxed out under stressful conditions while manual channel is heavily loaded. During all of these processes auditory channel is usually only lightly loaded.

Data entry is one of the essential problems is aircraft systems. Although keypads and keyboards are easy to use in common, in aircrafts keyboards and keypads are too small and compressed than normal desktop versions. Typing longs strings, working manually with keyboards and keypads, especially under stressful flights decreases safety level of flight.

- Direct Aircraft Systems Queries – Rather than step through menus to query specific aircraft systems or scan a specific instrument, a pilot could simply ask the aircraft what he wants to know, much as he would a copilot or flight engineer.
- Data Entry for FMS, Autopilot, Radio Frequencies – Updating the flight profile in flight now becomes easier and safer, as there is far less likelihood of speaking the wrong lat/long or radio frequency than there is in inputting the incorrect data.
- Checklist Assistant – The synthetic speech system leads the pilot through the checklist without the need to refer to its printed version. As the pilot reports compliance, the checklist assistant automatically moves to the next item. Again, these features would provide significant benefit in emergency situations and abnormal flight conditions.

Considering the factors mentioned above, the phraseology mentioned earlier was converted into a BNF grammar. The commands that we included are shown below. Some of the commands have different pronunciation in aviation, that's why they are mentioned with respective transcriptions.

Figure 3.1 List of Commands

Command	Special Pronunciation
Cockpit	
Cockpit checklist completed	
Preflight inspection	
Towbar	
Weight and balance	
A_C Documents	[ˈeə.krɑːft ˈdɒk.jə.mənt]
Circuit Breakers	
Seats & Belts	[siːts ənd belts]
Cabin doors	
Fuel Selector	
All switches	
Completed	
On	
Closed	
Checked	
Off	
Fuel Shutoff Valve	
Shut-off cabin heat	
Alternate air door	
Battery+Main bus	[ˈbæt.ər.i plʌs meɪn bʌs]
Fuel Quantity	
Fuel Temperature	
Flight Controls	
1	
2	[tri:]
3	[faʊə]
4	
5	
6	
7	
8	[ˈzɪr.oʊ]
9	[ˈdes.ɪ.məl]
0	
decimal	
removed	[əˈdʒʌstid ənd lɒkt]
aboard	
adjusted & locked	
locked	
in	
open	
sufficient	
first take	
second take	

D. Requirement analysis

Requirement analysis is essential part in developing process. Which is the process of determining user expectations for a new or modified product. This section will provide the information about what product is expected to do in detail. Functional and non-functional specifications will be described.

1) Functional Specifications

Functional specifications are the services that system is able to provide. It describes how system response to inputs in particular situations. Functional specifications given below define the requirements for Speech Recognition in Flight Simulation:

- The system will read .vaw file
- The system will write .vaw file
- The system will read from streaming audio input
- The system will allow stop streaming
- The system will recognize input file
- The system will recognize streaming audio input
- The system will allow cut files
- The system will allow delete files
- The system will allow define length
- The system will let extract features
- The system will provide list of commands

2) Non-functional Specifications

Non-functional specifications are the requirements that are not concerned with the abilities and functionalities that system provides. Non-functional specifications describe User Interface, Security, Safety, Availability and etc. to describe general specifications of the system. Below are non-functional specifications of our system:

- User Interface
  User-friendly and responsive design will be developed
  The system will let user operate comfortably with buttons
  The system will provide easily-understandable interface
- Availability
  The system will be able to operate in any situation
  The system may have issues if non-accepted commands are spoken
- Security
  The system will be available only for limited users
- Safety
  The system will sustain safe flights
  The system will let pilot to operate and manage easily
  The system will decrease the workload of pilot

All of the requirements were considered during the development process, and the final product responses to all functional and non-functional requirements. The whole system has been written from scratch in C# programming language. Several essential Speech Recognition algorithms have been implemented which will be mentioned later in this article. Data collection and analysis process was realized by special tool named nAudio to collect voice data from different people to train the system. With the help of volunteers who agreed to give their voice data pronouncing given commands for the system. Great amount of data was collected to work on it, and after collection process, this data was analyzed, and useful ones were chosen and developed to train the system. As a result of collecting great amount of data, system is able to recognize and response to all input commands.

After development process, experimentations were organized according to initial plans. In laboratory environment, system was tested and highly accurate expected results were achieved. All requirements and specifications had been met accordingly.

E. Architecture, Model, Diagram description

1) How speech recognition works

Two types of speech recognizers are known They are: speaker-dependent, and speaker-independent. If we consider the speaker-dependent one, it is visible that a concrete person is needed here. System is configured to recognize only speaker's voice. This approach has a high-level accuracy, that's why in past decades, only this approach was used. In speaker-independent speech recognition approach, system is trained using many people. These kind of applications occur in civil aviation reservation.

Dictionary: Recognizer is trained to learn different words. Having different capacities, in small databases 100, average databases 1000, large database 10.000 words are contained, there are even much larger databases containing 64.000 words. However, there is no limitation for larger ones.

Work conditions for speech recognition: There are two types of conditions for speech recognition: Continuous speech recognition and isolated speech recognition. The main idea behind the continuous speech recognition is whenever speaker starts to speak, recognition engine starts working immediately and realizes recognition. In this case, there are some points that requires attention: identifying start points and end points at recognition time, and the speed of speech flow. System should be able to identify the start and end points during continuous speech. Words will have different start and end points depending on covered phonemes. This is called "co-articulation". Speaker's speech speed affects to the recognition accuracy. Speech recognition accuracy will decrease if speech speed is high. Discrete or isolated speech recognition systems stands to the ideology – recognizing one word in given time. In this approach, system requires some break time between each word. Break time can be in different intervals, nevertheless, recognition is done based on the interval in which break time is declared. This type of recognition is the easiest one to execute, because in this approach it is easy to identify the end point, and spoken word's having the same pronunciation with other words is approximately impossible. Users of this kind of system will have to speak with breaks.

Methods to decrease errors: There is no real standard that its application will decrease the speech recognition identification errors. Bustle causes decreasing accuracy percentage. There are four types of errors generally which affects the execution of NT:

substitution errors
insertion errors
rejection errors
operator errors

Substitution errors: Let's assume that pilot commands: "Turn COM one two three decimal nine", but NT understands this command as "Turn NAV one two three decimal nine" by mistake. In this case, specialist working on this system has entered both phrases to the dictionary, and as a result of wrong recognition, the phrase COM pronounced by pilot is replaced with another phrase – NAV and is executed. This kind of error is called intersection error.

Insertion errors: There is a possibility that any kind of bustle or any word in board during speaker's pronunciation may be accepted as the one entered to system vocabulary. This kind of error is called insertion error. To avoid the kind of errors special microphones or "push to talk" method can be used.

Rejection errors: There may be some engines using which speaker has pronounced the word or phrase correctly, but as a result of engine's not responding to command recognition is not done. This may be also understood as rejection of engine. This type of error is called rejection error.

Operator errors: This is considered exactly the speaker's error who uses engine, meaning he/she tries to execute the command using another word instead of the word declared in dictionary. For example, this is recognized by system - "Change radio frequency 1 2 3 .4", however pilots says: "Change COM to 1 2 3 .4". This type of error is called operator error.

2) Modules

Characteristics calculation module. This module is created to verify the parameters that are used during the calculation of characteristics of speech. Following tasks are done here:

TABLE I. Entering speech to computer: speech is converted to numeric data in analog numeric converter. Speech changes approximately between (60 – 4000Hz) low frequency interval. According to Kotelnikov theorem, discrete frequency considered for speech signal should be taken at least 8kHz. That's why, in FSR, user is provided with three discrete frequencies (8kHz, 11.025kHz, 16kHz) to write speech to computer. Speech signal is written in mono-channel style. In ANTS, user is able to enter speech directly or in a form of file. User can also do some operations here, such like extra speech pronouncing, pause, stop, get the time description of speech and analyze it using magnifier. While speech is written, its time continuity is also noted in application. Cleaning bustle in speech: As speech signal changes in low frequency, for cleaning high frequency bustle of it, speech is filtered in first adjusted high frequency FIR. ANTS provides user with a chance of changing parameter of this filter.

Let's apply this filter to word "Absheron". Image 1. a) The first visualization of speech signal in application, Image 1. b) Visualization of speech signal after filtered in FIR. It is visible that after the application of filter, the amount of high frequency bustles has decreased considerably.

