VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JNANA SANGAMA", BELGAUM – 590014

A Project Report on

"A Robust, Low Cost Method for Digitizing Classroom Experiences"

Submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Engineering

in

Information Science & Engineering

Submitted by:

KAVITHA G PURANIK	1PI12IS039
LAKSHMI V ANTIN	1PI12IS045
MANASA H G	1PI12IS052

Under the guidance of

Internal Guide

Dr. Viraj Kumar

Professor

Department of CSE

PESU

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

PES INSTITUTE OF TECHNOLOGY

100 Feet Ring Road, BSK 3rd Stage, Bengaluru – 560085

January 2016 – May 2016

PES INSTITUTE OF TECHNOLOGY

100 Feet Ring Road, B S K 3rd Stage,

Bengaluru-560085

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project work entitled **"A Robust, Low Cost Method for Digitizing Classroom Experiences"** carried out by **Kavitha G Puranik**, bearing USN **1PI12IS039**, **Lakshmi V Antin**, bearing USN **1PI12IS045**, **Manasa H G**, bearing USN **1PI12IS052**, are bonafide students of **PES INSTITUTE OF TECHNOLOGY**, Bangalore, an autonomous institute, under VTU, in partial fulfillment for the award of degree of **BACHELOR OF ENGINEERING IN INFORMATION SCIENCE & ENGINEERING** of **Visvesvaraya Technological University, Belgaum** during the year **2016**. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the above said degree.

Dr. Viraj Kumar	**Dr. Shylaja S S**	**Dr. K. S. Sridhar**
Professor	Professor and Head,	Principal
Department OF CSE	Department of ISE	PESIT
PESU	PESIT	

External Viva

Name of the Examiners	**Signature with Date**
1._____	_____
2._____	_____

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

P.E.S. Institute of Technology

Department of Information Science and Engineering

Bengaluru – 560085

DECLARATION

We, **Kavitha G Puranik, Lakshmi V Antin and Manasa H G**, students of Eighth Semester B.E., in the Department of Information Science and Engineering, **P.E.S. Institute of Technology, Bangalore** declare that the project entitled **"A Robust, Low Cost Method for Digitizing Classroom Experiences"** has been carried out by us and submitted in partial fulfillment of the course requirements for the award of degree of **Bachelor of Engineering** in **Information Science and Engineering of Visvesvaraya Technological University, Belagavi** during the Academic year **2015-16**. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

Name and USN	**Signature**
Kavitha G Puranik 1PI12IS039	
Lakshmi V Antin 1PI12IS045	
Manasa H G 1PI12IS052	

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible, and whose constant guidance and encouragement helped us in completing the project successfully. We consider it a privilege to express gratitude and respect to all those who guided us throughout the course of the completion of the project.

We extend our sincere thanks to **Dr. Viraj Kumar** for his invaluable suggestions, constant guidance, encouragement, support and invaluable advice without which this project would not have been completed.

We would like to express our heartfelt thanks to **Dr. Mamata H.R.**, Project coordinator, for coordinating the course of project work in a planned manner.

We express our gratitude to **Dr. Shylaja S. S**, Head of the Department, Information Science, PESIT, whose guidance and support has been invaluable and for including the project as part of the course.

We would like to express our heartfelt thanks to **Prof. M. R. Doreswamy**, PESIT founder, **Prof. D. Jawahar**, CEO and **Dr. K. S. Sridhar**, Principal, for providing us with a congenial environment for carrying out the project.

Last, but not the least, we would like to thank our friends whose invaluable feedback helped us to improve the application by leaps and bounds, and our parents for their unending encouragement and support.

Abstract

There are times when a course or professor is difficult to understand. In these cases and others, such as illness, teachers may want to record the lecture so that anyone can listen to it later to study for a paper or an exam. There are many advantages of recording the lecture. With video lectures students can learn anywhere from their mobile devices like laptops, tablets, Smartphone etc. They can also learn whenever they want to and also can learn at their individual pace. Video lectures can be used by teachers also, so that teacher can look at his/her presenting skills and find out what can be improved. Traditional way of recoding the lecture would require a professional camera that would capture the video of the lecture. In the above app, while the teacher is writing on the paper, the app captures her/his voice using microphone. After the teacher has written, he/she takes a single picture of the paper with the phone. The app now uses image processing techniques to segment the picture into lines and extract the contents written on the lines. The app animates the picture word by word (left to right per line) and plays the audio alongside. This will create **writing on a paper** experience. Instead of using a video, it will be just one image along with audio and control file.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

In the field of education video lectures are very popular. They are intuitive means of learning. With video lectures, students can learn anywhere from their mobile devices like laptops, tablets or Smartphone. But video lectures are normally very large in size (in hundreds of MBs) which makes it difficult for transferring the video file over social media. To watch the video, students have to download this huge video file and to send the video to students, teacher has to upload the video. Both teachers and students have to utilize a lot of bandwidth.

This project shows that how with just an image file, an audio file and a control file (which are just few kilobytes), and image processing techniques the classroom experience can be recreated which is not only advantageous for students but also for teachers. The compression ratio achieved by the app is significantly high.

The teacher takes a single image of the board/paper and records the audio of his/her lecture. Then this image and audio is given to the app which applies suitable image processing algorithms to animate the text written on the image and plays the audio alongside. The app has two parts one for students and other for teachers. Teacher's part of the app provides tools to sync the animation and audio and generates the control file which has this information and students' part uses all three image, audio and control file to do the animation.

Chapter 2

Problem Definition

Recording a video lecture in a class requires high quality expensive camera. The size of the video made will also be large. This video file has to be shared with students. As the video file will be too large, the video cannot be sent through social media like WhatsApp. The size of a video file that can be sent through WhatsApp is limited to 16MB. While capturing the video of teacher explaining the concept on paper, there would be disturbance like teachers hand coming in between the video, etc. A lot of internet data will also be required as large video has to be transferred by teacher and downloaded by the student. So this app aims at providing a robust and low-cost method for digitizing classroom experiences.

The app can be used to transfer classroom materials like video. The teacher just has to send the image of the paper where concept is explained, the audio file recorded during lecture and a control file. The student can receive all these files through WhatsApp as the size would be very less compared to that of video file explaining same concept. The app loads all the 3 files. It decodes the control file and animates the text written in image so that the speed of animation matches the speed of audio. This would create writing on a paper experience.

So this app aims at creating an effect similar to that of video using just an image, audio and control file. The app tries to attain as much compression as possible through this method.

Chapter 3

Literature Survey

3.1 Instructional video in e-learning: Assessing the impact of interactive video on learning effectiveness, Information Management, January 2006

Interactive video in an e-learning system allows proactive and random access to video content. Our empirical study examined the influence of interactive video on learning outcome and learner satisfaction in e-learning environments. Four different settings were studied: three were e-learning environments—with interactive video, with non-interactive video, and without video. The fourth was the traditional classroom environment. Results of the experiment showed that the value of video for learning effectiveness was contingent upon the provision of interactivity. Students in the e-learning environment that provided interactive video achieved significantly better learning performance and a higher level of learner satisfaction than those in other settings. However, students who used the e-learning environment that provided non-interactive video did not improve either. The findings suggest that it may be important to integrate interactive instructional video into e-learning systems.

3.2 The flipped classroom: A survey of the research

JL Bishop, MA Verleger - ASEE National Conference, 2013

Recent advances in technology and in ideology have unlocked entirely new directions for education research. Mounting pressure from increasing tuition costs and free, online course offerings is opening discussion and catalyzing change in the physical classroom. The flipped classroom is at the center of this discussion. The flipped classroom is a new pedagogical method, which employs asynchronous video lectures and practice problems as homework, and active, group-based problem solving activities in the classroom. It represents a unique combination of learning theories once thought to be incompatible—active, problem-based learning activities founded upon a constructivist ideology and instructional lectures derived from direct instruction methods founded upon behaviorist principles.

