



Presented to the **Electronics and Communication Engineering Department**

De La Salle University - Manila

Term 2, A.Y. 2022 - 2023

In partial fulfillment

of the course

**In SIGNALS, SPECTRA AND SIGNAL PROCESSING LABORATORY (LBYEC4A -
EK1)**

Letter image identification to FSL and Vice Versa

Submitted by:

Carrasca, Justine Christian H.

Mansilungan, Chrismon Elijah Q.

Subong, Reinald Jan M.

Submitted to:

Engineer Stephen Ruiz

April 21, 2023

I. Abstract

- A. Currently most algorithms present are for detecting letters to ASL and vice versa. The current gap in this field is development for letters to FSL conversion and vice versa.
- B. This paper created an image detection software that will convert letters to the appropriate FSL equivalent and vice versa.
- C. The letters and FSL gestures were detected by comparing input images to reference images. The images were evaluated using ssim, psnr, and immse comparison methods to see which reference image has the most similarity with the input image.
- D. The output of the letter to FSL algorithm was highly dependent on the reference images. Where if the input image slightly deviated from the font of the reference image it would detect it as the wrong letter and display the incorrect FSL equivalent.
- E. The output of the FSL to letter algorithm was highly dependent on the reference images where the image identification of the input image detects similarity of the input image, it will detect it even if it has slight differences displaying the letter equivalent.
- F. This algorithm could be used as a basis for future developments in FSL teaching techniques.

II. Introduction

The use of letter to sign language in communicating with deaf people is evident where there are many applications and systems that were used to evaluate the sign input. Sign language is a system that is composed of formal gestures, mimic, hand sign, figure spelling. Sign language can represent a whole idea or phrase in which this visual language is a method of communication. [1] The techniques that were used include the use of MATLAB where the system evaluates the sign language input with the built-in function of MATLAB where it just used to recognize sign language. [1] Technologies were utilized in using Artificial Neural Networks (ANN), evolutionary algorithms, and hidden Markov models that are employed to recognize sign language in a wide range of theoretical and experimental studies. [2] Another technique done in image acquisition such as the gloved based approach uses electromechanical devices to provide exact hand configuration, and position but it is expensive and not user friendly. The next approach is the vision based approach where a type of vision based approach uses color gloves for gesture recognition. The last approach uses the bare hand as input which is user friendly, but has challenges like complex background, environment variations and presence of

other skin color objects with the hand object. [3] The last approach was used in getting the image acquisition which the approach of this paper was to create an image detection that will convert alphabetical letters to Filipino Sign Language (FSL) equivalents where the other papers used only in the recognition of the Sign Language.

III. Theoretical Consideration

A. Letter to FSL and FSL to Letter

1. Rgb2gray

It converts the truecolor image RGB to the grayscale image I. The `rgb2gray` function converts RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance. Using the Parallel Computing Toolbox being installed, `rgb2gray` can perform this conversion on a GPU. [4]

2. Graythresh

It computes a global threshold T from grayscale image I, using Otsu's method where it chooses a threshold that minimizes the intraclass variance of the thresholded black and white pixels. The global threshold T can be used with “`imbinarize`” to convert a grayscale image to a binary image. [5]

3. Imbinarize

It creates a binary image from 2-D or 3-D grayscale image by replacing all values above a globally determined threshold with 1s and setting all other values to 0s. It uses the Otsu's method which automatically chooses the threshold value to minimize the intraclass variance of the thresholded black and white pixels and also uses a 256-bin image histogram to compute Otsu's threshold. [6]

4. gpuArray

It represents an array stored in GPU memory. A large number of functions in MATLAB and in other toolboxes support `gpuArray` objects, allowing you to run your code on GPUs with minimal changes to the code. [7]

5. Gather

It can operate on the following array data on a `gpuArray` where it transfers the given elements from the GPU to the local workspace and assigns them to another variable. [8]

6. Imresize

It returns image “B” that is scale times the size of image “A”. The input image “A” can be a grayscale, RGB, binary, or categorical image. If “A” has more than two dimensions, then it only resizes the first two dimensions. If scale is between 0 and 1, then B is smaller than A. If scale is greater than 1, then B is larger than A. By default, it uses bicubic interpolation. [9]

7. `ssim`

It calculates the structural similarity (SSIM) index for grayscale image or volume “A” using `ref` as the reference image or volume. A value closer to 1 indicates better image quality. [10]

8. `psnr`

It calculates the peak signal-to-noise ratio (PSNR) for the given image, with the image `ref` as the reference. A greater PSNR value indicates better image quality. [11]