Identification of the end points of speech: Identifying the start and end points of pronounced speech is one of the main problems. Inaccurate identification of speech's end points depends on the speech diagnostic's recognition quality. VAD (Voice Activation Detection) is one of the methods used to identify the end points of speech signal. ANTS provides user with three parameters to identify the end points of speech accurately.

1. The number of blocks that speech is divided to identify the end points of speech: In our system, for silence 100 partitions have been spared. Having high value helps to get the pauses between speech's phonemes.
2. The number of blocks in which environmental sounds occur: In our system, 5 blocks have been spared for silence (30msecs).
3. Speech separation scalar: Changing this scalar according to speech's writing sensitivity, we can test the correctness of identification of speech's end points.

Dividing words which establish speech into classes, and finding their average length: As the recognition system that we create is realized with neuron network model, the enterance of neuron should be collection of data taken in equal amount. That's why speech parts with different length are resulted in matching length. Lagrange's interpolation issue is used here. ANTS allows user to enter speech's average length value. For the silence, the average length value is considered as the mean value of the copies that will be learned in system.

Depending on the pronunciation of different words, speech gets values from different sides of identified borders. To avoid this, the learning and recognition of words which are in intersection interval in 2 neighbor classes is realized.

From the comparison of obtained recognition results, it is assumed that, it is not desirable to have less than 3 classes. The results that are obtained taking more classes is practically same. That's why optimal number of classes is 3.

Speech framing: According to the practices, speech signal keeps its stationarity in some level, for approximately 20 ms. Using speech signal's kvazisterial characteristics, to analyze speech, it is divided into frames. In the next steps, before applying discrete Fur-ye conversion to signal, to decrease the spreading of specter, signal is multiplied by mass function or the function called "Windows" (Hamming function). After Hamming window application in signal's time region, information decreases considerably. That's why, to avoid the risk of loosing useful information of speech signal, frames are approximately overlapped. ANTS suggests user the chance to change the frames' length and frame's step, depending on discretization frequency. As speech discretization for silence is $f_s = 16\,\text{kHs}$, frame length is 400, ad frame step is considered 160 partitions.

Token calculations from speech frames: In ANTS, two types of token calculation algorithms are used parallelly: these are MFCC and LPC tokens.

To get MFCC(Mel Frequency Cepstral Coefficients) cepstrals, discrete Fur-ye is applied to framed speech signal. To decrease the amount of information that we got more, Mel filter is used. Using the Mel filter, it is possible to decrease the amount of data without loosing useful information. Because of that reason signal is filtered in Mel filter. The primary parameter in filter the number of its channels. Considering that, ANTS provides user with the chance to change the number of those channels. For silence, the number of channels is 24. After applying logarithm to the signal, we get MFCC cepstrals applying reverse discrete Fur-ye. Depending on the user's choice ANTS also provides the function to enter MFCC cepstrals' speed and acceleration tokens to token vector.

To calculate the LPC (Linear Predictive Coding) tokens, first of all LPC scalars describing each frame are calculated with the help of Levinson-Durbin algorithm. ANTS lets user to change the number of components in LPC tokens vector according to each frame. For silence, the number of components in LPC tokens vector is 12. With the help of calculated LPC scalars, frames' cepstrals are calculated. Applying average subtraction to calculated 12 LPC ceepstrals in the next step, they are added to token vector.Cleaning the speech from channel effect: While recording speech signal some bustles occur, which is called channel effect. During system training and afterward usage of it different channel effects decreases the system's rcognittion accuracy. To avoid this, cepstral average subtraction is used.

System training module. Using this module, user enters the computer the speech samples which will be trained. ANTS identifies the useful parameters for system processing these samples.

Depending on the pronunciation of speech, system divides speech samples to classes according to their length: short, average, long. Afterwards to train the computers with calculated tokens, artificial neuron network model is used. For this a seperate network is created for each class. There are exit neurons according the number of words in class, and enterance neurons according to the number of tokens matching to words. The neuron network that we use is connected graph. Neuron network is trained using joint gradient. Minimization error is accepted 0.01. As created neuron network is not accepted one digit independent of start point, each network, starting from different start points, is trained multiple times.

Creating module realizing recognition. As mentioned above, in the speech's initial processing block, its MFCC and LPCC tokens are calculated. Training and recognition are done based on both two tokens. Results are compared based on recognition according to both two tokens.

The system that works using MFCC tokens is named MFCC-based, and the system that is working with LPC tokens is called LPC-based system. In our system, we suggest the usage of both two tokens at the same time. That's why speech recognition system consists of two MFCC-based and LPC-based subsystems. These subsystems are trained with MFCC and LPC network model separately.

Recognition process is done in two steps:

1. Parallel recognition processes are done in MFCC-based and LPC-based subsystems.
2. Calculated recognition results are compared in MFCC-based and LPC-based subsystems, and speech recognition system decides on the result that both two subsystem decides.

IV. IMPLEMENTATION

A. *Software Design*

Implemented Algorithms in Project

1) *Fast Fourier Transform [8]*

[The shortened version of the *Fast Fourier Transform is called FFT*. Essentially, the FFT is still the DFT for transforming the discrete-time signal from time domain into its frequency domain. The difference is that the FFT is faster and more efficient on computation. The most widely used FFT algorithm is the *Radix-2 FFT Algorithm.*

Since FFT is still the computation of DFT, so it is convenient to investigate FFT by firstly considering the N-point DFT equation:

$$X(k) = \sum^{N-1} x(n) W_N^{kn}, \; k = 0, 1, 2 \ldots N-1$$

Firstly separate x(n) into two parts: x(odd)=x(2m+1) and x(even)=x(2m), where m=0, 1, 2 ,..., N / 2 - 1. Then the N-point DFT equation also becomes two parts for each N/2 points:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} = \sum_{m=0}^{N/2-1} x(2m) W_N^{2\,mk} + \sum_{m=0}^{N/2-1} x(2m+1) W_N^{(2\,m+1)\,k} = \sum_{m=0}^{N/2-1}$$

where m $= 0, 1, 2, \ldots., \mathrm{N} / 2 - 1$

Since:

$e^{j\omega_k n} = \cos(\omega_k n) + j \sin(\omega_k n).$

$e^{j(\omega_k + \pi)n} = \cos[(\omega_k + \pi)n] + j \cdot \sin[(\omega_k + \pi)n]$

14 $-\cos(\omega_k n) - j \cdot \sin(\omega_k n) = -[\cos(\omega_k n) + j \cdot \sin(\omega_k n)] = -e^{j\omega_k n}$

That is: $e^{j(\omega_k + \pi)n} = -e^{j\omega_k n}$

So when the phase factor is shifted with half period, the value of the phase factor will not change, but the sign of the phase factor will be opposite. This is symmetry property of the phase factor. Since the phase factor can be also expressed as

$W_N^{kn} = e^{j\omega_k n}$, so:

$W_N^{(k+\frac{N}{2})n} = -W_N^{kn}$ *AND*

$$(W_N^{kn})^2 = -W_{N/2}^{kn} = e^{\frac{4\pi k_j}{N} n}$$

The N-point DFT equation finally becomes:

$$X(k) = \sum_{m=0}^{N/2-1} x_1(m) W_{N/2}^{mk} + W_N^k \sum_{m=0}^{N/2-1} x_2(m) W_{N/2}^{mk} = X_1(k) + W_N^k X_2(k), \mathrm{k} =$$

$$X(k + N/2) = X_1(k) - W_N^k X^2(k), \; \mathrm{k} = 0, 1, 2 \ldots \mathrm{N} / 2$$

So N-point DFT is separated into two N/2-point DFT. From equation (21), X_1(k) has (N/2) · (N/2) = $(N/2)^2$ complex multiplications. $W_N^k X_2(k)$ has N/2+$(N/2)^2$ complex multiplications.

So the total number of complex multiplications for X(k) is $2 \cdot (N/2)^2 + N/2 = N^2/2 + N/2$. For original N-point DFT equation (14), it has N^2 complex multiplications. So in the first step, separating x(n) into two parts makes the number of complex multiplications from N^2 to $N^2/2+N/2$. The number of calculations has been reduced by approximately half.

