A Project Report on

"REAL TIME IMAGE TEXT EXTRACTION AND SPEECH SYNTHESIS USING MATLAB"

Submitted in partial fulfillment of the requirements of the degree of

Bachelor of Technology

Submitted by

B.Sarath -(O180081)

B.Renuka Devi -(O180873)

T.Krishna Veni -(O180366)

P.Naveen -(O180892)

Under the Supervision of

Mr. G.Mala Kondaiah M.Tech

Assistant Professor

Department of ECE

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

ONGOLE CAMPUS

2023-24

APPROVAL SHEET

This report entitled "Real-Time Image Text Extraction and Speech Synthesis Using MATLAB" by B.Sarath -(O180081), B.Renuka Devi -(O180873), T.Krishna Veni - (O180366), P. Naveen -(O180892) is approved for the degree of Bachelor of Technology in Electronics and Communication Engineering.

Examiner(s)

_____

_____

_____

Supervisior(s)

_____

_____

_____

Chairman _____

_____

Date: _____

Place:_____

CANDIDATE'S DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included. I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Signature

B.Sarath -(O180081) _____

B.Renuka Devi -(O180873) _____

T.Krishna Veni -(O180366) _____

P.Naveen -(O180892) _____

Date:_____

DEPARTMENT OF ELECTRONICS ANDCOMMUNICATION ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

ONGOLE CAMPUS

Certificate

This is to certify that the report entitled "Real-Time Image Text Extraction and Speech Synthesis" Using MATLAB"" submitted by B.Sarath - (O180081), B.Renuka Devi- (O180873) T.Krishna Veni- (O180366) and P.Naveen- (O180892) in partial fulfilment of the requirements for the award of Bachelor of Technology in Electronics and Communication Engineering is a bonafide work carried by them under my supervision and guidance.

Project Internal Guide

Mr.G.Malakondaiah M.Tech

Assistant Professor

Department of ECE

Head of the Department

Mr.G.Bala Nagireddy M.Tech

Assistant professor

Department of ECE

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **Mr.G.Malakondaiah M.Tech,** my project guide for valuable suggestions and keen interest throughout the progress of my course and research.

I am grateful to, **Mr.G.Bala Nagireddy M.Tech HOD** of **Electronics & Communication Engineering**, for providing excellent computing facilities and a congenial atmosphere for progressing with my project.

I would like to thank **Prof. B.Jayarami Reddy Garu**, director of **RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES, ONGOLE** for providing all the necessary resources for the successful completion of my course work. At last, but not the least I thank my teammates and other students for their physical and moral support.

With Sincere Regards,

B.Sarath -O180081

B.Renuka Devi -O180873

T.Krishnaveni -O180366

P.Naveen -O180892

Date:_____

ABSTRACT

In this project, we aim to implement a text to speech synthesis which makes the user improve efficiency in delivering messages in a proficiency and straight-forwarded entry, where this project undergoes in three steps pre-processing, processing and post processing. This addresses the need to extract text by edge based algorithm, mathematical morphology method and connected component algorithm which are previously used.

Now, we use Novel text detection algorithm which employs edge-enhanced maximally stable external regions as basic letter candidates. These candidates are then filtered using geometric and stroke width information to exclude non-text objects. And these algorithms are done in pre-processing and processing. In post processing we extract text from text regions which are done in above two methods by using Optical character recognition (OCR).Finally, extracted and recognized text is converted to speech.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

1.1 Introduction

In recent years, advancements in artificial intelligence and computer vision have revolutionized the way we interact with digital information. This project focuses on the integration of real-time image text extraction and speech synthesis using MATLAB, a powerful and versatile platform for scientific computing and algorithm development. By leveraging MATLAB's extensive toolsets, this project aims to create an efficient and accurate system capable of extracting textual information from images or videos in real-time and converting it into natural speech, enhancing accessibility and user interaction in various applications.

The project revolves around the development of a robust MATLAB-based solution that seamlessly combines real-time image text extraction and speech synthesis technologies. The system will utilize advanced image processing techniques and Optical Character Recognition (OCR) algorithms to extract text from images or videos in real-time. The extracted text will then be processed through MATLAB's speech synthesis capabilities, transforming it into clear, human-like speech output.

Image

An image is defined as a two-dimensional function F(x,y), where x and y are spatial coordinates, and the amplitude of F at any pair of coordinates (x,y) is called the intensity of that image at that point. When x,y, and amplitude values of F are finite, we call it a digital image

Image processing

Image processing is a method to convert an image into digital formant perform some operations on it, in order to get an enhanced image or to extract some useful information from it. It is a type of signal dispensation in which input is image, like video frame or photograph and output may be image or characteristics associated with that image. Usually, Image Processing system includes treating images as two dimensional signals while applying already set signal processing methods to them. It is among rapidly growing technologies today, with its applications in various aspects of a business. Image Processing forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

- Importing the image with optical scanner or by digital photography.
- Analysing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not human eyes like satellite photographs.
- Output is the last stage in which results can be altered image or report that is based on image analysis.