This paper provides a comprehensive survey of prior and ongoing research of the flipped classroom. Studies are characterized on several dimensions. Among others, these include the type of in-class and out-of-class activities, the measures used to evaluate the study, and methodological characteristics for each study. Results of this survey show that most studies conducted to date explore student perceptions and use single-group study designs. Reports of student perceptions of the flipped classroom are somewhat mixed, but are generally positive overall. Students tend to prefer in-person lectures to video lectures, but prefer interactive classroom activities over lectures. Anecdotal evidence suggests that student learning is improved for the flipped compared to traditional classroom. However, there is very little work investigating student learning outcomes objectively

3.3 Supported Media Formats in Android

3.3.1 Audio Formats

Table 3.1 Supported Audio Formats in Android

Format/ Codec	Details	Supported File Type(s) / Container Formats
AAC LC	Support for mono/stereo/5.0/5.1 content with standard sampling rates from 8 to 48 kHz.	• 3GPP (.3gp) • MPEG-4 (.mp4, .m4a) • ADTS raw AAC (.aac, decode in Android 3.1+, encode in Android 4.0+, ADIF not supported) • MPEG-TS (.ts, not seekable, Android 3.0+)
HE-AACv1 (AAC+)		
HE-AACv2 (enhanced AAC+)	Support for stereo/5.0/5.1 content with standard sampling rates from 8 to 48 kHz.	
AAC ELD (enhanced low delay AAC)	Support for mono/stereo content with standard sampling rates from 16 to 48 kHz	
AMR-NB	4.75 to 12.2 kbps sampled @ 8kHz	3GPP (.3gp)
AMR-WB	9 rates from 6.60 kbit/s to 23.85 kbit/s sampled @ 16kHz	3GPP (.3gp)
FLAC	Mono/Stereo (no multichannel). Sample rates up to 48 kHz (but up to 44.1 kHz is recommended on devices with 44.1 kHz output, as the 48 to 44.1 kHz	FLAC (.flac) only

	downsampler does not include a low-pass filter). 16-bit recommended; no dither applied for 24-bit.	
MP3	Mono/Stereo 8-320Kbps constant (CBR) or variable bit-rate (VBR)	MP3 (.mp3)
MIDI	MIDI Type 0 and 1. DLS Version 1 and 2. XMF and Mobile XMF. Support for ringtone formats RTTTL/RTX, OTA, and iMelody	• Type 0 and 1 (.mid, .xmf, .mxmf) • RTTTL/RTX (.rtttl, .rtx) • OTA (.ota) • iMelody (.imy)
Vorbis		• Ogg (.ogg) • Matroska (.mkv, Android 4.0+)
PCM/WAVE	8- and 16-bit linear PCM (rates up to limit of hardware). Sampling rates for raw PCM recordings at 8000, 16000 and 44100 Hz.	WAVE (.wav)
Opus		Matroska (.mkv)

3.3.2 Video Formats

Table 3.2 Supported Video Formats in Android

Format/ Codec	Details	Supported File Type(s) / Container Formats
H.263		• 3GPP (.3gp) • MPEG-4 (.mp4)
H.264 AVC	Baseline Profile (BP)	• 3GPP (.3gp) • MPEG-4 (.mp4) • MPEG-TS (.ts, AAC audio only, not seekable, Android 3.0+)
H.265 HEVC	Main Profile Level 3 for mobile devices and Main Profile Level 4.1 for Android TV	• MPEG-4 (.mp4)
MPEG-4 SP		3GPP (.3gp)
VP8	Streamable only in Android 4.0 and above	• WebM (.webm) • Matroska (.mkv, Android 4.0+)
VP9		• WebM (.webm) • Matroska (.mkv, Android 4.0+)

3.4 Introduction to Image Processing Library OPENCV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV provides a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products.

OpenCV is written natively in C++ but it has C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available.

3.4.1 Mat - The Image Container

Mat is a class which represents image. It has two data parts the matrix header (containing information such as the size of the matrix, the method used for storing, at which address is the matrix stored, and so on) and a pointer to the matrix containing the pixel values (taking any dimensionality depending on the method chosen for storing) . OpenCV uses a reference counting system. The idea is that each *Mat* object has its own header, however the matrix may be shared between two instances of them by having their matrix pointers point to the same address. Moreover, the copy operators will only copy the headers and the pointer to the large matrix, not the data itself.

3.4.2 Adaptive Thresholding

Image thresholding segments a digital image based on a certain characteristic of the pixels. The goal is to create a binary representation of the image, classifying each pixel into one of two categories, such as “dark” or “light”. Adaptive thresholding is a form of thresholding that takes into account spatial variations in illumination.

The function transforms a grayscale image to a binary image according to the formulae:

- THRESH_BINARY

$$\texttt{dst}(x,y) = \begin{cases} \texttt{maxValue} & \text{if } \texttt{src}(x,y) > T(x,y) \\ 0 & \text{otherwise} \end{cases}$$

- THRESH_BINARY_INV

$$\texttt{dst}(x,y) = \begin{cases} 0 & \text{if } \texttt{src}(x,y) > T(x,y) \\ \texttt{maxValue} & \text{otherwise} \end{cases}$$

where $T(x,y)$ is a threshold calculated individually for each pixel.

3.4.3 Morphological Operations

A set of operations that process images based on shapes. Morphological operations apply a *structuring element* to an input image and generate an output image.

The most basic morphological operations are two: Erosion and Dilation. They have a wide array of uses, i.e.:

- Removing noise
- Isolation of individual elements and joining disparate elements in an image.
- Finding of intensity bumps or holes in an image

Dilation:

- This operation consists of convoluting an image A with some kernel B, which can have any shape or size, usually a square or circle.
- The kernel B has a defined *anchor point*, usually being the center of the kernel.
- As the kernel B is scanned over the image, we compute the maximal pixel value overlapped by B and replace the image pixel in the anchor point position with that maximal value. As you can deduce, this maximizing operation causes bright regions within an image to "grow" (therefore the name *dilation*).

Erosion:

- This operation is the sister of dilation. What this does is to compute a local minimum over the area of the kernel.
- As the kernel B is scanned over the image A, we compute the minimal pixel value overlapped by B and replace the image pixel under the anchor point with that minimal value.

3.5 Async Task in Android

AsyncTask enables proper and easy use of the UI thread. This class allows performing background operations and publishing results on the UI thread without having to manipulate threads and/or handlers.

AsyncTask is designed to be helper class around Thread and Handler and does not constitute a generic threading framework. AsyncTask should ideally be used for short operations (a few seconds at the most.) If it is needed to keep threads running for long periods of time, it is highly recommended to use the various API's provided by the java.util.concurrent package such as Executor, ThreadPoolExecuter and Future Task.

An asynchronous task is defined by a computation that runs on a background thread and whose result is published on the UI thread. An asynchronous task is defined by 3 generic types, called
Params, Progress and Result, and 4 steps, called onPreExecute, doInBackground, onProgressUpdate and onPostExecute.

Async Task's generic types
The three types used by a synchronous task are the following

1. Params, the type of the parameters sent to the task upon execution.

2. Progress, the type of the progress units published during the background computation.

3. Result, the type of the result of the background computation.

Not all types are always used by an asynchronous task. To mark a type as unused, simply use the type Void.

private class MyTask extends AsyncTask<Void, Void, Void> { ... }

When an asynchronous task is executed, the task goes through 4 steps:

1. onPreExecute(), invoked on the UI thread before the task is executed. This step is normally used to setup the task, for instance by showing a progress bar in the user interface.