This is the process for reducing the calculations from N points to N/2 points. So continuously separating the x_1(m) and x_2(m) independently into the odd part and the even part in the same way, the calculations for N/2 points will be reduced for N/4 points. Then the calculations of DFT will be continuously reduced. So if the signal for N-point DFT is continuously separated until the final signal sequence is reduced to the one point sequence. Assuming there are $N=2^s$ points DFT needed to be calculated. So the number of such separations can be done is $s=\log_2(N)$. So the total number of complex multiplications will be approximately reduced to (N/2) $\log_2$ (N). For the addition calculations, the number will be reduced to N $\log_2$ (N) [2]. Because the multiplications and additions are reduced, so the speed of the DFT computation is improved. The main idea for Radix-2 FFT is to separate the old data sequence into odd part and even part continuously to reduce approximately half of the original calculations] [8].

Spectrum Normalization

After doing FFT calculations, the investigated problems will be changed from discrete-time signals to the frequency domain signals X(ω). The spectrum of the X(ω) is the whole integral or the summation of the all frequency components. When talking about the speech signal frequency for different words, each word has its frequency band, not just a single frequency. And in the frequency band of each word, the spectrum ($X(\circ)$) or spectrump ower ($X(\circ)^2$) has its maximum value and minimum value. When comparing the differences between two different speech signals, it is hard or unconvincing to compare two spectrums in different measurement standards. So using the normalization can make the measurement standard the same.

In some sense, the normalization can reduce the error when comparing the spectrums, which is good for the speech recognition [3]. So before analyzing the spectrum differences for different words, the first step is to normalize the spectrum X(。) by the linear normalization. The equation of the linear normalization is as below:

y=(x-MinValue)/(MaxValue-MinValue)

After normalization, the values of the spectrum X(。) are set into interval [0, 1]. The normalization just changes the values' range of the spectrum, but not changes the shape or the information of the spectrum itself. So the normalization is good for spectrum comparison. Using MATLAB gives an example to see how the spectrum is changed by the linear normalization. Firstly, record a speech signal and then apply the FFT to the speech signal. Then take the absolute values of the FFT spectrum. The FFT spectrum without normalization is as below:

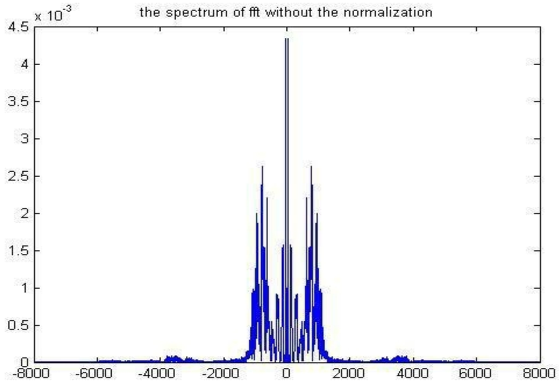


Figure 4.1 Absolute values of the FFT spectrum without normalization

Secondly, normalize the above spectrum by the linear normalization. The normalized spectrum is as below:

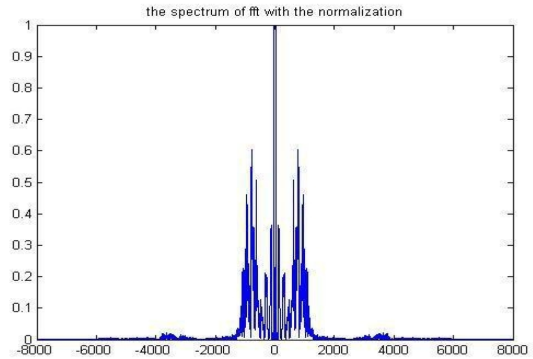


Figure 4.2 Absolute values of the FFT spectrum with normalization

From the Fig.4 and the Fig.5, the difference between two spectrums is only the interval of the spectrum X(。) values, which is changed from [0, 4.5×10^{-3}] to [0, 1]. Other information of the spectrum is not changed. After the normalization of the absolute values of FFT, the next step of programming the speech recognition is to observe spectrums of the three recorded speech signals and find the algorithms for comparing differences between the third recorded target signal and the first two recorded reference signals.

2) MFCC an LPC Algorithms Combined

At first the speech signals is transformed into electric oscillation by the sound recorders (for example, microphone). Later the signal passed over analog-digital converter is transformed into digital form at some sampling frequency f_d and quantization level. The sampling frequency - analog signal without losing its important information determines the necessary frequency for sampling.

The main part of speech recognition system consists of training and recognition processes. Initially basic features characterizing speech signal are computed in both processes. The efficiency of this stage is one of the significant factors affecting behavior of the next stages and exactness of speech recognition. Using the time function of the signal as feature is ineffective. The reason for this is that when the same person says the same word, its time function varies significantly.

At present the methods of calculating MFCC (Mel Frequency Cepstral Coefficients) and LPC (Linear Predictive Coding) are widely used in speech recognition as speech features.

Let's explain the essence of these methods, separately:

The model of speech generation consists of two parts: the generation of the excitation signal and the vocal tract filter. The excitation signal is spectrally shaped by a vocal tract equivalent filter. The outcome of this process is the speech. If $e(n)$ denotes a sequence of the excitation signal and

$\theta(n)$ denotes the impulse response of the vocal tract equivalent filter, a sequence of the speech is then equal to the excitation signal convolved with the impulse response of the vocal tract filter as shown in equation.

$$s(n)=e(n)*\theta(n)$$

A convolution in the time domain corresponds to a multiplication in the frequency domain:

$$S(\omega)=E(\omega)\cdot\theta(\omega)$$

In MFCC method using the logarithm of equation, the multiplied spectra becomes additive

$$\log|S(\omega)|=\log|E(\omega)\cdot\theta(\omega)|=\log|E(\omega)|+\log|\theta(\omega)|$$

It is possible to separate the excitation spectrum $E(\omega)$ from the vocal system spectrum $\theta(\omega)$ by remembering that: $E(\omega)$ is responsible for the "fast" spectral variations, $\theta(\omega)$ is responsible for the "slow" spectral variations. Frequency components corresponding to $E(\omega)$ appear at "large values" on the horizontal axis in the "new frequency domain", whereas frequency components corresponding to $\theta(\omega)$ appear at "small values". The new domain found after taking the logarithm and the inverse Fourier transform is called the cepstrum domain, and the word quefrency is used for describing the "frequencies" in the cepstrum domain.

As same way Z transform is applied to the convolution in the time domain in method LPC:

$$S(z)=E(z)\cdot\theta(z)$$

The main idea behind linear prediction is to extract the vocal tract parameters. Given a speech samples at time n, $s(n)$ can be modeled as a linear combination of the past p speech samples, such that:

$$s(a;n)=\sum_{k=1}^{p}a_p(k)\cdot s(n-k)$$

where $a_p=(a_p(1),a(2),\dots,a_p(p))$ are unknown LPC coefficients $(p\in[8,12])$.

Summing the real and predicted samples we get the following signal:

$$e(a;n)=s(n)+s(a;n)=s(n)+\sum_{k=1}^{p}a_p(k)\cdot s(n-k)$$

Apply to this signal the $z-$ transform:

$R(z)=S(z)A(z)$.

The filter $A(z)=1+\sum_{k=1}^{p}a_p(k)z^{-k}$ is called the predicting error filter. This filter is equal to the inverse value of vocal tract equivalent filter.

$$A(z)=\frac{1}{\theta(z)}\ .$$

To find the vocal tract filter $\theta(z)$, we must first find the LPC coefficients a_p. By this aim, the following function is minimized

$$\varepsilon_p(a)=\sum_{n=1}^{M}|e(a;n)|^2\to\min$$

where M is a number of frames.

We use the necessary condition of minimum to solve the problem:

$$\frac{\partial\varepsilon_p(a)}{\partial a_p(k)}=\frac{\partial}{\partial a_p(k)}\sum_{n=1}^{M}|e(a;n)|^2=2\sum_{n=1}^{M}e(a;n)\frac{\partial}{\partial a_p(k)}e(a;n)=$$

$$¿2\sum_{n=1}^{M}e(a;n)\frac{\partial}{\partial a_p(k)}\left[s(n)+\sum_{l=1}^{p}a_p(l)s(n-l)\right]=$$

$$2\sum_{n=1}^{M}e(a;n)s(n-k)=0\ ,\ k=1,2,\dots,p$$

Then we get

$$\sum_{n=1}^{M}\left[s(n)+\sum_{l=1}^{p}a_p(l)s(n-l)\right]s(n-k)=0$$

Let's denote $r_x(k)=\sum_{n=1}^{M}s(n)s(n-k)$.