Purpose of image processing

The purpose of image processing is divided into 5 groups.

- Visualization – Observe the objects that are not visible.
- Image sharpening and restoration – To create a better image
- Image retrieval – Seek for the image of interest.
- Measurement of pattern – Measures various objects in an image
- Image Recognition – Distinguish the objects in an image.

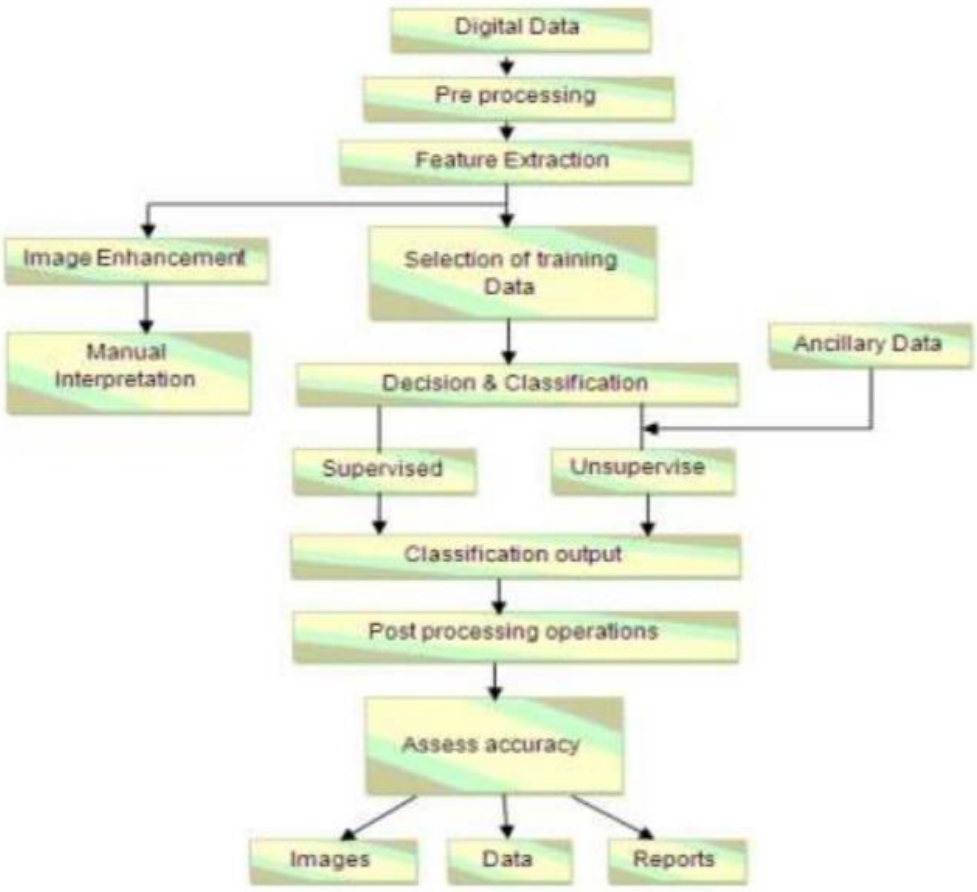

Fig:1.1. Block diagram of image processing

1.2 Incidents where it is used:

1. **Assistive Technologies**: Real-time image text extraction and speech synthesis have been utilized in assistive devices for individuals with visual impairments. These technologies enable users to capture text from various sources like books or signs and have it read aloud in real-time, enhancing accessibility and independence.

2. **Language Translation Apps**: Many language translation apps use real-time image text extraction to capture foreign language text from images and then utilize speech synthesis to translate and read the translated text aloud, facilitating communication in different languages.

3. **Mobile Applications**: Numerous mobile applications employ these technologies for functionalities like translating restaurant menus, street signs, or product labels. Users can capture the text using their phone's camera, and the app provides instant audio feedback by synthesizing the text into speech.

4. **Education and Learning Tools**: Real-time text extraction and speech synthesis are integrated into educational apps and tools. These applications help students with learning disabilities by converting textbooks and educational materials into audio formats, making learning more accessible and engaging.

5. **Accessibility in Public Spaces**: Public spaces such as airports and museums use real-time image text extraction and speech synthesis to assist visitors. For instance, visitors can capture text-based information about exhibits or flight schedules using their smartphones, and the content is converted into speech for easy comprehension.

   These real-life incidents illustrate the practical applications of real-time image text extraction and speech synthesis, showcasing how these technologies are making a significant impact in various fields and improving accessibility and user experience.

CHAPTER-2

LITERARURE

2. LITERATURE

2.1 Literature

Title: "A Comparative Study of Text Detection Algorithms in Images."

Description: This research paper compares different algorithms used for text detection in images. It discusses the strengths and weaknesses of each method, providing valuable insights into the effectiveness of various techniques in the context of image text extraction.

Title: "A Survey of Text Detection and Recognition in Images and Videos."

