DOCUMENT READING SYSTEM FOR BLIND PEOPLE (READING EYE)

***Abstract*—Vision impairment prevents people reading books, newspapers and so forth. Accordingly, a visually impaired person needs to use alternatives when reading documents. This paper presents a mobile-based audio assistance system to read documents which contains images, tables, equations, and charts along with textual contents as a state-of-the-art solution to that problem of reading.**

***Keywords*— Text, Table, Chart, Mathematical Equation, Region, Image base content identification, cloud computing, mobile development**

I. INTRODUCTION

Throughout the history, many inventions have found their way into the market which were recognized as useful for disable people to accomplish their duties. Electronic books were also one of those early innovations. The history of electronic books goes as far as 1930s, when the American Foundation for the Blind (AFB) started recoding books for their blind consumers[1]. Though these books were available at that time, using them was laborious at best. As a solution for this matter, George Kerscher created the first Talking Book in 1985[1]. After his breakthrough, digital talking books became popular all over the blind community. Even nowadays both unsighted and sighted people enjoy listening to audio books because of their convenience. Nonetheless most of the existing audio books, e-reading apps can read only text-based contents in documents and therefore unable to recognize graphical contents like images, equations, tables and charts. This is clearly a disadvantage for blind people as it prevents them from getting further knowledge by reading documents contained with equations, charts, and so on. In addition to that, availability of Braille books with the same contents is hugely lacking as it is extremely difficult and expensive to design such books in Braille format[2]. Mainly students cannot find suitable books in this format. Therefore, as a solution this paper presents a mobile-based application named as "Reading Eye" that provides audio assistance to navigate through mobile application, detection and reading aloud image, table, equation and chart contents descriptively to the blind user. Reading Eye mobile application will assist visually impaired people to read documents independently without anyone else's support.

II. LITERATURE REVIEW

Along with major developments of technology and the researches carried out in the field of assistive technology in preceding years, people have developed several products to make reading easier and more enjoyable for unsighted people. However, to improve the reading ability and to gain knowledge for academic purposes for the visually impaired people, there is a major impediment because people have limited resources available for many types of documents, specially books.

There is a research conducted for this same purpose called Schmoozer.[3] It eliminates some of the issues in existing applications. It can read images, tables, and equations in a document except charts.

Table based content identification function in Schmoozer is to recognize the type of data table (whether it is 2-columns or 3-columns) and convert table data into meaningful digitized text. When implementing this function, authors had used HOG algorithm and SVM algorithm to extract features and to train the database. Text contents had been read using MATLAB OCR function.

A Simple Equation Region Detector for Printed Document Images in Tesseract is one of the researches that had conducted for the equation detection, which recognizes equations using Tesseract OCR engine [4]. This will basically classify the text symbols and then identify text regions and expand the text regions to read and detect the equations. However, there was no other researches carried out on solely for equation detection, since this feature compromise of new methodologies and techniques for chart-based content detection.

There were no other researches to read chart contents in a document, but one research had conducted to classify chart types using Mask RCNN [5]. In this research, authors had successfully identified chart region and the type of the chart (e.g. pie, bar, scatter plot, etc.).

III. METHODOLOGY

Methodology section describes each functionality in the mobile application in more details. Particularly the process of detecting each graphical content in a document, analyzing & reading the contents, security aspects of the application and the technologies used in implementation.

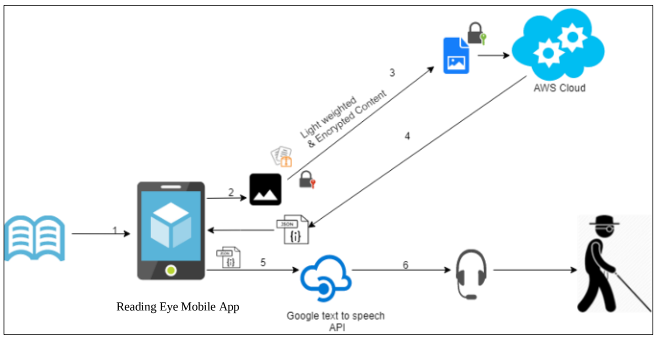


Figure 1. System overview of the Reading-Eye application

Figure 1 shows how the application works. As the outcome of this application, it generates the audio of the detailed description of the identified content for the user to listen.