Consequently we can write the equation as following form.

$$r_x(k)+\sum_{l=1}^{p}a_p(l)r_x(l-k)=0 \quad \text{or}$$

$$\sum_{l=1}^{p}a_p(l)r_x(k-l)=-r_x(k)\ ,$$

$$k=1,\dots,p$$

This equation is called the normal equation or the Yule-Walker equation.

Then using previous equations, we get:

$$\varepsilon_p(a)=\sum_{n=1}^{M}|e(a;n)|^2=\sum_{n=1}^{M}e(a;n)e(a;n)=$$

$$\sum_{n=1}^{M}e(a;n)\left[s(n)+\sum_{k=1}^{p}a_p(k)s(n-k)\right]=$$

$$=\sum_{n=1}^{M}e(a;n)s(n)+\sum_{k=1}^{p}a_p(k)\sum_{n=1}^{M}e(a;n)s(n-k)$$

While $\sum_{n=1}^{M} e(a;n)s(n-k)=0$, we can write:

$$\varepsilon_{p,\min}(a)=\varepsilon_p(a)=\sum_{n=1}^{M} e(a;n)s(n)=\sum_{n=1}^{M}\left[s(n)+\sum_{k=1}^{p} a_p(k)s(n-k)\right]s(n)=$$

$$=r_x(0)+\sum_{k=1}^{p} a_p(k)r_x(k).$$

The coefficients $a_p(k)$, which giving the minimum to the functional is found by using following Levinson- Durbin recursion.

a. a) $a_0(0)=1$ b) $E_0=r_x(0)$

b. For $j=0,1,\dots p-1$ calculated the following expressions:

a) $\gamma_j=r_x(j+1)+\sum_{i=1}^{j} a_j(i)r_x(j-i+1)$

b) $\Gamma_{j+1}=-\gamma_j/E_j$

a. $i=1,2,\dots,j$

$a_{j+1}(i)=a_j(i)+\Gamma_{j+1}a_j(j-i+1)$

b. $a_{j+1}(j+1)=\Gamma_{j+1}$

c. $E_{j+1}=E_j\left[1-\left|\Gamma_{j+1}\right|^2\right]$

Calculating of MFCC features.

Fast Fourier transform: **A**pplying by FFT to windowing frames are calculated spectrum of frames.

$$\mathrm{bin}_k=\left|\sum_{n=1}^{N} s_w(n)e^{-i(n-1)k\frac{2\pi}{N}}\right|,k=0,1,2,\dots,N-1$$

.

Mel filtering. The low-frequency components of the magnitude spectrum are ignored. The useful frequency band lies between $64\,\mathrm{Hz}$ and half of the actual sampling frequency. This band is divided into 23 channels equidistant in mel frequency domain. Each channel has triangular-shaped frequency window. Consecutive channels are half-overlapping.

The choice of the starting frequency of the filter bank, $f_{\text{start}}=64\,\mathrm{Hz}$, roughly corresponds to the case where the full frequency band is divided into 24 channels and the first channel is discarded using any of the three possible sampling frequencies.

The centre frequencies of the channels in terms of FFT bin indices (cbin_i for the i -th channel) are calculated as follows:

$$\mathrm{Mel}(x)=2595\lg\left(1+\frac{x}{700}\right),x=700\cdot\left(10^{\frac{\mathrm{mel}}{2595}}-1\right)$$

,

$$f_{c_i}=\mathrm{Mel}^{-1}\left\{\mathrm{Mel}\{f_{\text{start}}\}+\frac{\mathrm{Mel}\{f_s/2\}-\mathrm{Mel}\{f_{\text{start}}\}}{\mathrm{NF}}i\right\},$$
$$i=1,2,3,\dots,\mathrm{NF}-1$$

,

$$\mathrm{cbin}_i=\mathrm{round}\left\{\frac{f_{c_i}}{f_s}N\right\}$$,

where $\mathrm{round}()$ stands for rounding towards the nearest integer. $\mathrm{NF}=24$ -is the number of channels of filter.

The output of the mel filter is the weighted sum of the FFT magnitude spectrum values (bin_i) in each band. Triangular, half-overlapped windowing is used as follows:

$$fbank_k=\sum_{i=cbin_{k-1}}^{cbin_k}\frac{i-cbin_{k-1}+1}{cbin_k-cbin_{k-1}+1}bin_i+$$
$$+\sum_{i=cbin_k+1}^{cbin_{k+1}}\left(1-\frac{i-cbin_k}{cbin_{k+1}-cbin_k+1}\right)bin_i,\ k=1,2,\dots,NF-1.$$

where cbin_0 and cbin_{24} denote the FFT bin indices corresponding to the starting frequency and half of the sampling frequency, respectively,

$$\mathrm{cbin}_0=\mathrm{round}\left\{\frac{f_{\text{start}}}{f_s}N\right\}$$;

$$\mathrm{cbin}_{24}=\mathrm{round}\left\{\frac{f_s/2}{f_s}N\right\}=\frac{N}{2}$$

Non-linear transformation. The output of mel filtering is subjected to a logarithm function (natural logarithm)

$f_i=\ln(\mathrm{fbank}_i),i=1,2,\dots,\mathrm{NF}-1$.

Cepstral coefficients. 12 cepstral coefficients are calculated from the output of the non-linear transformation block.

$$C_i=\sum_{j=1}^{\mathrm{NF}-1} f_j\cdot\cos\left(\frac{\pi\cdot i}{\mathrm{NF}-1}(j-0.5)\right),i=1,..,12$$

.

We apply to these 12 LPC cepstrals the cepstral mean subtraction and enter to the feature vector in next step.

Cepstral Mean Subtraction (CMS). A speech signal may be subjected to some channel noise when recorded, also referred to as the channel effect. A problem arises if the channel effect when recording training data for a given person is different from the channel effect in later recordings when the person uses the system. The problem is that a false distance between the training data and newly recorded data is introduced due to the different channel effects. The channel effect is eliminated by subtracting the mel-cepstrum coefficients with the mean mel-cepstrum coefficients:

$$\mathrm{mc}_j(q)=C_j(q)-\frac{1}{M}\sum_{i=1}^{M} C_i(q),q=1,2,,\dots,12$$

***Calculating of LPC features**.*

The LPC coefficients of each frame are found by applying Levinson-Durbin algorithm and following cepstrals are calculated .

.

We apply the cepstral mean subtraction to these 12 LPC cepstrals and enter to the feature vector in next step.

3) *Voice Activation Detection Algorithm*

1. *Pre-processing.* The amplitude spectrum of a speech signal is dominant at "low frequencies" (up to approximately 4 kHz). The speech signals is passed through a first-order FIR high pass filter:

$$s_p(n)=s_{\text{in}}(n)-\alpha\cdot s_{\text{in}}(n-1)$$

where $\alpha-$ is the filter coefficient $(\alpha\in(0,95;1))$, $s_{\text{in}}(n)-$ is the input signal.

2. *Voice activation detection (VAD).* The problem of locating the endpoints of an utterance in a speech signal is a major problem for the speech recognizer. An inaccurate endpoint detection will decrease the performance of the speech recognizer. Some commonly used measurements for finding speech are short-term energy estimate E_s , or short-term power estimate P_s , and short term zero crossing rate Z_s . For the speech signals $s_p(n)$ these measures are calculated as follows:

$$E_s(m)=\sum_{n=m-L+1}^{m} s_p^2(n) \quad ,$$

$$P_s(m)=\frac{1}{L}\sum_{n=m-L+1}^{m} s_p^2(n) \quad ,$$

$$Z_s(m)=\frac{1}{L}\sum_{n=m-L+1}^{m}\frac{|\operatorname{sgn}(s(n))-\operatorname{sgn}(s_p(n-1))|}{2}$$

where

$$\text{¿}$$
$$1, s(n)\geq 0,$$
$$-1, s_p(n)<0.$$
$$\text{¿}\operatorname{sgn}(s_p(n))=\{$$
$$\text{¿}$$

For each block of $L=100$ samples these measures calculate some value. The short term zero crossing rate gives a measure of how many times the signal, $s_p(n)$, changes sign. This short term zero crossing rate tends to be larger during unvoiced regions.

