LBYEC4A – EK2

Signals, Spectra and Signal Processing Laboratory

Final Project Proposal

Implementing Text Detection with Audio Output

Ira Third L. Burgos
Alfred Felix A. Daanoy
Michaello J. Hermogenes

PROJECT DESCRIPTION

In this project, the students desire to integrate the concepts of image and audio processing. Thus, text detection with audio output is the paramount representation of the endeavor. The programming structure of the code will entail the principles of OCR, which is utilized to recognize the text within an image using optical character recognition, and TTS, which illustrates text-to-speech recognition. Therefore, the entire process will start from MATLAB reading the text inscribed in the image using **imread**, specifically road signs, to outputting the encompassed text as audio [1][2][3].

- **imread**

A = imread(filename)
A = imread(filename,fmt)
A = imread(___,idx)
A = imread(___,Name,Value)
[A,map] = imread(___)
[A,map,transparency] = imread(___)

However, Artificial Intelligence (A. I.) installed within MATLAB is not capable of detecting fonts with myriads of variations. Thus, it cannot detect some letters due to its unfamiliarity with the showcased fonts. The students plan to combat this hindrance by training the A. I. of MATLAB by accessing the OCR Trainer application available in the software. With this, the individuals can input images of disparate versions or styles of texts to allow the A. I. recognize the photos. A block diagram is represented below to illustrate the process of training the A. I. of MATLAB [4][5].

Figure 1.1. OCR Training Block Diagram [1]

Another possible limitation of the A. I. is the inefficiency when the images have a background. Sequentially, the students can mitigate the effects of this predicament by removing the background by utilizing the function **im2gray**. This compels MATLAB to lessen the saturation of the image to underscore the texts visible. The codes below are possible implementations of the operation **im2gray** [6]. The operations **strel** and **edge** can also be utilized to amplify the legibility of the generated images [67[8]. The following syntaxes of the codes are showcased below [6][7][8].

- **im2gray**

I = im2gray(RGB)

- **strel**

SE = strel(nhood)

SE = strel("diamond",r)
SE = strel("disk",r)
SE = strel("disk",r,n)
SE = strel("octagon",r)
SE = strel("line",len,deg)
SE = strel("rectangle",[m n])
SE = strel("square",w)

SE = strel("cube",w)
SE = strel("cuboid",[m n p])
SE = strel("sphere",r)

- **edge**

BW = edge(I)
BW = edge(I,method)
BW = edge(I,method,threshold)
BW = edge(I,method,threshold,direction)
BW = edge(___,"nothinning")
BW = edge(I,method,threshold,sigma)
BW = edge(I,method,threshold,h)

[BW,threshOut] = edge(___)
[BW,threshOut,Gx,Gy] = edge(___)

In light of the aforementioned information, this project should produce a product that proficiently detects the fonts exhibited in the photos without any errors in the process. This will mainly be applied to road signs to benefit individuals who have poor eyesight. Henceforth, with the aid of outputting the words stated, the populace will have an easier experience walking or driving in public.