A. *Digital Content Identification (CBCI)*

Identify text, images, charts, tables, and the mathematical equations uniquely and create another digital image for different identical matches.

1) *Graphical content's regions identification*

a. *Segmentation model*

Firstly, load the image as OpenCV cv2 matrix. After that color adaptation to grayscale and Gaussian Blur, Canny, Dilate, Erode functioning. After identifying the context Nodes. Then, stored nodes pass the 2D array and draw Bounding Boxes. Then crop by bounding box outline. Rename and save temp jpg file. Also, parallelly maintain an order of the bounding box node index and save to temp file.

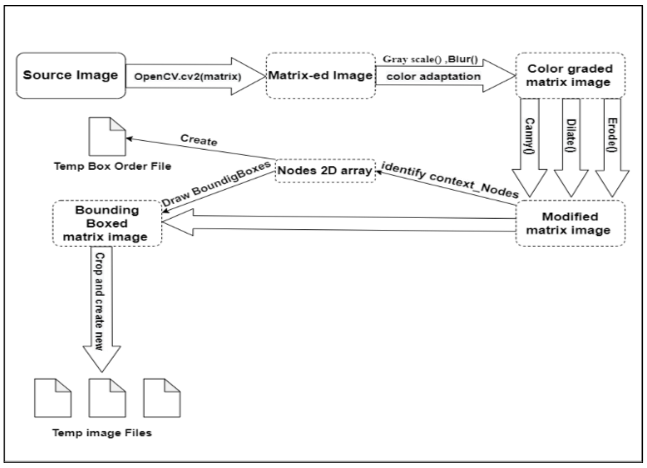


Figure 2. Segmentation model

b. *Identification model*

The main task of the identification model is to identify the type of images. Moreover, to remove the impediments of having no fully filled datasets and variation of the accuracy levels. Therefore, selected deep learning based convolutional neural networks (CNN) classification methods used for the idenication.

As the initial step, dataset was created. It includes images of paragraphs, basic mathematical equations, bar charts, pie charts, tables and natural images. Moreover, for training and validation purpose the entire dataset was divided into two parts. Labeling and annotating of the all images done using the VGG Image Annotator.

Then a new model was training using pre-trained COCO weights and after that applied ImageNet model and, PubLayNet model. After that trained model size was reduced using MobileNet.

Finally, load the model and pass the created temp image files for the prediction. After the correct prediction, file moves to the related repository.

B. *Graphical image-based content identification (GIBICI)*

This feature is used to generate a textual description about an image in a document. In here, a Convolutional Neural Network (CNN) model was trained using Flickr8K dataset to generate the description. Initially, the data (both photos and captions) should be preprocessed.
InceptionV3 pre-trained model was used to extract features from photos. After that, features were saved in a separate file to be used for training the model later. These image features are a 1-dimentional 2048 element vector. The dataset contains multiple descriptions for each photograph and these descriptions need to be cleaned before feeding the model as they could contain one letter words, punctuations and words with numbers. After cleaning, the textual data were tokenized and created a vocabulary using those words.

The CNN model contains a photo feature extractor, a sequence processor(encoder) and a decoder. Photo feature extractor uses the extracted features predicted by this model as input. Encoder is a word embedding layer for handling the text input, followed by a Long Short-Term Memory (LSTM) recurrent neural network layer. Lastly, the Decoder model merges the vectors from both input models using an addition operation. This is then fed to two LSTM layers and then to a final output Dense layer that makes a Softmax prediction over the entire output vocabulary for the next word in the sequence.

The model had to be trained for 300 epochs to increase the accuracy and to reduce the validation loss using GPUs. The trained model could be used to predict new descriptions for images. Then that description will be used to concatenate with other outputs to make the audio file later.

C. *Mathematical equations -based content identification (MEBCI)*

The purpose of this feature is to identify mathematical equations and recognize the numerical constants, characters and operators contained in the equation. Then the function will generate a description about the equation.

The function uses images saved in the equations folder for reading. The equation is read using Tesseract OCR engine. After that, image will be converted to the grayscale using thresholding [6] for dimension reduction and model complexity reduction. Furthermore, image noise will be removed by applying the erosion and dilation (morphological operations) using OpenCV library.

Finally, function will generate a description about the equation and saves it in a JSON file with separately identified numerical constants, characters and mathematical operators.