2. doInBackground(Params…), invoked on the background thread immediately after onPreExecute() finishes executing. This step is used to perform background computation that can take a long time. The parameters of the asynchronous task are passed to this step. The result of the computation must be returned by this step and will be passed back to the last step. This step can also use publishProgress(Progress...) to publish one or more units of progress. These values are published on the UI thread, in the onProgressUpdate(Progress...) step.

3. onProgressUpdate(Progress...), invoked on the UI thread after a call to publishProgress(Progress...). The timing of the execution is undefined. This method is used to display any form of progress in the user interface while the background computation is still executing. For instance, it can be used to animate a progress bar or show logs in a text field.

4. onPostExecute(Result), invoked on the UI thread after the background computation finishes. The result of the background computation is passed to this step as a parameter.

3.6 Importing OpenCV to Android Studio

Android Studio is the official IDE for Android app development, based on IntelliJ IDEA.

1. Create a new Android Studio project using the project wizard (Menu:/File/New Project):
    - Call it "*cvtest1*"
    - Form factor: *API 19, Android 4.4 (KitKat)*
    - *Blank Activity* named *MainActivity*
2. Verify that app runs correctly.
3. Download the OpenCV package for Android v3.1.0 and unzip it in some temporary directory somewhere. (Make sure it is the package specifically for Android and not just the OpenCV for Java package.) let the directory be "*unzip-dir*"
4. From Android Studio import OpenCV into the project as a module: Menu:*/File/New/Import_Module*:
    - Source-directory: *{unzip-dir}/sdk/java*
    - Module name: Android studio automatically fills in this field with *openCVLibrary310*
    - Click on *next*. A screen with three checkboxes and questions about jars, libraries and import options. All three should be checked. Click on *Finish.*
5. Open the project structure dialogue (Menu:/File/Project_Structure). Select the "app" module, click on the Dependencies tab and add: openCVLibrary310 as a module dependency. After selecting Add/Module_Dependency it should appear in the list of modules you can add. It will now show up as a dependency but you will get a few more cannot-find-android-14 errors in the event log.
6. Look in the build.gradle file for app module. There are multiple build.gradle files in an Android project. It looks like build.gradle (Module: app). Note the values of these four fields:
    - compileSDKVersion
    - buildToolsVersion
    - minSdkVersion
    - targetSdkVersion

7. Copy the *{unzip-dir}/sdk/native/libs* directory (and everything under it) to Android project, to *cvtest1/openCVLibrary/src/main/*, and then rename copy from *libs* to *jniLibs*. Resynch your project and this directory should now appear in the project view under **openCVLibrary31**

Chapter 4

Software Requirement Specification

4.1 Hardware Requirements

Recommended

- 2 GB RAM
- 80 GB hard disk space
- 1.5 GHz Pentium Dual Core processor
- Android Phone with Kitkat OS(API 19) , microphone and 8 MP camera should be present in the phone

4.2 Software Requirements

Recommended

- Operating System : Windows 7 or 8
- Jdk1.7.0_51
- Android Studio 1.5.1
- OpenCv 3.1.0

4.3 Functional Requirements

- The app must provide a means to browse the file system, to select the required image, audio along with its control file.
- Once the required folder is selected, the app should be navigated to a separate window.
- In the second window, the app should provide an interface to display the image being animated.
- The second window should also have start, pause/play, and stop buttons.
- On clicking the start button the image along with its respective audio and control file should get animated.
- On clicking the pause/play button the animation of the image should be paused or resumed. Care should be taken to pause/play the audio when the animation is paused/resumed.

- On clicking the stop button the entire animation along with its audio should be stopped.
- Validations like clicking the start button while the image is still being animated or clicking the stop button while the animation is paused should be taken care.

4.4 Non-functional Requirements.

4.4.1 Performance

The overall performance of the app should be good. There should be very little delay between accepting the files and animating them on students' version of the app.

4.4.2 User Interfaces

The interface should be easy to understand. The teacher will be given a user manual to understand how to generate the control file using the GUI given to the teachers.

4.4.3 Reliability

The app should be reliable by teachers so that they can use it in place of videos.

4.3.4 Dependencies

The Android phone's camera used by teacher should be functioning properly. The microphone should be working properly. The camera should be working properly.

4.3.5 Assumptions

The Teacher writes the JSON file correctly.

Chapter 5

GANTT chart

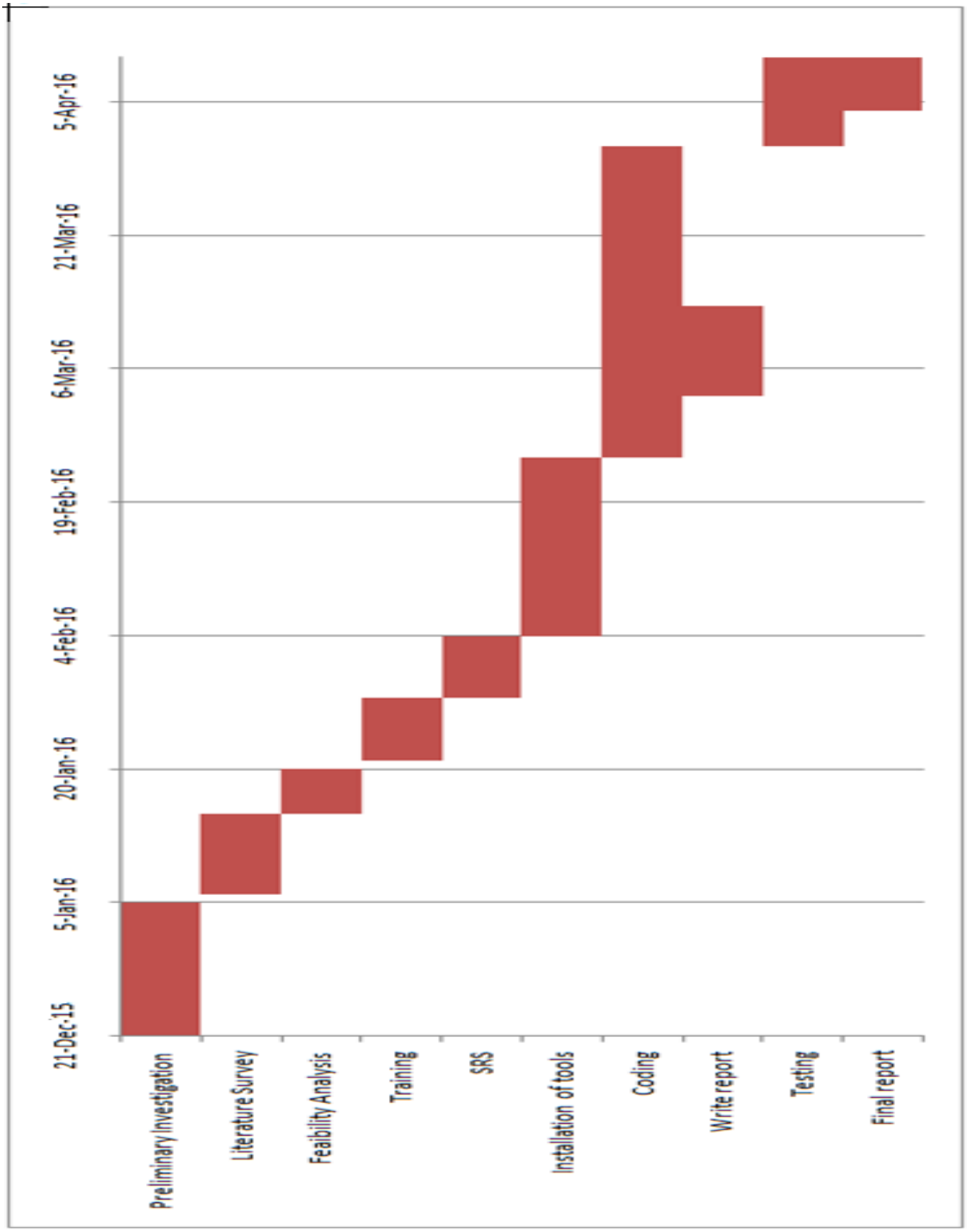

Figure 5.1 Gantt chart

Chapter 6

System Design

6.1 System Architecture

The Student receives the Image, audio and control file from the teacher through some media. These 3 files will be in a single folder. The App allows the student to pick the folder containing these 3 files. Then App animates the image along with audio by using the control file with adjusted speeds.

