Islamic University of Gaza
Deanery of Higher Studies
Faculty of Information Technology
Information Technology Program

الجامعة الإسلامية –غزة
عمادة الدراسات العليا
كلية تكنولوجيا المعلومات
برنامج تكنولوجيا المعلومات

Detecting Inactivity Segments in E-learning Video Recordings (DISE)

Prepared by

Sahar S. Hammo
220100065

Supervised by

Dr. Ashraf Alattar

A Thesis Submitted as Partial Fulfillment of the Requirements for the

Degree of Master in Information Technology

July

2015

سورة النحل – الآية ٥٣

ABSTRACT

Lecture video recordings used in e-learning typically contain segments of inactivity which are better removed in order to save storage and bandwidth load, and to decrease viewing time and effort. This problem is also common in other areas such as multimedia.

This research proposes a method for detecting segments of inactivity in video sequences and marking them for further processing. The method is known as DISE.

The core of our approach are two modules one for sound inactivity detection and the other for video inactivity detection. Inactivity segments detected by the sound module are forwarded to the video module for video inactivity processing.

The method has been developed into a system (DISE) and tested using a collection of real lecture videos obtained from lecture recordings at University of Palestine for different courses under various recording conditions. The method performs close to human detection as it achieves accuracy of 84.4%. We believe that our method will add value to the process of recording lectures in the e-learning domain in educational institutes .

Keywords: *E-learning, video inactivity detection , video sequences.*

تحديد المقاطع غير النشطة في فيديوهات المحاضرات في التعليم الالكتروني

يعتبر نظام التعليم الالكتروني طريقة فعالة لإثراء عملية التعليم والتدريس، حيث أن الإقبال على استخدام الفيديو كوسيلة تعليمية قد ازداد في الآونة الأخيرة لما له من دور في تسهيل مهمة التعليم.

تسجيلات الفيديو قد تحتوي على مقاطع غير نشطة، من الأفضل إزالتها لتوفير الوقت والجهد ومساحة التخزين، يظهر ذلك جلياً في تسجيلات الفيديو الخاصة بالمحاضرات بالإضافة إلى ظهورها في مجالات أخرى.

يشمل هذا البحث على عرض وتصميم وإنشاء نظام يقوم بتحديد هذه المقاطع غير النشطة في تسجيلات الفيديو ومن ثم عرضها للمستخدم لاتخاذ القرار بشأن إزالتها أو الإبقاء عليها حسب وجهة نظره، مع الحفاظ على استمرارية الفيديو وجودة الصوت والصورة .

يتكون هذا النظام من جزئيتين جوهريتين، الأولى هي تحديد المقاطع الصوتية الصامتة، والثانية تحديد المقاطع المرئية غير النشطة، المقاطع التي تنتج عن الجزئية الأولى يتم تمريرها للجزئية الثانية لمعالجتها والخروج بالنتيجة النهائية .

تم تجربة هذا النظام على فيديوهات حقيقة لمحاضرات تم تسجيلها في جامعة فلسطين لمواد مختلفة وتحت ظروف محيطة مختلفة .

في عملية تقييم النظام، توصلنا أن نسبة دقة النظام هي ٨٤.٤ %، مما يعني تقارب بين نتيجة النظام والنتيجة الصادرة عن الاستخدام البشري للنظام.

الكلمات المفتاحية : التعليم الالكتروني، تسلسل مقاطع الفيديو، تحديد المقاطع غير النشطة.

Dedication

To My great MOM..

To My DAD..

To The Memory Of My Father-In-Law

To My Soul Mate ,, IYAD ..

To My Heroes ,, YACOUB N' MOHAMMED

To My Beautiful Princess ,, HIAM ..

To My Sisters and Brother ..

Acknowledgment

Thanks for Allah for all the blessings that he gave me , and to give me the strength and courage to complete this thesis and research.

Gratitude to my supervisor Dr. Ashraf Al Attar, for providing me the opportunity to develop this work over the master years.

Thanks for MOM n' DAD for encouraging me ,

Thanks for my HUSBAND for supporting me , and for being patient and understanding all the time ,

Thanks for my children for giving happiness to my life ,

I am also grateful to my friends and family , for their care , I deeply appreciate it. I would like to extend sincere gratitude and appreciation to the people who helped me to accomplish this thesis.

I would like to thank all my academic teachers, staff members and colleagues for their support throughout my study in Islamic university .

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND GLOSSARIES

DISE :	**Detecting Inactivity Segments in ELearning videos**
E-learning :	**Electronic learning**
JDK :	**Java Development Kit**
P :	**Precision**
R :	**Recall**
TP :	**True Positive**
TN:	**True Negative**
FP:	**False Positive**
FN:	**False Negative**
R_b:	**Bit Rate**

LIST OF APPENDICES

CHAPTER 1
INTRODUCTION

This chapter introduces the thesis by stating the underlying concepts of e-learning, lectures recorded videos and the requirements for detecting and deleting the idle segments in those videos, also it talks about the thesis problem, the research objectives, the research importance, the research scope and limitations, as well as the research methodology.

1.1 Overview for E-Learning

We live in a world that is constantly changing. The presence of computers has revolutionized the world. Computers have brought in a host of new technologies for education. Learning has changed as well; starting from the formal classroom learning in schools to distance education, the process of learning in education has come a long way.

E-Learning is the use of technology to enable people to learn anytime and anywhere. E-learning (or eLearning) refers to the use of electronic media and information and communication technologies in education. E-learning is broadly inclusive of all forms of educational technology in learning and teaching. E-learning includes many types of media that deliver text, audio, images, animation, and streaming video, and includes technology applications and processes such as audio or video tape, satellite TV, CD-ROM, and computer-based learning, as well as local intranet/extranet and web-based learning[1].

1.2 Lectures recorded videos

In recent years, lecture capture has gone from a helpful learning aid to an essential utility that more and more universities are using as part of the learning experience. Recent studies have demonstrated that recorded lectures improve student engagement and

achievement by providing individual control over the pace of learning and the ability to review complex topics after class.

The value of using videos as a teaching tool in E-learning has increased due to its ability to facilitate teaching and learning. Lecture recording refers to the process of recording and archiving the content of a lecture, conference, or seminar. [2] It consists of hardware and software components that work in synergy to record audio and visual components of the lecture.

Sometimes, the lecturer may use visual aids to support their speech, such as slide shows, which are presented to the public with some kind of projector; in this case such slideshows can also be recorded. Once captured, the data is then either stored directly on the capture hardware or sent to a server over a LAN or the Internet. After some processing to adapt the video formats to the desired distribution mechanism, viewers whose are almost students are then able to remotely access the recording, either in real time or ex post facto.

Those videos, and because the relatively long period time of the lectures that may reach to two or three hours per lecture, will suffer from large sizes which annoys the users when uploading , downloading or watching them Also they may contain sections of idleness doesn't contain any lecture activity such as lecturer lecturing or discussions, but they increase the total time of the lecture and fake the actual lecture time given by the lecturer, which may influence the evaluation of the performance of the lecturer in the field of quality in universities and this certainly leads to unnecessary increase in the total size of the video.

In this work, a complete, easy to use and effective end user system is developed to reduce the large size for those videos and detecting the actual period of time of the lectures. The user of the system inputs the video and some configuration values, and the system outputs the video without any idleness sections, where the trimmed sections are outputted in an external file for archiving or later viewing.

1.3 Problem Statement

Although video recordings of lectures can serve as an important tool for facilitating teaching and learning in higher educational institutes, they suffer a number of challenges. One set of challenges stem from their typical large sizes which result in consuming scarce storage space and network bandwidth when these files are uploaded or downloaded. Other challenges are related to the fact that these lecture files typically contain sparse segments of inactivity (idle segments). These idle segments result in significant waste of time for students when viewing these videos, and raise questions about how to fairly assess and manage lecturer performance.

Therefore, the problem can be stated as: how to identify and remove segments of inactivity in lecture videos to overcome the above mentioned challenges.

This main problem can be divided into the following set of **sub problems**:

1) How to correctly detect idle segments (determining features/characteristics of idle segments)?
2) Examine possible pre-processing steps necessary before the detection step, and choose the most efficient.
3) How to use the chosen crop method to delete the inactivity in video sequences?
4) What is the most appropriate programming language to use for implementation phase?
5) What are the best evaluation methods to assess the performance of the proposed solution?

1.4 Objectives

The objectives of this research are expressed in a main objective and a set of secondary objectives.

1.4.1 Main Objective

The main objective of this research is to develop an efficient method for detecting inactivity in video sequences and delete them. The method should be implemented within a complete end-user solution. This method will be known as DISE.

1.4.2 Secondary objectives

The main objective is achieved by fulfilling the following sequence of secondary objectives:

1) Collect an adequate set of lecture videos to identify one or more features that characterize idle segments.
2) Develop a detection module to mark the beginning and end of idle segments.
3) Test the module and evaluate its performance (based on manual evaluation measure).
4) Package the module as a end-user application.

1.5 Significance of the thesis

1) Saving time and effort for the users of the system when viewing, downloading or uploading videos.
2) Improving the measurement of the actual time of the lecture given by the teacher in order to evaluate the teacher performance in the field of quality control at universities.
3) Reducing the size of the storage space of the eLearning video.
4) Encouraging the using of videos in the e-Learning systems.
5) Reducing the noise in the videos that results from the surrounding environment or the recording media.

1.6 Scope and limitation

This work is expected to be developed under some constraints and limitations such as:

1) The video can be in any type such as mp4 , wmv or mpeg.
2) The video must be at least at the medium quality, to be able to distinguish between voice and noise.
3) The surrounding environment must not contain significant noise, and this can be satisfied when recording a video in a lecture room, located within a building with easy access by students and equipment, isolated from noisy gathering places, and concentrated on the lower floors of buildings to provide an easy avenue for students.

1.7 Research Methodology

This research follows a methodology that consists of four stages as follows:

1) **Exploring the theoretical foundation:** A deep reading, understanding and analyses of the pedagogical theories and the related works which were done in this field will be performed in order to form a clear orientation of the theoretical framework for detecting inactivity segments in E-learning video recordings system.
2) **Designing the method functions:** The features of the inactivity segments are defined, and then the process of deleting those segments is designed based on the proposed algorithms.
3) **Implementing the method:** The method is implemented using java language.
4) **Evaluating the method:** The detecting inactivity segments system will be evaluated in order to ensure its accuracy and precision. Those characteristics are measured manually by viewing the video before and after cropping, this will be done by the help of a specialist in multimedia and e-learning fields .

1.8 Thesis format

This thesis has been divided into six major chapters, which are structured around the objectives of the research. The dissertation is organized as follows:

Chapter 2, Presents Literature Review of lectures recorded videos and popular techniques used for reducing their large size, inactivity segments features and detection approaches. Also, this chapter presents details about voice , motion detection techniques and noise reduction techniques.

Chapter 3, Presents some related work of motion detection techniques, voice detection techniques, and highlights its main shortages which are to be avoided and solved in our work.

Chapter 4, Includes the methodology steps and the architecture of the method. An explanation about the data sets used in the experiments, preprocessing of these data set, and the experiment cases is included as well.

Chapter 5, Give the details about the sets of experiments, and analyze the experimental results and the evaluations.

Chapter 6, Will draw the conclusion and summarize the research achievement of experiments and suggests future work.

CHAPTER 2
LITERATURE REVIEW

In this chapter, we will review the usage of videos in education and popular techniques for reducing their large size. Then we will describe the inactivity segments features and detection approaches from two views, voice and motion. Finally we will discuss noise definition and noise reduction techniques.

2.1 Use of video in education

Researches have shown that the levels of attention of students in a traditional lecture will tend to decline rapidly after 20 minutes. Hence, students who attend lectures may have difficulty in focusing on a 1-hour or 2-hour lectures in the University. Foreign students, who may not be conversant in the mother language , often find it even more difficult to focus during lectures as they have to comprehend the content and language at the same time.

The video provides a face with expressions, gestures and a human voice to what is usually 'faceless' online content, which according to the social-cue hypothesis stimulates students interest and communication, and therefore influences learning in a positive manner . A system for video recorded lectures allows learners to access any segment of the lectures easily through a table of content. Learners can easily click onto any particular topic under this table of content to view the segment. Learners are also able to re-play, start or pause the video streaming [3].

Recorded videos in e-learning have a number of benefits and challenges:

Common benefits include:

1) Replay sequences.
2) Drawing attention to details some students may have missed, or where the interpretation difference among students.
3) Providing recordings of lectures and other teaching materials in audio or video format can help all students consolidate their learning, and is of particular benefit to

students whose first language is not the speaker language or who have difficulties taking notes.

4) It can be used as a measurement for the actual time of the lecture given by the lecturer in order to evaluate the teacher performance in the field of quality control at universities.

Common challenges include:

1) Copyright considerations.
2) Open access to a previously private activity.
3) Videos with large size use more bandwidth and take longer to be uploaded or downloaded.

2.2 Techniques for reducing videos size

Concentrating on the last point in the challenges in the previous section, there are many techniques can be used to reduce the large video sizes. We can mention from them:

1- Reduce video resolution, frame rate and bit rate

The video resolution is associated with video quality and video size. The higher resolutions mean that more pixels are used to display video, resulting in the larger video file size. We can change the resolution of the original video file to a smaller resolution. Generally, HD videos can be as high as 1920X1080P, but you can reduce to 1280X720, 640×480, 640×360, and more resolutions as shown in figure 2.1.

Figure 2.1 : video resolutions

If the video size is not small enough, you could also lower frame rate and bit rate. The frame rate has a lot to do with the file size, which displayed images of a video per second.

Lowering the frame rate will directly lead to the decrease of video file size. With regard to the bit rate, it mainly describes video or audio quality. If the video file is compressed at higher bit rates, the higher video quality and larger file size will be, and vice versa.[4].

2- Change video format

This way will focus on how to convert video files to other formats with smaller size. The frequently used formats include AVI, WMV, MP4, MOV, RMVB, 3GP, MKV etc. The best quality format for HD video should be MP4, but it has maximum video resolution and frame rate to upload to. AVI is a multimedia container format, which is very flexible. But the downside is that it is too large to compress. Amongst all accepted formats, WMV and FLV provide relatively small size. WMV is a good choice for video email and storage on your hard disk while FLV is a suitable format for web video[4].

3- Video compression

Video compression uses modern coding techniques to reduce redundancy in video data. Most video compression algorithms and codecs combine spatial image compression and temporal motion compensation. Video compression is a practical implementation of source coding in information theory. In practice, most video codecs also use audio compression techniques in parallel to compress the separate, but combined data streams as one package.[29]

But all those techniques mentioned earlier, works on one idea, how to decrease the size of the video, no matter the actual duration of the video, and this doesn't fit our main objective where we want these videos to be a measurement for the actual time of the lecture given by the lecturer in order to evaluate the teacher performance in the field of quality control at universities. So another technique must be used in for fulfilling this objective, which is to get rid of idle video parts.

4- Trim unwanted video parts

The professional video editor enables you to trim unwanted video clips from your big video file. Through video editing, you can change the video file into several parts and cut off unnecessary clips. After saving, the video size will be compressed. Also, you can save away each spliced video clip. This way can make it easy to get decreased video file. Many applications are available for this purpose such as windows video maker.

In this way you need to be aware or to know what parts exactly you want to delete / trim, this means that you must watch the entire video, then to detect which parts to delete depending on your watching, but actually this will not work effectively if your video is too long or if you have too many videos to deal with such as daily lectures videos in a university, so there must be a technique or a system to cut unwanted -idle- parts automatically without the need to view the entire video .

In this research, an efficient video processing system is developed to detect and delete those unwanted parts automatically from videos of recorded lectures in the field of e-learning, and make sure that this will not affect the continuance and the quality of the video or audio. Those unwanted parts which are useless but increased storage and bandwidth load, and viewing time and effort are called in our research "inactivity segments ".

2.3 Detecting the inactivity segments

We define lecture activity as the presence of a) motion, b) human voice. Thus, inactivity segments can be defined as a group of consecutive frames in the e-learning video recordings frames, where there is an absence of motion and human voice for a specified period. Total absence is not necessarily required but only to a certain extent that can be defined using some thresholds.

Detection of inactivity is a type of intelligent system processing, and therefore could classified under video analytics. Video analyze is a technology that is used to analyze video for specific events, alarms, conditions or objects. It is comprised of software algorithms to analyze the contents of video to extract information about the

content of that video, and this will free the users or the video viewers from human vigilance.

Video analytics applications that will be used in detecting inactivity in lectures recorded videos are: motion detection and voice detection. Motion detection will automatically detects motion in a field of view, and voice detection will automatically detect the presence of human voice, therefore when both motion and human voice are not detected, this situation will alert the existence of inactivity segments.

2.3.1 Motion detection

Motion detection is the process of detecting a change in position of an object relative to its surroundings or the change in the surroundings relative to an object. Motion detection can be achieved by both mechanical and electronic methods. [5]

Several methods have been proposed to detect the motion objects in an image sequence automatically. They can be classified into three major approaches. Each approach has its advantages and shortcomings.

1- Background subtraction based approach:

The fundamental process of the background subtraction based approach is to subtract the current image with a pre-selected background image. Then, the subtraction result is analyzed to find the motion objects. The major disadvantage of the background subtraction based approach is that the background may change with time lightly. The subtraction result may include some additional noise. Wren [6] tried to method the noise by using a mixture of Gaussian distributions at each pixel.