D. *Table based content identification (TBCI)*

This feature is used to read contents of the table images. It can read tables with any number of columns without adhering to only 2 or 3 columns table. The only constraint is that table should have horizontal and vertical lines. The technique used here relies on identifying the table structure hence, it does not work on tables with no border lines.

To read the text from the table image, the Tesseract OCR tool was used. Initially before reading the text, the image must be converted into grayscale, threshold and inverting. Detection of boxes in the image is the next step. Morphological operations (erode, reconstruct, dilate are examples for morphological operations) were used to complete that task. By using OpenCV library morphological operations can be done on the image. Two rectangular kernels were defined with the length based on the width of the image. First kernel is to detect horizontal lines and second to detect vertical lines. After defining kernels erode operation to detect vertical and horizontal lines in the table. Then, the result should be saved as separate images.

The final image will only contain boxes excluding text in the original table image. As a result, noise will be eliminated from the image. The next step is to find contours in the table using findContours() method in OpenCV.

It finds all the boxes in the image. These boxes should be sorted from top to bottom. Now loop over all the contours, find the location of all the boxes and crop the part which has a rectangle and save it into the folder. By cropping the box, it can ensure the text inside the box will be read accurately and in order because when reading the table directly by OCR engine it reads the words horizontally regardless of word order (especially in columns). If we directly read images using OCR, there is no way to differentiate columns from rows as all information are read as text line by line. By using this method, not only we can separate columns and rows we can use them to generate meaningful descriptions at the same time. Then the column cells (cropped cells) and row cells should be read iteratively using OpenCV and read the text in the cells using Tesseract OCR engine.

E. *Chart based content identification (CBCI)*

The CBCI function focuses on reading pie and bar charts. As the final output, the function will generate a description using the contents of each chart type.

Initially, the obtained image files will be preprocessed before proceeding to the main identification stage. Data inside the image files will be read using OCR engine. For the dimension reduction and model complexity reduction the read image will be converted to the grayscale using thresholding technique. Furthermore, the morphological operators such as, erosion and dilation will be applied to remove the noise of the image because result in pixel values that do not reflect the true intensities of the real scene with the noise of image. There are two main identification stages.

1) *Pie chart identification*

Here, several assumptions were made that pie charts do not contain 3D effects, each wedge has a distinct color, and the corresponding labels are presented adjacent to the wedge.

As the first step the loaded prepossessed image file should be used to detect the edges. The edge detection is performed through Hough transform pattern recognition algorithm [7]. When detecting the edges of pie charts Hough transform pattern recognition algorithm use to detect circle positions and match their edges.

Given in a black circle on a white background, first we should guess its radius or a range of radiuses to construct a new circle. This circle is applied on each black pixel of the original picture and the coordinates of this circle are voting in an accumulator. From this geometrical construction, the original circle center position receives the highest score and most prominent circles.

Furthermore, scikit-image open-source image processing library and the SciPy package of key algorithms and functions applied for get the count of the founded segments and segments values with the percentage of the pie chart.
Finally, by creating the subplots and unpack the created array and generates a description of the chart. It contains the identified pie chart segments with the percentage values.

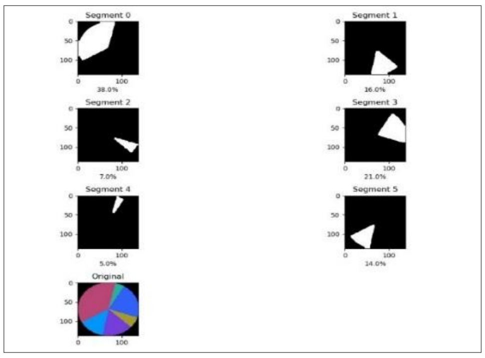


Figure 3. Pie chart segmentation and percentages with original image

2) *Bar chart identification*

Here also some assumptions were made. Such as bar chart orientation should be either horizontal or vertical, charts do not contain any 3D effects no texts inside the stacked bars and each stack has a distinct color.

As the first step the loaded prepossessed image file of the bar chart with correct orientation. Otsu Binarization is applied for threshold [6] with using various filters to smooth the image. It is done by applying the Tesseract OCR and OpenCV image processing algorithms. After that the image will be inverted.