9. `Immse`

It calculates the mean-squared error (MSE) between two different arrays. A lower MSE value indicates greater similarity between two images. [12]

10. Conditional statements

It enables you to select at run time which block of code to execute. The `if` else statement is the simplest conditional statement where it executes the code corresponding to the first true condition, and then exits the code block. Each conditional statement requires the `end` keyword. [13]

IV. Methodology

A. Letter to FSL

```

sl3GreenProjectLetter2FSL.mlx
1 [y,yrgb]=imread("sample image.png");
2 ygray=rgb2gray(y);
3 level = graythresh(ygray);
4 BW=imbinarize(ygray,'global');
5 name=["A.png" "B.png" "C.png" "D.png" "E.png" "F.png"];
6 name=name';
7 size=size(BW);
8 BW=gpuArray(BW);
9 ressim=zeros([1 size(name)]);
10 resnr=zeros([1 size(name)]);
11 resimm=zeros([1 size(name)]);
12 for a=1:size(name)
13 file='Letter reference images\' +name(a);
14 imref=imread(file);
15 imref=rgb2gray(imref);
16 imref=imbinarize(imref,'global');
17 imref=imresize(imref,size);
18 imref=gpuArray(imref);
19 sim1=ssim(double(BW),double(imref));
20 sim1=gather(sim1);
21 sim2=psnr(double(BW),double(imref));
22 sim2=gather(sim2);
23 sim3=immse(double(BW),double(imref));
24 sim3=gather(sim3);
25 ressim(a)=sim1;
26 resnr(a)=sim2;
27 resimm(a)=sim3;
28 % figure()
29 % imshowpair(imref,BW);title(a)
30 end

31 [Mag1,Pos1]=max(ressim);Mag1
32 name(Pos1)
33 [Mag2,Pos2]=max(resnr);Mag2
34 name(Pos2)
35 [Mag3,Pos3]=min(resimm);Mag3
36 name(Pos3)
37
38 fsrname=["Ah.png" "Bh.png" "Ch.png" "Dh.png" "Eh.png" "Fh.png"]
39 % Poss=round(mean([Pos1 Pos2 Pos3]),0)
40 if (Pos1==Pos2)&&(Pos2==Pos3)&&(Pos1==Pos3)
41 Poss=Pos1;
42 elseif (Pos1~=Pos2)&&(Pos2==Pos3)&&(Pos1~=Pos3)
43 Poss=Pos2;
44 elseif (Pos1==Pos2)&&(Pos2~=Pos3)&&(Pos1~=Pos3)
45 Poss=Pos1;
46 end
47 file2='FSL reference images\' +fsrname(Poss);
48 file3='Letter reference images\' +name(Poss);
49 figout= tiledlayout(2,2);
50 nexttile
51 imshow(file3);title('Detected letter')
52 nexttile
53 imshow(y);title('Written letter')
54 nexttile([1 2])
55 imshow(file2);title('Corresponding FSL Letter')
56

```

Figure 4.1. Code for Letter to FSL

The code in Figure 4.1 is what processes an input letter image and outputs the corresponding FSL gesture. The first step in determining the FSL gesture of the input letter image is by converting it into a black and white image. This is necessary as the

reference images used are in black and white as well. Where the black and white color conversion occurs during lines 1 to 4 in Figure 4.1. Second, the reference image filenames are initialized in an array which is seen in lines 5 and 6. This array would be used in the *for loop* later on. Then, the input image is converted into a gpuarray so that the calculation is offloaded into the GPU instead of the CPU which is seen in line 8. Next, three arrays are formed to store the similarity indices used in “ssim()”, “psnr()”, and “immse()” seen in lines 9 to 11.

Next a *for loop* is created to compare each reference image to the input image provided. The filenames are read, made to be the same size as the input image, and is also converted into a gpuarray in lines 13 to 18. Then, the comparison of the images are done in lines 19 to 24 for the ssim, psnr, and immse respectively. Finally, lines 25 to 27 store the similarity indices in the three arrays initialized earlier in lines 9 to 11.

Then, in lines 31 to 36 the similarity indices with the magnitude of the similarity index of the ssim, psnr, and immse are analyzed. The highest values obtained from ssim and psnr would mean that it has the most similarity with the input image, while the immse would have the lowest value if it is the most similar to the input image.