Other challenges in developing a good background subtraction algorithm are:

- It must be robust against changes in illumination.
- It should avoid detecting non-stationary background objects such as moving leaves, rain, snow, and shadows cast by moving objects.
- Its internal background method should react quickly to changes in background such as starting and stopping of moving objects [7].

2- Temporal difference approach:

To reduce the background changes, the temporal difference approach [8] detects motions by taking the absolute difference of consecutive images. Lipton et al. [9] extracted all moving objects by using a temporal difference algorithm. They applied the temporal differencing and adopted many variants on this method. The disadvantage of the temporal difference approach is that it may miss the detection of an object that stops motion in the image frame.

3- Probability based approach:

The probability approach uses the observed information to obtain a classification equation of probability to segment image. This approach suffers from the problem of high computational complexity. So it is hard to be used in real- time surveillance application.[10].

Table 2.1 shows a comparison between the three approaches[36].

Table 2.1:Comparison between the three approaches

Approach	method	characteristics	shortcomings
Background subtraction	Comparison between the images and a fixed reference background image.	Work well if the illumination of the images is constant.	Fail when there is a change in the background such as ambient lighting, weather , etc.
Probability	Used the observed information to obtain a classification equation of probability and statics problem , and used the observed information to obtain a classification equation of probability to	They are perfect in tracking moving objects.	They suffer from high computing cost and hard to be used in real time applications and videos.

	segment image.		
Temporal difference	Computes difference between two successive frames	Simple method for detecting moving objects in a static environment.	The too high or the too low speed of the moving object may be wrong detected.

According to the comparison in table 2.1 . And according to our situation where we are using videos for lectures recordings in a university where one lecturer is speaking in front of a camera in a lecture room. The background subtraction approach is not suitable to be used because any change in the background such as ambient lighting, weather and this is so popular may lead to wrong results. Also probability approach is not suitable to be used because the high computing cost. So we have used in our research, a motion detection method based on temporal differencing technique called blocks-average technique to detect motion and objects movement, This algorithm is simple , suitable for the input videos and takes less processing time and , which increases the speed and also the detection rate .

2.3.2 Voice detection

As mentioned previously, inactivity also is the absence of human voice beside the absence of motion.

Voice activity detection (VAD), also known as speech activity detection or speech detection, is a technique used in speech processing in which the presence or absence of human speech is detected.

The classification of the speech signal into voiced, unvoiced, and silence provides a preliminary acoustic segmentation of speech, which is important for speech analysis. The nature of the classification is to determine whether a speech signal is present and, if so, whether the production of speech involves the vibration of the vocal folds. [11]. Voiced speech usually has a higher amplitude than unvoiced speech, and silent regions

are distinguished by almost no amplitude, which makes the energy very suitable for a first classification.[12]. A voiced sound is one in which the vocal cords of the speaker vibrate as the sound is made, and unvoiced sound is one where the vocal cords do not vibrate. A given speech utterance could contain a mix of different voiced and unvoiced segments [13]. It should be clear that the segmentation of the waveform into well-defined regions of silence, unvoiced, signals is not exact; it is often difficult to distinguish a weak, unvoiced sound from silence, or weak voiced sound from unvoiced sounds or even silence.[14]

A Speech Classification Algorithm will be used in our system to detect human voice based on the volumes levels of the speech waveforms and recognizing different sound types based on certain threshold values. The algorithm will be discussed in details in section 4.1.3 .

2.4 Noise definition and reduction techniques

This section is divided into two subsections , the first talks about the noise in general and definition from different views . The second section discuses some popular techniques and algorithms for reducing noise as much as possible.

2.4.1 Noise definition

In audio , noise is generally any unpleasant sound and, more technically, any unwanted sound that is unintentionally added to a desired sound. Ambient sound itself is a series of changes in air pressure transmitted in waves from the sound source to a human. Sound waves are expressed as a series of analog waves. The combination and blend of these waves gives sounds their individual characteristics, making them pleasant or unpleasant to listen to. In recording sound, noise is often present on analog tape or low-fidelity digital recordings. The standard audio cassette includes a layer of hiss on every recording. When doing digital recording, the conversion of a sound file from 16-bit to 8-bit adds a layer of noise.[15].

One popular type of noise , is a white noise. White noise is a sound that contains every frequency within the range of human hearing . It has a typical frequency of 40-

80 hz. White noise is a type of noise that is produced by combining sounds of all different frequencies together. If you took all of the imaginable tones that a human can hear and combined them together, you would have white noise.

The adjective "white" is used to describe this type of noise because of the way white light works. White light is light that is made up of all of the different colors (frequencies) of light combined together (a prism or a rainbow separates white light back into its component colors). In the same way, white noise is a combination of all of the different frequencies of sound. You can think of white noise as 20,000 tones all playing at the same time. White noise can be generated on a sound synthesizer. Sound designers can use this sound, with some processing and filtering, to create a multitude of effects such as wind, surf, space whooshes, and rumbles.[30]

2.4.2 Noise reduction algorithms

There are two types of noise reduction systems:

1) Modulation detection
2) Synchrony detection

Underlying algorithms may vary between manufacturer's but overall aim is to provide improved listener comfort and possibly improved speech intelligibility in background noise. Common to all algorithms is this need to identify which signal is noise and which is speech.[31]

2.4.2.1 Modulation Detection

A speech/non speech detector analyses fluctuations in signal amplitude. Speech & noise envelopes fluctuate in a well characterised manner:

- Speech modulations tend to be slow and have big amplitude fluctuations.
- Noise modulations tend to be more constant with rapid and smaller fluctuations.

The precise way in which the hearing aid processes the two signals will differ depending on the algorithm. One common approach is to:

1) Estimate when the speech signal is present in each channel.

2) If the amplitude envelope at a channel's output is characteristic of speech the gain in that channel remains unaltered.
3) If the envelope is relatively constant then the signal is assumed to be noise and the gain for that channel is reduced.
4) If more noise is detected than speech at a certain channel, then the gain will also be reduced.[32]

Important points to note:

- If the unwanted signal has similar fluctuations to speech, then it is unlikely to be attenuated.
- Best results occur with steady state noise that has a narrow frequency band, or is of low frequency (to reduce the risk of upward spread of masking)
- If any other 'constant' amplitude signals (e.g. pure tones) are identified then they will also be treated as a 'noise' signal and be attenuated. This means that pure tones should not be used to either set or test the hearing aid, unless the noise reduction feature has been turned off.

Advantages and Limitations:

- Advantages: Channel specific gain reductions to ensure comfort whenever noise is present
- Limitations: Cannot distinguish between noise only versus speech plus noise.[33]

2.4.2.2 Synchrony Detection

On this type of detection, you can find speech versus noise but you cannot separate them. So it is designed to look for the unique structure of speech (energy). It maintains full response whenever speech is present and only goes into comfort mode when speech is no Longer present.

Advantages & Limitations

- Advantages: Can protect audibility of speech whenever speech is present
- Limitations: Anytime speech is present there will be no comfort-based changes, even if noise present and high level.[34][35].

CHAPTER 3

RELATED WORKS

The related work chapter will be divided into two sections; the first section will summarize the related works that solve the motion detection problem, the second section will summarize the related works that solve the voice detection problem.

3.1 Motion detection related works

Kausalya and Chitrakala [16] proposed a method to detect idle object and applied it for the analysis of suspicious events. Partitioning and Normalized Cross Correlation (PNCC) is used for detecting and tracking moving object in an image sequence and for indicating the presence of moving object. This system is used to detect suspicious tracking of human behavior in video surveillance for security purpose in ATM application. This algorithm takes less processing time, which increases the speed and also the detection rate. Two consecutive frames from image sequence are partitioned into four quadrants and then the Normalized Cross Correlation (NCC) is applied to each sub frame. The sub frame which has minimum value of NCC, indicates the presence of moving object. If the motion of the
moving object is constant for a particular period of time (constant in some sequence of frames) then it is idle, the idle action represents the object is doing suspicious action inside the ATM machine. Then Alarm is indicated for idle object identification. The shortcoming of this system that it can consider an object in the background as a suspicion such as a tree or a street light that is because it doesn't split between background and foreground.

Cucchiara et. al. [17] described an approach for Moving Visual Objects segmentation in an unstructured traffic environment. It considers a complex situation with moving people, vehicles, infrastructures that have different aspect method and motion method. His approach is based on background subtraction with a statistic and

knowledge-based background update is given. This approach allows a limited number of frames for the background update suitable for real time computation. He defined a specific approach called S&KB Background update based on background subtraction with a statistically adaptive and knowledge based selective background updating. The statistical adaptation copes with the changing appearance of the scene during time, while the knowledge-based selection gives a feedback to the perceptual level in order to improve separation of foreground objects from background. The shortcoming of this approach that it is weak when there is changes in illumination

Heisele et. al. [18] presented an algorithm for tracking moving objects in a sequence of colored images. In this object parts are determined by a divisive clustering algorithm, which is applied to all pixels in the first image of the sequence. For each new image the clusters of the previous frame are adapted iteratively by a parallel k-means clustering algorithm. This algorithm being complex requires more computational time. So there is a requirement to increase the speed.

Ssu-Wei Chen et. al. [19] combined a background subtraction method with a temporal difference method . They proposed a new tracking method that uses Three Temporal Difference (TTD) and the Gaussian Mixture Method(GMM) approach for object tracking. TTD method is the use of a continuous image subtraction. The GMM approach consists of three different Gaussian distributions, the average, standard deviation and weight respectively. There are two important steps to establish the background for method, and background updates which separate the foreground and background. The advantages of TTD quick calculations and the disadvantage is a lack of complete object tracking. The advantage of GMM is a complete result of the operation . But the disadvantage is not a complete object tracking, also it includes more computing for a long time with more noise. So we need here to make a choice between one of the methods to reduce computing complex.

Shujie and Fred [20] proposed a novel approach to motion detection and estimation based on visual attention. Their method used two different thresholding techniques and comparisons are made with Black's motion estimation technique based on the measurement of overall derived tracking angle. They tested their method on various

video data and the results show that the new method can extract motion information. They used visual attention (VA) methods to define the foreground and background information in a static image for scene analysis. Those regions which are largely different to most of the other parts of the image will be salient and are likely to be in the foreground. The discrimination between foreground and background can be obtained using features such as color, shape, texture, or a combination. The concept has been extended into the time domain and is applied to frames from video sequences to detect salient motion.

The shortcoming of this method is that the results are less certainty with particular emphasis on addressing noise arising from background motion and changes in illumination.

Bouthemy et. al.[21] proposed a novel probabilistic parameter-free method for detecting independently moving objects using the Helmholz principle. Optical flow fields were estimated without making assumptions on motion presence and allowed for possible illumination changes. The method imposes a requirement on the minimum size for the detected region and detection errors arise with small and low contrasted objects.

Black and Jepson [22] proposed a method for optical flow estimation based on the motion of planar regions plus local deformations. The approach used brightness information for motion interpretation by using segmented regions of piecewise smooth brightness to hypothesize planar regions in the scene. The proposed method has problems dealing with small and fast moving objects. It is also computational expensive.

Kellner and Hanning [23],proposed a method for motion detection based on a modified image subtraction approach to determine the contour point strings of moving objects. The proposed algorithm works well in real time and is stable for illumination changes. However, it is weak in areas where a contour appears in the background which corresponds to a part of the moving object in the input image. Also some of the contours in temporarily non_moving regions are neglected in memory so that small broken contours may appear.

We summarized the motion detection related works in Table 3.1.

Table 3.1:summarizing motion detection related works

Related work	Detection method and techniques	Shortcomings
Kausalya and Chitrakala [16]	Partitioning and Normalized Cross Correlation	doesn't split between background and foreground.
Cucchiara et. al. [17]	S&KB Background update based on background subtraction	weak when there is changes in illumination
Heisele et. al. [18]	divisive clustering algorithm	complex and requires more computational time
Ssu-Wei Chen et. al.[19]	Three Temporal Difference (TTD) and the Gaussian Mixture Method(GMM) approach	not a complete object tracking
Shujie et. al. [20]	visual attention	results are less certainty when there is noise
Bouthemy et. al. [21]	novel probabilistic parameter-free method	errors arise with small and low contrasted objects
Black and Jepson [22]	optical flow estimation method	- problems dealing with small and fast moving objects. - computational expensive.
Kellner and Hanning [23]	modified image subtraction approach	it is weak in areas where a contour appears in the background which corresponds to a part of the moving object in the input image.

3.2 Voice detection related works

Saha and Chakroborty [24] presented a method to detect silence/unvoiced part from the speech sample using uni-dimensional Mahalanobis algorithm. This algorithm uses statistical properties of background noise as well as physiological aspect of speech production and does not assume any adhoc threshold. The Algorithm passes over two steps : In Step 1 they use statistical property of background noise to make a sample by as voiced or silence/unvoiced. In step 2 they use physiological aspects of speech production for smoothening and reduction of probabilistic errors in statistical marking of step I. The shortcoming of this method that it makes a classification of Voiced part of a speech from silence/unvoiced part, where it doesn't make the classification between silence part and unvoiced part.

Jacobs et. al. [25] posed the silence detection problem in a multimedia communications environment. These environments are characterized by their packet based data handling and communication facilities as well as a varying degree of QoS support (i.e. bandwidth and delay).They had identified several requirements for efficient use of silence detection namely elimination of only inter sentence or (longer) silence low complexity and low decision delay. They had presented a novel algorithm for silence detection based on the small and large signal behavior of the speech waveform in the law domain. The shortcoming of this method that it makes a classification of silence part of a speech, where it doesn't make the classification between voiced part and unvoiced part.

Rabiner and Sainbur [26] presented a novel approach to the voiced-unvoiced-silence detection problem , in which a spectral characterization of each of the 3 classes of signal is obtained during a training session, and LPC distance metric and an energy distance are nonlinearly combined to make the final discrimination. The method for discriminating among these three signal classes is to use a level test to discriminate silence from speech, and then discriminate between voiced speech and unvoiced speech by a logical decision based on the values of certain measured features of the signal - e.g., energy, zero crossings, etc. When used in conjunction with pitch detection, features of the pitch detector are often used to supplement the voiced-unvoiced decision. But this algorithm is limited on specific application such as telephone lines.

Atal and Rabiner [27] proposed a statistical decision approach to voiced-unvoiced-silence classification in which a set of measured features were combined using a non-Euclidean distance metric to give a reliable decision. This method was optimized for telephone line inputs. Their results showed that reliable discrimination between voiced and nonvoiced speech could be obtained over telephone lines using the statistical approach; however the overall error rate for the 3-class decision was fairly high (l1.7%) .

A novel approach was suggested by McAuley [28] in which a matched digital Wiener filter was designed for each of the signal classes, and the signal was processed by each of these fitters. Based on the signal output from each of the filters, a distance was computed representing how closely the input signal was matched to the filter, and the minimum distance was used to make the final classification. Although this approach shows promise it requires a large amount of signal processing, and has not as yet been extensively tested.

We summarized the voice detection related works in Table 3.2.

Table 3.2:summarizing voice detection related works

Related work	Method & Detection Technique	Shortcomings
Saha and Chakroborty [24]	uni-dimensional Mahalanobis algorithm	doesn't make the classification between silence part and unvoiced part.
Jacobs et. al. [25]	algorithm for silence detection based on the small and large signal behavior of the speech waveform in the law domain	doesn't make the classification between voiced part and unvoiced part
Rabiner and Sainbur [26]	approach to the voiced-unvoiced-silence detection problem , in which a spectral characterization of each of the 3 classes of signal is obtained during a training session,	limited on specific application such as telephone lines
Atal and Rabiner [27]	statistical decision approach to voiced-unvoiced-silence using a non-Euclidean distance metric	error rate was fairly high (l1.7%)
McAuley [28]	matched digital Wiener filter was designed for each of the signal classes	large amount of signal processing

CHAPTER 4

METHODOLOGY AND PROPOSED METHOD

In this chapter, we presented and explained the proposed method (DISE) and methodology which we followed in this research. DISE method is described using flowcharts and figures.

The method tasks are introduced as well as the requirements specification, analysis and design of these tasks.

4.1 The Proposed DISE Method

Building an accurate method for detecting inactivity segments requires the integration of pedagogical perspective as well as technical perspective.

Figure 4.1 provides a top level flowchart for the overall proposed method for detecting inactivity segments that we build.

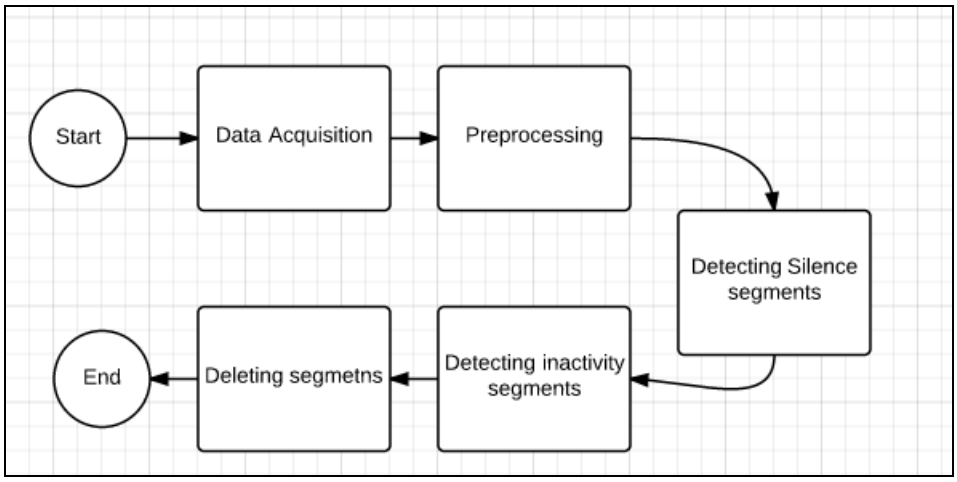

Figure 4.1 : Top level flowchart of the proposed method

The method has five basic steps which are : Data acquisition, Pre-processing such as extracting audio from the video and reducing noise, detecting the silence period of time, detecting the inactivity segments and finally deleting them.