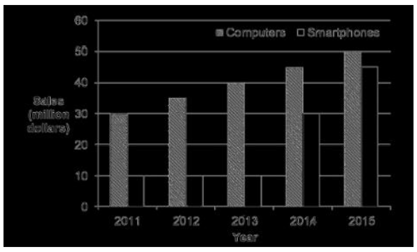


Figure 4. Otsu Binarization applied bar chart image for thresholding

Perfectly prepossessed image file should confine our search space to the bars extracted from the connected component analysis. For that purpose, local search algorithm is used as follows. It will generate the X and Y scale dimension of the bars.

- As the beginning, the midpoint of the length of two edges of the rectangle should find along with its length.

X-axis is used for the horizontally and Y-axis is used for vertically for the lengthwise in the bar chart.

- With the identification of start and end points we move along with the line made by them and detect sudden changes in the pixel values between the two adjacent points. The border or the adjacent point that separates two adjacent stacks is taken as the center of the pixel points.
- The process should continue until the finding the all the stacks inside all the bars.

Hough transform pattern recognition algorithm [7] and box detection algorithm [8] are used to detect rectangular bars separately in the bar chart. The process of applying the above algorithms and detecting the bar chart from the prepossessed image as follows.

- Defining two kernels to detect horizontal lines and vertical lines. It will define rectangular kernel with the length based on the width of the preprocessed image.
- After defining the kernels, morphological operations of the image processing are applied to detect horizontal and vertical lines.
- Getting rid of grid lines accurately to detect the bars and then no noise will occur for false bar extraction.
- Adding waiting parameters as alpha = 0.5 and beta = 1-0.5. Then it will make the quantity of an image to be added to make new image and this function helps to add two images with specific weight parameter to get a third image as summation of two image.

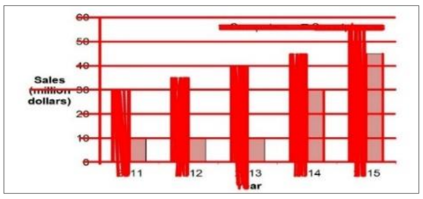


Figure 5. Horizontal and vertical lines detected, and weighted parameters applied image

As the next step, contours for image, this will detect all the bars and sort them from top to bottom by sorting counters using OpenCV.

Furthermore, loop over all the counters to find the all the boxes and crop the part which has a rectangle and save those into a new folder. The folder contains the exact detected bars of images.

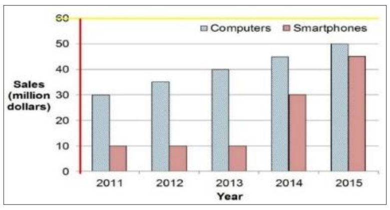


Figure 6. Bounding box identified final image

Finally, generates a description using the identified bounding boxes of the bar chart. It contains the identified bar charts with maximum and minimum scaled values of the X and Y axes. The description will be merged with outputs of other function as necessarily to generate the final audio file.

F. *Securing the client-server connection (SCSC)*

AES (Advanced Encryption Standard) is a strong encryption that can be used for securing the client server communication. However, such strong algorithms depend on more processing power. When implementing this algorithm to a mobile application, the efficiency must be considered. Therefore, application was implemented with hybrid AES algorithm with more efficiency than the original.

AES has four main functions as such as,

1) SubBytes.	2) ShiftRows.
3) MixColumns.	4) AddRoundKey.

Each of the mentioned function needs different processing power. Out of those functions MixColumns function needs more processing power because of the complicated arithmetic operations performed by the function.

It operates column-by-column on the block, treating each column as four-term polynomial (polynomials over GF(28) and multiplied modulo x 4 + 1 with a fixed polynomial a(x). following figure 7 shows how bytes in a column get replaced.

$$
\begin{aligned}
s'_{0,c} &= (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c} \\
s'_{1,c} &= s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c} \\
s'_{2,c} &= s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c}) \\
s'_{3,c} &= (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}).
\end{aligned}
$$

Figure 7. Replacement of four bytes in a column

Every secure encryption algorithm must have two main functions according to NIST.

1. Confusion 2. Diffusion

In AES, confusion part is done by the SubBytes function and diffusion part is done by the ShiftRows and MixColumns functions. In DES Diffusion part is done by the permutation table. This permutation function of DES is efficient than the AES's MixColumn function. Therefore, the replacement done as the MixColumns function with Permutation function of the DES. This modification makes AES algorithm far more efficient.