Description: This comprehensive survey paper explores text detection and recognition techniques in both images and videos. It provides an overview of the existing methods, their applications, and challenges. The survey offers a holistic understanding of the field, aiding researchers in making informed choices for their text extraction projects.

Title: "Stroke Width Transform for Text Localization in Natural Images."

Description: This paper introduces the Stroke Width Transform (SWT), a technique used for text localization in natural images. By analyzing stroke widths, SWT helps identify text regions in complex backgrounds. The method's effectiveness in handling text of varying sizes and orientations makes it a valuable resource for text detection projects.

Title: "Scene Text Recognition: No Country for Old Men?"

Description: This research paper focuses on scene text recognition, addressing the challenges faced when recognizing text in natural scenes. It discusses advanced techniques, including deep learning models, designed specifically for reading text in complex environments. The paper provides insights into cutting-edge methods for text recognition, which can be relevant for real-time applications.

Title: "Reading Text in the Wild with Convolutional Neural Networks."

Description: This paper presents Convolutional Neural Networks (CNNs) tailored for reading text in natural scenes. It delves into the application of deep learning techniques for text recognition, highlighting the use of CNNs in handling various fonts, sizes, and styles. The research showcases the potential of neural networks in improving the accuracy and robustness of text recognition systems.

33These papers offer in-depth knowledge and methodologies related to text detection, recognition, and advanced algorithms, providing a solid foundation for your real-time image text extraction and speech synthesis project.

CHAPTER-3

EXISTING METHOD

3. EXISTING METHOD

3.1 Edge detected algorithm

Edge Detection:

Edge detection is a technique of image processing used to identify points in a digital image with discontinuities, simply to say, sharp changes in the image brightness. These points where the image brightness varies sharply are called the edges (or boundaries) of the image.

It is one of the basic steps in image processing, pattern recognition in images and computer vision. When we process very high-resolution digital images, convolution techniques come to our rescue.

Methods of Edge Detection:

There are various methods, and the following are some of the most commonly used methods-

Sobel edge detection

Laplacian edge detection

Canny edge detection

Sobel Edge Detection: This uses a filter that gives more emphasis to the centre of the filter. It is one of the most commonly used edge detectors and helps reduce noise and provides differentiating, giving edge response simultaneously.

Fig:3.1.1Sobel Edge Detection

Laplacian Edge Detection:

The Laplacian edge detectors vary from the previously discussed edge detectors. This method uses only one filter (also called a kernel). In a single pass, Laplacian detection performs second-order derivatives and hence are sensitive to noise. To avoid this sensitivity to noise, before applying this method, Gaussian smoothing is performed on the image.7

Canny Edge Detection:

This is the most commonly used highly effective and complex compared to many other methods. It is a multi-stage algorithm used to detect/identify a wide range of edges.

Convert the image to grayscale

Reduce noise – as the edge detection that using derivatives is sensitive to noise, we reduce it.

Calculate the gradient – helps identify the edge intensity and direction.

Non-maximum suppression – to thin the edges of the image.

Double threshold – to identify the strong, weak and irrelevant pixels in the images.

Hysteresis edge tracking – helps convert the weak pixels into strong ones only if they have a strong pixel around them.