According to the presented DISE method, the five basic steps are:

4.1.1 Data acquisition:

This consists of user input data which is a video with any size or type . Also it consists from four values of thresholds. We have been reached to those values for thresholds after many numbers of attempts and experiments with different values in different situations and environments.

Thresholds are:

1) Min Silence Length : indicates how long the silence length must be at least. The value is entered by seconds. This value is chosen according to the institution policies using this method , for example some institutions allow long silence periods that reach to 5 minutes. Such as events or conferences recordings , so the user will enter the threshold value = 300 seconds. Other institutions maybe so aware to prevent silence more than 2 minutes , so the user will enter 120 s (2*60 = 120 s) such as lectures recordings in universities.
2) Noise threshold : indicates the degree of the allowed noise , where choosing the highest value means the method is more sensitive to noise and vice versa. This threshold value is detected by a professional after a number of experiments on some samples from the same environment.

The user will choose from three choices as shown in table 4.1.

Table 4.1 : The Noise threshold value

The value	The degree of noise	Type of reduced noise
0.01	Delete the minimum noise frequencies	Hush and Him resulted from using microphones
0.1	Delete the high noise frequencies	Low whispers between students , or voices for people or vehicles outside the room

3) Silence level threshold : detects the silence segments when comparing it with the audio volumes. This threshold value is detected by a professional after a number of experiments on some samples from the same environment.
The user will choose from three choices as shown in table 4.2 :

Table 4.2: The Silence threshold values

The value	The degree of Silence
Low = 0.0001	Detect low silence levels , for example : whispering of the speaker is not silence.
Medium = 0.000001	Detect medium silence , for example : breathing of the speaker is not silence.
High = 0.00000001	Detect complete silence , for example : whispering and breathing of the speaker is silence.

4) Frame compare threshold : detects how is the similarity between frames when comparing the images to find motion , this depends on the camera recording the video and its ability to maintain satiability of the images while recording the lectures.
When using a stable and a high quality cameras , the user can lower the value of the frame compare threshold , otherwise when recoding using humans or by a low quality cameras , user most raise the value of the threshold , to not give the opportunity to the human vibration to affect the result of the system.
The user will choose from three choices as shown in table 4.3 :

Table 4.3:The Frame compare threshold values

The value	The degree of similarity
Low (30 %)	Detect similarity between images up to 70%
Medium (20%)	Detect similarity between images up to 80%
High(10%)	Detect similarity between images up to 90%

4.1.2 Preprocessing

Preprocessing starts after choosing the video file by the user and then choosing the thresholds. Preprocessing includes:

4.1.2.1 Extracting audio from the video

After choosing the video and the thresholds , the audio is extracted from the video file and saved into a wav file. DISE can extract audio from any of the many input sources it supports, and write this audio to an audio-file in a variety of formats. In other words, it discards any video content from the input source, and it converts the audio content to the desired format.

Using FFmpeg , which is an extremely powerful and versatile command line tool for converting audio and video files and available for Windows, Mac and Linux machines, we can extract the audio component from a video file.

The FFmpeg command is called within the java code using the external command call routine runtime.exec().

Runtime.exec() returns a Process object that provides you access to three streams:

1- InputStream: this is the stream to which the executing process writes its output to
2- ErrorStream: this is the stream to which the executing process writes errors to
3- OutputStream: this is a stream to which you can send input and interact with the executing process.

The –vn switch in ffmpeg , extracts the audio portion from a video and we are using the -ab switch to save the audio as a wav audio file named sound.wav. The command used is:

```
ffmpeg –I video.mp4 –vn –ab 256 sound.wav
```

where video.mp4 is the input video , and sound.wav is the output audio file.

4.1.2.2 Reducing noise

Noise in audio, recording, and broadcast systems refers to the residual low level sound (usually hiss and hum) that is heard in quiet periods of a program.
In audio engineering it can refer either to the acoustic noise from loudspeakers, or to the unwanted residual electronic noise signal that gives rise to acoustic noise heard as 'hiss'.
SoX is a power-packed command-line tool for various types of audio processing. It's very useful as an audio format converter, and it can be used for removing noise from audio files.

To use it, first run SoX with the noiseprof effect on a section of audio that ideally would contain silence but in fact contains noise –such sections are typically found at the beginning or the end of a recording. Noiseprof will write out a noise profile to profile-file. The command is:

```
# Create background noise profile from mp3:
/usr/bin/sox noise.mp3 –n noiseprof noise.prof
```

Where noise.prof is the noise profile that is generated from the audio file noise.mp3.

To actually remove the noise, run SoX again, this time with the noisered effect; noisered will reduce noise according to a noise profile (which was generated by noiseprof). The command is:

```
# Remove noise from mp3 using profile:
/usr/bin/sox sound.wav noise.wav noisered noise.prof
```

Where sound.wav is the input audio file , noise.wav is the output audio file without noise.
But creating a background noise profile can be somehow complex specially when using the system for a large number of videos, So we can avoid this step by using a threshold , which can be detected by a professional after a number of experiments and entered as an input in the interface, hence the command is:

```
# Remove noise from mp3 using a threshold:
/usr/bin/sox sound.wav noise.wav noisered noise.prof 0.01
```

Where sound.wav is the input audio file , noise.wav is the output audio file without noise.

And 0.01 is the threshold entered by user. Now the noise.wav file is ready to the next step to detect the silence segments.

After reducing noise , the preprocessing step finishes , and prepare the video to next steps.

4.1.3 Detecting silence from the audio file

Detecting silence from the audio file is one of the main functions that the method does, and where the further functions depends on its results.The algorithm is shown in table 4.5.

The input is the audio file named noise.wav free from noise resulted from the previous steps.

This step is running as follows:

1) Initiate a sound volume file named sound.dat that contains a table from the following rows:

Time	Ch 1	Ch2

Time: The time by milliseconds.
Ch1 : The volume in the left speaker.
Ch2 : The volume in the right speaker.
Where at normal cases Ch1 must be equal to Ch2., as shown in figure 4.2.

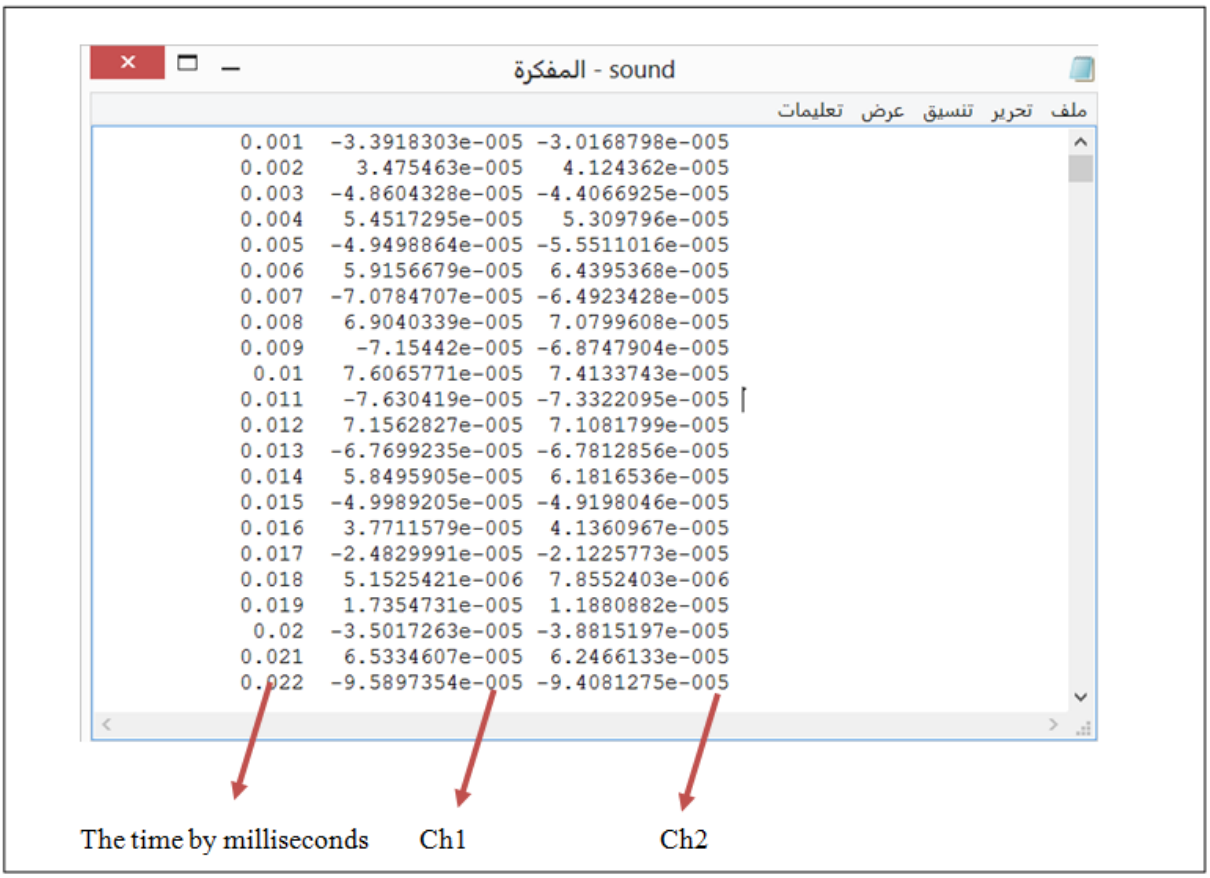

Figure 4.2: The file sound.dat that contains the volumes

2) Suppose using Ch2 values , a comparison is done between Ch2 values and the silence level threshold entered by the user as shown in the previous section, the result of the comparison will be saved in another file as as shown in table 4.4:

Table 4.4 :The result of comparing sound volume and threshold

Ch2 values	Result	Explanation
More than the threshold	1	No silence
Equal to the threshold	0	No silence
Less than the threshold	-1	Silence

3) Now , depending on the Min Silence Length threshold , the silence segments are accepted if the silence segment length is more than the threshold. For each silence segment , a start pointer (S1) is pointing the second at the start of the silence segment .An end pointer (Sn) is pointing the second at the end of the silence segment. Where n is the number of seconds between the start and end pointer , in other words n detects the length of the silence segment.

Table 0.5:Detecting silence algorithm

Algorithm 4.1: detecting silence

Input: audio file, Min Silence Length threshold , Silence level threshold

Output: SilenceB (Boolean)

```
1.  Silence[ ]
2.  T ← 0                    // Time
3.  L = AudioFile.Lenght
4.  X =0        //Counter
5.  For T ← 0 , to L
6.       x1 ← ch1.volume(T)
7.       x2 ←ch2.volume(T)
8.       If x1 > Silence level threshold
9.          Silence[T]=0
10.      else
11.       Silence[T]=1
12.      End if
13.     T=T+1
14. End for
15. For x=0 , to L
16.       If Silence[x] =1
17.      Count = count+1
18. End for
19. If count < Min Silence Length threshold
20.     SilenceB = False
21.     Break
22. Else
23.     SilenceB = True
24.     Startframe = x.getsnapshot()   //  taking a snapshot from the frame
25.     Endframe = count.getsnapshot()
26. End if
```

4.1.4 Detecting the inactivity segments

Using a technique called blocks-average method based on temporal differencing according to the comparison in table 2.1 . And according to our situation where we are using videos for lectures recordings in a university where one lecturer is speaking in front of a camera in a lecture room. The background subtraction approach is not suitable to be used because any change in the background such as ambient lighting, weather and this is

so popular may lead to wrong results. Also probability approach is not suitable to be used because the high computing cost. This algorithm is simple , suitable for the input videos and takes less processing time and , which increases the speed and also the detection rate.

The algorithm is discussed in details in the subsection 4.1.4.1 , the method detects motion and objects movement. Similarity is required between frames and not coincidence. A comparison is done between the frame that lies in the second of the start of the silence segment (S1) and the next frame (S2) which follows it. Frames are transformed into gray scale .If the two frames are similar then a compare will be held between S1 and S3 , S4 , … the comparison will stop in two cases :

Case 1 : Reach the frame lies at the end millisecond of the silence segment (Sn). In this case the segment between (S1) and (Sn) is to be defined as an Inactivity Segment . This segment is copied and saved as an individual video file in an external folder created automatically when first running the system.

Case 2 :The comparison between two frames result a non – similarity. Then the length between the start frame and the last same frame is computed , if the length is less than the Min Silence Length threshold , then the silence segment is discard , otherwise it is defined also as an Inactivity Segment , copied and saved as an individual video file in the same external folder.

This step is looped for all the silence segments results from the fourth step.

4.1.4.1 Blocks-average method

This section will discuss how the blocks-average method works for comparing images . The algorithm is shown in table 4.6.

First of all , it only works on gray scale images , so images must be transformed into gray scale before comparing them together using this method .

In this method the pixels of the image are collected into groups called blocks , there are ten blocks horizontally and ten blocks vertically. Where the correspondent pixels are aggregated in the same correspondent blocks in the compared images, then for each block , as shown in Figure 4.3.

- The number of pixels in each block is calculated using this equation :

No of pixels in a block = (image.getWidth/10) * (image.getHeight /10)

Where N : Number of pixels in an individual block.

W: the width of the image.

H: the height of the image.

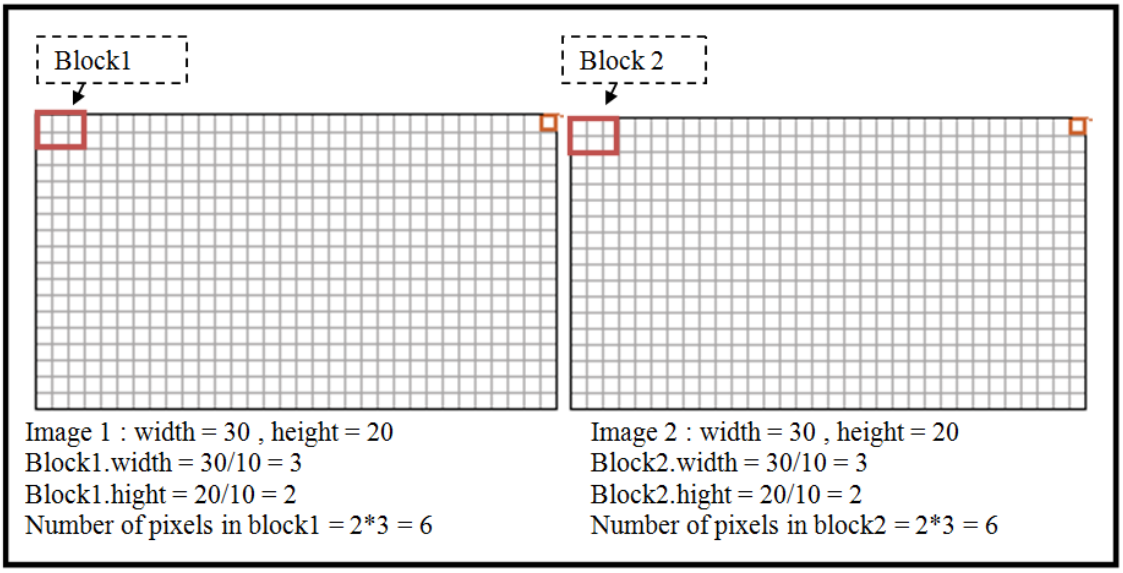

Figure 4.3: Selecting blocks using Blocks-average method

- The average of the images pixels is calculated by using a normal average equation:

$V=\sum p / N$

Where V : The average of the pixels value in an individual block

P : pixel value from each pixel in the block

N : Number of pixels in an individual block.

Now for two consecutive frames:

Avg1 = sum of the pixels values in b1 / number of the pixels in b1 Avg2 = sum of the pixels values in b2 / number of the pixels in b2

Where b1 ; The block from the first image.

b2: The block from the second image.

- Then B is calculated , where B is the degree of similarity between blocks b1 and b2 ,it is calculated as the absolute num for avg1 from avg2 , where B =| avg2 – avg1|.

If B is bigger than the frame compare threshold which has been entered by the system user, this means the difference between the blocks is too large , so the method stops comparing those images and consider them as not similar.

Otherwise , if B is less than the threshold , a loop through whole image is done and compared all individual blocks of images. If all the comparing values are less than the threshold then those compared images are considered as similar , as shown in Figure 4.4.