These measures will need some triggers for making decision about where the utterances begin and end. To create a trigger, one needs some information about the background noise. This is done by assuming that the first 5 blocks are background noise. With this assumption, the mean and variance for the measures will be calculated. To make a more comfortable approach, the following function is used:

$$W_s(m)=P_s(m)\cdot(1-Z_s(m))\cdot S_c \quad .$$

Using this function both the short-term power and the zero crossing rate will be taken into account. S_c is a scale factor for avoiding small values, in a typical application is $S_c=1000$. The trigger for this function can be described as:

$$t_W=\mu_W+\alpha\delta_W$$

the μ_w is the mean and δ_w is the variance for $W_s(m)$ calculated for the first 5 blocks. The α term is constant that have to be fine tuned according to the characteristics of the signal. After some testing the following approximation of α will give a pretty good voice activation detection in various level of additive background noise.

$$\alpha=0,2\cdot\delta_W^{-0,4} \quad .$$

The voice activation detection function, $\text{VAD}(m)$, can be found as:

$$\text{¿}$$
$$1, W_s(m)\geq t_W,$$
$$0, W_s(m)<t_W.$$
$$\text{¿}\,\text{VAD}(m)=\{$$
$$\text{¿}$$

By using this function we can detect the endpoints of an utterance.

1. *Framing.* The input signal is divided into overlapping frames of N samples.

$$s_{\text{frame}}(n)=s_p(n)\cdot w(n) \quad ,$$

$$\text{¿}$$
$$1, K\cdot r<n\leq K\cdot r+N, r=0,1,2,\ldots,M-1,$$
$$0,\text{otherwise},$$
$$\text{¿}\,w(n)=\{$$
$$\text{¿}$$

where M is the number of frames, f_s is the sampling frequency, t_{frame} is the frame length measured in time, and K is the frame step.

$$N=f_s\cdot t_{\text{frame}} \quad .$$

We use the $f_s=16\text{ kHs}$ sampling frequency in our system.

2. *Windowing.* There are a number of different window functions to choose between to minimize the signal discontinuities. The Hamming window is one of the mostly used methods for windowing speech signal before Fourier Transformation:

$$s_w(n)=\left\{0,54-0,46\cos\left(\frac{2\pi(n-1)}{N-1}\right)\right\}s_{\text{fram}}$$

4) *Construction of Neuron Network*

There are various mathematical models which form the basis of speech recognition systems. The widely used model is Multilayer Artificial Neural Network (MANN). Let’s briefly describe the structure of MANN.

Generally, MANN is incompletely connected graph. Let L – quantity of MANN’s layers, N_ℓ - neuron quantity on layer l , $l=1..L$; I_{lj}^{-} - set of neurons of layer $(l-1)$, which connected to the neuron j on layer l ; θ_j^l - bias of neuron j on layer l ; w_{ij}^{ℓ} - weighted coefficient (synapse) of connection between of neuron i on layer $(l-1)$ and neuron j on layer l ; $s_{j,p}^{\ell}$ and $y_{j,p}^{\ell}$ - state and output value of neuron j on layer l for input signal $x_p\in X$ of MANN.

Forward propagation of MANN for $x_p\in X$ input signal has been described by the following expressions

$$s_{j,p}^{l}=\sum_{i\in I_{\text{lj}}^{-}} w_{\text{ij}}^{l}\cdot y_{i,p}^{l-1}+\theta_j^l,$$

$$y_{j,p}^{l}=f(s_{j,p}^{l}),\, j=1,\dots,N_l, l=1,\dots,L,$$

$$y_{j,p}^{0}=x_{j,p},\, j=1,\dots,N_0,$$

where $f(\cdot)$ - given nonlinear activation function. As activation function logistic or hyperbolic tangent functions can be used:

$$f_{\log}(z)=\frac{1}{1+e^{-\alpha z}}, \qquad f_{\tan}(z)=\frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}.$$

Their derivation can be calculated by function value:

$$\frac{\mathrm{df}_{\log}(z)}{\mathrm{dz}}=\alpha\cdot f_{\log}(z)\cdot(1-f_{\log}(z)) \quad ,$$

$$\frac{\mathrm{df}_{\tan}(z)}{\mathrm{dz}}=1-f_{\tan}^{2}(z) \quad .$$

Let, the training set of $\{x_p,d_p\}, p=1..P$ pairs are given, where $d_p=(d_{1,p},\dots,d_{N_L,p})$ – desired output for x_p input signal. The training of MANN consists in finding such w_{ij}^{ℓ} and θ_j^{ℓ} $i\in I_{\text{lj}}^{-}, j=1,\dots,N_l, l=1,\dots,L$, herewith on x_p input signal that MANN has output y_p , which maximal closed to desired output d_p . Usually, training quality is defined by mean square error function:

$$E(w,\theta;x,s,y)=\frac{1}{P}\sum_{p=1}^{P}\eta_p E_p(w,\theta;x_p,s_p,y_p),$$

$$E_p(w,\theta;x_p,s_p,y_p)=\frac{1}{2}\sum_{j=1}^{N_L}\left(y_{j,p}^{L}-d_{j,p}\right)^2,$$

where η_p – coefficient, which determine the belonging “quality” of input x_p to its “ideal” pattern $p=1,\dots,P, j=1,\dots,N_L$.

The task of MANN training constitutes minimization of criterion (4.4) according to parameters (w,θ) with (4.1)-(4.3) conditions. The MANN of developed system was trained by conjugate gradient method.

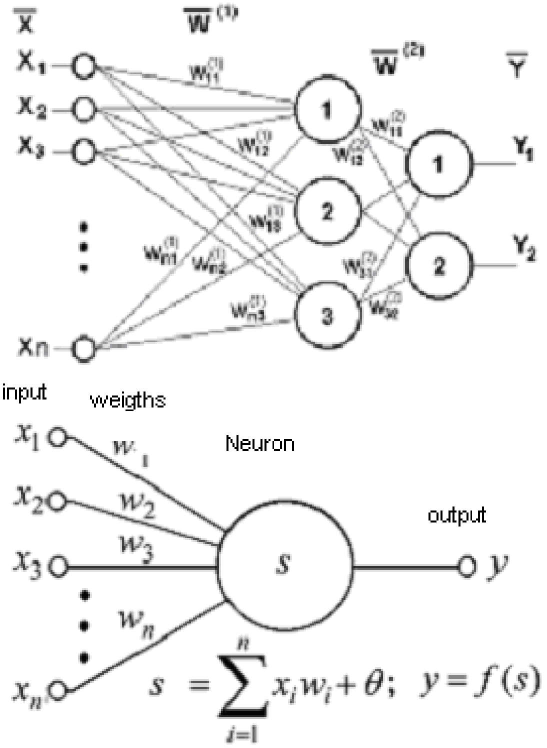


Figure 4. 3 MANN with two layers

B. *Timeline of Grantt Chart*

We have started our work on project with research about the technologies that are needed to use for the development of software product. All frameworks and programming languages have been reviewed and the most optimal ones have been chosen. There were lots of algorithms useful for implementation of Speech Recognition. Researching special algorithms and analyzing them were the most time consuming processes at initial steps. We decided to use VAD, FFT, MFCC, LPC, Lagrange algorithms to use in implementation of Speech Recognition. As constructing Neuron Network is the most essential part in Speech Recognition, it has been developed firstly with the help of provided algorithms. After research process, developing program code was our next step, which we realized in Visual Studio in programming language C#. However, we cannot say that research was done already, we still saw a need for research, and our team was researching and studying technologies continuously throughout the whole project. At the same time, since we did not have long time, and as training the system was significant for our project, together with development processes, we started collecting, analyzing and sorting data that would be useful for us to teach newly developed system to realize recognition. We decided not to wait until last minute to have experiments on our product, so we were testing program whenever there was a modification. However, we also had final experimentation in which we achieved acceptable results in our work. After a few experiments, as the initial prototype was ready, we started optimization processes. We realized the optimization of implemented algorithms, the optimization of Object Oriented Programming in code, and the optimization of User Interface. After completing all of these steps, we started integration of open source APIs, the process that is still in progress and is considered one of the main plans to be completed in short term.