The following diagram explains the architecture

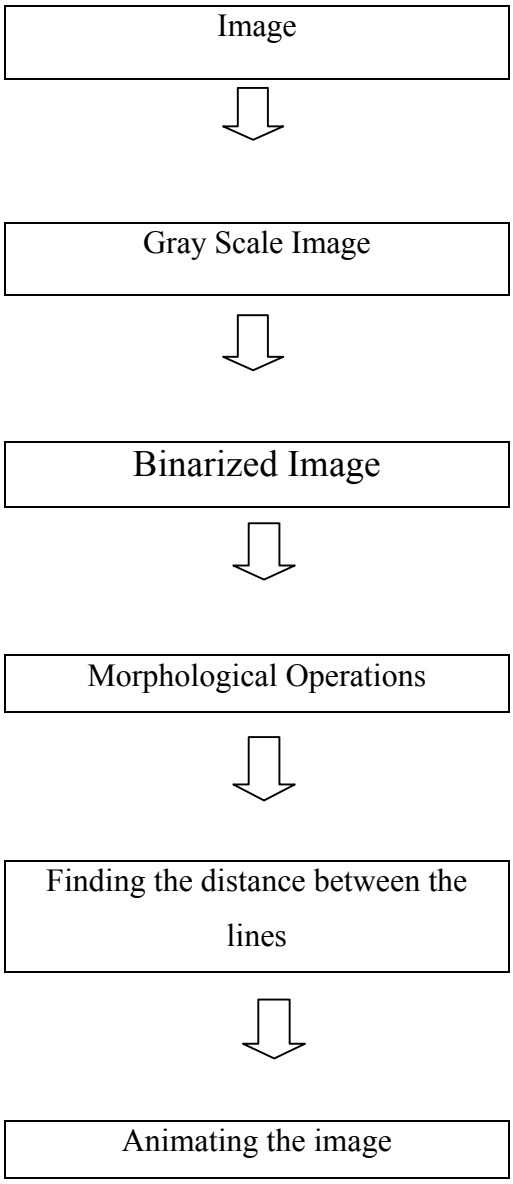

Figure 6.1 System Architecture

6.2 Data Flow Diagram

6.2.1 Context-level Data Flow Diagram

The context-level data flow diagram shows the interaction between the system and external agents which act as data sources and data sinks. On the context diagram the system's interactions with the outside world are modeled purely in terms of data flow across the system boundary. The context diagram shows the entire system as a single process, and gives no clues as to its internal organization.

The data flow diagram is represented up to level 2

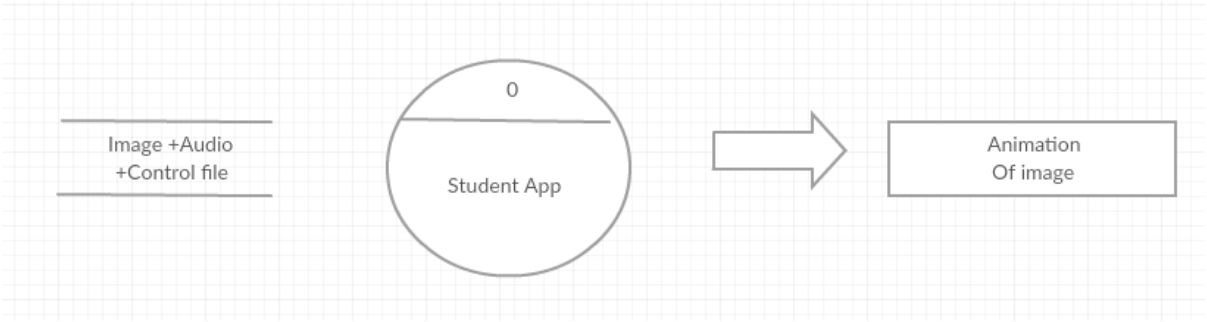

Figure 6.2 Contextual Level Data Flow Diagram

The figure shows the context level data flow diagram. The input will be Image, audio and a control file. When these are given as an input to the app, the does the processing and animates the image.

LEVEL 1:

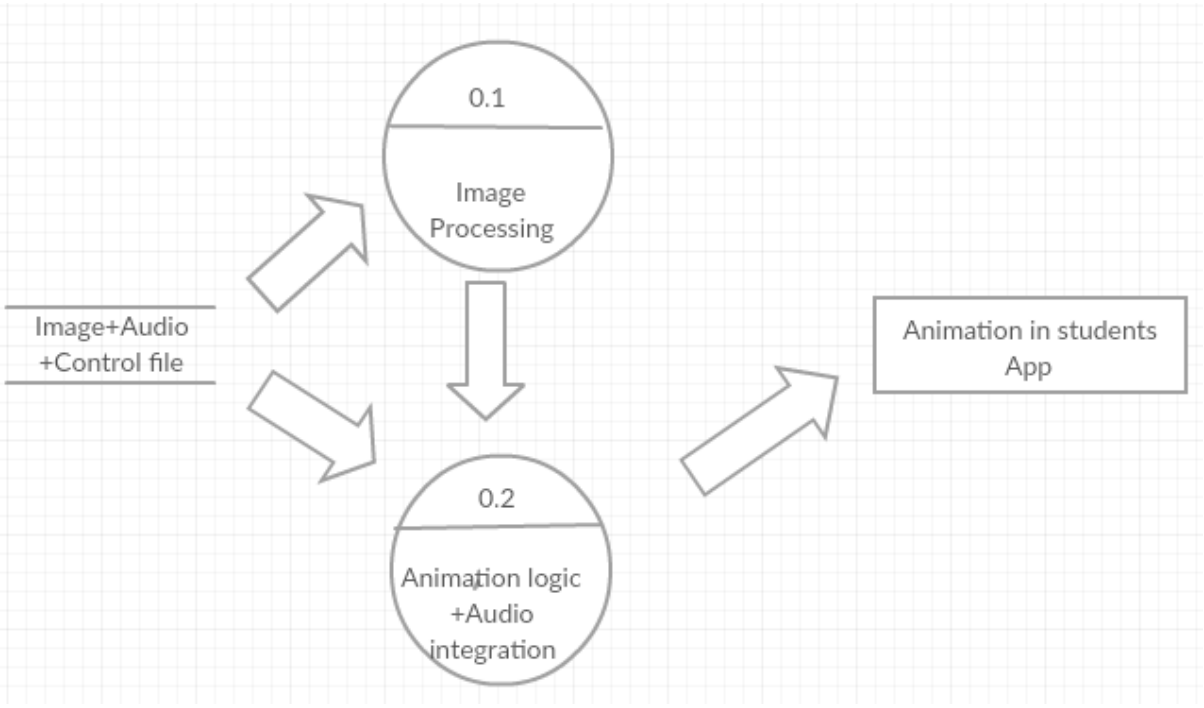

Figure 6.3 Level 1 Dataflow Diagram

This is an extension of data flow diagram at level 0.Here interaction between the modules can be seen. The image processing techniques are applied to image to obtain the text from the image. Then animation logic along with audio file combined together gives the effect of video.