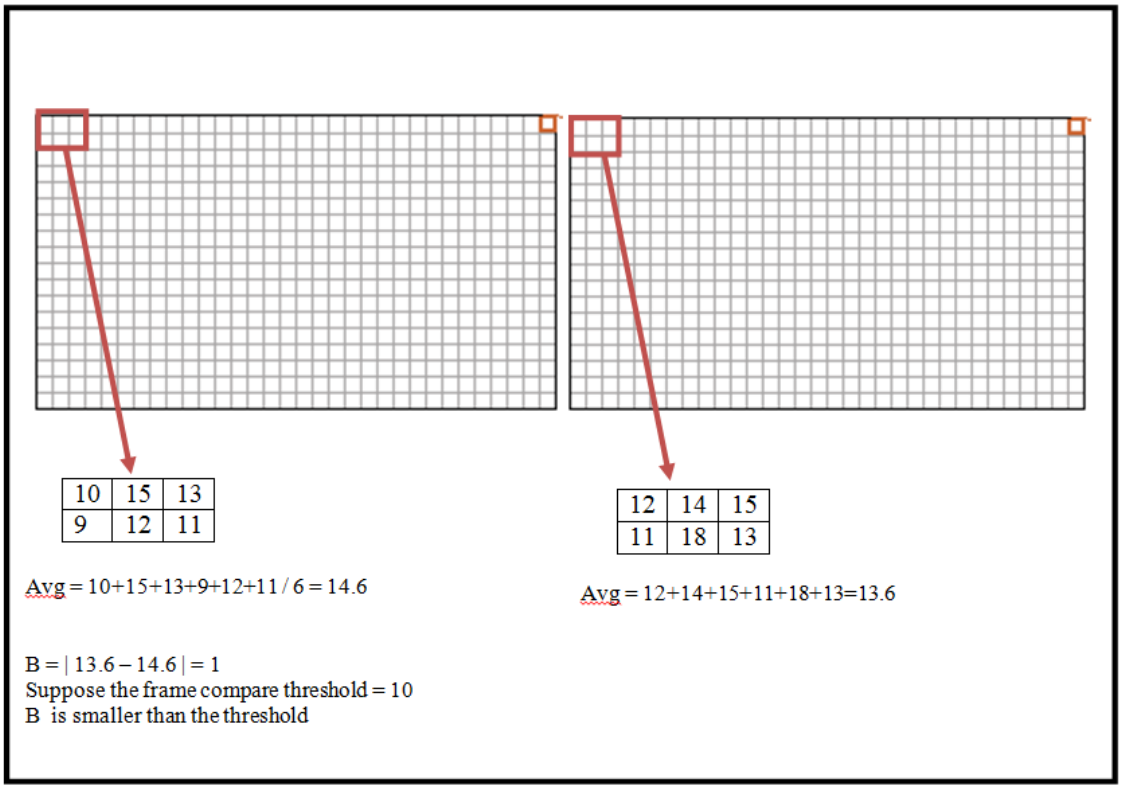

Figure 4.4 : Calculating similarity using Blocks-average method

Table 0.6 : Blocking average algorithm

Algorithm 4.2: Blocking and comparing

Input: img1, img 2 , frame compare threshold

Output: Similarity (Boolean)

```
1.  x1 ← img1. Height
2.  y1 ←img1.width
3.  x2 ← img2. Height
4.  y2 ←img2.width
5.  blockx1 ← x1/10
6.  blocky1 ← y1 /10
7.  blockx2 ← x2/10
8.  blocky2 ← y2 /10
9.  Sum1 ← 0
10. Sum2 ← 0
11. for I ← 0 to blockx1
12.     for j ← 0 to blocky1
13.         if blocky2 > img1.width
14.             blocky2 ← 0
15.         end if
16.         sum1 ← sum1 + pixelvalue(j)
17.         sum2 ← sum2 + pixelvalue(j)
18.     end for
19.     b1 ←sum1
20.     b2 ←sum1
21.     b ←b2-b1
22.     if b > frame compare threshold
23.         match ← false
24.     break
25.     else
26.         match ← true
27.         I ← blockx1+1
28.         blockx1 ← blockx1 + blockx1
29.     end if
30. end for
```

4.1.5 Delete inactivity segments

Deleting the inactivity segments after detecting them most take into account two factors, when both of them are satisfied , we can say that , deleting was successful , those factors will be evaluated manually by the user after watching the resulted video from the system. Those factors are :

1) Deleting without affecting the continuous of the video :

When deleting the inactivity segments, the major important point is to maintain the continuous of the video after merging the active segments , that means :

- No black segments in the output video.
- No cutoff in the output video.
- Confirm that there is no missed active segment not included in the output video

2) Deleting without affecting the quality of the video

Also it is important to not affect the quality of the output video , and that means:

- No derangement in the images.
- Coordinate between the image and the voice.

The system operates in two modes:

a) Fully automatic: In this mode the user specifies a set of configuration values related to motion and human voice such as the video and audio thresholds that we discussed earlier in section 4.1.1. Based on this set of configurations the system detects inactivity segments, delete those segments, and returns a "clean" video; while idle segments are output each as a separate file. In other words, the system in this mode takes independent action.

b) Semi-automatic: Similar to mode (a) the user specifies a configuration set, and the system detects inactivity segments. However, the system does not take independent action. The system only presents the detected segments to the user via a table that presents the segments , their lengths , the start second ,the end second and the type of the segment if it is active or inactive. The user can watch each segment separately and then decides whether to delete or keep them .

For both modes , the chosen inactive segments are deleted from the video , then the active segments are merged out to produce a clean , full active , high quality and a continuous output video , while keeping the deleted segments in the external folder for further watching.

4.1.5.1 Cutting a section of a movie

As an example, if I want to cut off from 0 to 7.5 second at the start of a video named Yellowstone.mp4, , and chop off the tail from 4:05 to the end.
Here is the ffmpeg command:

```
ffmpeg –I Yellowstone.mp4 –ss 00:00:07.5 –to 00:04:05 -c copy
Yellowstone_cut.mp4
```

Where – I specify the name of the video.
- ss specify the start point of the video. In the form of hh:mm:ss
- to specify the end point of the video . In the form of hh:mm:ss

This code copies the video named Yellowstone.mp4 from the second 7.5 to the second 4.05 to a new video named Yellowstone_cut.mp4. Since it's just making a copy, it's quick.
Now , to extract only a small segment in the middle of a movie, it can be used in combination with –t which specifies the duration, like –ss 60 –t 10 to capture from second 60 to 70. Or you can use the –to option to specify an out point, like –ss 60 –to 70 to capture from second 60 to 70. –t and –to are mutually exclusive. If you use both, -t will be used.
Note that if you specify –ss before –I only, the timestamps will be reset to zero, so -t and –to have the same effect:

```
ffmpeg -ss 00:01:00 -I video.mp4 -to 00:02:00 -c copy cut.mp4
ffmpeg -I video.mp4 -ss 00:01:00 -to 00:02:00 -c copy cut.mp4
```

Here, the first command will cut from 00:01:00 to 00:03:00 (in the original), whereas the second command would cut from 00:01:00 to 00:02:00, as intended.

4.2 The detailed flowchart for DISE

Figure 4.5 explains the detailed flowchart of DISE method, starting with inserting the video by the user using the interface of the system, and entered the required thresholds. Then DISE method makes preprocessing for the video that includes separating video from audio and reducing noise then saving the audio volumes in a txt file. After that the system will detect the silence segments depending on the threshold and compare frames among the silence frames, when frames are similar to a degree that is identified through the threshold then the segment is inactive , then it is moved to a window to be viewed by the user , whose in turn decides either to maintain or delete. After taking decision , the output video is ready , without any noise , inactive and unwanted segments.

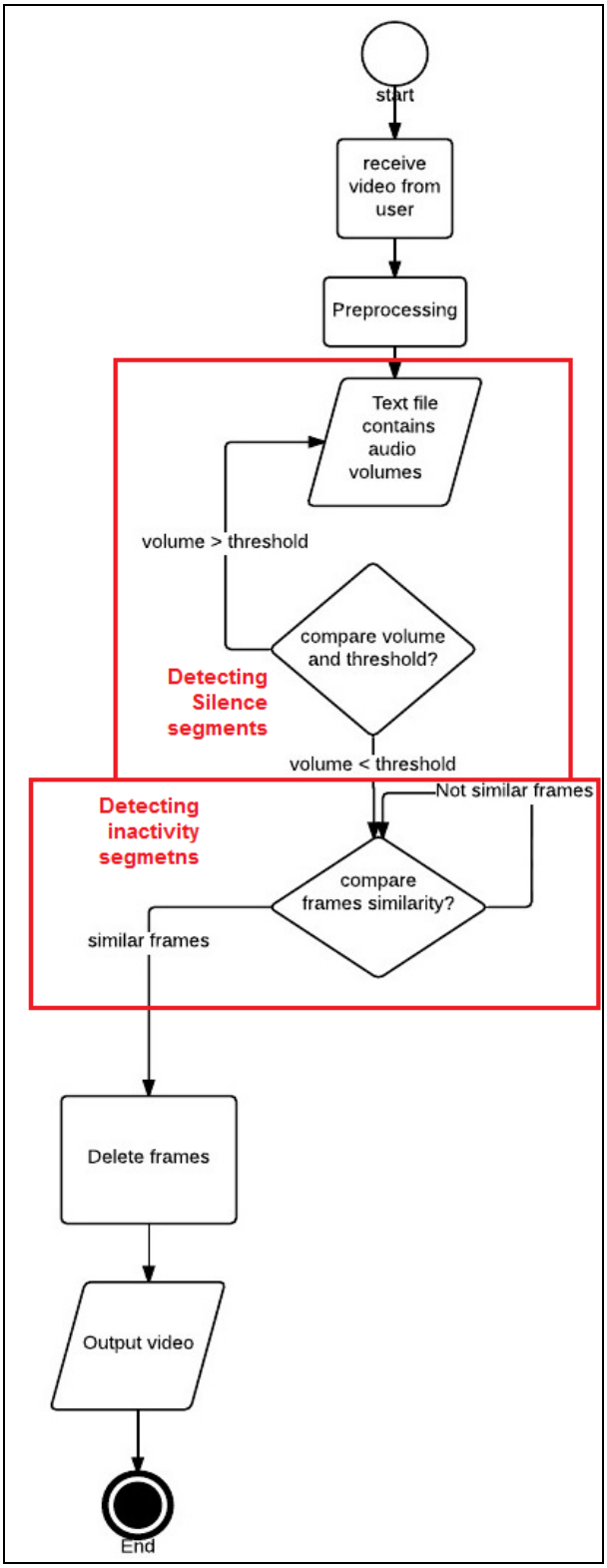

Figure 4.5 : DISE flowchart

4.3 Conclusion

In this section we will talk about the conclusion of the proposed DISE method used to detect inactive segments in video .

Five stages are implemented in the method which are:

1. Data acquisition: This consists of user input video and four threshold values.
2. Pre-processing such as extracting audio from the video and reducing noise.
3. Detecting silence from audio file , by first generating a dat file contains the volumes at each second , and then compare the volumes with a threshold to detect the silence segments.
4. Detecting the inactivity segments by comparing frames that lies in the silence segments driven from the previous step.
5. Taking action after the decision of the user by deleting unwanted frames.

In the following Table 4.7 the summary of the processes for the proposed method is shown briefly.

Table 4.7 : Summary for DISE method steps

Step #	Input	Process	Output
Step 1	- Video from a user - Thresholds values	Receive video and thresholds values	-
Step 2	Received Video	Preprocessing (extract audio from video)	Audio file : sound.wav
	Audio file : sound.wav	Preprocessing (reducing Noise)	Audio file : noise.wav
Step 3	Audio file : noise.wav	Detecting silence from audio file (writing volumes in a dat file)	Dat file : sound.dat
	Dat file : sound.dat	Detecting silence from audio file (comparing volumes with threshold)	Silence segments
Step 4	Silence segments	Comparing frames among each silence segments produced from step 3 using Avg- blocks algorithm	Inactive segments
Step 5	Inactive segments	Deleting inwanted segments after viewing it from user	Output clean , no noise , no inactive segments video

CHAPTER 5

EXPERIMENTAL RESULTS AND EVALUATION

In this chapter we will present the implementation and the experiments on DISE . The first section will state the implementation of the method, states the programming language and tools used to develop the proposed system , the interface of the system and the videos used to test the system . The second section will briefly describes the experimental procedures and results. Finally, the third section will discuss the processing time and its factors.

5.1 Implementation

This section contains the system interface , the specifications of the input videos and the tools and programs used to implement the method.

5.1.1 System Interface

Our system interface is very simple , consists of one primary windows . An example for using the interface with more details is explained in Appendix A . The interface contains the following items as shown in figure 5.1:

1- File browser button: to allow user to choose the video file which need to be analyzes to detect inactive segments.
2- Sound wave form: Represent the audio frequencies in a waveform.
3- The min silence length threshold .
4- The noise level threshold.
5- The silence level threshold.
6- The frame compare threshold.
7- The start frame and the end frame for each silence segments which is in process using the avg-compare algorithm.
8- Under the frames , a text shows what is in process.

9- A table that shows the detected segments , their start , duration , type and size.

10- Play bottom : will play the selected segment from the above table.

11- Delete bottom : will delete the selected segment from the above table.

12- Open folder bottom : will open the directory of the input video and the output video and segments.

13- Run bottom : will merge the remain segments after deleting the unwanted , and then run the output video.

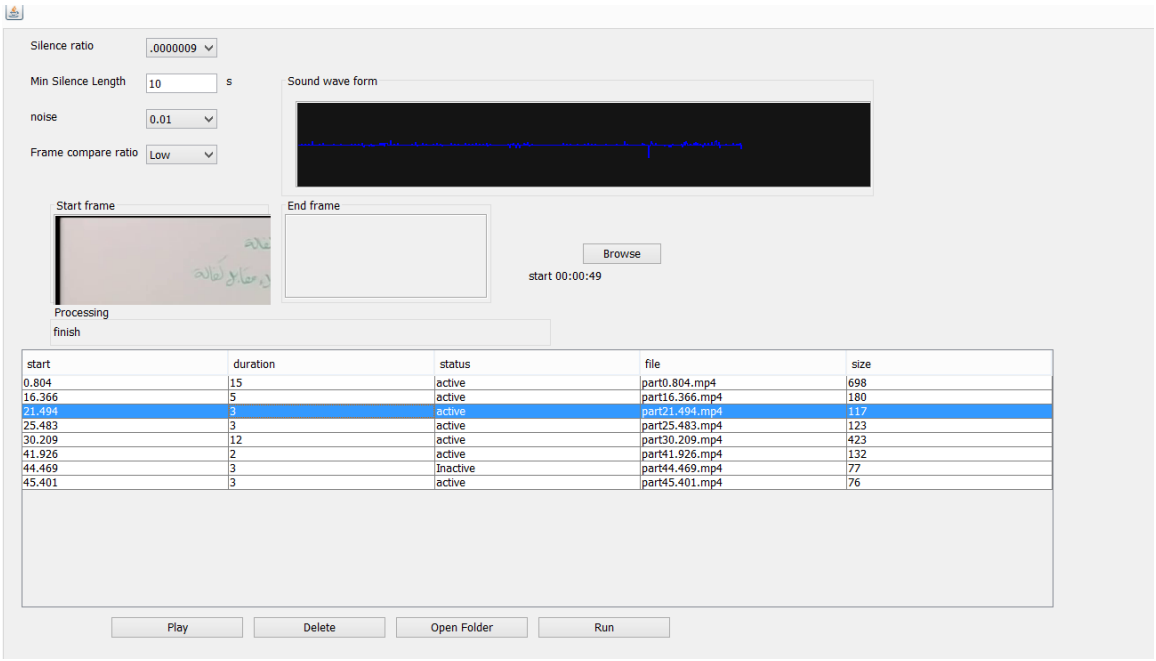

Figure 5.1:The system interface

5.1.2 Input videos

For our DISE we collect videos for lectures taken in Palestine University (see Appendix B) . A randomly 24 videos are selected between January 2011 and January 2015.

The input video can be with any type such as : mp4 , wmv or mpeg. The video must be at least at the medium quality, to be able to distinguish between voice and noise. The system can accept a video with any size , but user must take into account that , increasing the size of the video will increase time of processing as discussed later in section 5.2.

5.1.3 Tools and programs

To implement the DISE, we used java programming language because most of preprocessing tools are found in java language beside special tools such as ffmpeg and sox.

Table 5.1 displays part of the code with some explanation; we shall introduce other code examples with its explanations in Appendix C.

Table 5.1: Avg-blocks algorithm

```
public void compare( ) {
    // setup change display image
    imgc = imageToBufferedImage(img2);
    Graphics2D gc = imgc.createGraphics();
    gc.setColor(Color.RED);
    // convert to gray images.
    Img1 = imageToBufferedImage(GrayFilter.createDisabledImage(img1));
    img2 = imageToBufferedImage(GrayFilter.createDisabledImage(img2));
    // how big are each section
    int blocksx = (int) (img1.getWidth() / comparex);
    int blocksy = (int) (img1.getHeight() / comparey);
    // set to a match by default, if a change is found then flag non-match
    this.match = true;
    // loop through whole image and compare individual blocks of images
    for (int y = 0; y < comparey; y++) {
      if (debugMode > 0) {
        System.out.print("|");   }
      for (int x = 0; x < comparex; x++) {
        int b1 = getAverageBrightness(img1.getSubimage(x * blocksx, y * blocksy,
        blocksx – 1, blocksy – 1));
        int b2 = getAverageBrightness(img2.getSubimage(x * blocksx, y * blocksy,
       blocksx – 1, blocksy – 1));
        int diff = Math.abs(b1 – b2);
```

```
            if (diff > factorA) { // the difference in a certain region has passed the threshold
           value of factorA
             // draw an indicator on the change image to show where change was detected.
               Gc.drawRect(x * blocksx, y * blocksy, blocksx – 1, blocksy – 1);
               this.match = false; }
            if (debugMode == 1) {
               System.out.print((diff > factorA ? "X" : " "));
            }
            if (debugMode == 2) {
               System.out.print(diff + (x < comparex – 1 ? "," : ""));
            }          }
         if (debugMode > 0) {
            System.out.println("|");
         }
      }   }
```

Explanation: This function is to compare two images using Avg-blocks algorithm , then compare the result with a threshold.

