Lip Reading Application

ARD

December 2012

Sagi Bernstein

Dor Leitman

Dagan Sandler

Supervisors: Dr. Kobi Gal

Dr. Gavriel Kohlberg

Table of Contents

Chapter 1 - Introduction

1 Vision:

Speech is the main form of human communication. We rely on it to communicate on a daily basis and it helps us get our message through even when a person is not paying direct attention to us.

But what happens when one can't vocalize his words?

Such is the case of people who go through Laryngectomy procedures.

Laryngectomy is a medical procedure for people who suffer from different conditions such as laryngeal cancer. The procedure consists of the removal of the larynx and separation of the airway from the mouth, nose and esophagus. This cripples the patient's ability to vocalize words, and in some cases is a permanent condition.

Today, these patients are required to spell out their sentences or write them down, and need to make sure people are paying attention first.

The goal of the project is to provide an easy and simple way for Laryngectomy patients to speak out their words using only lip movements. The system will work on a relatively low-end system equipped with a camera such as a tablet, smartphone or a laptop, and will transform the input video of the lip movements into words.

The project will include a learning mechanism, which will allow the training and improving of the software over time.

2 Problem Domain:

Most of the data of human speech is obtained from voice. However, a lot of data that the naked human eye can't perceive can be extracted from lip movements during speech.

The problem domain is to analyze video inputs of words being spoken, and transform the data gathered from the video to actual words.

Our solution is to extract features, which are not always noticed by the naked eye, from the video input, and then use those features to compare against a training set using different algorithms that will choose the word that was most likely spoken.

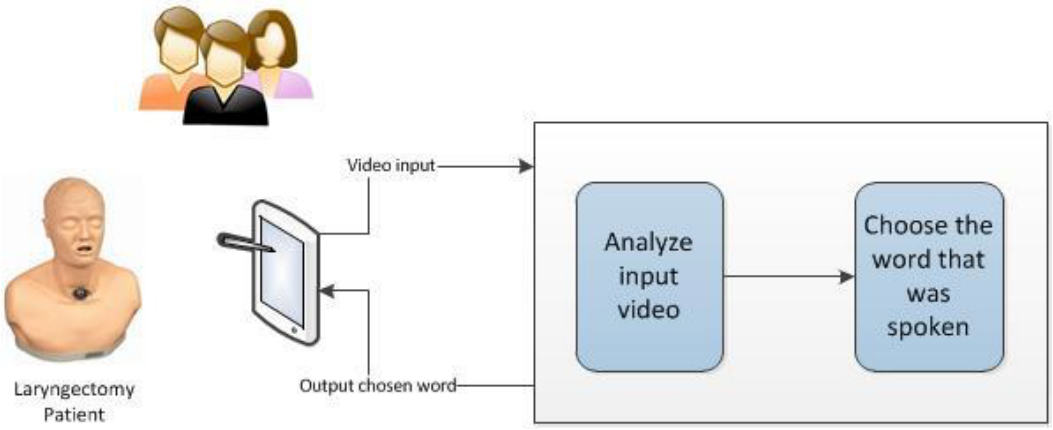


Fig 1. : General interaction

3 Stakeholders

1. Experts:
   The system is developed according to the specifications and requirements of Gavriel Kohlberg, M.D. Columbia University Medical Center, and with professional and academic guidance of Dr. Kobi Gal, Information Systems Engineering department at Ben-Gurion University.
2. Users:
   The users of the system will be laryngectomy patients and other people with difficulty speaking.

4 Software Context:

The software will be used in hospitals by patients, on daily basis, to capture the video of their lip movements, and output the words they spoke.

Our system will be divided into the following general components:

1. The training and learning database – May be a simple file-based database that will hold data of words and their representation in the system.
2. The client – a simple user client that will allow the users to interact with the system and capture their lip movements and transform the video into data.
3. The algorithm layer – will process the data from the client using different methods such as Neural Networks, HMM and DTW and output the matching words. This is the area in which most of the project work will be done.