METHODOLOGY

System Flowchart

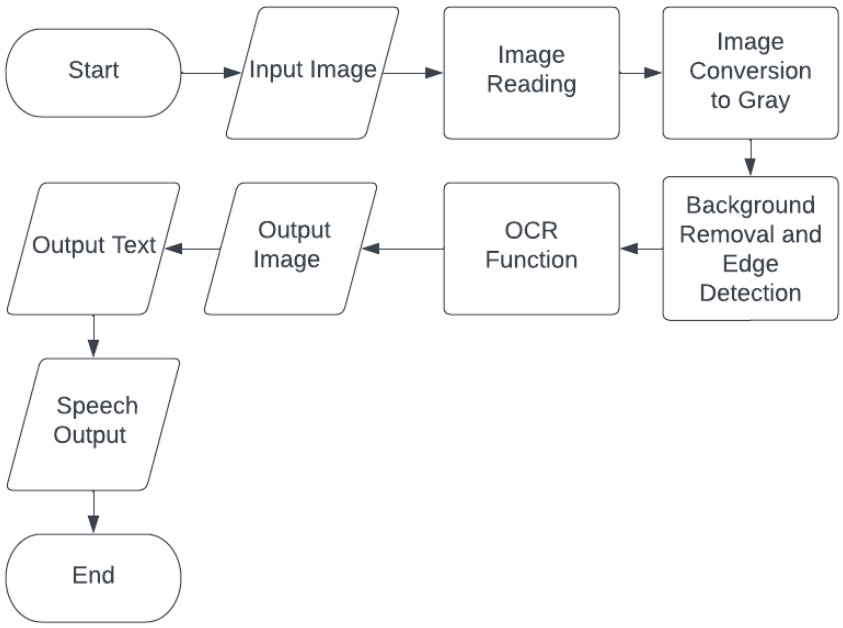

Figure 2.1. System Flowchart.

As seen in the flowchart above, the program of this project takes an input image, then processes it to extract its text. In the context of this group, images from road signs are taken. After processing, the software then outputs the filtered image alongside what the text detection algorithm has detected. This is then fed to another function that allows the detected text to be spoken by a text-to-speech algorithm within MATLAB.

Schematic Diagram of OCR Training

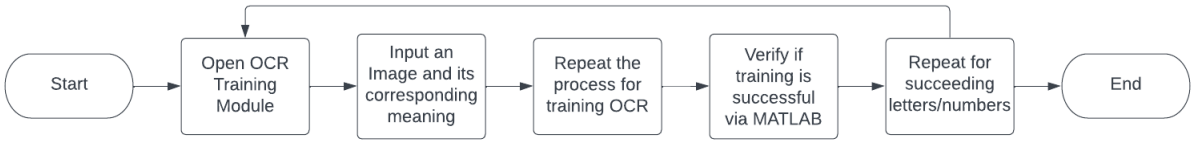

Figure 2.2. Schematic Diagram of OCR Training [1]

In this project, the system diagram above will be the basis for the program. Adding images and editing labels are both part of the code itself. Training the OCR algorithm is also an important task, as an untrained OCR may not be able to recognize or detect text that should have been recognized. After inputting images and training the OCR, the trained OCR must then be tested inside MATLAB to determine the training's effectiveness. This then should be repeated until all desired letters and/or numbers have been trained onto the OCR.

Initial Methodology

For this project, the students decided to utilize three distinct parts for organization purposes. The first part is coding the program itself in MATLAB. The second part is training the OCR function to the required fonts and numbers. The last part is finalization and creation of the paper and presentation for the project.

Part 1: MATLAB Coding

This part focuses on the creation of the program of the project. This is the most important part of the project. The steps are as follows:

1. Create a new livescript in MATLAB. This consists of the group members as well as their details.
2. Inside the livescript, write down the code that reads, decodes, and displays an image.
    a. This uses the command **imread**. This allows MATLAB access to the image that shall be used for processing.
3. Next, filtering should now be used. These include, but are not limited to:
    a. im2gray. This converts the image given to grayscale.
    b. edge. This allows edge detection within the image. This will be useful at differentiating background from foreground, especially with text.
    c. strel. This allows further text detection within an image.
4. After filtering has been applied, the OCR function should now be utilized. This allows the built-in MATLAB function to detect text, based on its training.
    a. This can be done using the **ocr()** command. It shall then be saved to a variable that stores the text detected.
5. The next step after OCR is to check whether or not the text detected by OCR is satisfactory. If not, the OCR algorithm should be trained further. This will be done in part 2 of the experiment.
6. If the OCR text detection algorithm outputs a satisfactory result, the resulting variable wherein the text is stored should now then be dictated by a text-to-speech function.
    a. This can be done using the following commands:
        i. **caUserInput = text**

ii. **NET.addAssembly('System.Speech')**
iii. **obj = System.Speech.Synthesis.SpeechSynthesizer**
iv. **obj.Volume = 100**
v. **Speak(obj, caUserInput)**

7. The last step is to verify if what was dictated is the same as what was recognized by OCR. If all are complete, the project should now move to the next parts.