1. Initializing arrays where similarity indices are stored
2. Converting reference array images into gpuarrays
3. Comparing of sample image to each reference images using for loop
([Reference images citation](#))
 - a) Comparing using
 - (1) [Ssim](#)
 - (2) [Psnr](#)
 - (3) [Immse](#)
4. Determining FSL output
 - a) Ssim - bigger value better
 - b) Psnr - bigger value better
 - c) Immse - smaller value better
5. Conditional statements comparing ssim, psnr, immse
6. Output of input letter, detected letter, and corresponding FSR letter

B. FSL to Letter

sl3GreenProjectFSL2Letter.mlx

```
1 [y,yrgb]=imread("C:\Users\acer\Desktop\O
2 ygray=rgb2gray(y);
3 level = graythresh(ygray);
4 BW=imbinarize(ygray,'global');
5 name=["Ahand.jpg" "Bhand.jpg" "Chand.jpg
6 name=name';
7 sizen=size(BW);
8 BW=gpuArray(BW);
9 ressim=zeros([1 size(name)]);
10 resnr=zeros([1 size(name)]);
11 resimm=zeros([1 size(name)]);
12 for a=1:size(name)
13
14     file='C:\Users\acer\Desktop\DL SU SUBJECT
15     imref=imread(file);
16     imref=rgb2gray(imref);
17     imref=imbinarize(imref,'global');
18     imref=imresize(imref,sizen);
19     imref=gpuArray(imref);
20     sim1=ssim(double(BW),double(imref));
21     sim1=gather(sim1);
22     sim2=psnr(double(BW),double(imref));
23     sim2=gather(sim2);
24     sim3=immse(double(BW),double(imref));
25     sim3=gather(sim3);
26     ressim(a)=sim1;
27     resnr(a)=sim2;
28     resimm(a)=sim3;
29 end
```



```

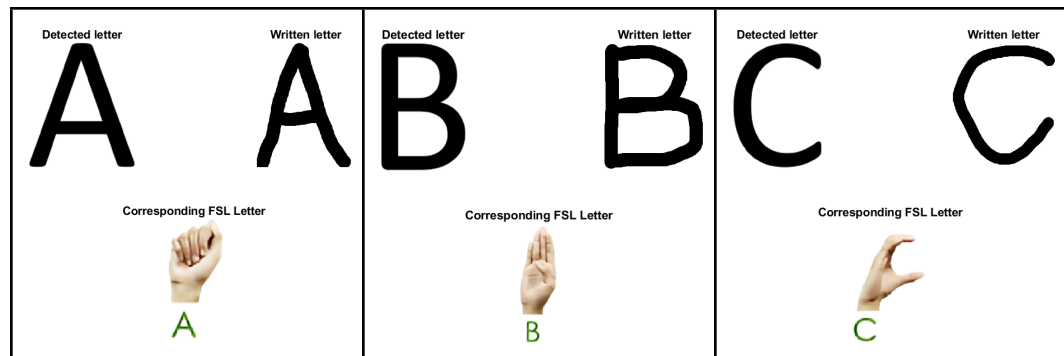
30 [Mag1,Pos1]=max(ressim);Mag1
31 name(Pos1)
32 [Mag2,Pos2]=max(resnr);Mag2
33 name(Pos2)
34 [Mag3,Pos3]=min(resimm);Mag3
35 name(Pos3)
36
37 fsrname=["A.png" "B.png" "C.png" "D.png" "E.png"
38
39 if (Pos1==Pos2)&&(Pos2==Pos3)&&(Pos1==Pos3)
40     Poss=Pos1
41 elseif (Pos1~=Pos2)&&(Pos2==Pos3)&&(Pos1~=Pos3)
42     Poss=Pos2;
43 elseif (Pos1==Pos2)&&(Pos2~=Pos3)&&(Pos1~=Pos3)
44     Poss=Pos1
45 end
46 file2='C:\Users\acer\Desktop\DLSU SUBJECTS\7TH T
47 file3='C:\Users\acer\Desktop\DLSU SUBJECTS\7TH T
48 figout=tilde layout(2,2);
49 nexttile
50 imshow(file3);title('Detected FSL Letter')
51 nexttile
52 imshow(y);title('Given FSL Letter ')
53 nexttile([1 2])
54 imshow(file2);title('Corresponding Letter')
55
56
57