Permutation table of DES is a Metrix that has different letters. The permutation function replaces plaintext blocks' letters with permutation table's letters. The creation of the permutation table is done specifically for the AES because the direct application to AES is difficult since it uses 128-bit blocks.
The creation of the table is done as an array because it is much easier than the permutation with a matrix. Following figure 8 shows created permutation table for AES.

```
pc1 = [6, 4, 15, 7, 13, 9, 11, 2, 5, 14, 8, 10, 0, 12, 1, 3]
fp  = [12, 14, 7, 15, 1, 8, 0, 3, 10, 5, 11, 6, 13, 4, 9, 2]
```

Figure 8. Permutation and reverse permutation tables for AES

After the ShiftRows function the block of matrix convert to an array. As the next step, permutation function applied. The permuted array will convert back to 128-bit matrix. The 256-bit length key was used because the length of the key is a critical fact when it’s come to the security of the algorithm. When the key length is low the possibility of brute forcing the algorithm will be high.

G. *Light weighting the images*

The captured picture it will be resized into 1000*1000 pixels that will be divided into 200*200 pieces and those image segments encrypted with 256-bit length key and will be sent to the servers. These image segments will be the size of 10KB to 15KB.Therefore the time for the encryption will be around 0.10 to 0.30 seconds per block.

IV. RESULT AND DISCUSSION

This section is allocated to discuss about the results obtained from the implemented features and accuracy of them.

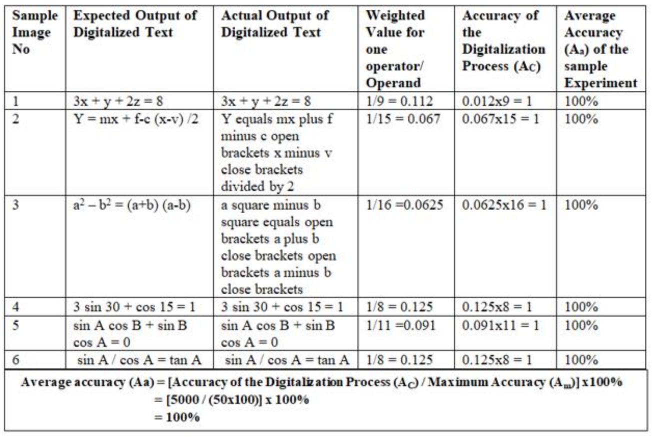
A. *MEBCI Function*

50 input images were tested to calculate the average accuracy of the mathematical equations-based content identification function. It is based on the following proposed weighted matrix.

Average accuracy (Aa) = [Correct occurrences of digitalization process (AC) / Max Accuracy (Am)] x100%

Below table contains the samples of tested experiment images with the calculation of the accuracy of the function.

Table 1. Accuracy of MBCI



Sample Image No	Expected Output of Digitalized Text	Actual Output of Digitalized Text	Weighted Value for one operator/ Operand	Accuracy of the Digitalization Process (Ac)	Average Accuracy (A_a) of the sample Experiment
1	3x + y + 2z = 8	3x + y + 2z = 8	1/9 = 0.112	0.012x9 = 1	100%
2	Y = mx + f-c (x-v) /2	Y equals mx plus f minus c open brackets x minus v close brackets divided by 2	1/15 = 0.067	0.067x15 = 1	100%
3	$a^2 – b^2$ = (a+b) (a-b)	a square minus b square equals open brackets a plus b close brackets open brackets a minus b close brackets	1/16 =0.0625	0.0625x16 = 1	100%
4	3 sin 30 + cos 15 = 1	3 sin 30 + cos 15 = 1	1/8 = 0.125	0.125x8 = 1	100%
5	sin A cos B + sin B cos A = 0	sin A cos B + sin B cos A = 0	1/11 =0.091	0.091x11 = 1	100%
6	sin A / cos A = tan A	sin A / cos A = tan A	1/8 = 0.125	0.125x8 = 1	100%

Average accuracy (Aa) = [Accuracy of the Digitalization Process (A_C) / Maximum Accuracy (A_m)] x100%
= [5000 / (50x100)] x 100%
= 100%

B. *CBCI Function*

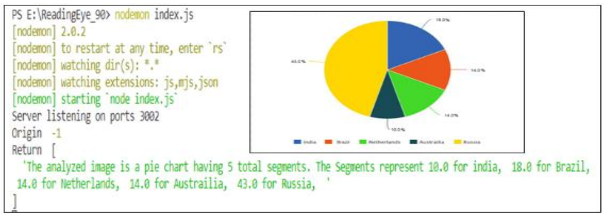
Charts -based content identification function results were generated using 40 samples of input images data as 20 pie charts and 20 bar charts. The compound function of chart detection including pie charts detection and bar chart detection shows it average accuracy as 92.5%.