The implementation were conducted using an Intel® Core™i7 CPU 2.00GHz with 4.0GB RAM. Special tools and programs are used to complete the implementation of DISE and documentation of the thesis , such as:

1) VLC media player : is a free and open source cross-platform multimedia player and framework, that plays most multimedia files.
2) Net Beans IDE 8.0: this is the program which helps us to build interface and finish the system implementation using java language.
3) Java Development Kit (JDK) 1.6: A software development package from Sun Microsystems that implements the basic set of tools needed to write, test and debug Java applications.
4) Microsoft Word 2007: the main program used to write the documentation of the system.

5.2 Experiments Setup

This section describes the setting of experiments for evaluating our proposed approach. Its primary purpose is to empirically validate the advantages and to notice the shortcomings of our proposed method. Our approach is expected to detect inactive segments in videos without affecting accuracy and quality. The following sections describe experimental procedures and results.

5.2.1 Experiment Procedure

The goal of experiments is to observe the system performance.

The used evaluation technique is calculating precision, recall , accuracy and error for each output video to evaluate our work. All measurements are applied manually and results are summarized from table 5.2 to 5.6.

Four important measures are commonly used, precision, recall , accuracy and error.

Before we go to the equations , some points must be explained:

1- True positive (tp): are correct results generated from the system.

2- False positive (fp): are wrong results generated from the system.

3- True negative (tn): are correct results not generated from the system.

4- False negative (fn): are wrong results not generated from the system

Where all the numbers of the results (n) = tp + fp + tn + fn eq (5.1)

Precision is a measure of how much of the returned information by the system is correct. ***Precision*** is the fraction of correct results (where ***tp*** is the true positive and ***fp*** is the false positive) to summation of correct results and unexpected results as follows:

Precision =

$$\frac{tp}{tp + fp} \qquad \text{eq. (5.2)}$$

Recall is a measure of the coverage of the system. ***Recall*** is the fraction of correct results (where ***TP*** is the true positive) to summation of correct results and missing results (where ***FN*** is false negative) as follows:

$$\text{Recall} = \frac{tp}{tp + fn} \qquad \text{eq. (5.3)}$$

Usually Recall and Precision are antagonistic to one another. A system strives for coverage will get lower precision and a system strives for precision will get lower recall. To measure Accuracy Eq. (5.4) is used, which is the fraction of correct results and correct absence (true positive and true negative) to the summation of correct results, unexpected results, missing results and correct absence (true positive ***TP***, true negative ***TN***, false positive ***FP*** and false negative ***FN*** respectively which are known as binary evaluation) .To measure error rate Eq (5.5) is used . Error is the fraction of wrong results and wrong absence (false positive and false negative) to the summation of correct results, unexpected results, missing results and correct absence (true positive ***TP***, true negative ***TN***, false positive ***FP*** and false negative ***FN*** respectively).

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn} \qquad \text{eq. (5.4)}$$

$$\text{Error} = \frac{fp + fn}{tp + tn + fp + fn} \qquad \text{eq. (5.5)}$$

The evaluation steps that used in the evaluation technique:

1) After applying inactive segments detection , some inactive segments are correctly detected by the system. But there still some inactive segments that are not detected. So, we count correct inactive segments detected by the system in output result as correct result (TP).
2) Some segments in the result window which are labeled as inactive are not inactive segments and should not be labeled as inactive . Also some active segments in the

result window are labeled active where they are inactive. So, we are counting as unexpected output (FP).

3) The inactive segments are correct absence from the output result we are counting as correct absence (TN).
4) Some items are selecting from the beginning incorrect , but some rules prevent it from appearance in the output result, and we are counting as missing (FN).
5) Then we are counting the correct results, missing results, unexpected results and correct absence results for each video to compute precision, recall and accuracy for each video.

5.2.2 Experimental results

In this section we will show our work results on different types of difficulties. These difficulties come from combination between background noise and speaker voice or low quality images (see table 5.2). Such as video has high noise, or the sound of the speaker is too low.

Table 5.2 : The types of videos difficulties

Video type	Noise level	Speaker level	Quality of the image
1	Low	High	High
2	Low	High	Low
3	Low	Low	Low
4	Low	Low	High
5	High	High	High
6	High	High	Low
7	High	Low	Low
8	High	Low	High

We are using 24 videos , 3 from each type from the types that are shown in the table above.

For each video , we calculated tp , fp , tn and fn. Then we calculated precision, recall , accuracy and error. See tables from 5.3 to 5.6.

Table 5.3 calculated *tp* and *fp* for the test videos . Table 5.4 calculate *tn* and *fn* according to human views. Table 5.5 calculates precision, recall , accuracy and error.

Table 5.3:Calculating tp and fp

No	Video type	Period in minutes	inactive segments	correct inactive segments	active segments	correct active segments	Total correct segments (tp)	Total wrong segments (fp)
1	1	5:12	5	5	9	9	14	0
2		1:40	4	3	12	12	15	1
3		37:56	8	7	14	13	20	2
4	2	29:57	4	3	9	8	11	2
5		2:00	2	2	8	6	8	2
6		15:1	7	6	18	17	23	2
7	3	37:09	7	5	14	11	16	5
8		24:00	3	2	8	7	9	2
9		3:28	2	2	7	6	8	1
10	4	10:06	3	3	10	9	12	1
11		12:12	3	2	8	8	10	1
12		1:17	8	8	20	18	26	2
13	5	13:0	4	4	15	14	18	1
14		5:30	2	1	5	4	5	2
15		1:25:05	2	2	7	5	7	2
16	6	37:11	8	7	20	19	26	2
17		25:0	11	9	25	23	32	4
18		15:0	4	2	9	5	7	6
1 9	7	51:43	5	5	11	11	16	0
20		14:1	4	2	15	14	16	3
21		30:0	8	7	25	23	30	3
22	8	17:4	5	5	12	11	16	1
23		21:3	2	2	19	16	18	3
24		14:4	2	0	10	10	10	2

Table 5.4: calculating tp , tn , fp and fn for each test video

No	Video type	Period of video in min	inactive segments	Missed correct inactive segments	active segments	Missed correct active segments	Total segments by system	Total correct segments by system (tp)	Total correct segments by human	Total missed correct segments (tn)	Total wrong segments (fp)	Total missed wrong segments (fn)
1	1	5:12	5	0	9	1	14	14	14	1	0	1
2		1:40	4	1	12	1	16	15	17	2	2	0
3		37:56	8	1	14	2	22	20	23	3	3	0
4	2	29:57	4	2	9	2	13	11	14	4	3	1
5		2:00	2	2	8	2	10	8	10	4	2	2
6		15:1	7	1	18	3	25	23	26	4	3	1
7	3	37:09	7	2	14	4	21	16	18	6	2	4
8		24:00	3	1	8	2	11	9	10	3	1	2
9		3:28	2	1	7	3	9	8	8	4	0	4
10	4	10:06	3	1	10	1	13	12	12	2	0	2
11		12:12	3	1	8	0	11	10	10	1	0	1
12		1:17	8	2	20	4	28	26	27	6	1	5
13	5	13:0	4	2	15	1	19	18	18	3	0	3
14		5:30	2	1	5	2	7	5	6	3	1	2
15		1:25:05	2	0	7	4	9	7	8	4	1	3
16	6	37:11	8	1	20	1	28	26	27	2	1	1
17		25:0	11	1	25	3	36	32	34	4	2	2
18		15:0	4	2	9	2	13	7	11	4	4	0
19	7	51:43	5	2	11	2	16	16	16	4	0	4
20		14:1	4	2	15	4	19	16	17	6	1	5
21		30:0	8	1	25	1	33	30	30	2	0	2
22	8	17:4	5	2	12	0	17	16	17	2	1	1
23		21:3	2	1	19	2	21	18	20	3	2	1
24		14:4	2	0	10	4	12	10	10	4	0	4

Table 5.5 : Calculating precision, recall , accuracy and error.

No	Video type	Period of video in minutes	Precision	recall	Accuracy	Error
1	1	5:12	100	93	94	6
2		1:40	88	100	89	11
3		37:56	87	100	88	12
4	2	29:57	79	92	79	21
5		2:00	80	80	75	25
6		15:1	88	96	87	13
7	3	37:09	89	80	79	21
8		24:00	90	82	80	20
9		3:28	100	67	75	25
10	4	10:06	100	86	88	13
11		12:12	100	91	92	8
12		1:17	96	84	84	16
13	5	13:0	100	86	88	13
14		5:30	83	71	73	27
15		1:25:05	88	70	73	27
16	6	37:11	96	96	93	7
17		25:0	94	94	90	10
18		15:0	64	100	73	27
19	7	51:43	100	80	83	17
20		14:1	94	76	79	21
21		30:0	100	94	94	6
22	8	17:4	94	94	90	10
23		21:3	90	95	88	13
24		14:4	100	71	78	22
AVG			91%	86%	84.83%	15.16%

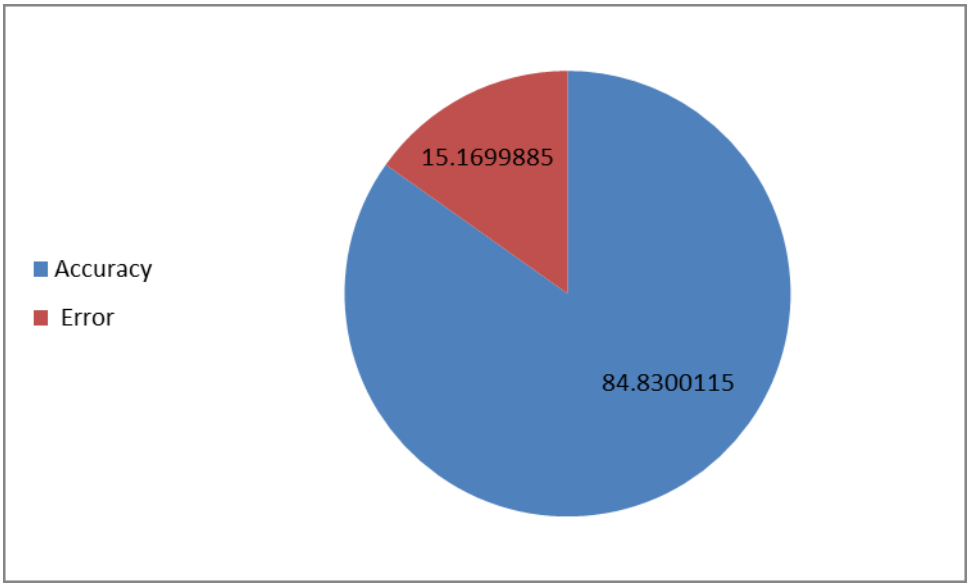

Figure 5.2 : Accuracy and error ratio for all video types

For each video type , we calculate the accuracy and the error rate . the results are shown in table 5.6 . Figure 5.3 shows a comparison between the types.

Table 5.6 : The accuracy and error ratio for the video types

Video level	Accuracy	Error
1	98.61%	1.38%
2	90.98%	9.01%
3	75.38%	24.61%
4	86.34%	13.65%
5	76.46%	23.53%
6	77.63%	22.36%
7	88.95%	11.04%
8	84.25%	15.74%

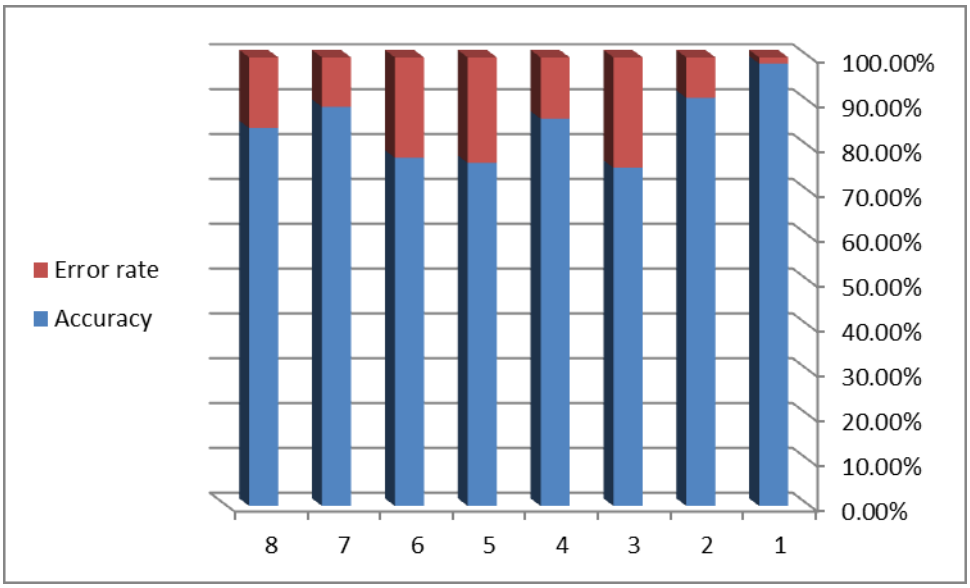

Figure 5.3: A comparison between the video types

5.3 Processing Time

Now we will compute the processing time for each video , and find the relation between processing time and the size or the duration of the video.

Table 5.7 shows the results for detecting inactivity segments for each video .

By referring to the table , we notice that videos with large sizes take more processing time regardless their duration , while the duration of the video has a slight effect on the processing time .

By referring to section 2.2 , the frame rate has a large effect on the file size. Frame rate, also known as frame frequency, is the frequency (rate) at which an imaging device displays consecutive images called frames. The term applies equally to film and video cameras, computer graphics, and motion capture systems. Frame rate is expressed in frames per second (FPS). Lowering the frame rate will directly lead to the decrease of video file size. With regard to the bit rate, it mainly describes video or audio quality.

Table 5.7:Relation between processing time with size and duration

No	Video type	Period of video in min	Size of the video In MB	Processing time in minutes
1	1	5:12	13	00:10
2		1:40	3	00:04
3		37:56	4.79	00:24
4	2	29:57	82.34	09:15
5		2:00	5.8	00:05
6		15:1	7	00:20
7	3	37:09	177	14:03
8		24:00	2.4	00:15
9		3:28	46	01:30
10	4	10:06	26.08	00:47
11		12:12	31	02:01
12		1:17	100	04:02
13	5	13:0	4	00:08
14		5:30	8.2	00:12
15		1:25:05	60	03:44
16	6	37:11	98	13:31
17		25:0	41	02:27
18		15:0	30.21	01:12
19	7	51:43	33	01:51
20		14:1	24	01:15
21		30:0	47.3	01:34
22	8	17:4	32.2	12:24
23		21:3	20.1	01:19
24		14:4	4.25	00:58

5.4 Discussion

From the previous experiments and comparisons we can find that:

1- DISE is very similar to human detection as it achieves accuracy of 84.4% with our used evaluation technique .

2- From the table 5.6, we notice that , the video level 3 , is the lowest percentage of accuracy , and by returning to table 5.2, level 3 is the level where the noise is low , the speaker sound is low and the quality of the images is also low.

3- On the other hand , level 1 has the highest accuracy value , where the noise level is low , the speaker sound is high and the quality of the images is also high.

4- The duration and the size of the video doesn't affect the accuracy of the system . From table 5.5 and table 5.7 , the video no. 19 with period = 51:43 minutes has an accuracy = 83% , and a video with a period = 37 minutes , has an accuracy = 93% . That is because we used avg-blocks algorithm in comparing images , which is fast and has less computational operations.

5- From experiments , if the noise level is near to the sound level (both of them low or both of them high) , the process of reducing noise becomes harder and it may cause some segments of speaker voice to be eliminated.

6- Our system , can be a way to reduce noise in addition to its primary task. And in some situations , when there is a video with music in background , the system can delete the music when choosing a high level for the noise threshold.

7- From our experiments , the reason where some black segments may occur in the output video refers to the fact that those segments are videos less than a second , where the player can't handle a video that is less than a second , so it plays it as a black segment.

8- From table 5.7 , the size of the video has a large effect on the processing time , while the duration has a slight affect , that is because videos with large sizes have a high frame rate and by the way needs a large amount of processing especially the process of comparing the sequential frames in order to detect inactive segments .

9- DISE has some drawbacks, that it may not give accurate results when the speaker voice is too low, and it may consider it as a noise.

10- Finally , there are many ways for evaluating, one can choose the way where he feels it is more suitable for his system . We choose precision and recall to evaluate our work , then we calculated accuracy and error rate.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This chapter is the last chapter in our thesis which presents our conclusion as a summary for all our work. Then present our future work.

6.1 Conclusion

Our approach is applied on lectures videos in e-learning systems in universities , that contains inactivity segments to extract them then delete them upon the user request . Our approach is summarized in three steps. 1) Pre-processing, 2) segments detection, and 3) Post-processing. Preprocessing contains the processes of extracting audio from video and then reducing noise. The next step is detecting the inactive segments by first detecting the silence segments and then comparing frames that lies in a silence segments. The post processing step is deleting the inactive segments according to the user decision. The segments are deleted and saved into an external folder for further watching. We used a blocks-avg algorithm in comparing frames with an error ratio depends on a threshold entered by a user.

We evaluate the proposed approach on a 24 video samples distributed on 8 types of videos according to the noise , speaker volume and image quality rate. Evaluation is done by calculating the overall precision 86.75%, recall 94.49%, accuracy 84.83% and error rate 15.16%.

We find some difficulties when processing videos with high noise and low speaker voice , or in other words , if the noise and the speaker voice are relatively in the same range. Also comparing frames was somehow difficult when the quality of the image is poor.