LEVEL 2:

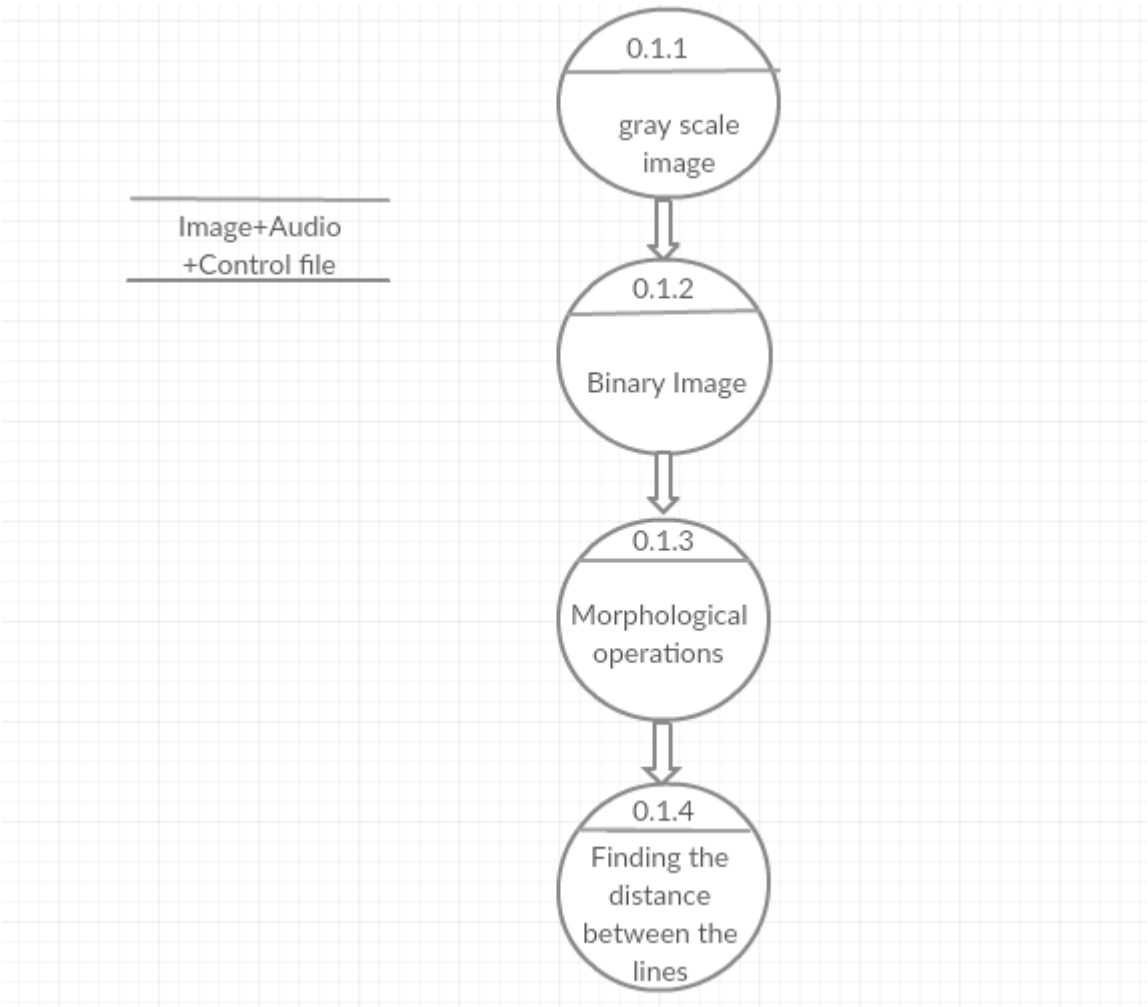

Figure 6.4 Level 2 Dataflow Diagram (a)

Here is a more detailed data flow diagram. The image to be processed is first converted into a gray scale image. The gray scale image is binarized using the inverted adaptive threshold method. Here the threshold value is specified. Pixels above the threshold value are assigned the value of black pixel and pixel below the threshold value are assigned the value of white pixel. Once this is done, the binarized image is subjected to a set of morphological operations. Morphological operations like erode and dilate are used. These set of operations gives out all the horizontal lines in the image. Once the horizontal lines are detected, distance between the consecutive lines is found out. This gives the gap between the lines.

LEVEL 2:

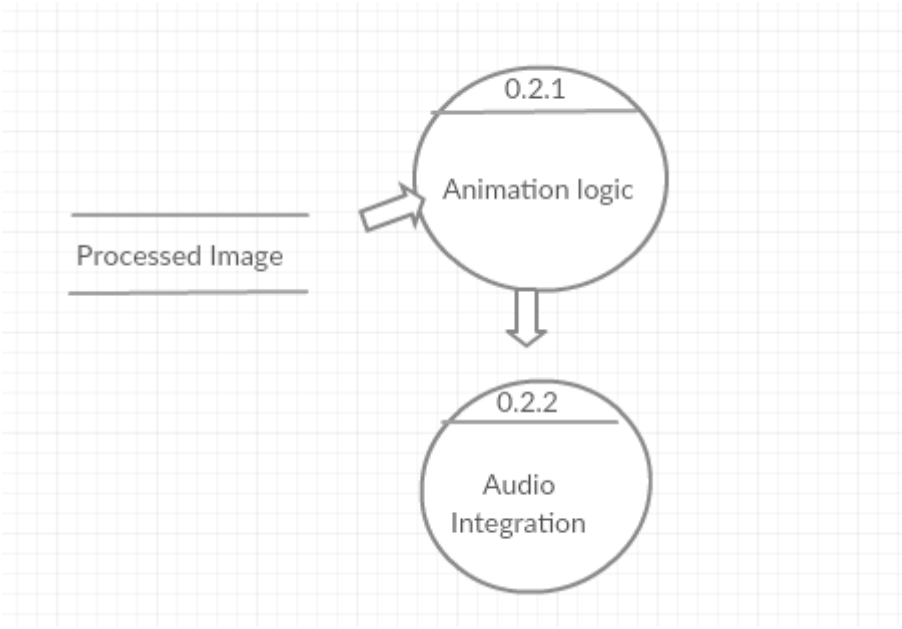

Figure 6.5 Level 2 Dataflow Diagram (b)

In Figure 6.5, the gap between the successive lines is found. This is fed to the algorithm which animates each line column wise. The animated image is integrated with the audio. Once the animated image is synced with the audio, speed of each line that is being animated can be controlled by means of a control file.

Chapter 7

Implementation

7.1 Keywords

In OpenCV image is stored using Mat object. *Mat* is basically a class with two data parts: the matrix header (containing information such as the size of the matrix, the method used for storing, at which address is the matrix stored, and so on) and a pointer to the matrix containing the pixel values (taking any dimensionality depending on the method chosen for storing). The matrix header size is constant, however the size of the matrix itself may vary from image to image and usually is larger by orders of magnitude.

OpenCV is an image processing library. It contains a large collection of image processing functions. To solve a computational challenge, most of the time you will end up using multiple functions of the library. Because of this, passing images to functions is a common practice. We should not forget that we are talking about image processing algorithms, which tend to be quite computationally heavy. The last thing we want to do is further decrease the speed of your program by making unnecessary copies of potentially *large* images.

To tackle this issue OpenCV uses a reference counting system. The idea is that each *Mat* object has its own header, however the matrix may be shared between two instances of them by having their matrix pointers point to the same address. Moreover, the copy operators **will only copy the headers** and the pointer to the large matrix, not the data itself.

Example: a.

Mat src = Imgcodecs.imread(pathtoimage);

Here Imgcodecs is the Opencv 3.0 library which contains the image processing functions.

Example: b

```
Mat gray = new Mat(src.size(), CvType.CV_8U);
```

Here src is the input image, and gray is a Mat object with same size as src. CvType.CV_8U specifies the type of the image and the number of channels in the image.