Average accuracy (Aa) = [Correct occurrences of input data sample / Total input data sample] x100%

Aa Pie charts = [18/20] x 100% = 90%
Aa Bar chart = [19/20] x 100% = 95%

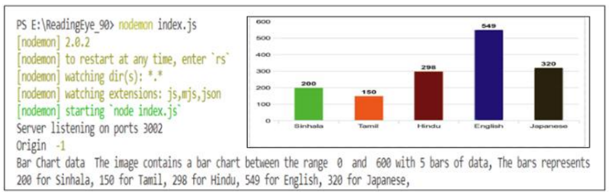
Average accuracy chart Identification = [(90+95) / 2] x 100%
= 92.5%

The sample result of the pie chart and bar chart identification function is as follows.



```
PS E:\ReadingEye_90> nodemon index.js
[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Server listening on ports 3002
Origin  -1
Return  [
  'The analyzed image is a pie chart having 5 total segments. The Segments represent 10.0 for india,  18.0 for Brazil,
 14.0 for Netherlands,  14.0 for Austrailia,  43.0 for Russia,  '
]
```

Figure 9. Sample output result of pie chart identification

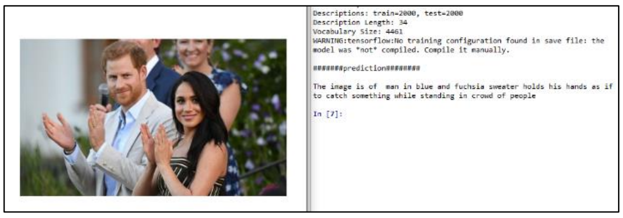


```
PS E:\ReadingEye_90> nodemon index.js
[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Server listening on ports 3002
Origin  -1
Bar Chart data  The image contains a bar chart between the range  0  and  600 with 5 bars of data, The bars represents
200 for Sinhala, 150 for Tamil, 298 for Hindu, 549 for English, 320 for Japanese,
```

Figure 10. Sample output result of bar chart identification

C. *GIBCI Function*

Below are the results that we obtained for this function.



```
Descriptions: train=2000, test=2000
Description Length: 34
Vocabulary Size: 4461
WARNING:tensorflow:No training configuration found in save file: the
model was *not* compiled. Compile it manually.

#######prediction########

The image is of  man in blue and fuchsia sweater holds his hands as if
to catch something while standing in crowd of people

In [7]:
```

Figure 11. Sample output description for an image

No. of train images in the dataset: 6000
No. of test images: 1000
Epochs: 300 Batch Size: 512 Optimizer: Adam
Loss: 0.6002
Accuracy of the model: 0.849

D. *TBCI Function*

When evaluating how good this function is, 35 images were tested, and the results are given below. Only tables with border lines were tested as this function does not work well with tables without borders.

Table 2. Accuracy of TBCI

Table Type	*#No Inputs*	*#No Correct Outputs*
2 column tables with lines	10	10
3 column tables with lines	10	10
4 or more column tables with lines	15	13

Average Accuracy (AC) = (Number of correct inputs / Number of total inputs) * 100%
= (33/35) *100%
= 94.3%

E. SCSC Function

Table 3. Time elapsed for encryption & decryption

Size of the image (KB)	Time elapsed for encryption		Time elapsed for Decryption	
	AES Original (In Seconds)	AES Hybrid (In Seconds)	AES Original (In Seconds)	AES Hybrid (In Seconds)
10	0.24	0.17	0.37	0.16
18	0.42	0.31	0.57	0.31
51	1.23	0.93	1.84	0.86
150	3.90	2.67	4.99	2.56
550	14.43	10.20	18.48	9.47
1702	44.47	31.30	58.31	30.60
4180	111.26	76.96	150.25	78.19
4892	128.07	89.68	174.06	92.23

Table 3 shows time elapsed for encryption with 256bit length key and time elapsed for decryption with the same key. Since we are sending images as block stream it will only take around 0.10 to 0.30 seconds to encrypt and decrypt a block. Because size of every block is around 10kb to 18kb.