Chapter 2 - Functional Requirements

- The application will be able to capture user's video and transform it into words or phrases.
- Allowing more than one user to use the application (at different times).
- Allowing the user to start the program with his own user profile to give him better output.
- The user will be able to configure his video input at each session, such as sticker colors, and test that his lips are recognized properly.
- The application will have a training mode in which the user can train the application to learn new words or to improve its accuracy.
- The application will allow the user to give feedback on successful and unsuccessful recognitions.

Chapter 3 - Non-functional requirements

1 Performance constraints

1 Speed

- Single word recognition process will complete in less than one second.

2 Capacity

- The CPU usage by the system should be under 90%
- The system should take up to 500MB RAM when processing the data.

3 Portability

- Windows or Linux with at least 2GB RAM and a 2 GHz processor with Java Runtime environment version 6 or later, OpenCV version 2.4.3 or later and ffmpeg version 1.0 or later installed. A camera that produces at least 24 frames per second and each frame with at least 320 x 240 pixels, is optional for live input.
- Mobile application: should run on any Android device with version 4.0.0 and up, 1GB of RAM and at least a 1GHz single core processor. The device should also be equipped with a front facing camera that produces at least 24 frames per second and each frame with at least 320 x 240 pixels.

4 Usability

- The system GUI should be user-friendly and easy to use.
- All options will be available to the user under no more than 3 clicks.
- Learning how to use the system, should not take more than 10 minutes.
- The system will support English speaking users.

5 Reusability - Modularity of the system

- The system should have easily replaceable modules for video feature extraction, various data normalization processes and classifiers.

2 Platform constraints

The system will be developed in Java with Windows and Linux operating systems. We will use eclipse as our IDE as it is the most mature offering and as it is the most familiar to us. For compilation we shall employ the Maven system as it makes it easier to add existing components to the application and also automates compilation and testing on all operating systems and architectures. For video processing we will use OpenCV as it is the most mature offering. For code hosting, version control, and issue management

we’re using github. For testing we use an automated Travis - continuous integration system that tracks our commits and runs the test suites for every commit.

All the tools have been selected since they are most widely used and hence the best documented products. Also they allow us to develop for free since all are open source with free software license. This is in order to keep our application also open source for free use by free clinics.

The tools we'll be using to develop the project are:

1. Oracle Java development kit 7
2. Apache Maven 3.0.4
3. Eclipse Juno 4.2 SR1
4. github.com
5. travis-CI.

1 Project constraints

- The system is interactive. The input is a video sample of a spoken word. The input will at first be from a self produced data set of video files.
- Since the premise of this project is to use a cheap camera we do have access to the required device and the data will be actual.
- For testing and development we will use xml files to represent a extracted features of a video sample. and finally we will use live input from a camera feed.
- The system will be tested automatically after each commit. The data set for the system will be available online and the tests will get the needed file inputs this way. Work on live camera feed will be tested manually.
- The final system will be presented by a user interacting with it by saying silently one or more of the word from the domain and displaying the output of the system.

3 Special restrictions & limitations

- Since our search for a free real time capable lip tracking software has left empty handed, we currently support video inputs in which the speaker has colored stickers sticked around his lips.