Speech Recognition in Flight Simulation	**Start**	**End**
Research Research and Study Algorithms	September 2017	March 2018
Development of Software Product Implementation of VAD, FFT, MFCC, LPC, Neuron Network, and etc.	November 2017	February 2018
Data Analysis Data collecting, Data analyzing, Data sorting	November 2017	February 2018
Optimization Optimization of algorithms, Optimization of OOP Optimization of UI	February 2018	March 2018
Integration of Open source APIs Integration of Google Cloud Speech API, Microssoft Bing Speech API, and Windows Media Speech Recognition API	March 2018	Ongoing

V. CONCLUSION

A. *Discussion of Results*

While developing speech recognition applications, following logical basics should be considered: What kind of **hardware and software limitations** will occur for the usage of application? Which principle will be used for speech recognition, isolated or continuous speech recognition? Is speaker-based recognition more suitable, or is speaker-independent recognition is needed with limitation of decreasing accuracy? Will the system be able to process required information in given time? **Security** – Will system be able to be a part of equipment that supports security? How much will mistakenly recognized phrased will affect to the security of flight? Is there any equipment, if yes, which alternative management applications will be used when a problem occurs in engine?

Training the system: The main idea is training in an environment which will be used. Engine works highly accurately in laboratory environment, however, there is a possibility that it will not be useful in board environment. The main concept behind this point of view is systems trained in ideal framework, its accuracy level may decrease when it is out of that ideal framework(having bustle in board and etc.). That’s why recognition training should be done in a framework which is mostly like real environment where NT will be used – meaning the environment of usage or another similar framework.

Do not try to apply speech recognition where it is not to possible to apply it: Do not rush to apply speech recognition where its application is impossible. It brings to important risks and high responsibility. Application of the speech recognition systems in the fields such like changing radio converter, navigation functions, FMS functions, changing display mode is much more suitable. Situations which risks human life or flight security are bad kind of usage sectors.

Incorporate error correction mechanisms: There are some repeater systems which responds to pilot with recognition results using voice or display, and the pilot may accept or reject the recognition result. Whenever recognizer finds suspicious parts in information that it accepts and processes, it is able to ask pilot to repeat the same command one more time, and reminds the pilot that result is not valid printing that result on display.

B. *Future Work*

Considering the need for the concept of Speech Recognition in Flight Simulation, the software product is expected to be expanded and developed in future. Optimization of algorithms, and designing a better User Interface are the main priorities that are expected to be done in future. Additionally, integration of additional open source SDKs is in progress. Google Cloud Speech API, Microssoft Bing Speech API, and Windows Media Speech Recognition API are the planned integrations for project.

VI. REFERENCES

[1] Weasson, Pearson, R., Gary. (2006). Voice-Activated Cockpit for General Aviation. *Final Report for SBIR Contract # DTRT57- 06-C-1 0009..*

[2] Cordero, Dorado, Miguel de Pablo (2012). “Automated Speech Recognition in ATC Environment”. ATACCS’2012 | RESEARCH PAPERS.

[3] Bell, G., Schultz, M. C., & Schultz, J. T. (2000). “Voice Recognition in Fighter Aircraft.Journal of Aviation/Aerospace Education &Research, 10(1)”.

[4] Karlsson (1990). “The Integration of Automated Speech Recognition into the Air Traffic Control System”. Flight Transportation LaboratoryDepartment of Aeronautics and Astronautics,M.I.T. Cambridge, Massachusetts.

[5] K.R.Ayda-zade, S.S.Rustamov. Research of Cepstral Coefficients for Azerbaijan speech recognition system. Transactions of Azerbaijan National Academy of sciences.”Informatics and control problems”. Volume XXV, №3. Baku, 2005, p.89-94.

[6] Bengt Mandersson. Chapter 4. Signal Modeling. Department of Electroscience. Lund University. August 2005.

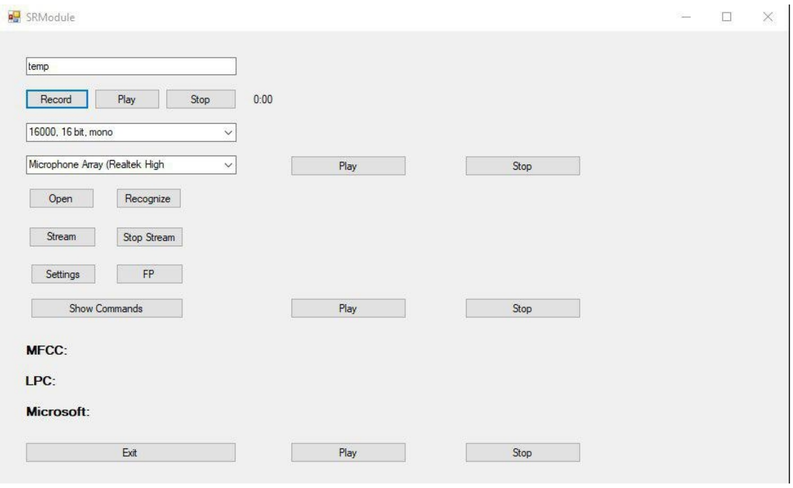
[7] Group 622 On Speaker Verification” 2004. 198 p.

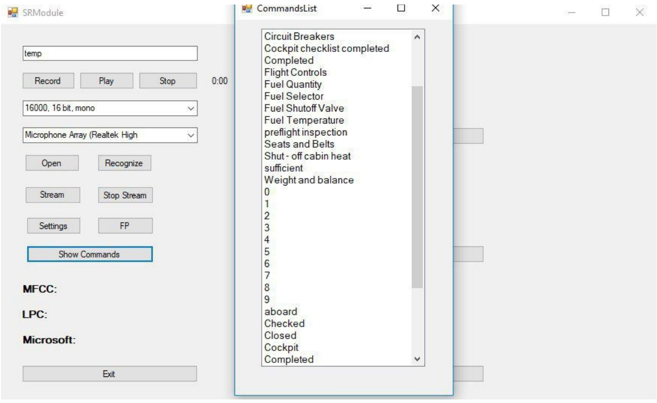
[8]Yang, T. (2012). The Algorithms of Speech Recognition, Programming and Simulating in MATLAB.

[9] Lokhande, Nehe, Vikhe (2011). Voice Activity Detection Algorithm for Speech Recognition Applications. International Conference in Computational Intelligence (ICCIA) 2011 Proceedings published in International Journal of Computer Applications® (IJCA

VII. APPENDICES

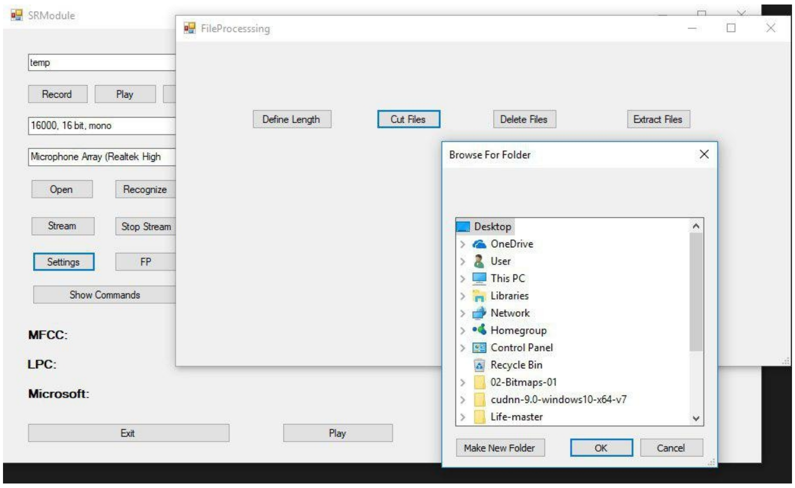
A. Screen shots of the software interface

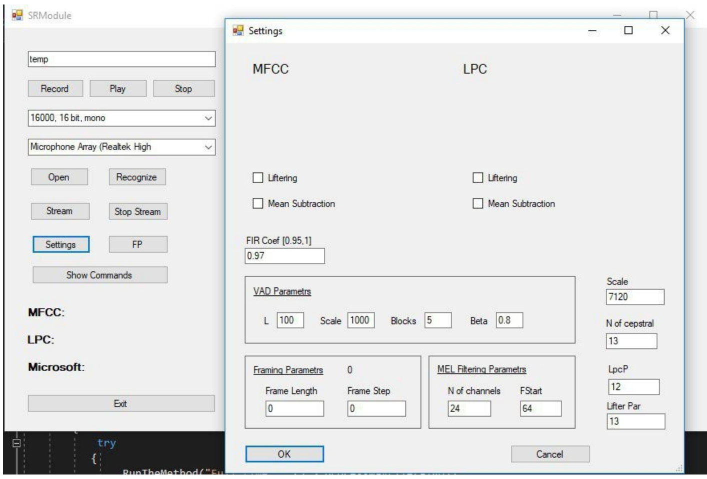




CommandsList

Circuit Breakers
Cockpit checklist completed
Completed
Flight Controls
Fuel Quantity
Fuel Selector
Fuel Shutoff Valve
Fuel Temperature
preflight inspection
Seats and Belts
Shut - off cabin heat
sufficient
Weight and balance
0
1
2
3
4
5
6
7
8
9
aboard
Checked
Closed
Cockpit
Completed