```

Figure 4.2 - FSL to Letter Code

V. Result

A. Letter to FSL Results



<p>Detected letter</p> <p>D</p> <p>Written letter</p> <p>D</p> <p>Corresponding FSL Letter</p> <p>D</p>	<p>Detected letter</p> <p>E</p> <p>Written letter</p> <p>E</p> <p>Corresponding FSL Letter</p> <p>E</p>	<p>Detected letter</p> <p>F</p> <p>Written letter</p> <p>F</p> <p>Corresponding FSL Letter</p> <p>F</p>
<p>Detected letter</p> <p>G</p> <p>Written letter</p> <p>G</p> <p>Corresponding FSL Letter</p> <p>G</p>	<p>Detected letter</p> <p>H</p> <p>Written letter</p> <p>H</p> <p>Corresponding FSL Letter</p> <p>H</p>	<p>Detected letter</p> <p>I</p> <p>Written letter</p> <p>I</p> <p>Corresponding FSL Letter</p> <p>I</p>
<p>Detected letter</p> <p>J</p> <p>Written letter</p> <p>J</p> <p>Corresponding FSL Letter</p> <p>J</p>	<p>Detected letter</p> <p>K</p> <p>Written letter</p> <p>K</p> <p>Corresponding FSL Letter</p> <p>K</p>	<p>Detected letter</p> <p>L</p> <p>Written letter</p> <p>L</p> <p>Corresponding FSL Letter</p> <p>L</p>
<p>Detected letter</p> <p>M</p> <p>Written letter</p> <p>M</p> <p>Corresponding FSL Letter</p> <p>M</p>	<p>Detected letter</p> <p>N</p> <p>Written letter</p> <p>N</p> <p>Corresponding FSL Letter</p> <p>N</p>	<p>Detected letter</p> <p>Ng</p> <p>Written letter</p> <p>Ng</p> <p>Corresponding FSL Letter</p> <p>Ng</p>

<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p>


<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 		
--	--	--

Table 1. Letter to FSL Results

<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 
<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 	<p>Detected letter</p> <p>Written letter</p> <p>Corresponding FSL Letter</p> 









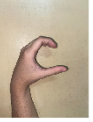















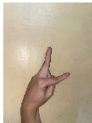



















<p>Detected letter</p> <p>P</p> <p>Written letter</p> <p>B</p> <p>Corresponding FSL Letter</p>  <p>P</p>	<p>Detected letter</p> <p>P</p> <p>Written letter</p> <p>R</p> <p>Corresponding FSL Letter</p>  <p>P</p>	<p>Detected letter</p> <p>T</p> <p>Written letter</p> <p>I</p> <p>Corresponding FSL Letter</p>  <p>T</p>
---	---	---

Table 2. Letter to FSL Errors

B. FSL to Letter Results

<p>Detected FSL Letter</p>  <p>A</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>A</p>	<p>Detected FSL Letter</p>  <p>B</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>B</p>	<p>Detected FSL Letter</p>  <p>C</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>C</p>
<p>Detected FSL Letter</p>  <p>D</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>D</p>	<p>Detected FSL Letter</p>  <p>E</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>E</p>	<p>Detected FSL Letter</p>  <p>F</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>F</p>
<p>Detected FSL Letter</p>  <p>G</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>G</p>	<p>Detected FSL Letter</p>  <p>H</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>H</p>	<p>Detected FSL Letter</p>  <p>I</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>I</p>

<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>J</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>K</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>L</p>
<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>M</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>N</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>N</p>
<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>Ng</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>O</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>P</p>
<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>Q</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>R</p>	<p>Detected FSL Letter</p>  <p>Given FSL Letter</p>  <p>Corresponding Letter</p> <p>S</p>






















<p>Detected FSL Letter</p>  <p>T</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 	<p>Detected FSL Letter</p>  <p>U</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 	<p>Detected FSL Letter</p>  <p>V</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 
<p>Detected FSL Letter</p>  <p>W</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 	<p>Detected FSL Letter</p>  <p>X</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 	<p>Detected FSL Letter</p>  <p>Y</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 
<p>Detected FSL Letter</p>  <p>Z</p> <p>Given FSL Letter</p>  <p>Corresponding Letter</p> 		

Table 3. FSL to Letter

VI. Discussion

A. Letter to FSL Results

Given the sample image, prior to determining the corresponding FSL letter, it is processed first and converted into a binary image. `Rgb2gray(I)` is utilized to convert the colormap of the image into grayscale, which is then further processed by `imbinarize(I)`, which converts all the values above the set threshold to 1 and the rest to 0, where 1 is for white and 0 is for black. However, given that `rgb2gray(I)` is reliant upon the quality of the image, for example, the colormap will retain the luminance of the signals but would be prone to error due to noise found within the image. There is also a certain limit to the threshold settings where some pixels might be mistakenly converted to the wrong binary value in `imbinarize(I)`. This is prevalent when the background is found to be darker than the letter. To mitigate this type of error, all the reference images had a white

background since this provided a clear contrast to the black letter. It can be seen that the letter to FSL detection is constrained by being reliant on the reference image within the set array. This is due to how the Letter to FSL detection is written, which relies on determining the most similar image within the array to the input image. It depends on the values of `ssim()`, which measures the similarity index between grayscale images, `psnr()`, which measures the peak signal-to-noise ratio, and `immse()`, which measures the mean squared error between the arrays. This limits the input image to being of similar font styles and uniform in nature.