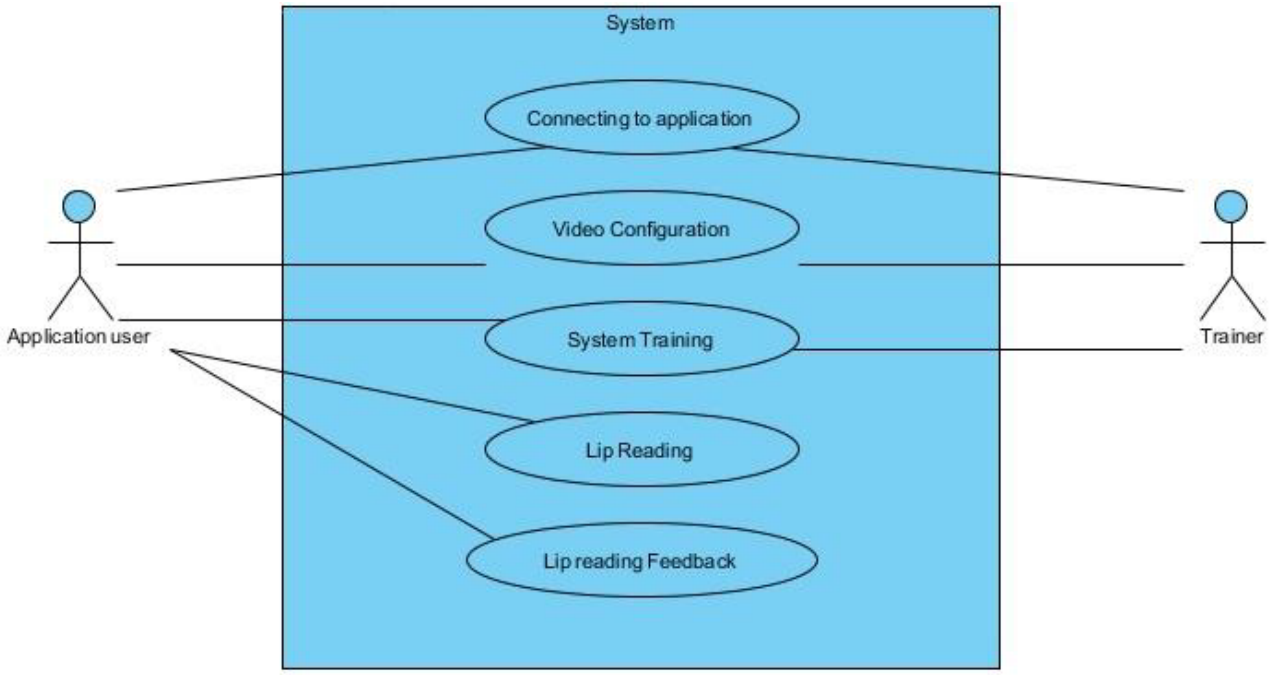
Chapter 4 - Usage Scenarios

1 User Profiles – The Actors

- Application user - Laryngectomy patient:
    - This actor is the main user of the system. This user will use the application for input his lip video & get a voice output to the soundings.
    - User Profile:
        - Laryngectomy patients has no physical limitations of using the application.
        - The user will use the application mainly in hospital environment.
    - All use cases below are relevant for this user.

- Trainer:
    - This user will only train the system.
    - Only following use cases are relevant for this user:
        - Connecting to application
        - Video Configuration
        - System Training
        - Closing application

2 Use-cases



1 Use Case Name: Connecting to application

Description: When user opens application he reaches a connection screen. User inputs his/her name & clicks "OK" button. This Use Case is important for the application to know which training data to match the user.

Actors: All application users.

Triggers:

- User opens the application.

Preconditions:

- Application is closed.

Postconditions:

- Application is open & ready for use.

Normal Flow:

1. User opens the application.
2. Welcome screen is displayed, asks the user to choose his name from users list.
3. If it is the first time the user open the application, he clicks on "New User", insert his name in a text box & clicks "OK".
4. Application displays "Welcome" message & opens main application screen.

2 Use Case Name: Video Configuration

Description: Configuration lips pattern of user. User has 4 colored stickers on his lips. In this use case the user will indicate which sticker is in which color. The user

chooses a sticker & clicks on the frame on one of this sticker`s pixel. This pixel RGB data will be saved by the application.

Actors: All application users.

Triggers:

- User click on the button "Video Configuration".

Preconditions:

- User has 4 stickers on his lips (2 stickers on upper & lower lips, 2 sides stickers in connection point of upper & lower lips).
- User`s device includes camera, identified by application.

Postconditions:

- Colors of stickers are known & saved in application.

Normal Flow:

1. User click button "Video Configuration".
2. Device`s camera real time video is displayed in application UI. The user is asked to point with the mouse on the upper lip sticker.
3. User clicks on upper lip sticker with the mouse.
4. Application locates the pixel & retrieves RGB data of this pixel. This data is saved.
5. Graphical circle in sticker`s colored is drawn on video frame in sticker location.
6. Steps 2-5 are performed 3 more times for lower sticker & sides stickers.
7. User approves that all graphical entities are drawn in stickers' places by clicking an "OK" button.

Alternate flow:

- If no stickers are required in final version, this use case is aborted.

3 Use Case Name: System Training

Description: User records lip videos of words or phrases according to application demand. This feature helps to improve the accuracy for a particular individual by learning his typical lips movements.