Part 2: OCR Training

This section is an essential part of the previous section. This is because an untrained OCR will output erroneous or wrong information, or may not even detect text. To train the OCR algorithm:

1. Open the OCR trainer module by typing in a livescript **ocrTrainer.**
2. This opens a separate window containing the OCR training application.
3. Within the application, open an image file containing a letter for training.
4. Manually label this letter or number.
5. Continuously do this to properly train the OCR.
6. After the session, make sure to save the OCR file.

Part 3: Finalization

The final step for this project is to finalize the remaining requirements. These include the final paper, the final code, and the final presentation. All of these can be done using respective applications, such as Microsoft Word, Powerpoint, and GitHub.

SCHEDULE OF ACTIVITIES

Gantt Chart

Task Name	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13
Tentative Project Proposal Submission							
Final Project Submission							
Optical Character Recognition (OCR) Training							
Code Implementation and Debugging							
Document Making							
Project Document Review							
Project Presentation and Demo Making							
Project Demonstration							

Time Table

Day	Agenda	Notes	Person/s In-Charge
February 28	Tentative Project Proposal Submission	The initial project proposal will be submitted.The group's Trello, GitHub and Document files will be created.	Third Burgos Alfred Daanoy Mico Hermogenes
March 14	Final Project Proposal Submission	The Final Project Proposal Document will be submitted. The document will be accomplished, which consists of description, methodology, and schedule of activities with regards to the project.	Third Burgos Alfred Daanoy Mico Hermogenes
March 15 - 19	Optical Character Recognition (OCR) Training	The OCR will be given several images that will further develop its character recognition.	Third Burgos Alfred Daanoy Mico Hermogenes
March 20 - 27	Code Implementation and Debugging	The students will be completing the code for the image detection and voice output.	Third Burgos Alfred Daanoy Mico Hermogenes
	Document Making	For the project, the students will be working on the final document, which includes the theoretical consideration, methodology, discussion, results, and analysis	Third Burgos Alfred Daanoy Mico Hermogenes
March 28	Project Document Review	The document will be reviewed by the Instructor.	Third Burgos Alfred Daanoy Mico Hermogenes
March 29 - April 10	Project Presentation and Demo Making	The students will work on the PowerPoint	Third Burgos Alfred Daanoy

		presentation of the project as well as the video demo presentation.	Mico Hermogenes
April 11	Project Demonstration	The student's video demo will be presented.	Third Burgos Alfred Daanoy Mico Hermogenes

REFERENCES

[1] Mathworks. “ocr.” *Mathworks*. Recognize text using optical character recognition - MATLAB ocr (mathworks.com) [accessed Mar. 3, 2023].

[2] Mathworks. “Text-to-Speech Conversion.” *Mathworks*. Text-to-Speech Conversion - MATLAB & Simulink (mathworks.com) [accessed Mar. 3, 2023].

[3] Mathworks. “imread.” *Mathworks*. Read image from graphics file - MATLAB imread (mathworks.com) [accessed Mar. 14, 2023].

[4] Mathworks. “OCR Trainer.” *Mathworks*. Train an optical character recognition model to recognize a specific set of characters - MATLAB (mathworks.com) [accessed Mar. 3, 2023].

[5] Mathworks. “Train Optical Character Recognition for Custom Fonts.” *Mathworks*. Train Optical Character Recognition for Custom Fonts - MATLAB & Simulink (mathworks.com) [accessed Mar. 3, 2023].

[6] Mathworks. “im2gray.” *Mathworks*. Convert RGB image to grayscale - MATLAB im2gray (mathworks.com) [accessed Mar. 14, 2023].

[7] Mathworks. “strel.” *Mathworks*. Morphological structuring element - MATLAB (mathworks.com) [accessed Mar. 14, 2023].

[8] Mathworks. “edge.” *Mathworks*. Find edges in 2-D grayscale image - MATLAB edge (mathworks.com) [accessed Mar. 14, 2023].