B. FSL to Letter Results

The FSL to letter detection functions similarly to the FSL to letter detection, but it varies in a few aspects. One is the input image, where it requires the FSL image from a preset array of images; the other is the array assignment, where a new set of reference images is prepared. Lastly, it also requires that the input image be taken with ample lighting and a clear background to minimize errors. Overall, in terms of processing and functionality, it is essentially the same as the letter to FSL coding.

VII. Conclusion

In conclusion, the program was able to correctly identify and output the majority of the Filipino alphabet with their FSL, while also being able to identify and output all the FSL images and their equivalent Filipino letters. The arrays were correctly indexed and functioned properly. The output images were placed and labeled correctly in MATLAB. And the program was consistent in that it was able to run repeatedly and output the same output over the course of several runs.

The project is limited to selecting an image from the existing arrays for the program and having the reference images be from existing arrays. There was no dynamic identification of FSL hand signs. This allows the development of additional features like the use of computer vision to identify the FSL hand signs and written letters on the spot. The project is also limited by its detection algorithm, which is biased toward the reference images found in the program's array, limiting the possible input images it can correctly identify. This would provide an opportunity to add the prospect of training the program via machine learning so that it will not be limited by what has been added to its database array of images as a basis, as well as not be hindered by a lack of uniformity and image quality.

VIII. Authors Contribution

Justine Christian Carrasca - Introduction, Theoretical Considerations, Results

Chrismon Elijah Mansilungan - Abstract, Methodology, Results

IX. References

- [1] B. Hema, S. Anjum, U. Hani, P. Vanaja, and M. Akshatha, "Sign Language and Gesture Recognition for Deaf and Dumb People.," 2019.
<https://www.irjet.net/archives/V6/i3/IRJET-V6I3639.pdf>
- [2] G. K. Vaidhya and C. A. S. Deiva Preetha, "A Comprehensive Study on Sign Language Recognition for Deaf and Dumb people," *irojournals.com*.
<https://irojournals.com/tcsst/article/pdf/4/3/5>
- [3] A. S. Konwar, B. S. Borah, and C. T. Tuithung, "An American Sign Language detection system using HSV color model and edge detection," *IEEE Xplore*, Apr. 01, 2014.
<https://ieeexplore.ieee.org/document/6949942>
- [4] "Convert RGB image or colormap to grayscale - MATLAB rgb2gray," *www.mathworks.com*.
<https://www.mathworks.com/help/matlab/ref/rgb2gray.html>
- [5] "Global image threshold using Otsu's method - MATLAB graythresh," *www.mathworks.com*.
<https://www.mathworks.com/help/images/ref/graythresh.html#bvviewj6> (accessed Apr. 18, 2023).
- [6] "Binarize 2-D grayscale image or 3-D volume by thresholding - MATLAB imbinarize," *www.mathworks.com*. <https://www.mathworks.com/help/images/ref/imbinarize.html>
- [7] "Array stored on GPU - MATLAB," *www.mathworks.com*.
<https://www.mathworks.com/help/parallel-computing/gpuarray.html>
- [8] "Transfer distributed array, Composite array or gpuArray to local workspace - MATLAB gather," *www.mathworks.com*.
<https://www.mathworks.com/help/parallel-computing/gpuarray.gather.html> (accessed Apr. 18, 2023).
- [9] "Resize image - MATLAB imresize," *www.mathworks.com*.
<https://www.mathworks.com/help/matlab/ref/imresize.html>
- [10] "Calculate Structural Similarity Index (SSIM)," *Mathworks.com*, 2014.
<https://www.mathworks.com/help/images/ref/ssim.html>
- [11] "Peak Signal-to-Noise Ratio (PSNR) - MATLAB psnr," *www.mathworks.com*.
<https://www.mathworks.com/help/images/ref/psnr.html>
- [12] "Mean-squared error - MATLAB immse," *www.mathworks.com*.
<https://www.mathworks.com/help/images/ref/immse.html> (accessed Apr. 18, 2023).

[13] “Conditional Statements - MATLAB & Simulink,” *www.mathworks.com*.
https://www.mathworks.com/help/matlab/matlab_prog/conditional-statements.html