7.2 Gray scale image

Grayscale is a range of shades of gray without apparent color. The darkest possible shade is black, which is the total absence of transmitted or reflected light. The lightest possible shade is white, the total transmission or reflection of light at all visible wavelength s. Intermediate shades of gray are represented by equal brightness levels of the three primary colors (red, green and blue) for transmitted light, or equal amounts of the three primary pigments (cyan, magenta and yellow) for reflected light.

In the case of transmitted light (for example, the image on a computer display), the brightness levels of the red (R), green (G) and blue (B) components are each represented as a number from decimal 0 to 255, or binary 00000000 to 11111111. For every pixel in a red-green-blue (RGB) grayscale image, R = G = B. The lightness of the gray is directly proportional to the number representing the brightness levels of the primary colors. Black is represented by R = G = B = 0 or R = G = B = 00000000, and white is represented by R = G = B = 255 or R = G = B = 11111111. Because there are 8 bit s in the binary representation of the gray level, this imaging method is called 8-bit grayscale.

Since all image processing techniques are applied on the gray scale image, the given image is converted into gray scale image .This is done using the using one of the function call in opencv library.

Imgproc.cvtColor(src, des, Imgproc.COLOR_BGR2GRAY);

Here

-src is the source image

-des is the destination gray scale image

- Imgproc.COLOR_BGR2GRAY specifies that BGR image must be converted to Gray scale with 0-255 channels

The output would look like:

The output is the screen shot of the android app.

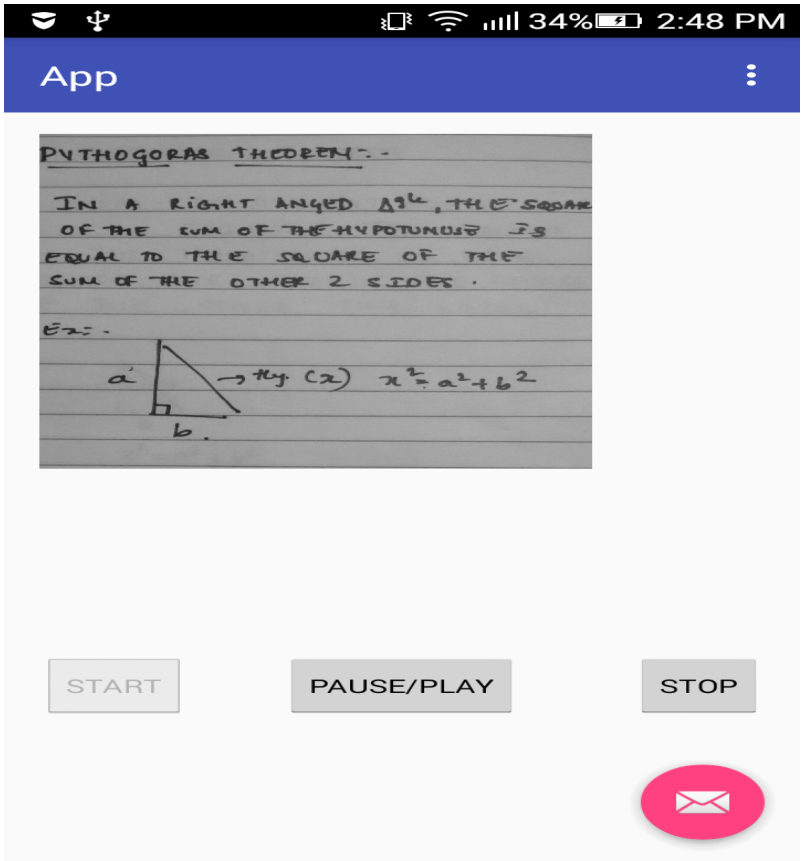

Figure 7.1 Gray Scale of the Image

7.3 Binary Image

The Gray scale image with pixel range 0-255 is converted to binary image by applying adaptive threshold .The pixels with values above the threshold value are assigned with pixel value 255(brighter pixel) and pixel below the threshold value are assigned 0(dark pixels).In this way
The contents of the paper are clearly visible.

The project uses inverted adaptive threshold method, where pixels above the threshold values are assigned with pixel value 0 and pixels below the pixel value are assigned with pixel value 255.This is done because the paper is generally white in color and the text is in black, blue etc. The function call looks like this

```
Imgproc.adaptiveThreshold(src, des, 255,
        Imgproc.ADAPTIVE_THRESH_MEAN_C,
        Imgproc.THRESH_BINARY_INV, 15, -2);
```

Here

src- is the source image

des- is the destination image

Imgproc.ADAPTIVE_THRESH_MEAN_C- threshold value is the mean of neighbourhood area.

Imgproc.THRESH_BINARY_INV- Inverted binary threshold

The Output image looks like this:

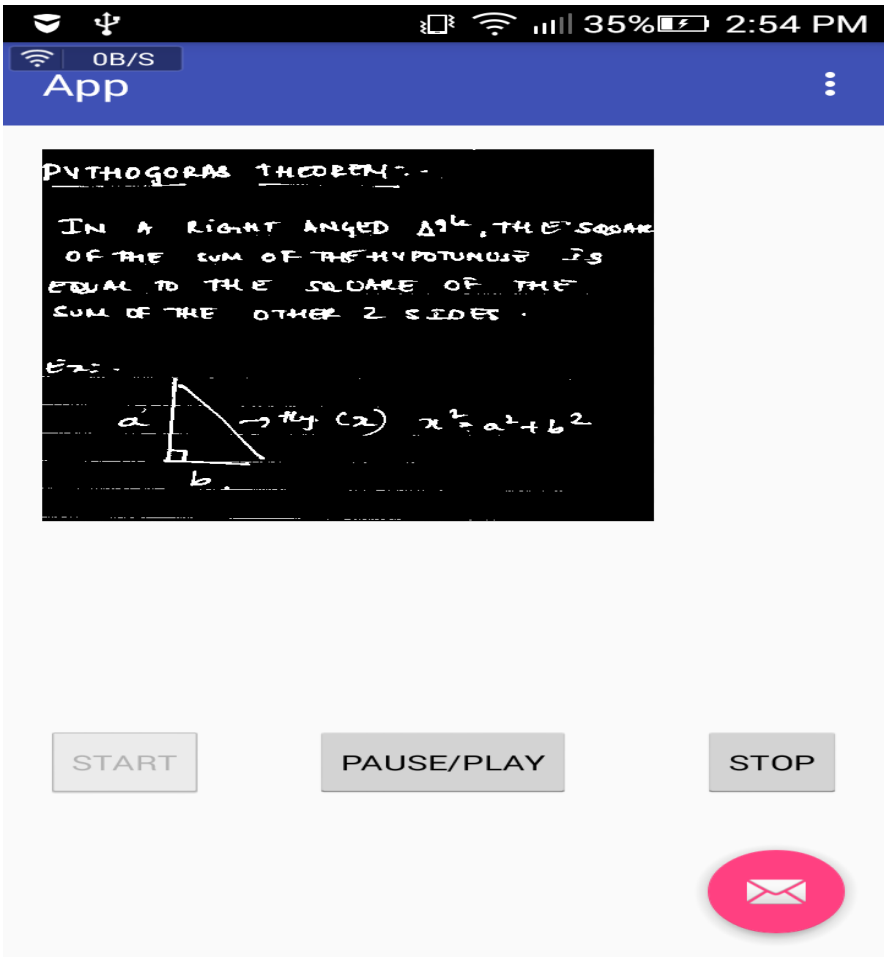

Figure 7.2 Binary of the Image

7.4 Morphological operations

The morphological operation erode is applied on the binary image obtained. This operation increases the darker regions over the brighter regions. In the binary image obtained, text constitutes the brighter region and background constitutes the darker region. When this morphological operation applied over the binary image using suitable structuring element the text is removed from the image. Then the morphological operation dilate is applied to the image obtained from above step. This operation increases the brighter regions over the darker regions. In the image obtained from above step brighter region constitutes line. When this operation is applied the lines dilate and the background with lines image is obtained.