Settings

MFCC

LPC

☐ Liftering ☐ Liftering

☐ Mean Subtraction ☐ Mean Subtraction

FIR Coef [0.95,1] 0.97

VAD Parametrs
L 100 Scale 1000 Blocks 5 Beta 0.8

Scale 7120

N of cepstral 13

Framing Parametrs 0
Frame Length 0 Frame Step 0

MEL Filtering Parametrs
N of channels 24 FStart 64

LpcP 12

Lifter Par 13

OK Cancel

B. Program Codes

- **VAD Algorithm Implementation in C#**

```
public void preProcessing()
    {
        PPsamples = new
double[originalWavSamples.Length];
        PPsamples[0] = originalWavSamples[0];
        for (int i = 1; i < originalWavSamples.Length; i++)
        {
            PPsamples[i] = originalWavSamples[i] - coefFIR
* originalWavSamples[i - 1];
        }
    }

    // Function to determine the energy of block of
samples
    public double energyEstimate(int m, int L)
    {
        double Es = 0; ;
        for (int i = m - L; i < m; i++)
        {
            Es += PPsamples[i] * PPsamples[i];
        }
        return Es;
    }

    // Function to determine the power of block of samples
    public double powerEstimate(int m, int L)
    {
        double Ps = energyEstimate(m, L) / L;
        return Ps;
    }

    // Function the determine how many times graph of
our wave crossed the zero
    public double zeroCrossingRate(int m, int L)
    {
        double Zs = 0;
        for (int i = m - L + 1; i < m; i++)
        {
        Zs += Math.Abs(sgn(PPsamples[i]) -
sgn(PPsamples[i - 1])) / 2;
        }

        return (Zs / L);
    }

    // Sign function
    public int sgn(double n)
    {
        if (n>= 0) return 1;
        else return -1;
    }

    // Function to determine WS
    public double findWs(int m, int L, int Sc)
    {
        double Ps = powerEstimate(m, L);
        double Zs = zeroCrossingRate(m, L);
        double Ws = Ps * (1 - Zs) * Sc;
       // Console.WriteLine("[" + m + "]"+ "Ps - " + Ps +
" Zs - " + Zs + " Ws = " + Ws);
        return Ws;
    }

    //Function to determine Tw
    public double findTw(int L, int Sc, int blocks, double
beta)
    {
        int m = (PPsamples.Length / L) * L;
       // Console.WriteLine("length = " +
PPsamples.Length + " M = " + m);
        double mv = mean(m, L, Sc, blocks);
        double bv = variance(m, L, Sc, blocks, mv);
        double alfa = 0.2 * Math.Exp(-beta*Math.Log(bv));
// -0.8 variable
       //  Console.WriteLine("bw" + bv);
       //  Console.WriteLine("mv" + mv);
        return mv + bv * alfa;
    }

    //Calculating mean Ws for the given number of blocks
    public double mean(int m, int L, int Sc, int blocks)
    {
        double mean = 0;
        for (int i = m - L * (blocks - 1); i <= m; i += L)
        {
            mean += findWs(i, L, Sc);
          //  Console.WriteLine("print="+ mean);
        }
        return mean / blocks;
    }

    // Calculating variance for the givent number of
blocks
    public double variance(int m, int L, int Sc, int blocks,
double mv)
    {
        double variance = 0;
        for (int i = m - L * (blocks - 1); i <= m; i += L)
        {
            variance += (findWs(i, L, Sc) - mv) * (findWs(i,
L, Sc) - mv);
        }
        return variance / (blocks - 1);
    }

    // Function to determine VAD
    public Boolean VAD(int m, int L, int Sc, double Tw )
    {
       double Ws = findWs(m, L, Sc);
       //  Console.WriteLine("Ws[" + m + "]" + Ws);
       if (Ws >= Tw) return true;
       else return false;
    }
```

- **FFT Algorithm Implementation in C#**

```
public void FastFourier()
   {
     bin = new double[frameMatrix.GetLength(0),
frameMatrix.GetLength(1)];
        for (int j = 0; j < frameMatrix.GetLength(1); j++)
        {
           complex[] a = new
complex[hammingFrameMatrix.GetLength(0)];
           for (int n = 0; n < frameMatrix.GetLength(0); n+
+)
           {
              a[n] = new complex(hammingFrameMatrix[n,
j], 0);
           }
           int cnt = 1;
           while (cnt < frameMatrix.GetLength(0)) cnt «= 1;
           FFT(a, cnt);
           for (int n = 0; n < frameMatrix.GetLength(0); n+
+)
           {
              var b = a[n];
              bin[n, j] = Math.Sqrt(b.real * b.real + b.image
* b.image);
           }
        }
   }

void FFT(complex[] a, int n, bool inv = false)
     {
        int i;
        if (n == 1) return;

        complex[] a0 = new complex[n / 2];
        complex[] a1 = new complex[n / 2];
        double angle = 2 * Math.PI / n * (inv ? -1 : 1);
     complex temp;
     complex w = new complex(1, 0);
     complex e = new complex(Math.Cos(angle),
Math.Sin(angle));

        for (i = 0; i < n / 2; i++) {
           a0[i] = a[2 * i];
           a1[i] = a[2 * i + 1];
        }

        FFT(a0, n » 1, inv);
        FFT(a1, n » 1, inv);

        for (i = 0; i < n / 2; i++)
        {
           temp = Multiply(w,a1[i]);
           a[i] = Add(a0[i],temp);
           a[i + n / 2] = Minus(a0[i],temp);
           if (inv)
           {
              a[i] = Divide(a[i],2);
              a[i + n / 2] = Divide(a[i + n / 2], 2);
           }
           w = Multiply(w,e);
        }
     }
```

- **Implementation of Hamming in C#**

```
public void hamming()
     {
        hammingFrameMatrix = new
double[frameMatrix.GetLength(0),
frameMatrix.GetLength(1)];

        for (int i = 0; i < frameMatrix.GetLength(0); i++)
        {
           for (int j = 0; j < frameMatrix.GetLength(1); j++)
           {
              hammingFrameMatrix[i, j] = frameMatrix[i, j]
* (double)(.54 - 0.46 * Math.Cos(2.0 * Math.PI * (i - 1) /
                 frameMatrix.GetLength(0)));

           }
        }
     }
```

- **Implementation of MFCC in C#**

```
public double[,] MFCC()
     {
        //MainWindow.RunTheMethod("fastFourier", ()=>
fastFourier());
        RunTheMethod("fourier", ()=>FastFourier());
        RunTheMethod("melFiltering",
()=>melFiltering());
        RunTheMethod("cepstarProcessing",
()=>cepstarProcessing());
        RunTheMethod("addingEnergy",
()=>addingEnergy());
        double[,] feature;
        feature = cepstralMFCC;
        if (liftMFCC || msMFCC) {
           if (liftMFCC)  {
              if (msMFCC) feature =
cepstralMeanSubstraction(liftering(cepstralMFCC));
              else feature = liftering(cepstralMFCC);
           }
           else feature =
cepstralMeanSubstraction(cepstralMFCC);
```