Actors: All application users.

Triggers:

- User clicks the button "System Training".

Preconditions:

- User`s device includes camera, identified by application.
- User performed "Video Configuration" Use Case.

Postconditions:

- User videos data is saved in the system.

Normal Flow:

1. Training UI contains a real time video frame of user.
2. Application training UI displays a word & asks the user to say it.
3. User makes sure his lips are in front of camera & he clicks on "Record Video" button.

4. User says the given word.
5. User clicks on "Stop Recording" button.
6. Steps 2-5 performed again for few given words.
7. Application display text: "Thank you for teaching me!"

4 Use Case Name: Lip Reading

Description: User presses "Record Video" button & says a phrase in front of device`s camera. Application`s output will be user`s phrase in Text & Voice.

Actors:

- Application user.

Triggers:

- User pressed "Record Video" button.

Preconditions:

- User`s device includes camera, identified by application.
- User performed "Video Configuration" Use Case.

Postconditions:

- User`s video data is saved in application database.
- Application will display an output of text & voice of user`s video interpretation.

Normal Flow:

1. User presses "Record Video" button on application UI.
2. The application displays the user real time video of him from device camera.
3. The application informs the user that it is recording.
4. User says his phrase (video only, without voice) in front of device camera, making sure his lips are inside the shown camera frame.
5. User presses "Stop Recording" button.
6. Application stops showing real time video in UI & prints "Analyzing video…" on screen.
7. User video data is analyzed and saved in Database.
8. Data is transferred to application Logic Layer.
9. An output is return from logic layer, displayed to user by text & voice output.

5 Use Case Name: Lip reading Feedback

Description: After performing "Lip Reading" Use Case, the user will be asked to give a feedback whether the application output was correct or. This use case is important for training the system according to users inputs.

Actors:

- Application user.

Triggers:

- "Lip Reading" Use Case is done.

Preconditions:

- "Lip Reading" Use Case.

Postconditions:

- User data is used to train the system.

Normal Flow:

1. After "Lip Reading" Use Case, the user is asked whether the output was correct. 2 buttons are displayed – "Yes", "No".
2. The user clicks on one of the buttons.
3. If the user clicks on "Yes" button, the application uses user`s video data to train the system of the specific output word.
4. If the user clicks on “No” button, the application asks the user about the input & train the system according to that input.
5. Feedback buttons disappear.

Extensions:

2.a. If user doesn’t click on any button. After the next action he takes in the application step 5 is performed.

4.a. Application displays “Enter original input phrase:”

4.b. User enters the correct text of the video input

4.c. The application analyzes input text, saves the video & trains the system according to that video data.

3 Special usage considerations

- Video Conditions:
  - Usage of this application needs to be done in a lighted room.
  - User`s face has to be align in front of device camera. User`s lips have to be in camera`s frame.
- User video recognition marks (might not be needed in final version):
  - User will have to put 4 colored stickers on his lips: central upper lip, central lower lip, left connection point of upper & lower lips and right connection point of upper & lower lips.
  - User will have to configure video & stickers colors first.

Chapter 5 - Risk assessment & Plan for the proof of concept

1 Risks

- Slow response time:
  The application is designed to work in real time. One of the main risks is that it won't be able to run in real time on every device. It might be caused either by heavy video processing or long classification process. A "Long processing time" exception will be thrown in exceptional cases like that.

- Low percentage of Lip Reading success:
  Might be caused by:
  - Bad video conditions: The user will be asked to find a better place.
  - Unrecognized lip motions: The user will be asked to do some training for the system before using its lip reading functionality.

2 Plan for the proof of concept

Proof of concept Prototype will include the following functionality:

- The prototype will include a training data base of lip videos of individual user saying 2 words: "Yes" and "No".
- Color stickers will be on trainer`s lips.
- The prototype will be able to extract stickers coordination points from training lips videos and train itself.
- Given test lip video files (not real time video), the prototype will recognize the word said in the video – "Yes" or "No".
- The prototype will achieve at least 75% accuracy of test videos.

The prototype will help us understand & choose the best recognition algorithm and machine learning mechanism. Moreover, it will teach us about success percents of our project & will reduce the risk of project failure.