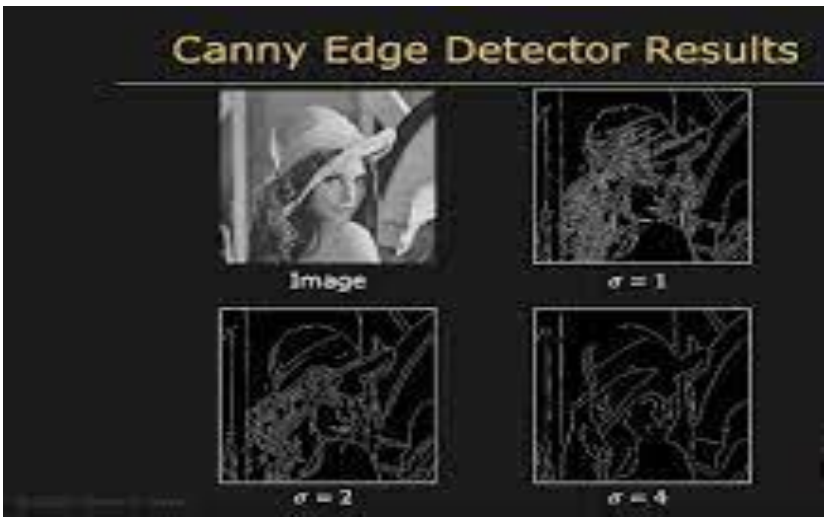

Fig:3.1.2 Canny edge detection

Flow chart of canny edge detection

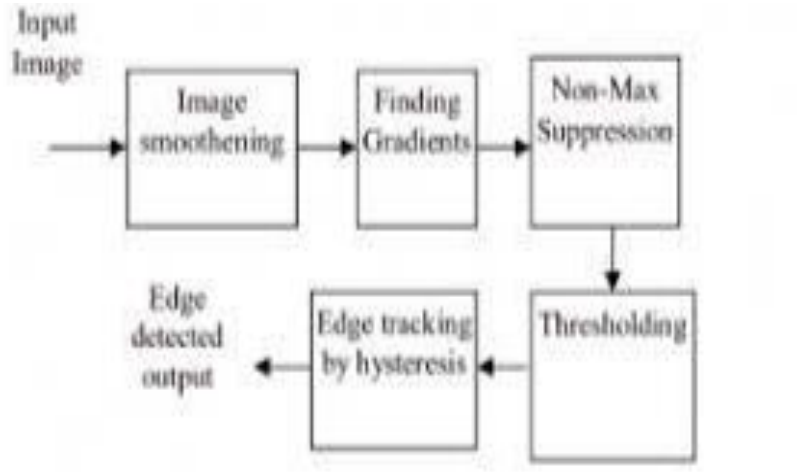

Fig : 3.1.3 flow chart of canny edge detection

Overview of edge detection

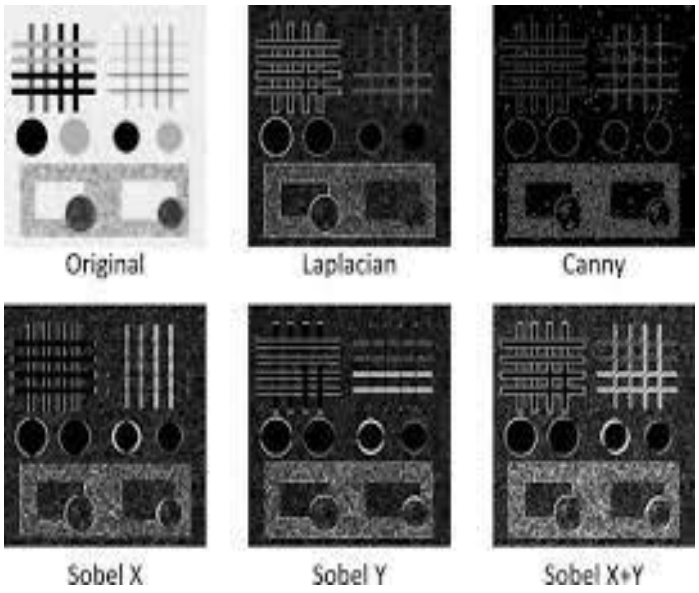

Fig 3.1.4: overview on types of edge detection

3.2 Mathematical Morphology:

The word 'Morphology' generally represents a branch of biology that deals with the form and structure of animals and plants. However, we use the same term in 'mathematical morphology' to extract image components useful in representing region shape, boundaries, etc.

Morphology is a comprehensive set of image processing operations that process images based on shapes . Morphological operations apply a structuring element to an input image, creating an output image of the same size. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors.

There is a slight overlap between Morphology and Image Segmentation. Morphology consists of methods that can be used to pre-process the input data of Image Segmentation or to post-process the output of the Image Segmentation stage. In other words, once the segmentation is complete, morphological operations can be used to remove imperfections in the segmented image and deliver information on the shape and structure of the image as shown in Fig-3.2.1

Terminologies in Morphological Image Processing

All morphological processing operations are based on mentioned terms.

Structuring Element: It is a matrix or a small-sized template that is used to traverse an image. The structuring element is positioned at all possible locations in the image, and it is compared with the connected pixels. It can be of any shape.

Fit: When all the pixels in the structuring element cover the pixels of the object, we call it Fit.

Hit: When at least one of the pixels in the structuring element cover the pixels of the object, we call it Hit.

Miss: When no pixel in the structuring element cover the pixels of the object, we call it miss.

Fig: 3.2.1 Mathematical morphology

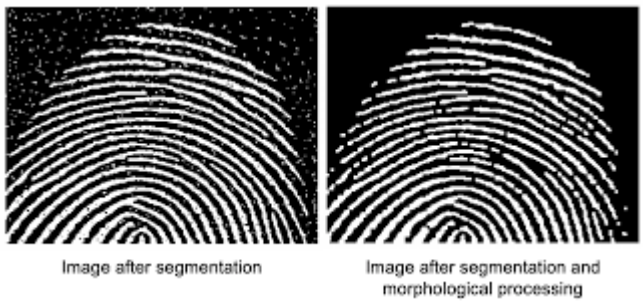

Fig 3.2.2 : Morphology

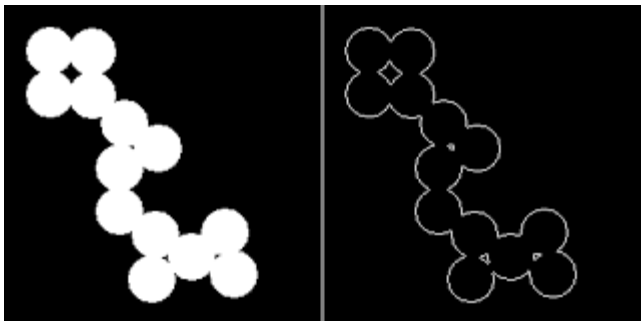

Fig:3.2.3: Mathematical morphology

3.3 Connected-component labelling:

Connected-component labeling (CCL), connected-component analysis (CCA), blob extraction, region labeling, blob discovery, or region extraction is an algorithmic application of graph theory, where subsets of connected components are uniquely labeled based on a given heuristic. Connected-component labeling is not to be confused with segmentation.