Videos with large sizes have a large effect on the processing time where increasing the size will highly increase the processing time and vice versa. While the duration time of the video haven't a large affect the processing time.

Finally, our system is optimized, easy to use, general to any domain area and able to deal with any video type or size. We expect the system to be used for a wide range of applications.

6.2 Future Work

- Detecting the presence of the teacher in front the camera and calculating the total time of it.
- Develop the system to deal with videos with high noise , by using an algorithm that can split speaker voice from noise and then reduce noise.
- Connecting the system to the Moodle , to upload videos after deleting the inactivity segments.
- Calculate the percentage of activeness to each lecture video , and then by connecting the system to the e-learning unit and the human resources system in the university , the total activeness degree to each lecturer can be calculated which can affect the percentage of the an annual assessment of the lecturers .

REFERENCES

[1] Richey C. "Reflections on the 2008 AECT Definitions of the Field". TechTrends volume 52, Issue 1 , pp 24-25 (2008).

[2] Kimberly C. , Kuanchin C. and David C. , " Distance Learning, Virtual Classrooms, And Teaching Pedagogy In The Internet Environment" , Technology in Society, (2004) .

[3] Swee K. , Alan S., Lay K. , "Impact of video recorded lectures among students", Proceedings of the 23rd annual ascilite conference(2008).

[4] Schwarz H. , Marpe D. and Wiegand T. "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard". IEEE Transactions on circuits and systems for video technology, vol. 17, (2007).

[5] Nave R. " Description Of Motion" . Georgia State University. January (2014).

[6] Wren C. , Azarbayejani A. , Darrel .T, Pentland and Pfinder A. " Real Time Tracking Of The Human Body" , IEEE Trans. Pattern Anal. Mach. Intell, pp780-785, (2007),

[7] Ching S. and Kamath C. , " Robust Background Subtraction with Foreground Validation for Urban Traffic Video" , EURASIP Journal on Applied Signal Processing, vol. 14, pp 1-11, (2005).

[8] Anderson C. , Burt P. and Van der Wal G. , " Change Detection And Tracking Using Pyramid Transformation Techniques" , Proceedings of SPIE – Intelligent Robots and Computer Vision, vol. 579, pp72-78, (2005),

[9] Lipton A., Fujiyoshi H. and Patil R.," Moving Target Classification And Tracking From Real- Time Video" , Proceedings of the 1998 DARPA Image Understanding Workshop.

[10] Mishra S. , Mishra P. , Chaudhary N. and Asthana P. , "A Novel Comprehensive Method For Real time Video Motion Detection Surveillance" , JSER, vol. 2, Issue 4 , April (2011).

[11] Yingyong Q. , and Hunt R. ," Voiced-Unvoiced-Silence Classifications of Speech Using Hybrid Features and a Network Classifier". 250 IEEE Transaction On Speech And Audio Processing, vol.1,no.2,april (1993).

[12] Hoelper C. , Frankort A. and Erdmann C. ," Voiced/Unvoiced/Silence Classification For Offline Speech Coding" , Proceedings of International Student Conference on Electrical Engineering (2003).

[13] Atal B. and Rabiner L. , “A Pattern Recognition Approach To Voiced-Unvoiced-Silence Classification With Applications To Speech Recognition” Acoustics, Speech, and Signal Processing , IEEE Transactions on , vol. 24 , pp: 201 – 212 , Jun (2006),

[14] Carrillo E. and Senevirathna B ,”Automatic Pitch Detection Using Speech Segmentation and Autocorrelation“, Montgomery College Student Journal of Science & Mathematics , A publication of the SEM Rockville, Montgomery College, (2004).

[15] Jochen B. ,Ulrich p. and Maciej O. , " Nonlinear Noise Reduction" , Proceedings of IEEE, vol. 90, no. 5, may(2002).

[16] K. Kausalya and S. Chitrakala. , “Idle Object Detection in Video for Banking ATM Applications “Easwari Engineering College, Anna University of Technology, Tamilnadu, India , Research Journal of Applied Sciences, Engineering and Technology,(2012).

[17] Cucchiara, R., Grana C. , Piccardi M. and Prati A. ,”Statistic and Knowledge-Based Moving Object Detection in Traffic Scenes”. Proceedings of Intelligent Transportation Systems , pp: 27-32 , (2000).

[18] Heisele B., Krebel U. and Ritter W. ,” Tracking Non-rigid, Moving Objects Based on Color Clusterflow “ , IEEE computer Society Conference on Computer Vision and Pattern Recognition, pp: 257 ,(1997).

[19] Ssu-Wei C. , Luke K., Hong Lan J., “Moving Object Tracking Based on Background Subtraction Combined Temporal Difference “ . International Conference on Emerging Trends in Computer and Image Processing (ICETCIP'2011) Bangkok Dec (2011).

[20] Shijie Z. , Fred S.,"Motion Detection Using a Model of Visual Attention" Department of Electronic and Electrical Engineering University College London.{j.zhang, f.stentiford}@adastral.ucl.ac.uk

[21] Veit T., Cao F. and Bouthemy P. , “Probabilistic parameter free motion detection,” in Proc. Of CVPR, Washington, DC, USA, vol. 1, pp. 715-721, June 27-July 2, (2004).

[22] Black M. and Jepson A. , “Estimating optical flow in segmented images using variable-order parametric models with local deformations,” IEEE Trans. On PAMI, vol. 18, Issue 10, pp.972-986, Oct.(1996).

[23] Kellner M. and Hanning T. , “Motion detection based on contour strings,” in Proc. Of ICIP, Singapore, vol. 4, pp. 2599-2602, Oct. (2004).

[24] Saha G . and Chakroborty S, " A New Silence Removal and Endpoint Detection Algorithm for Speech and Speaker Recognition Applications" , Department of Electronics and Electrical Communication Engineering Indian Institute of Technology, (2001).

[25] Jacobs S. , Eleftheriadis A. and Anastassiou D. , " Silence Detection For Multimedia Communication Systems " .Department of Electrical Engineering Columbia University New York NY. USA (1997).

[26] Rabiner L. and Sainbur M. , " Voiced-Unvoiced-Silence Detection Using the Itakura LPC Distance Measure " , Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '77. (Volume:2) pp : 323 – 326, May (1997).

[27] B. S. Atal and L. R. Rabiner, "A Pattern Recognition Approach to Voiced-Unvoiced-Silence Classification with Applications to Speech Recognition," IEEE Trans. On Acoustics, Speech, and Signal Processing, Vol. ASSP-24, No. 3, pp. 201-121, June(1996).

[28] McAulay R. , "Optimum Classification of Voiced Speech, Unvoiced Speech and Silence in the Presence of Noise and Interference," Lincoln Laboratory Technical Note 1976-7, June (1996).

[29] Manoj S., Madhuri A. and Ashok M. , " A Novel approach to Detect and Track Moving Object using Partitioning and Normalized Cross Correlation" , ICGST-GVIP Journal .ISSN : 1687-398x , Volume 9 . Issue 4, August (2009).

[30] Alawi M. , Khalifa O. and Islam M ," Performance Comparison of Background Estimation Algorithms for Detecting Moving Vehicle". World Applied Sciences Journal 21 ,Mathematical Applications in Engineering,(2013).

[31] Dinesh K. and Devi T , " Motion Detection and Object Tracking in Video frame sequence on IWT of Adaptive Pixel Based Prediction Coding". IRACST – Engineering Science and Technology: An International Journal (ESTIJ), ISSN: 2250-3498,Vol.2, No. 4, August (2012).

[32] Kogut, A., Lineweaver, C., and others , "Dipole Anisotropy in the COBE Differential Microwave Radiometers" . Astrophysical Journal 419 , (2003).

[33] DonVito M. and Schoenherr B. , "Subband Coding with Silence Detection" , Proceeding of the International Conference on Acoustics Speech and Signal Processing – ICASSP , pp 1433 – 1436 , (1998)

[34] Lynch J. , Josenhans L. and Crochiere R. ," Speech_Silence Segmentation for RealTime Coding via Rule Based Adaptive Endpoint Detection" , IEEE International Conference Acoustics Speech and Signal Processing _ICASSP, pp 1348 -1350, (2000).

[35] Un C. and Lee H. , " Voiced Unvoiced Silence Discrimination of Speech by Delta Modulation" , IEEE Transactions on Acoustics_ Speech_ and Signal Processing, pp 1254-1248, (1999)

[36] Abdul Sattar and Kang B., " AI 2006: Advances in Artificial Intelligence" , 19th Australian Joint Conference on Artificial Intelligence , pp 483 , (2006).

Appendix A: DISE Interface

The system interface from Figure A.1 to Figure A.9 is shown in this appendix with some explanations, Figure A.1 show the whole system interface.

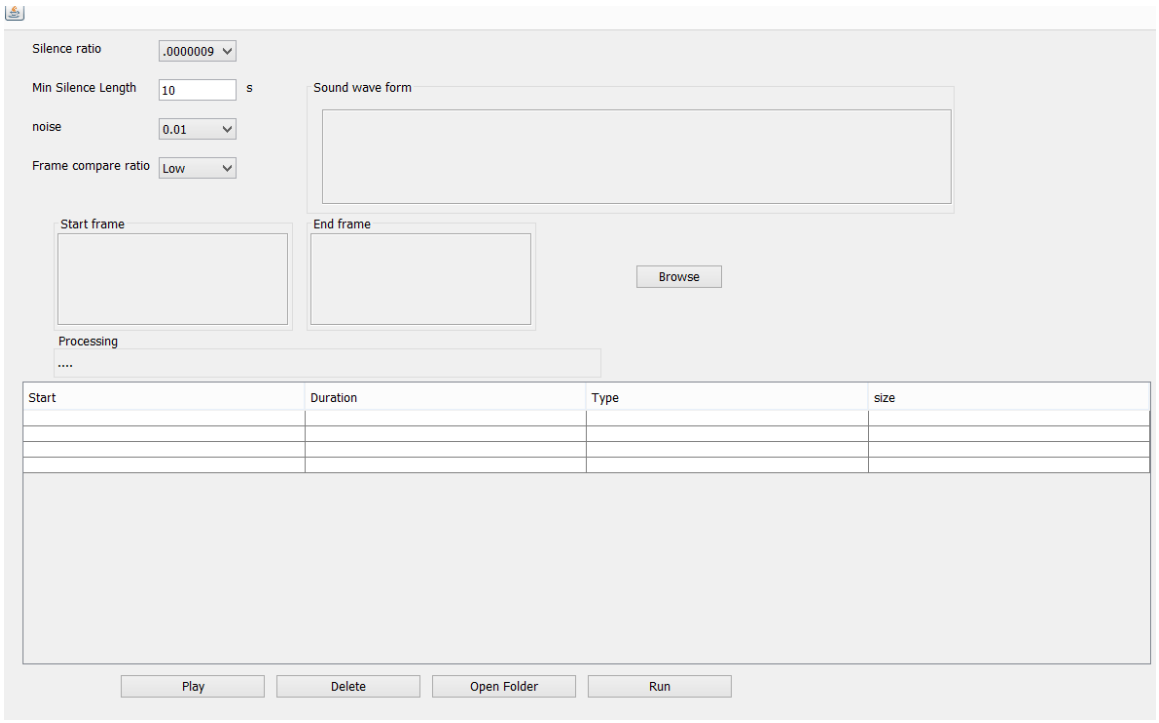

Figure 7.1: The system interface

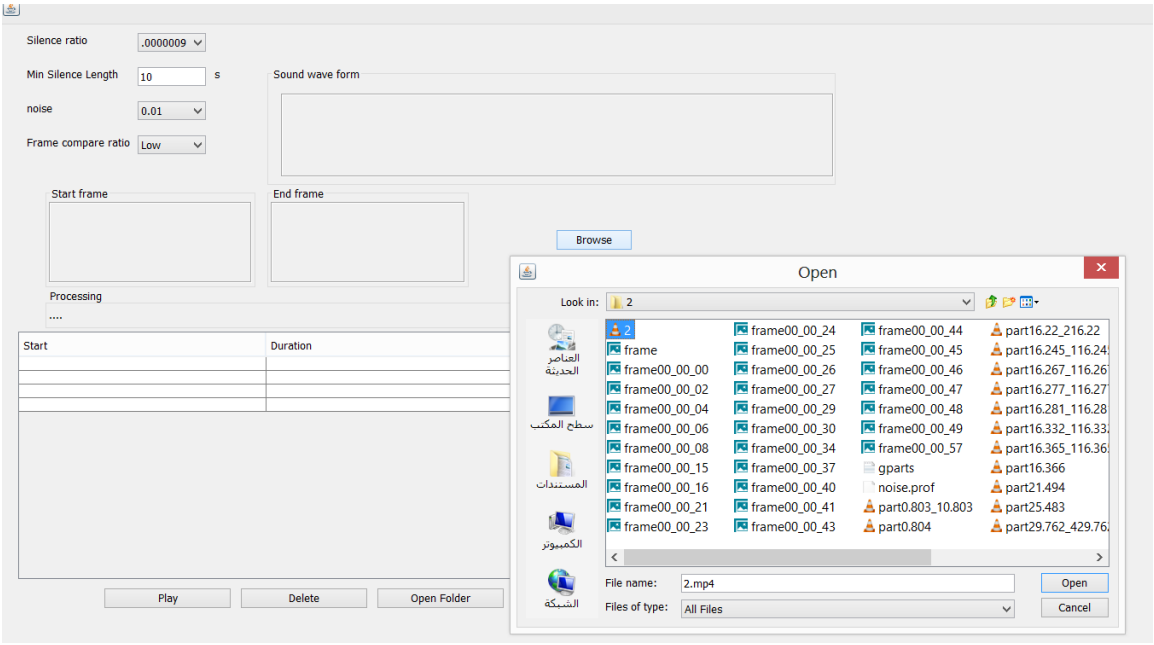

Figure 7.2 : Browsing the input video

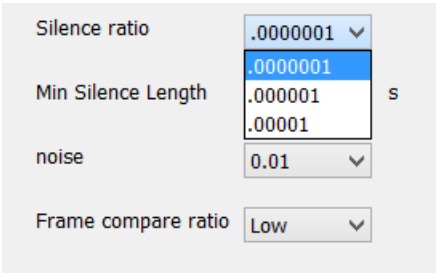

Figure 7.3 : Choosing the silence ratio

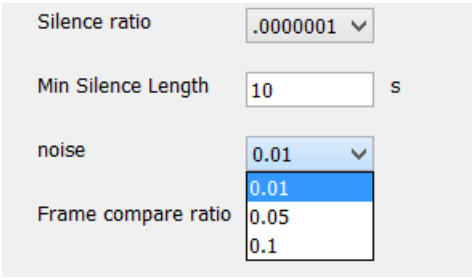

Figure 7.4: Choosing the noise ratio

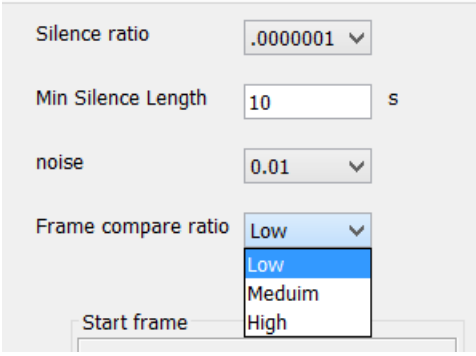

Figure 7.5 : Choosing the frames compare ratio

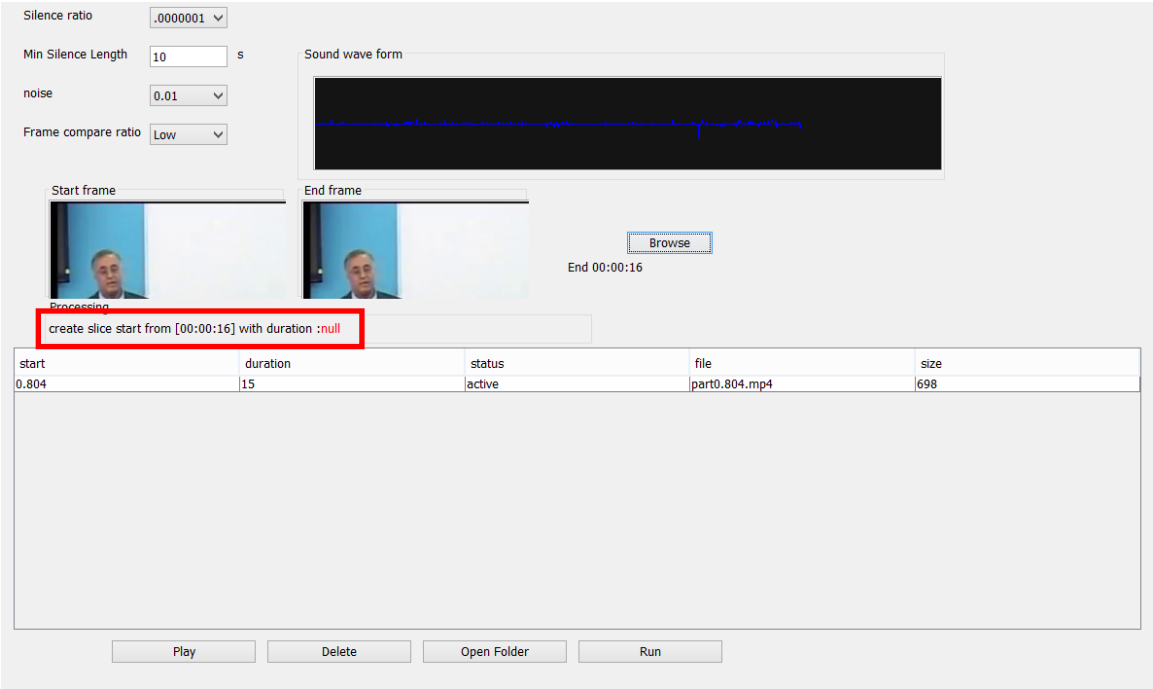

Figure A7.6 : The process which is actually running is showed in the rectangle , the start frame and the end frame are the frames which were compared at this moment .