The functions look like this:-

```
Mat horizontalStructure =   Imgproc.getStructuringElement(Imgproc.MORPH_RECT, new
Size(horizontalsize,1));

Imgproc.erode(src, des,horizontalStructure, new Point(-1,-1));

Imgproc.dilate(src ,des,horizontalStructure, new Point(-1,-1));
```

Here

horizontalStructure - is the input structuring element for dilate and erode methods

src- source image on which the functions must be applied

des-des image which has the destination image

point is the reference from where the structuring element should be taken

The output of the image looks like this:

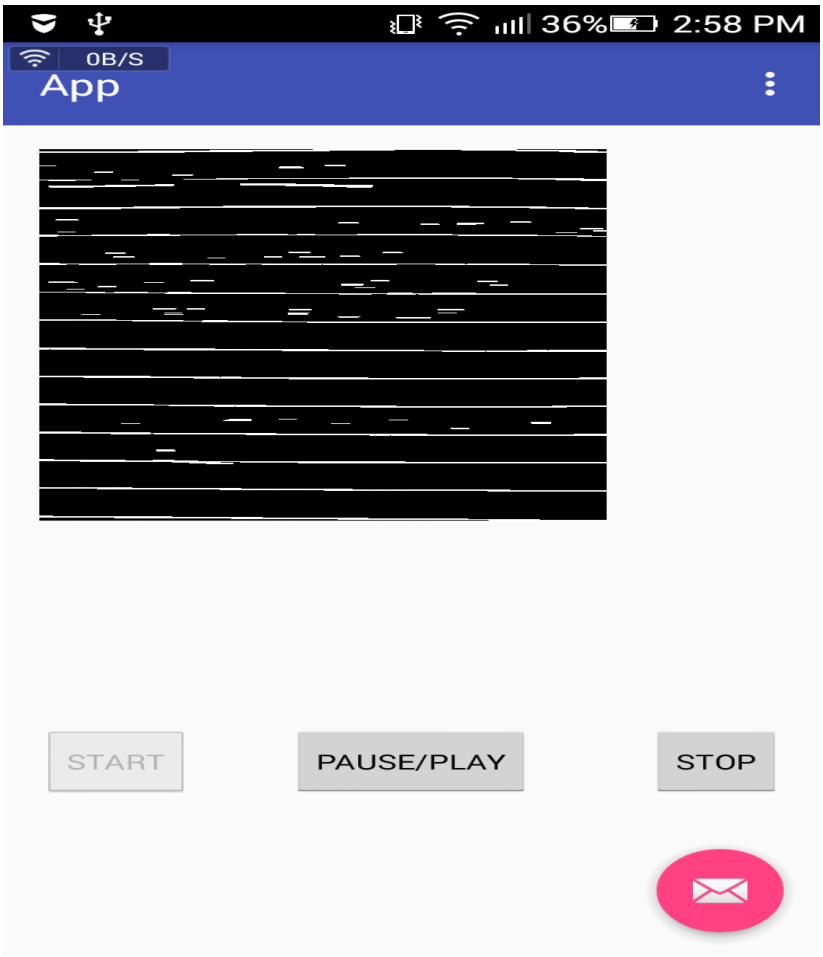

Figure 7.3 Image obtained after applying morphological operations

7.5 Finding the width between the lines

The dilated image from the previous step is traversed from left to right to detect the lines. Once two consecutive lines are detected their width is found out by subtracting their respective ‘y’ co-ordinates. Once the gap between the consecutive lines is found, the given image is animated line by line using the distance between the lines.

7.6 Animating the image

After finding the width between the lines, the contents within each line is animated column wise i.e. given a line, for each column in a line, the rows corresponding to that particular column is displayed .The pixels are written on to a plain background image and after a certain period of time the written pixels are displayed, to give a feel of the contents being animated. The thickness of the font is also taken into account.

Here is the pseudo code for displaying-

```
for k in lines:
    for i in columns:
        for j in rows:
            write (image[j][i],des)
        If i == delay:
            display the destination image
```

Displaying the image is done using AsyncTask in android. AsyncTask enables proper and easy use of the UI thread. This class allows performing background operations and publishing results on the UI thread without having to manipulate threads and/or handlers. Thread.sleep() in async task doesn't stop the main thread. Hence animation can be delayed by giving suitable parameters to Thread.sleep(time) method.

doInBackground(Params...) :- This method contains the code of animation.

onProgressUpdate(Progress...) :- It is invoked on the UI thread after a call to publishProgress(Progress...).This method is used to display image so after doing Thread.sleep() for appropriate time. This gives the feel of animation.

Android provides Bitmap class to handle images. This can be found under android.graphics.bitmap.

createBitmap(int width, int height, Bitmap.Config config)

-It is used to create a Bitmap of same size of that of image.

Utils.mapToBitmap(image, bitmapObject)

-It is used to convert opencv image to bitmap

7.7 Adjusted animation speed

The animation can be adjusted to suit the pace of the audio, by specifying the relative speed of animation in the control file. This speed is expressed in terms of percentage increase or decrease from the standard animation speed in the control file .The control file is written in json format and this file is read and parsed in MainActivity.java.

Example:

```
{  "arr":[
 {
  "lines": 2,//number of lines to be considered
  "speed": -10//fast i.e. decrease the speed of the animation by 10%
 },
 {
  "lines": 2,
  "speed":0//normal i.e. increase the speed of the animation by 0%
 },
```

```
{
 "lines": 2,
 "speed":20//slow i.e. modify the speed of animation by 20 %
}]}
```

7.8 Audio Integration

Audio is recorded at the beginning of the lecture .This audio is later integrated with the animated image to give a feel of watching a video.

7.9 Android Interface

An Android Interface is provided, where in a student can browse his file system and select the folder, containing image, its respective audio and control file to be animated. Once the required file is selected, opencv libraries are loaded for processing the selected image.

The app has start, stop and pause buttons .This gives a feel of watching a recorded lecture.

Chapter 8

Results and Analysis

Experiment No. 1: Lecture video of Pythagoras theorem

Language: English

A video (in WVGA mode) was recorded for 1 minute 30 seconds.

Size of the video was 24.82MB.

Audio was recorded alongside the video.

The size of the audio was 155 KB (@ 22 KHz).

At the end of the video, picture of the paper is taken.

The size of the image is 50 KB (640 x 480 pixels)

Control file is written for animation to happen correctly.

Size of the control file is 1 KB.

Compression Ratio Calculation:

Size of the files (audio + image+ control file):

- (155+50+1)/1024=0.2011 MB

Number of folds gained by our app:

- 24.82/0.2011 = 123.37708 ~ 123 times

That is the compression ratio of the app : video is 123**:** 1

Experiment No. 2: Lecture video of Kannada Poet Kumaravyasa

Language: Kannada

A video (in WVGA mode) was recorded for 3.32 minutes.

Size of the video was 53.24 MB.

Audio was recorded alongside the video.

The size of the audio was 333 KB (@ 22 KHz).

At the end of the video, picture of the paper was taken.

The size of the image is 50 KB.

Control file is written for the animation to happen correctly.

Size of the control file is 1 KB

Compression Ratio Calculation:

Size of the files (audio + image + control file):

- (333+50+1)/1024=0.375 MB

Number of folds gained by our app:

- 53.24/0.375=142.2395 ~ 142 times

That is the compression ratio of the app : video is 142**:** 1

Experiment No. 3: A student asking a doubt

Language: English

A video (in WVGA mode) was recorded for 31 seconds.

Size of the video was 8.12 MB.

Audio was recorded alongside the video.

The size of the audio was 43.38 KB (@ 22 KHz).

At the end of the video, picture of the paper is taken.

The size of the image is 50 KB (640 x 480 pixels)

Control file is written for animation to happen correctly.