Connected-component labeling is used in computer vision to detect connected regions in binary digital images, although color images and data with higher dimensionality can also be processed.When integrated into an image recognition system or human-computer interaction interface, connected component labeling can operate on a variety of information. Blob extraction is generally performed on the resulting binary image from a thresholding step, but it can be applicable to gray-scale and color images as well. Blobs may be counted, filtered, and tracked.

Blob extraction is related to but distinct from blob detection.

A graph, containing vertices and connecting edges, is constructed from relevant input data. The vertices contain information required by the comparison heuristic, while the edges indicate connected 'neighbors'. An algorithm traverses the graph, labeling the vertices based on the connectivity and relative values of their neighbors.

Connectivity is determined by the medium; image graphs, for example, can be 4-connected neighborhood or 8-connected neighborhood.

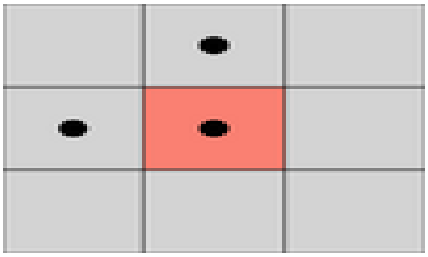

Fig:4-Connectivity

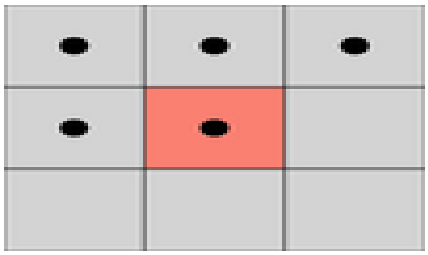

Fig:8-Connectivity

CHAPTER-4

PROPOSED ALGORITHM

4. PROPOSED ALGORITHM

4.1 MSER (Maximally Stable External Regions):

MSER is a method for blob detection in images. The MSER algorithm extracts from an imagea number of co-variant regions, called MSERs: an MSER is a stable connected component ofsome gray-level sets of the image .

- MSER is based on the idea of taking regions which stay nearly the same through a wide rangeof thresholds.
- All the pixels below a given threshold are white and all those above or equal are black.
- If we are shown a sequence of thresholded images It with frame t corresponding to
- threshold t, we would see first a black image, then white spots corresponding to local
- intensity minima will appear then grow larger.
- These white spots will eventually merge, until the whole image is white.
- The set of all connected components in the sequence is the set of all extremal regions.
- Optionally, elliptical frames are attached to the MSERs by fitting ellipses to the regions. Thoseregions descriptors are kept as features
- The word extremal refers to the property that all pixels inside the MSER have either higher
- (bright extremal regions) or lower (dark extremal regions) intensity than all the pixels on its
- outer boundary

This operation can be performed by first sorting all pixels by grayvalue and then incrementally adding pixels to each connectedcomponent as the threshold is changed. The area is monitored.

Regions such that their variation wrt the threshold is minimalare defined maximally stable:

- Let's make all the pixels below a threshold white. The
- others black
- Considering a sequence of thresholded images with increasing thresholds sweeping from black to white we pass from a black image to images where white blobs appear and grow larger by merging, up to the final image.
- Over a large range of thresholds the local binarization is stable and shows some invariance to affine transformation of image intensities and scaling.

MSER processing:

The MSER extraction implements the following steps:

- Sweep threshold of intensity from black to white, performing a simple luminance
- thresholding of the image
- Extract connected components ("Extremal Regions")
- Find a threshold when an extremal region is "Maximally Stable", i.e. local minimum of the relative growth of its square. Due to the discrete nature of the image, the region below above may be coincident with the actual region, in which case the region is still deemed maximal.
- Approximate a region with an ellipse (this step is optional)

- Keep those regions descriptors as features
- However, even if an extremal region is maximally stable, it might be rejected if:
- it is too big (there is a parameter MaxArea);
- it is too small (there is a parameter MinArea);
- it is too unstable (there is a parameter MaxVariation);
- it is too similar to its parent MSER.

4.2 Geometric properties

Eccentricity:

The eccentricity is a disparity on an image between the centre of the projected object and the projected location of the centre of an object. The correction of the eccentricity is considered to be necessary for highly accurate measurement.