F. DCI Function

125 input images were tested to calculate the average accuracy of the region identification function.

Table 4. Accuracy of DCI

Scenario	Number of occurrences	Number of Occurrence meets expected output
Identify image regions	25	24
Identify text regions	25	25
Identify chart regions	25	24
Identify table regions	25	25
Identify equation regions	25	25
Average Accuracy (AC) = (Number of occurrence / Number of Occurrence meets expected output) * 100% **= (98/100) *100% = 98%**		

The sample result of the DCI function as follows.

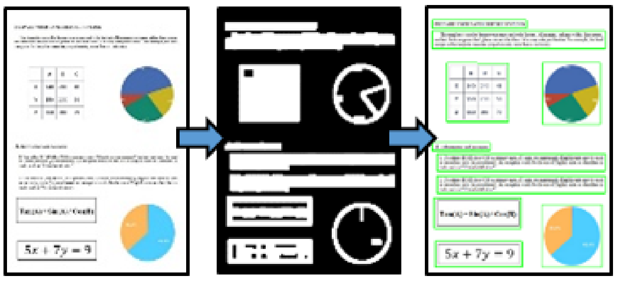


Figure 12. Sample output of DCI function

V. CONCLUSION AND FUTURE WORKS

This research was conducted to propose a solution that is capable of reading images, charts, equations and tables in a printed document. The process consists of region segmentation, secure client server connection, region analysis and content reading and so on. Authors successfully implemented the proposed solution with new functionalities which will help the visually impaired community immensely.

In future, this research can be further enhanced by extending the existing table reading function to read tables without border lines, extending the chart reading function to read other chart types such as line charts, area charts, scatter plots and including a new feature to read graphs.

REFERENCES

[1] "George Kerscher: A Pioneer in Digital Talking Books Still Forging Ahead | AccessWorld | American Foundation for the Blind", Afb.org. [Online]. Available: https://www.afb.org/aw/2/3/15031. [Accessed: 23- Aug- 2019].

[2] "Braille versions of textbooks help blind college students succeed - Marketplace", Marketplace, 2019. [Online]. Available: https://www.marketplace.org/2017/10/12/braille-versions-textbooks-help-blind-college-students-succeed/. [Accessed: 09-Aug-2019].

[3] N. Gamage, K. Jayadewa, S. Senanayake, K. Udeshitha and J. Jayakody, "Audio Assistance for Vision Impaired Individual To Recognize Graphical Content on Print Disable Documents", Annual conference 2017 - IET - Sri Lanka, 2017 [Accessed: 10- Aug- 2019].

[4] "A Simple Equation Region Detector for Printed Document Images in Tesseract - IEEE Conference Publication", Ieeexplore.ieee.org, 2019. [Online]. Available: https://ieeexplore.ieee.org/iel7/6627713/6628563/06628621.pdf. [Accessed: 22- Aug- 2019].

[5] J. Amara, P. Kaur, M. Owonibi and B. Bouaziz, "Convolutional Neural Network Based Chart ImageClassification.." [Accessed: 23-Aug-2019].

[6] Nelli, F. (2017). OpenCV & Python - The Otsu's Binarization for thresholding. [online] Meccanismo Complesso. Available at: https://www.meccanismocomplesso.org/en/opencv-python-the-otsus-binarization-for-thresholding/ [Accessed 1 Mar. 2020].

[7] Scikit-image.org. (n.d.). Circular and Elliptical Hough Transforms — skimage v0.12.2 docs. [online] Available at: https://scikit-image.org/docs/0.12.x/auto_examples/edges/plot_circular_elliptical_hough_transform.html [Accessed 2 Feb. 2020].

[8] Medium. (n.d.). A Box detection algorithm for any image containing boxes.. [online] Available at: https://medium.com/coinmonks/a-box-detection-algorithm-for-any-image-containing-boxes-756c15d7ed26 [Accessed 4 Feb. 2019].