Size of the control file is 1 KB.

Compression Ratio Calculation:

Size of the files (audio + image+ control file):

- (43.38+50+1)/1024=0.09216 MB

Number of folds gained by our app:

- 8.12/0.09216 =88.1046 times ~ 88 times

That is the compression ratio of the app: video is 88**:**1

Experiment No. 4: A chemistry teacher solving a problem

Language: English

A video (in WVGA mode) was recorded for 3 minutes 56 seconds

Size of the video was 59.29 MB.

Audio was recorded alongside the video.

The size of the audio was 370 KB (@22 kHZ).

At the end of the video, picture of the paper is taken.

The size of the image is 50 KB (640 x 480 pixels)

Control file is written for animation to happen correctly.

Size of the control file is 1 KB.

Compression Ratio Calculation:

Size of the files (audio + image + control file):

- (370+50+1)/1024=0.41113 MB

Number of folds gained by our app:

- 59.29/0.4113 =144.2356 times ~ 144 times

That is the compression ratio of the app : video is 144**:1**

Chapter 9

Conclusion and Future Work

In this project, we have implemented a robust, low cost method for digitizing classroom experiences. We are able to recreate the effect of video at much less storage. Instead of sharing a video file, now the teacher has to share 3 files. The size of the image, audio and control file together would be way less than the single video file explaining the same concept. As these files are small, they can be transferred through social media like whatsapp. Thus, overcoming the size restriction of whatsapp media files that can be transferred.

Firstly the image is converted to gray scale image. Then it is converted to binary image, and then morphological operations are applied to extract the text part from the image. The distance between 2 lines is calculated. Finally the image is animated line by line. The JSON file is the control file, which controls the speed of animation of the image. This gives the effect of video. Async task of android is used to carry out the animation as high computations of OpenCv are not carried out in UI thread and does not block the UI thread.

Future Work:

- Talking to the teachers to understand their expectations from the app.
- Providing an interface for the teacher to adjust the animation speed (Teachers' tools).
- Beautifying the app
- Increasing the efficiency of the app
- Increasing the robustness of the app

Chapter 10

User Manual

This app enables user to watch the lecture video without the overhead of storage .It has all the necessary infrastructure that a video player has.

Step 1: Browse the file system

- Click on the app.
- The app provides an android interface, where in the user can browse his file system. (Figure 10.1)
- Select the folder which has the image and its respective audio and control file.

Step 2: Animation of the Image

- Once the required folder is selected, the app takes you to a separate window.
- The start button enables the user to start the animation of the image. (Figure 10.2).
- In case the animation has to be paused, the pause/play button allows the user to pause the animation. Clicking the pause/play button again resumes the animation of the image. (Figure 10.2)
- The stop button permanently stops the animation. (Figure 10.2)

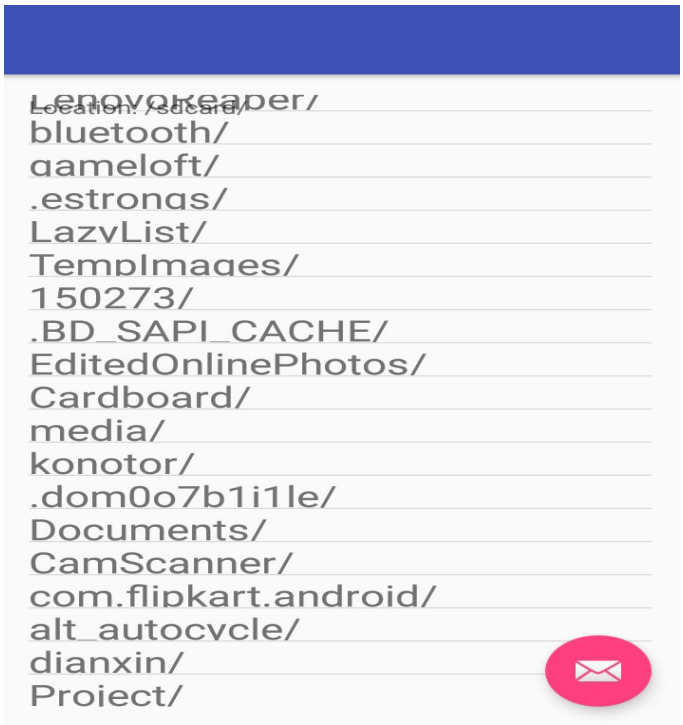

Figure 10.1 Interface to browse file system

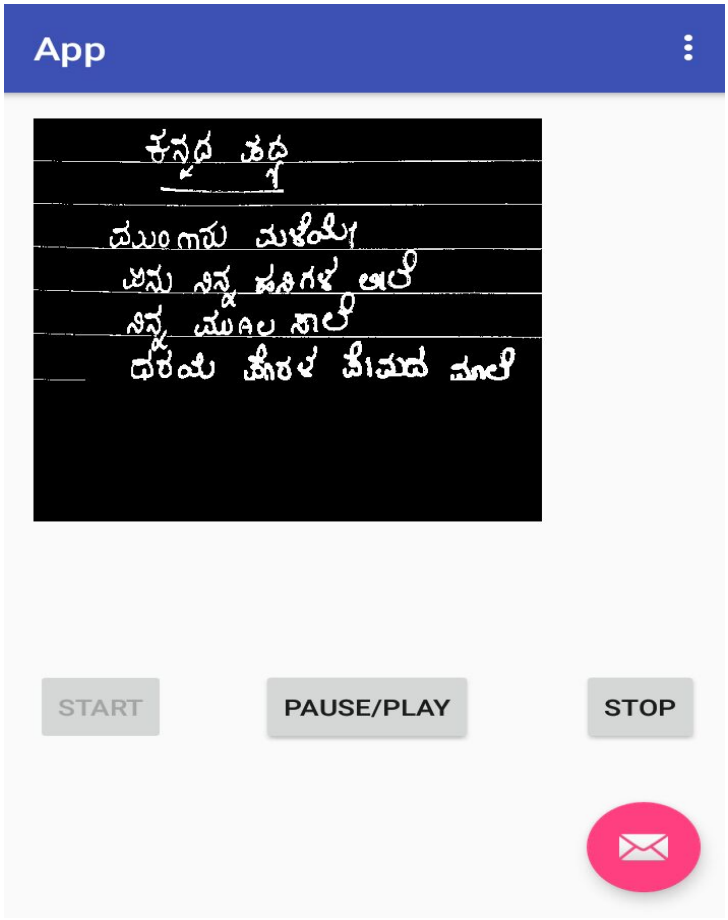

Figure 10.2 Interface to play animation

Chapter 11

References

[1] http://stackoverflow.com/questions/27406303/opencv-in-android-studio

[2] http://developer.android.com/intl/es/sdk/index.html

[3] http://developer.android.com/intl/es/reference/android/os/AsyncTask.html

[4] http://opencv.org/platforms/android.html

[5] http://opencv.org/

[6] http://developer.android.com/intl/es/reference/android/Manifest.permission.html#READ_EXTERNAL_STORAGE

[7] http://developer.android.com/intl/es/training/displaying-bitmaps/process-bitmap.html

[8] http://www.techotopia.com/index.php/An_Android_Studio_Recording_and_Playback_Example_using_MediaPlayer_and_MediaRecorder

[9] http://www.christophbrill.de/de_DE/how-to-create-a-android-file-browser-in-15-minutes/

[10] http://developer.android.com/guide/appendix/media-formats.html#recommendations

[11] http://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html

[12] Dongsong Zhanga, Lina Zhou, Robert O. Briggs, Jay F. Nunamaker Jr.: "**Instructional video in e-learning: Assessing the impact of interactive video on learning effectiveness**" in *Information & Management*, Volume 43, Issue 1, January 2006, Pages 15-27