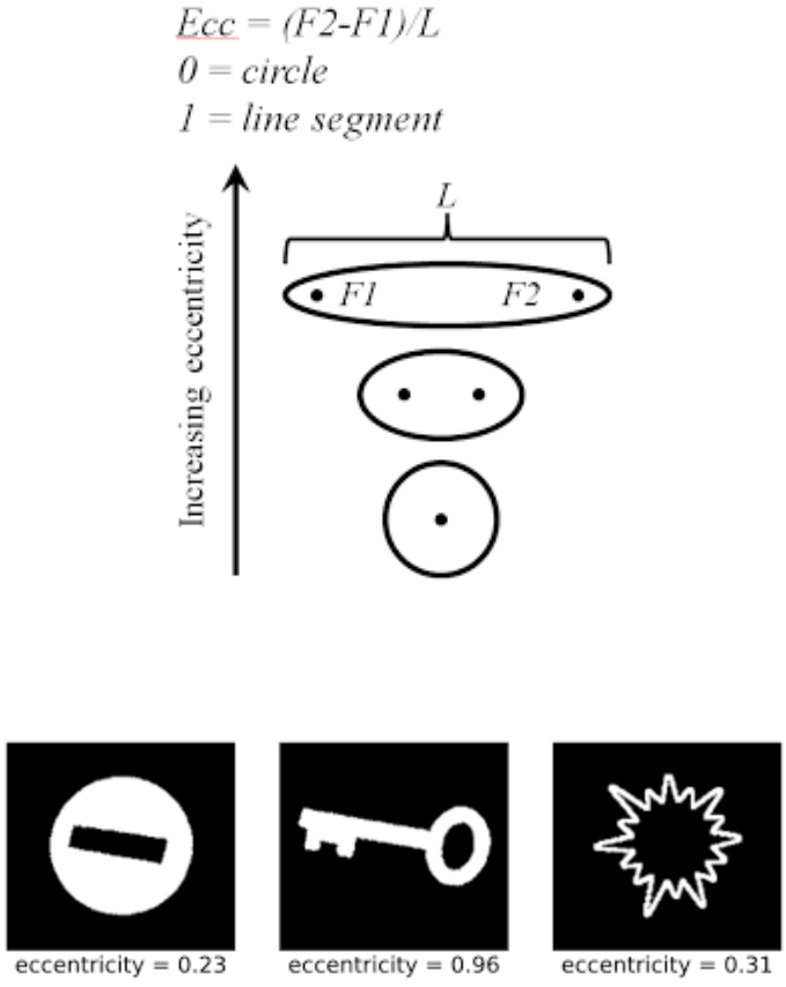

Fig 4.2.1:Eccentricity

Solidity:

Solidity is the ratio of pixels in the objects to pixels of the convex hull image. It gives a measure of the compactness of the object.

Fig4.2.2:Solidity

Euler Number:

The Euler number (also known as the Euler characteristic) is the total number of objects in the image minus the total number of holes in those objects. conn specifies the connectivity. Objects are connected sets of on pixels, that is, pixels having a value of 1.

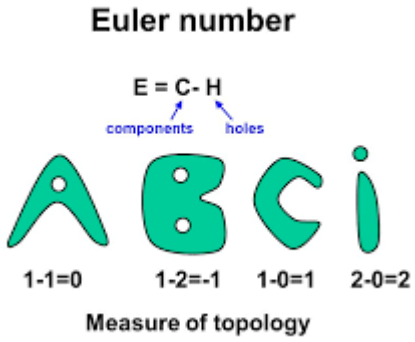

Fig4.2.3:Euler number

4.3 Stroke Width Variation:

Stroke width is another property that is used to distinguish (and filter) non-text regions from text-regions. The approach of using stroke width to aid in detecting text regions in images was introduced by Epshtein . The main idea behind using stroke width to filter non-text region is based around on the fact that usually text vs other elements in an image have constant stroke width. This means that binary images can be transformed into stroke width images (skeleton images) and these skeletal images are used to calculate the stroke width variation.

This ratio can be used with a max variance threshold to filter non-text regions. Because stroke widths are calculated on a per pixel basis text detection systems that use this technique can detect text in a way that is insensitive to font, color,size, language and orientation.

Below is a brief discussion of the stroke width related values that are calculated in the text detection system implemented for this project.