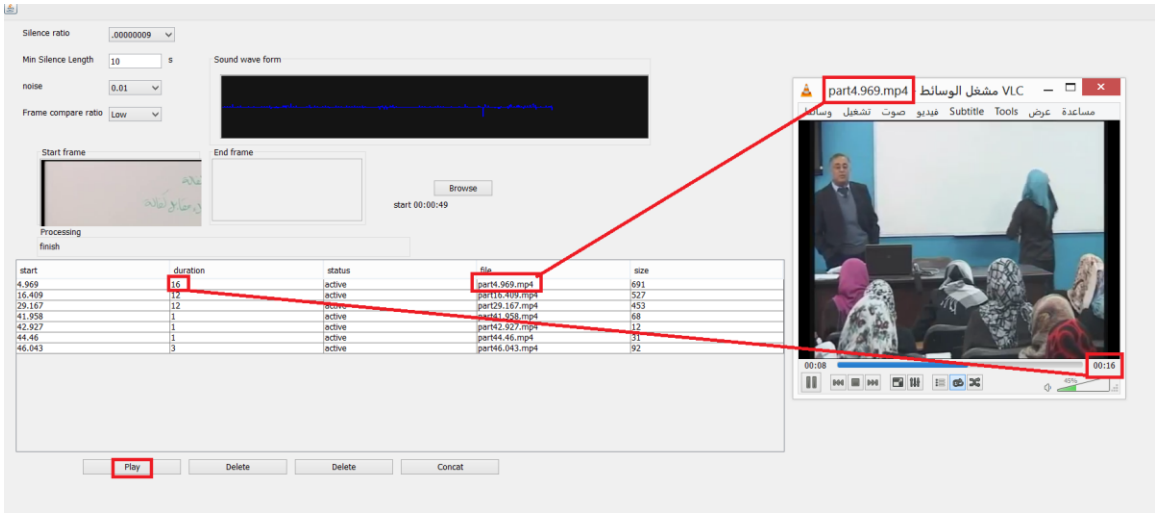

Figure A.7 : Playing a segment by the play bottom to take decision to delete or keep.

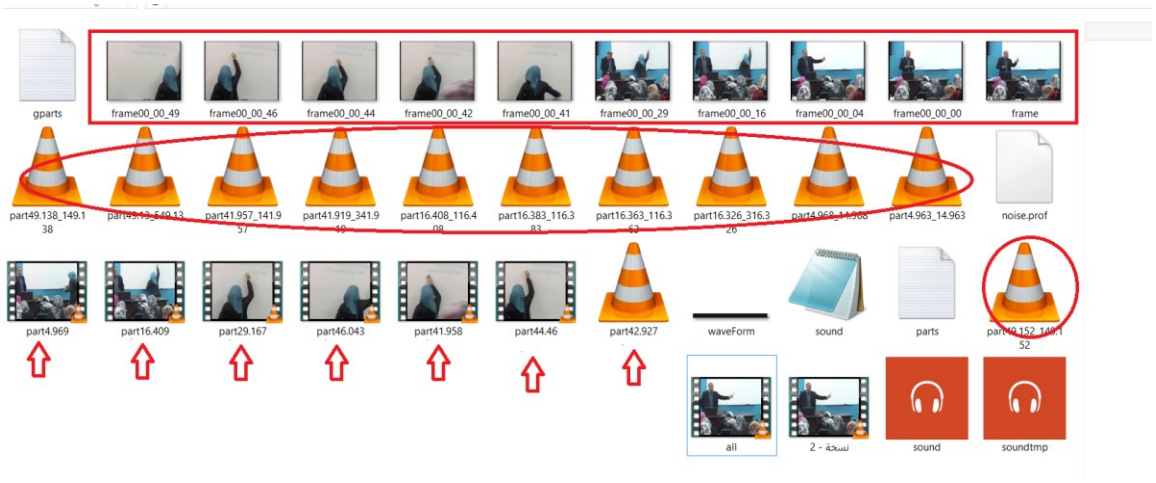

Figure A.8: The main folder

According to figure A.8 , the main folder contains the following:

1- Files in the rectangle are the start and end frames for each silence segment.
2- Files in the circles are the inactive segments ,files .
3- Files which are pointed by arrows are the active segments files .

Refers to the table in figure A.6 , the segments that are in the folder above are the same as shown in the table .

4- Video named (all) is the output video.

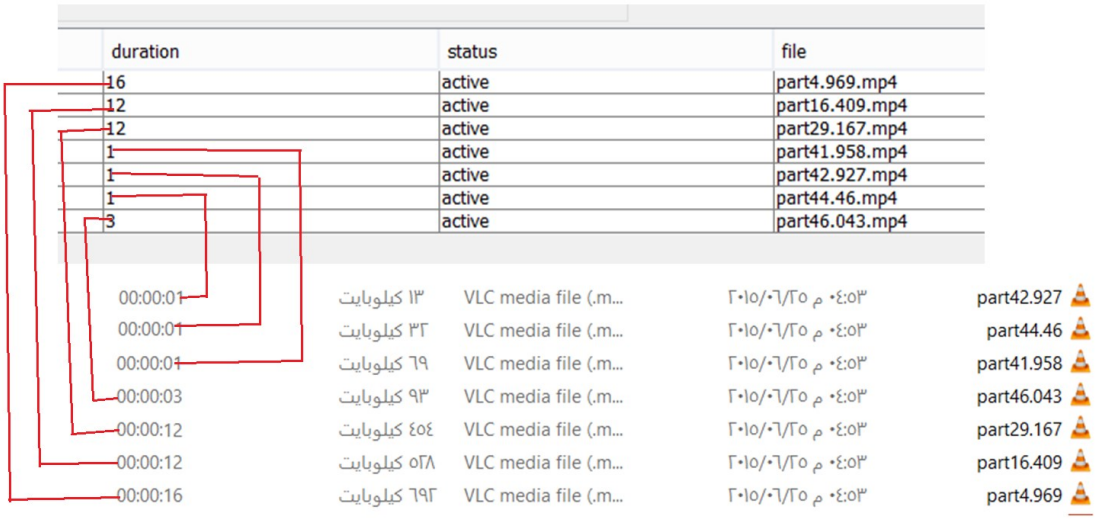

Figure A.9 : Matching between the table and the folder

Appendix B:About University Of Palestine

Palestine University is one of the Palestinian academic higher education institutions that was established to serve the Palestinian people in the inside and the outside in particular and the Arab students in general. This University carries on its shoulders a great message which is providing a high and modernistic level of the University education through creating an educational environment supported by an integral modern and electronic techniques with permissions and academic systems with world specifications , it also provides support and help for students to ensure a high level of creation and distinction and it is concerned of the scientific and knowledgeable research with the other civilizations and sciences of the world in order to stabilize the values of good citizenship, corporation and respecting others to accomplish the welfare and human happiness.

Palestine University was established by an approval of the late president Yasser Arafat Abu Ammar in 2003, and started to practice its role in the Palestinian society alongside with its components of the local Universities on the early of March 2005 at Gaza city, through preparing the files for the approval on the colleges and programs.

By the approval of his highness the president Mahmoud Abbas a 30 acres land has been allocated in Alzahraa city at the center of Gaza city, by the time the University got the land, it started to establish its buildings and settle in it. And after the continued work in preparing the University programs according to the needs of the higher education ministry and the national organization for approval, quality and variety of the higher education institutions, the University got the initial license then the general approval at 15/7/2007 to the final approval for its colleges which are:

1. Applied Engineering and Urban planning
2. Information Technology
3. Money and Business Administration
4. Law and Judicial Practice
5. Media and Communication
6. Education
7. Dental and Oral Surgery
8. Diploma

- **Vision and Objects**

Palestine University is a scientific institution that stands on the basis of integration and true partnership with its counterparts the educational institutions in the operation of building and developing the society of knowledge through the following:

Building and preparing generations of scientific innovation that are intellectually and technically qualified specifically in an educational and creative environment and highly qualified programs that combines between theoretical and applied sciences by using the information's techniques in its latest applications in a way that will work on transforming the university and students to a creative and producing person that is participating in the developing operation and its demands.

The University is seeking to be a scientific and cultural center and a scientific and civilized bridge between the Palestinian and Arab environment on the one hand and the International environment from the other hand through focusing and being interested to transforming and localizing the International Technology in collaboration with the International and original Universities and institution in order to serve the comprehensive and sustainable development.

The University is committed to all its academic, administrative and student elements by practicing and accomplishing the following civilized values:

1. The moral values for the Arab and Islamic civilization.
2. The values of the complete quality and distinction.
3. The values of collective work and hard work and responsibility.
4. Promoting critical, creative and modernistic thinking.
5. Promoting lifelong self-learning.
6. Promoting knowledge and special and professional creativity in the technology technique.

- **Message**

Palestine University is a Palestinian high education institution which aims to prepare cadres that are scientifically and professionally qualified and able to provide the society needs through preparing the universal environment according to the quality demands to keep up with the latest sciences and techniques and promoting the role of the scientific research and knowledgeable development to contribute in supporting the continuous development efforts. We also seek for contributing to draft the future map in the form of the original principles and values of our civilization.

- **Strategic Objects**

1. Promoting the organizational work in the University Administration to accomplish the University vision and its message.
2. Preparing the Universal environment, which satisfy the students expectations in order to reach the creative thinking and distinction levels.
3. Connect the University programs with the Market needs and develop it to suit these needs.
4. Applying the complete quality administration at the administrative, technical and financial services according to University standard.
5. Achieving the quality and variety standards for the Academic programs according to the local and International standards.
6. Caring for scientific research and keeping up with the latest technologies and using it in the educational and applied process to contribute building a creative and distinct student who contribute in the service of his society and his nation and directing him to serve the development cases and problems in the society, and to have the University as a bridge for transforming and localizing and strengthen the sciences and technologies and to promote the scientific and cultural openness.
7. Promoting the University's relation with its social and institutional environment and strengthen its links and its scientific and cultural relations.

Appendix C : Java Code

Tables from C.1 to C.7 shows the Java code that implements our system.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {


        final JFileChooser fileChooser = new JFileChooser();
        if (fileChooser.showOpenDialog(this) == JFileChooser.APPROVE_OPTION) {
           file = fileChooser.getSelectedFile();
           ext = fileChooser.getTypeDescription(file);
           System.err.println("-------------------\n" + ext +
"++++++++++++++++++++++++++++");
           try {

              new Thread() {
                 public void run() {
                    try {
                       removeParts();
                       stoped = false;
                       jLabel2.setText("Get Snapshot..");
                       Runtime.getRuntime().exec(ffmpegExe + " -ss 00:00:02 -i " + " " +
file.getAbsolutePath() + " -y -an -vframes 1 " + " " + file.getParentFile() + "/frame.png",
null, file.getParentFile()).waitFor();
                       jLabel1.setIcon(new ImageIcon(file.getParentFile() + "\\frame.png"));
                       jLabel2.setText("Extract sound file .. sound.wav");
                       System.err.println(ffmpegExe + " -i " + file.getAbsolutePath() + " -y -
vn -ar 44100 -ac 2 -ab 192k -f wav sound.wav");
                       Runtime.getRuntime().exec(ffmpegExe + " -i " + file.getAbsolutePath()
+ " -y -vn -ar 44100 -ac 2 -ab 192k -f wav sound.wav", null,
file.getParentFile()).waitFor();
                       /*
                        Choose a segment of the audio where there's no speech, only noise
(e.g. speaker was silent for a sec).
                        Generate a noise profile in sox:
                        $sox noiseaud.wav -n noiseprof noise.prof
                        */
                       jLabel2.setText("Generate noise profile (  )..");
                       System.err.println("Create noise profile");
                       // System.err.println(soxExe + " " + file.getParentFile() + "/sound.wav -
n noiseprof " + file.getParentFile() + "/noise.prof");
                       Runtime.getRuntime().exec(soxExe + " " + file.getParentFile() +
"/sound.wav -n noiseprof " + file.getParentFile() + "/noise.prof", null,
file.getParentFile()).waitFor();
                       /*
```

```
                    Clean the noise samples from the audio stream:
                    $sox tmpaud.wav tmpaud-clean.wav noisered noise.prof 0.21
                    */
                    jLabel2.setText("Clean the noise ( sensitivity " +
jComboBox1.getSelectedItem().toString() + " )..");
                    System.err.println("Clean the noise ( sensitivity " +
jComboBox1.getSelectedItem().toString() + "D )..");
                    Runtime.getRuntime().exec(soxExe + " " + file.getParentFile() +
"/sound.wav " + file.getParentFile() + "/soundtmp.wav noisered " + file.getParentFile() +
"/noise.prof " + jComboBox1.getSelectedItem().toString() "", null,
file.getParentFile()).waitFor();
                    jLabel2.setText("create sound table ");
                    Runtime.getRuntime().exec(soxExe + " " + file.getParentFile() +
"/soundtmp.wav -r 1000 " + file.getParentFile() + "/sound.dat", null,
file.getParentFile()).waitFor();
                    jLabel2.setText("clean file");
                    removeLineFromFile(file.getParentFile() + "/sound.dat", "");
                    jLabel2.setText("create Map");
                    Map<Integer, Double> map = new HashMap<Integer, Double>();
                    int c = 0;
                    for (Double i : startSignalList) {

                       map.put(c, i);
                       c++;           }
                    AudioWaveformCreator awc = new
AudioWaveformCreator(file.getParentFile() + "/soundtmp.wav", file.getParentFile() +
"/waveForm.png");
                    awc.createAudioInputStream();
                    jLabel4.setIcon(new ImageIcon(file.getParentFile() +
"/waveForm.png"));
                    jLabel2.setText("check sound table ");
                    datToString_Check(file.getParentFile() + "", map);

                    jLabel2.setText("create Graph");
                    //    JFrame frame = new JFrame("Test");
                    //   jPanel1.add(new JScrollPane(new Graph(map)));

                    jLabel2.setText("create video");

                    concatVideoParts();
                    jLabel2.setText("removing tmp Parts");
                    //      removeParts();
                    jLabel2.setText("finish");
                    stoped = false;
                } catch (Exception ex) {
                    //
Logger.getLogger(NewJFrame.class.getName()).log(Level.SEVERE, null, ex);
```

```
                }
            }
        }.start();
    } catch (Exception ex) {
        // Logger.getLogger(NewJFrame.class.getName()).log(Level.SEVERE, null,
ex); }}}
```

Table 9.1: Preprocessing the input video (remove noise and extract audio from video)

```
public void datToString_Check(String path, Map<Integer, Double> m) throws Exception
{
    File file = new File(new File(path), "sound.dat");
    System.err.println("" + file.getAbsolutePath());
    BufferedReader var2 = new BufferedReader(new InputStreamReader(new
FileInputStream(file)));

//      int datLen;
//      for (datLen = 0; var2.readLine() != null; ++datLen) {
//          ;
//      }
    BufferedReader var4 = new BufferedReader(new InputStreamReader(new
FileInputStream(file)));
    int var5 = 0;
    //    this.soundVolume = (String[][]) Array.newInstance(String.class, new int[]{var3,
5});
    Double cuurentSignal = new Double(0.0);
    String s[] = null;
    boolean isSilent = false;
    boolean flag = false;
    int silenceLen = 0;
    int soundLen = 0;
    String startSilence = "";
    String startSound = "";
    int counter = 0;
    int maxsilenceLen = 0;
    double limit = new Double(jTextField3.getText());
    parts = "";
    generatedParts = "";
    double vstart = 0d;
    String var6 = "";
    ImageCompare ic;
    DefaultTableMethod tableMethod;
    DefaultTableMethod method = (DefaultTableMethod) jTable1.getMethod();
```

```
        tableMethod = new DefaultTableMethod(new Object[]{"start", "duration", "status",
"file", "size"}, 0);
        jTable1.setMethod(tableMethod);
        long fileSizeInBytes = 0;

        long fileSizeInKB = 0;

        long fileSizeInMB = 0;
        File cf = null;
        method = (DefaultTableMethod) jTable1.getMethod();
        while (true) {

//          if (stoped) {
//              //  return;
//          }
            counter++;
            var6 = "";
            try {
               var6 = var4.readLine();
            } catch (Exception ex) {
               continue;
            }
            if (var6 == null) {
               //   System.err.println("maxsilenceLen=" + maxsilenceLen);
               var4.close();
               return;
            }
            s = var6.trim().split(" +");

            if (new BigDecimal(limit).compareTo(BigDecimal.valueOf(abs(new
Double(s[2])))) == 1) {
               //y s

               if (isSilent) {
                  isSilent = false;

                  startSound = s[0];

                  //         System.err.println(counter + ")silenceLen =[" + silenceLen + " ms]
start at time [" + startSilence + "] end at time [" + s[0] + "]");
                  flag = false;
                  endImage.setIcon(getSnapshot(millisToShortDHMS(counter)));
                  ic = new ImageCompare(iconToImage(startImage.getIcon()),
iconToImage(endImage.getIcon()));
                  // Set the comparison parameters.
                  //   (num vertical regions, num horizontal regions, sensitivity, stabilizer)
```