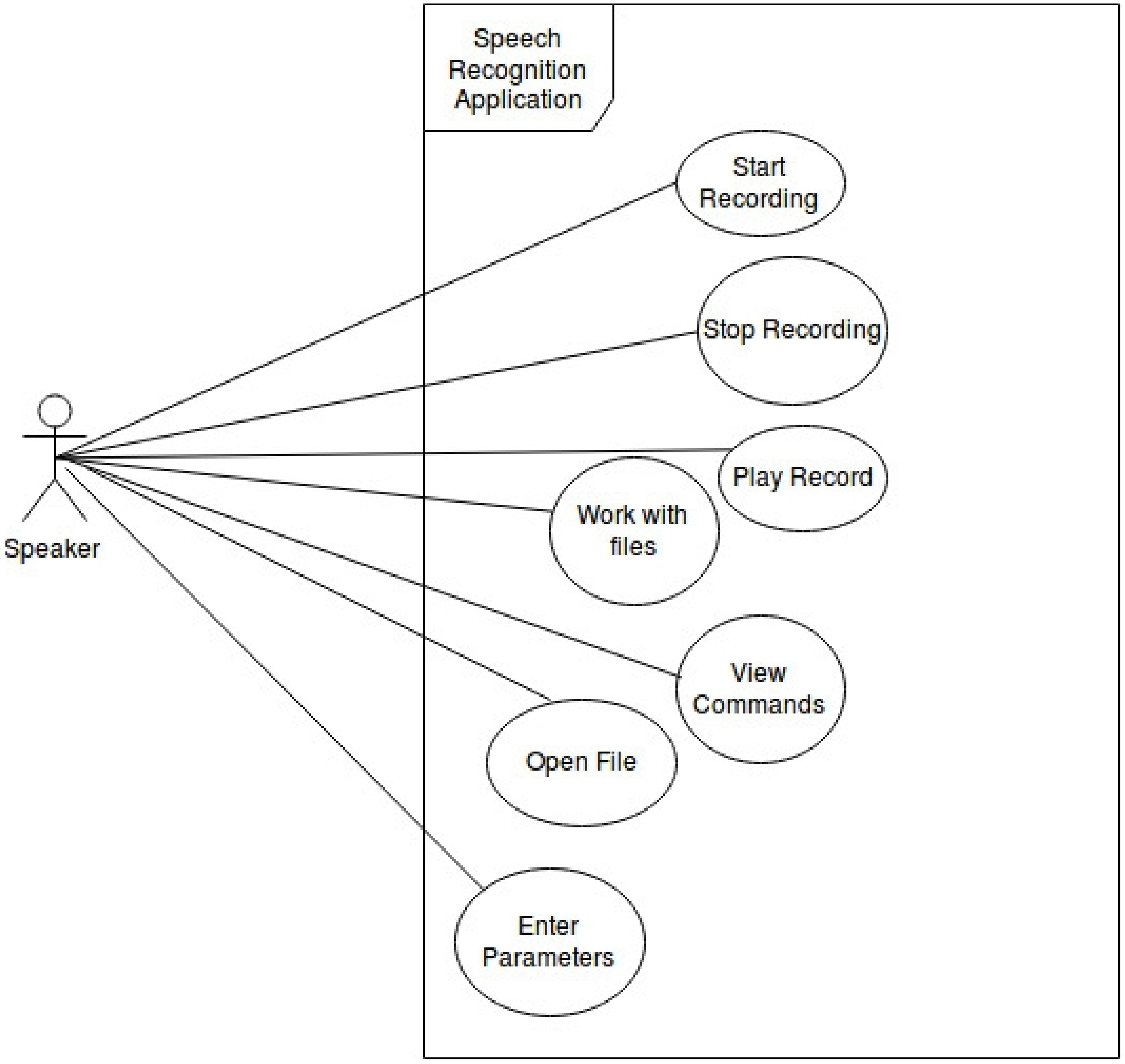
```
          }
          return feature;
      }
```

- **Implementation of LPC Algorithm in C#**

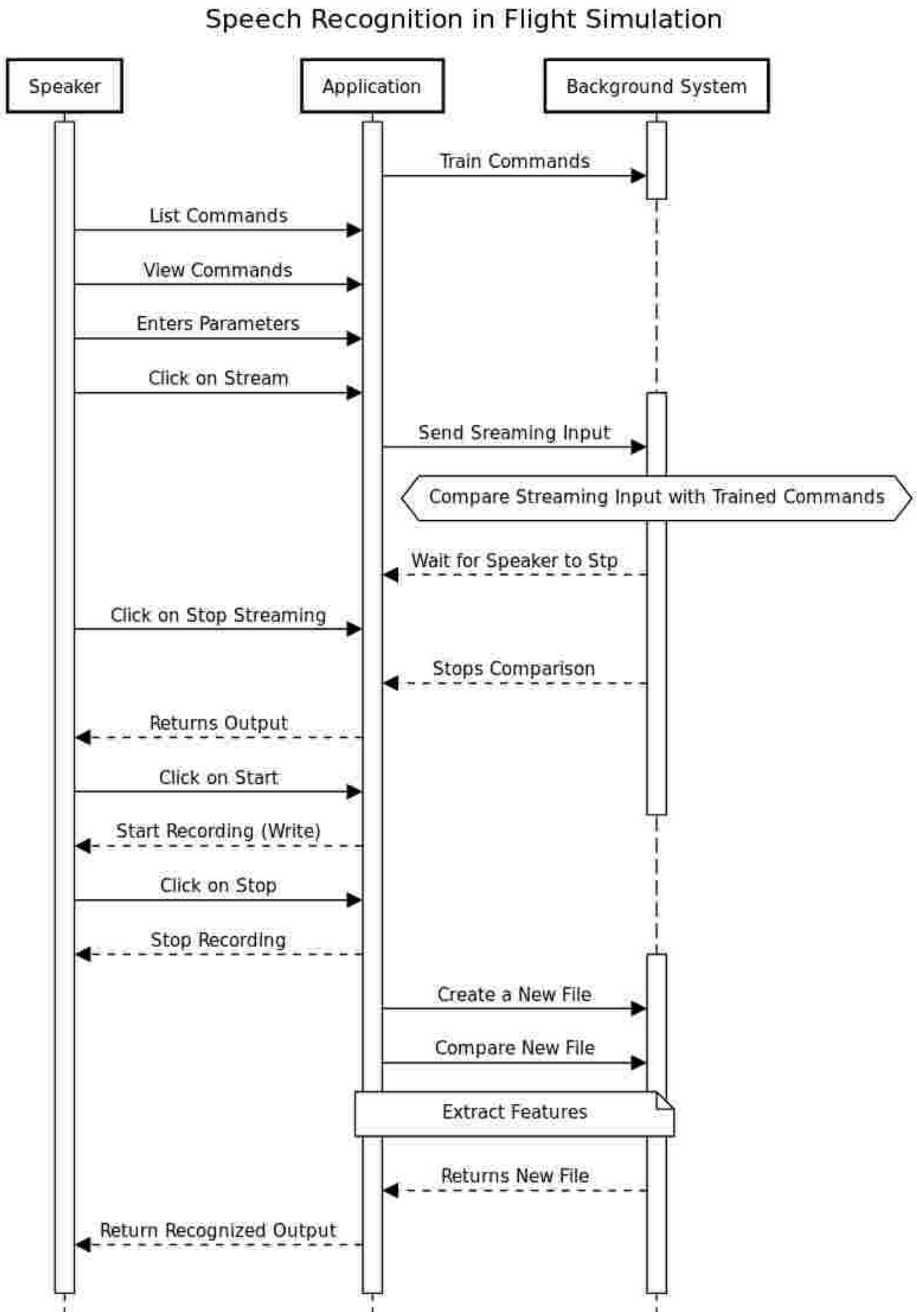
```
public double[,] LPC()
      {
          RunTheMethod("autoCorrelation",
()=>autoCorrelation());
          RunTheMethod("levinson", ()=>levinson());
          RunTheMethod("cepstralLpc", ()=>cepstralLpc());
          double[,] feature;
          feature = cepstralLPC;
          if (liftLPC || msLPC)
          {
             if (liftLPC)
      {
                if (msLPC) feature =
cepstralMeanSubstraction(liftering(cepstralLPC));
                else feature = liftering(cepstralLPC);
             }
             else feature =
cepstralMeanSubstraction(cepstralLPC);
          }
        //  Console.Write(liftLPC + " " + msLPC);
          return feature;
      }
```

C. Diagrams

1) Use-Case Diagram



2) Sequence Diagram


Speech Recognition in Flight Simulation
Speaker
Application
Background System
Train Commands
List Commands
View Commands
Enters Parameters
Click on Stream
Send Sreaming Input
Compare Streaming Input with Trained Commands
Wait for Speaker to Stp
Click on Stop Streaming
Stops Comparison
Returns Output
Click on Start
Start Recording (Write)
Click on Stop
Stop Recording
Create a New File
Compare New File
Extract Features
Returns New File
Return Recognized Output