Stroke Width Variation Metric**:**

Standard Deviation of Stroke Width Values/Mean of Stroke Width Values

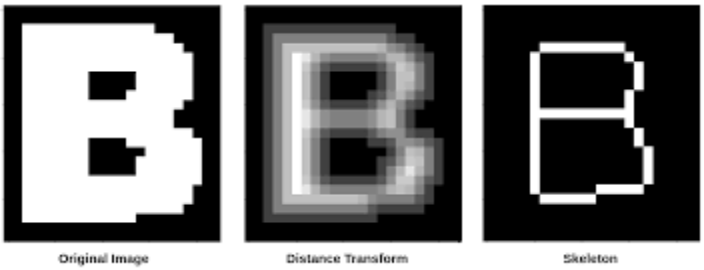

Fig 4.3.1 Stroke Width Transform-1

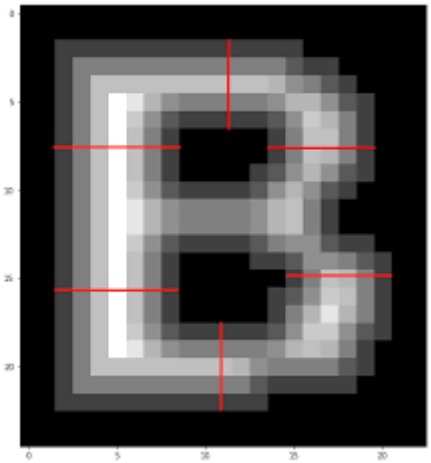

Fig4.3.2 Stroke Width Transform-2

4.4 Optical Character Recognisation:

Optical Character Recognition (OCR) is the process that converts an image of text into a machine-readable text format. For example, if you scan a form or a receipt, your computer saves the scan as an image file. You cannot use a text editor to edit, search, or count the words in the image file. However, you can use OCR to convert the image into a text document with its contents stored as text data.

How does OCR work?

The OCR engine or OCR software works by using the following steps:

Image acquisition

A scanner reads documents and converts them to binary data. The OCR software analyzes the scanned image and classifies the light areas as background and the dark areas as text.

Preprocessing

The OCR software first cleans the image and removes errors to prepare it for reading. These are some of its cleaning techniques:

Deskewing or tilting the scanned document slightly to fix alignment issues during the scan.

Despeckling or removing any digital image spots or smoothing the edges of text images.

Cleaning up boxes and lines in the image.

Script recognition for multi-language OCR technology

Text recognition

The two main types of OCR algorithms or software processes that an OCR software uses for text recognition are called pattern matching and feature extraction.

Pattern matching

Pattern matching works by isolating a character image, called a glyph, and comparing it with a similarly stored glyph. Pattern recognition works only if the stored glyph has a similar font and scale to the input glyph. This method works well with scanned images of documents that have been typed in a known font.

Feature extraction

Feature extraction breaks down or decomposes the glyphs into features such as lines, closed loops, line direction, and line intersections. It then uses these features to find the best match or the nearest neighbor among its various stored glyphs.

Postprocessing

After analysis, the system converts the extracted text data into a computerized file. Some OCR systems can create annotated PDF files that include both the before and after versions of the scanned document.