```
                ic.setParameters(9, 9, 10, 10);
                // Display some indication of the differences in the image.
                //   ic.setDebugMode(2);
                // Compare.
                ic.compare();

                // Display if these images are considered a match according to our
parameters.
                // System.out.println("Match: " + ic.match());
                jLabel10.setText("End " + millisToShortDHMS(counter));
                if (!ic.match()) {
                   slicePart(file, startSilence, silenceLen);
                   cf = new File(new File(file.getParent()), "part" + "" +
startSilence.replace(":", "_") + ".mp4");
                   System.err.println(file.getParent()+"part" + "" + startSilence.replace(":",
"_") + ".mp4");
                   fileSizeInBytes = cf.length();
                   System.err.println(fileSizeInBytes);
                   fileSizeInKB = fileSizeInBytes / 1024;
                   System.err.println(fileSizeInKB);
                   fileSizeInMB = fileSizeInKB / 1024;
                   System.err.println(fileSizeInMB);

                   try {
          method.addRow(new
Object[]{startSilence/*millisToShortDHMS(this.getSeconds(Long.valueOf(startSilence))
)*/,

silenceLen/1000/*millisToShortDHMS(this.getSeconds(Long.valueOf(silenceLen)))*/,
                         "active", "part" + "" + startSilence.replace(":", "_") + ".mp4",
fileSizeInKB});

                      jTable1.setMethod(method);
                                          //jTable1.setGridColor(Color.blue);
                                               //setSelectionForeground(Color.blue);
                                              // setSelectionBackground(Color.blue);

                   } catch (NumberFormatException numberFormatException) {
                      System.err.println("error:" + numberFormatException);
                   }

                   silenceLen = 0;
                   //   System.err.println("-------------------------------------------------" +
startSilence);
                   generatedParts += "file '" + file.getParentFile() + "/part" + (startSilence +
"")/*.replaceAll("\\.", "_") */ + ".mp4'|";
```

```
            }
            } else {

            }
            if (silenceLen > maxsilenceLen) {
               maxsilenceLen = silenceLen;
               //  System.err.println("maxsilenceLen=" + maxsilenceLen);
            }

            soundLen++;
            //    System.err.println("-Sound-->" + soundLen + " : " + s[0]);

         } else if (new BigDecimal(limit).compareTo(BigDecimal.valueOf(abs(new
Double(s[2])))) == -1) {
            //n s

            if (!isSilent) {
               startSilence = s[0];
               startImage.setIcon(getSnapshot(millisToShortDHMS(counter)));
               endImage.setIcon(new ImageIcon());
               jLabel10.setText("start " + millisToShortDHMS(counter));
               if (!flag) {
                  try {
                     if (silenceLen > (Integer.parseInt(jTextField2.getText()))) {
                        if (Double.valueOf(startSilence) > 0) {
                           try {
                              slicePartStr(file, startSound, soundLen + "", startSound + "_" +
soundLen);
                              cf = new File(new File(file.getParent()), "part" + startSound +
"_" + soundLen + startSilence.replace(":", "_") + ".mp4");
                              System.err.println(file.getParent()+"part" + "" +
startSilence.replace(":", "_") + ".mp4");
                              fileSizeInBytes = cf.length();
                              System.err.println(fileSizeInBytes);
                              fileSizeInKB = fileSizeInBytes / 1024;
                              System.err.println(fileSizeInKB);
                              fileSizeInMB = fileSizeInKB / 1024;
                              System.err.println(fileSizeInMB);
                              if (soundLen>5) {
                              method.addRow(new
Object[]{(millisToShortDHMS(this.getSeconds(Long.valueOf(startSilence))).intern()),

soundLen/1000/*millisToShortDHMS(this.getSeconds(Long.valueOf(silenceLen)))*/,
                                    "not active",
                                    "part" + startSound + "_" + soundLen +
startSilence.replace(":", "_") + ".mp4", fileSizeInKB});
```

```
                            jTable1.setMethod(method);
                            jTable1.setSelectionBackground(Color.red);
                                    }
                            } catch (NumberFormatException numberFormatException) {
                                System.err.println("error:" + numberFormatException);
                            }

                        }
                        soundLen = 0;
                    }
                } catch (Exception e) {
                }
                flag = true;
            }

            isSilent = true;
        }
        silenceLen++;

        //      System.err.println("-silent-->" + silenceLen + " : " + s[0]);
    }
    cuurentSignal = Double.valueOf(s[2]);
    m.put(var5, new Double(cuurentSignal));
    if (this.maxVol.compareTo(cuurentSignal) < 0) {
        maxVol = cuurentSignal;
    }
    if (this.minVol.compareTo(cuurentSignal) > 0) {
        minVol = cuurentSignal;
    }
    var5++;
  }

}

public int getSeconds(long millisec) {
  System.err.println("MS:--->" + millisec);
  Calendar c = Calendar.getInstance();
  c.setTimeInMillis(millisec);
  return c.get(Calendar.SECOND);

}

public Image iconToImage(Icon icon) {

  BufferedImage image = new BufferedImage(icon.getIconWidth(),
icon.getIconHeight(), BufferedImage.TYPE_INT_RGB);
  try {
```

```
            icon.paintIcon(new JPanel(), image.getGraphics(), 0, 0);
        } catch (Exception e) {
        }
        return image;
    }

    public void datToString(String path) throws Exception {
        File file = new File(new File(path), "sound.dat");
        BufferedReader var2 = new BufferedReader(new InputStreamReader(new
FileInputStream(file)));

        int len;
        for (len = 0; var2.readLine() != null; ++len) {
            ;
        }

        BufferedReader br = new BufferedReader(new InputStreamReader(new
FileInputStream(file)));
        int counter = 0;
        this.soundVolume = (String[][]) Array.newInstance(String.class, new int[]{len, 5});

        while (true) {
            String currentLine = br.readLine();
            //   System.err.println(var6);
            if (currentLine == null) {
                this.convertStringArraytoFloatArray();
                BigDecimal myNumber = new BigDecimal(this.soundVolume[0][2]);

                this.freq = myNumber.intValue();
                System.err.println(this.freq + " " + myNumber);
                this.duration = Float.valueOf((float) (len / freq));

                this.elementPartDurat = Float.valueOf(2.0F);
                //          this.partsAmount = Integer.valueOf((int) (this.duration.floatValue() /
this.elementPartDurat.floatValue()));
                return;
            }

            if (counter < this.soundVolume.length) {
                this.soundVolume[counter] = currentLine.trim().split("(; | +|;|\\s+)");
                //
System.err.println(this.soundVolume[var5][0]+"\t"+this.soundVolume[var5][1]+"\t"+this
.soundVolume[var5][2]);
            }
            ++counter;   } }
```

Table 9.2 : Detecting the silence segments

```
public void compare() {
        // setup change display image
        imgc = imageToBufferedImage(img2);
        Graphics2D gc = imgc.createGraphics();
        gc.setColor(Color.RED);
        // convert to gray images.
        img1 = imageToBufferedImage(GrayFilter.createDisabledImage(img1));
        img2 = imageToBufferedImage(GrayFilter.createDisabledImage(img2));
        // how big are each section
        int blocksx = (int) (img1.getWidth() / comparex);
        int blocksy = (int) (img1.getHeight() / comparey);
        // set to a match by default, if a change is found then flag non-match
        this.match = true;
        // loop through whole image and compare individual blocks of images
        for (int y = 0; y < comparey; y++) {
           if (debugMode > 0) {
              System.out.print("|");
           }
           for (int x = 0; x < comparex; x++) {
              int b1 = getAverageBrightness(img1.getSubimage(x * blocksx, y * blocksy,
blocksx - 1, blocksy - 1));
              int b2 = getAverageBrightness(img2.getSubimage(x * blocksx, y * blocksy,
blocksx - 1, blocksy - 1));
              int diff = Math.abs(b1 - b2);
              if (diff > factorA) { // the difference in a certain region has passed the threshold
value of factorA
                 // draw an indicator on the change image to show where change was
detected.
                 gc.drawRect(x * blocksx, y * blocksy, blocksx - 1, blocksy - 1);
                 this.match = false; }
              if (debugMode == 1) {
                 System.out.print((diff > factorA ? "X" : " "));
              }
              if (debugMode == 2) {
                 System.out.print(diff + (x < comparex - 1 ? "," : ""));
              }
           }
           if (debugMode > 0) {
              System.out.println("|");
           }    }
     }
```

Table 9.3 : Comparing frames

```
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            Pattern matcherPattern = Pattern.compile("'([^']*)'");
            Matcher matcher = matcherPattern.matcher(((CheckListItem1)
jList1.getMethod().getElementAt(jList1.getSelectedIndex())).toString() + "");
            String fname = "";
            while (matcher.find()) {
                fname = matcher.group();
            }
            System.err.println(fname);
            Desktop.getDesktop().open(new File(fname.replaceAll("'", "") + ""));
        } catch (IOException ex) {

        }
    }
```

Table 9.4: Play a segment

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String parts="";

        for (int i=0;i<jList1.getMethod().getSize();i++) {
            CheckListItem1 item=((CheckListItem1) jList1.getMethod().getElementAt(i));
            if(item.isSelected()){
             parts +=item.toString()+ "|";
            }
        }


        try {

            File file = new File(jf.file.getParent() + "/gparts.txt");

            // if file doesnt exists, then create it
            if (!file.exists()) {
                file.createNewFile();
            }

            FileWriter fw = new FileWriter(file.getAbsoluteFile());
            BufferedWriter bw = new BufferedWriter(fw);
            System.err.println(parts);
            bw.write(parts.replace("|", "\n\r"));
            bw.close();
```

```
            System.out.println("Done");

        } catch (IOException e) {
            e.printStackTrace();
        }
        try {
            Thread.sleep(3000);
        } catch (InterruptedException ex) {
            Logger.getLogger(NewJFrame1.class.getName()).log(Level.SEVERE, null, ex);
        }

        jf.doConcat(true);
    }
```

Table C.5 : Merge the segments

```
    private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            // TODO add your handling code here:

            Desktop.getDesktop().open(new File(this.fname).getParentFile());
        } catch (IOException ex) {
            Logger.getLogger(NewJFrame1.class.getName()).log(Level.SEVERE, null, ex);
        }
```

Table C.6 : Open The main directory

```
public String millisToShortDHMS(long duration) {
        String timeFormated = "";
        long days = TimeUnit.MILLISECONDS.toDays(duration);
        long hours = TimeUnit.MILLISECONDS.toHours(duration)
                - TimeUnit.DAYS.toHours(TimeUnit.MILLISECONDS.toDays(duration));
        long minutes = TimeUnit.MILLISECONDS.toMinutes(duration)
                -
TimeUnit.HOURS.toMinutes(TimeUnit.MILLISECONDS.toHours(duration));
        long seconds = TimeUnit.MILLISECONDS.toSeconds(duration)
                -
TimeUnit.MINUTES.toSeconds(TimeUnit.MILLISECONDS.toMinutes(duration));
        if (days == 0) {
            timeFormated = String.format("%02d:%02d:%02d", hours, minutes, seconds);
        } else {
```

```
            timeFormated = String.format("%dd%02d:%02d:%02d", days, hours, minutes,
seconds);
        }
        return timeFormated;
    }

    public void slicePart(File path, String start, int duration, String sname) {
        if (this.getSeconds(duration) < duLimit) {
            return;
        }
        System.err.println(getSeconds(duration));
        System.err.println("----------------------\n------------------------\ncreate slice start from
" + start + " with duration " + duration);
        System.err.println("start:" + start);
        System.err.println("duration:" + duration);
        try {
            jLabel2.setText("<html>create slice start from [<font color='grean'>" +
millisToShortDHMS((long) (new Double(start) * 1000)) + "</font>] with duration :<font
color='red'>" + duration + "</font></html>");
        } catch (Exception e) {
            System.err.println("Ex: double e");
        }
        try {
            //    Runtime.getRuntime().exec("ffmpeg -i part" + start + ".mp4 -filter:v
\"drawtext=fontsize=30:box=1:text='"+start+"-"+duration+"':x=(w-text_w)/2:y=(h-
text_h-line_h)/2\" part" + start + ".mp4");
            System.err.println(ffmpegExe + " -i " + file.getAbsolutePath() + "  -ss " + start + "
-t " + getSeconds(duration) + " -c copy part" + sname + start + ".mp4");
            Runtime.getRuntime().exec(ffmpegExe + " -i " + file.getAbsolutePath() + "  -ss "
+ start + " -t " + (duration / 1000) + " -c copy part" + sname + start.replace(":", "_") +
".mp4", null, path.getParentFile()).waitFor();
            parts += "file '" + file.getParentFile() + "/part" + (start + "")/*.replaceAll("\\.",
"_") */ + ".mp4'|";

        } catch (Exception ex) {
            System.err.println(ex);
        }

    }
```

Table 9.7 : Converting the time duration into the format(hh:mm:ss)

```
class Graph extends JPanel {

    protected static final int MIN_BAR_WIDTH = 8;
    private Map<Integer, Double> mapHistory;
```

```
    public Graph(Map<Integer, Double> mapHistory) {
        this.mapHistory = mapHistory;
        int width = (mapHistory.size() * MIN_BAR_WIDTH) + 11;
        Dimension minSize = new Dimension(width, 128);
        Dimension prefSize = new Dimension(width, 256);
        setMinimumSize(minSize);
        setPreferredSize(prefSize);
    }

    private Color colorFor(double value) {
        value = Math.max(0, Math.min(1, value));
        int red = (int) (value * 255);
        return new Color(red, 0, 0);
    }

    private double normalize(double min, double max, double value) {
        return (value - min) / (max - min);
    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        if (mapHistory != null) {
            int xOffset = 5;
            int yOffset = 5;
            int width = getWidth() - 1 - (xOffset * 2);
            int height = getHeight() - 1 - (yOffset * 2);
            Graphics2D g2d = (Graphics2D) g.create();
            g2d.setColor(Color.DARK_GRAY);
            g2d.drawRect(xOffset, yOffset, width, height * 2);
            int barWidth = 2/*Math.max(MIN_BAR_WIDTH, (int) Math.floor((float)
width / (float) mapHistory.size()))*/;
            int maxValue = 0;
            Double value = 0.0D;
            for (Integer key : mapHistory.keySet()) {
                value = mapHistory.get(key);
                maxValue = Math.max(maxValue, 1);
            }
            int xPos = xOffset;
            int counter = 1;
            long barHeight;
            int yPos;
            AffineTransform affinetransform = new AffineTransform();
            FontRenderContext frc = new FontRenderContext(affinetransform, true, true);
            int ms;
            Font f = new Font(Font.SERIF, Font.BOLD, 8);
            for (Integer key : mapHistory.keySet()) {
```

```
                value = mapHistory.get(key);
                barHeight = Math.round((value
                        / (float) maxValue) * height);

                g2d.setColor(colorFor(normalize(0, 220, value)));
                yPos = (int) (height + yOffset - barHeight);
                //Rectangle bar = new Rectangle(xPos, yPos, barWidth, barHeight);
                Rectangle2D bar = new Rectangle2D.Float(
                        xPos, yPos / 2, barWidth, barHeight);
                g2d.fill(bar);
                //  g2d.setColor(Color.DARK_GRAY);
                g2d.draw(bar);

                // g2d.setColor(Color.white);
                //  g2d.drawString(value + "", xPos, yPos - 5);
                AffineTransform org = g2d.getTransform();

                g2d.setFont(f);
                AffineTransform at = new AffineTransform();
                at.setToRotation(Math.toRadians(90), 10, 10);
                g2d.setTransform(at);

                if (-value.compareTo(Double.valueOf(Double.valueOf(0))) > 0) {
                   if (-
value.compareTo(Double.valueOf(jComboBox1.getSelectedItem().toString())) > 0) {
                      g2d.setColor(Color.white);
                   } else {

                      g2d.setColor(Color.blue);
                   }
                } else {

                   if
(value.compareTo(Double.valueOf(jComboBox1.getSelectedItem().toString())) < 0) {
                      g2d.setColor(Color.white);
                   } else {

                      g2d.setColor(Color.blue);
                   }

                }

//                if (value.compareTo(Double.valueOf(jTextField1.getText())) < 0) {
//                   if (value.compareTo(Double.valueOf(0)) < 0) {
//                      if (-value.compareTo(Double.valueOf(jTextField1.getText())) >0) {
//                         g2d.setColor(Color.white);
//                      } else {
```

```
//                          g2d.setColor(Color.blue);
//                       }
//                    } else {
//                       g2d.setColor(Color.blue);
//                    }
//                 } else {
//                    g2d.setColor(Color.blue);
//                 }
                ms = (int) (f.getStringBounds("" + value, frc).getWidth());
                g2d.drawString("" + BigDecimal.valueOf(abs(value)), yPos - ms - 10, -(xPos
- 16));
                //   g2d.drawString("Min:" + minVol, 20, -30);
                g2d.setTransform(org);

                xPos += barWidth;

                counter++;

                // Rotate 90 degree to make a vertical text
            }
            AffineTransform org = g2d.getTransform();
            g2d.setFont(new Font(Font.SERIF, Font.BOLD, 20));
            AffineTransform at = new AffineTransform();
            at.setToRotation(Math.toRadians(90), 10, 10);
            g2d.setTransform(at);
            g2d.setColor(Color.red);
            g2d.drawString("Max:" + maxVol, 20, 10);
            g2d.drawString("Min:" + minVol, 20, -30);
            g2d.setTransform(org);
            g2d.dispose();
        }
    }
}
```

Table 9: Methods used to prepare images before comparing