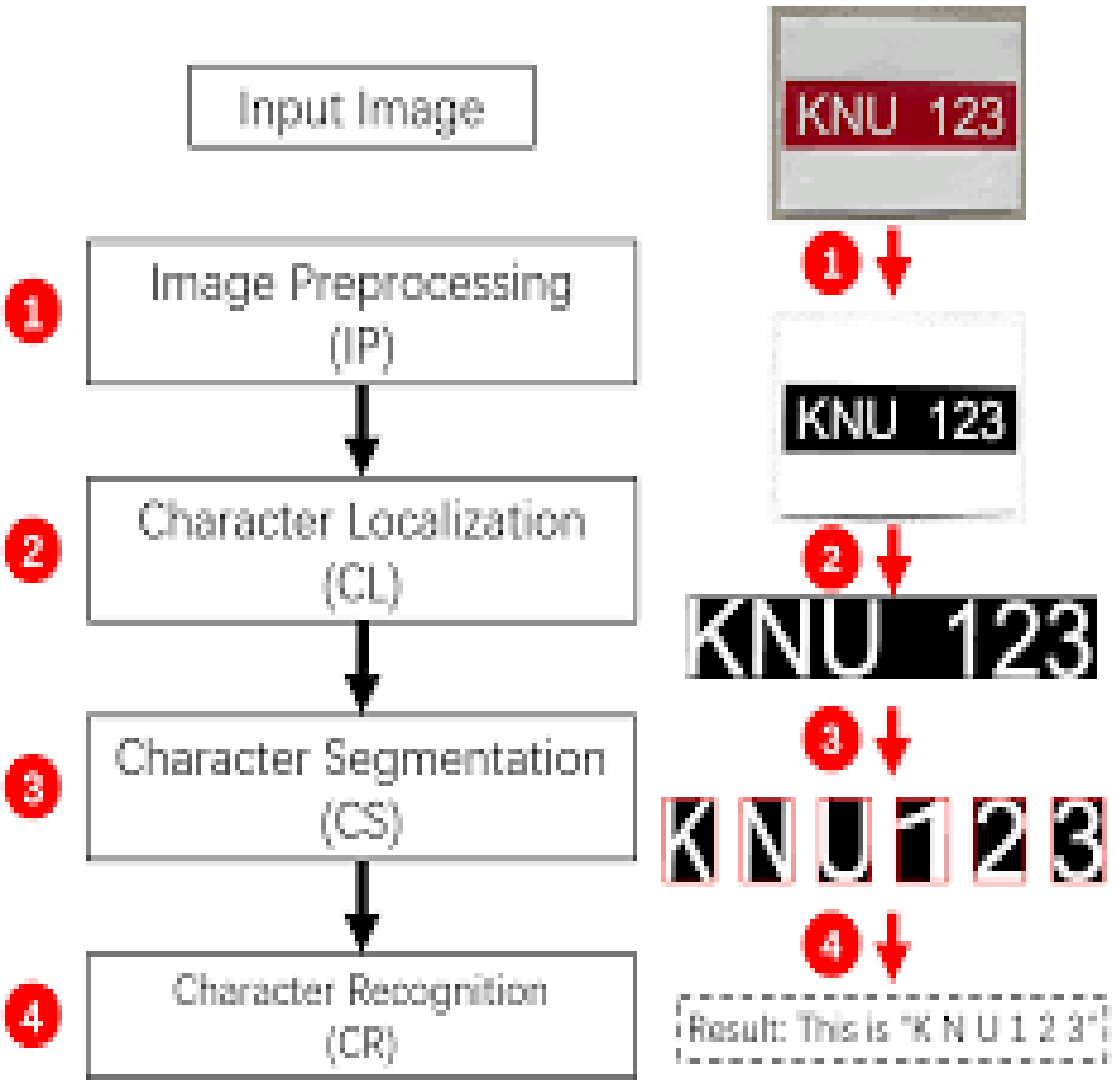

Fig 4.4.1: OCR Diagram

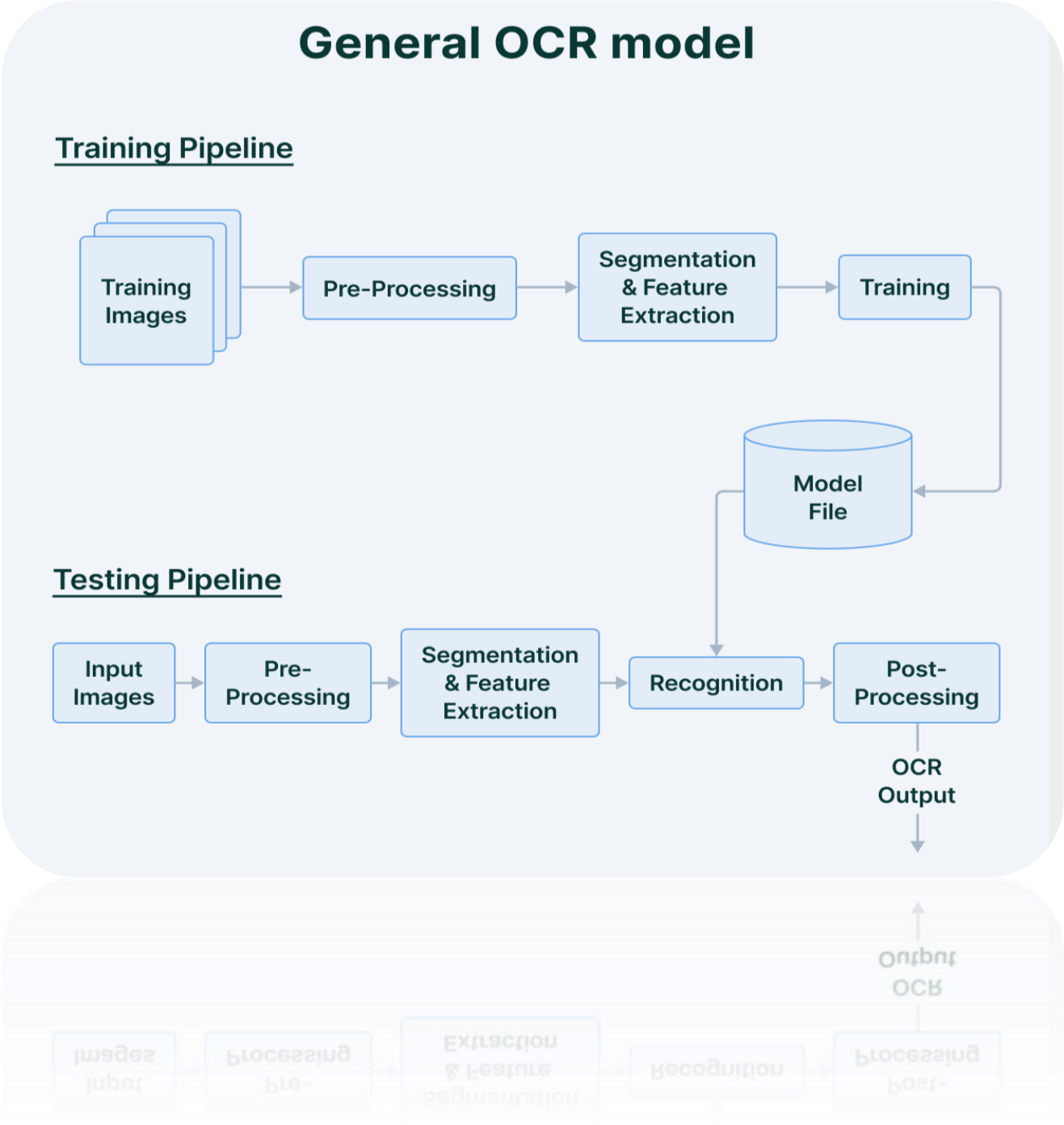

FIG.4.4.2. GENERAL FLOW MODEL

OVERALL IMPLEMENTATION

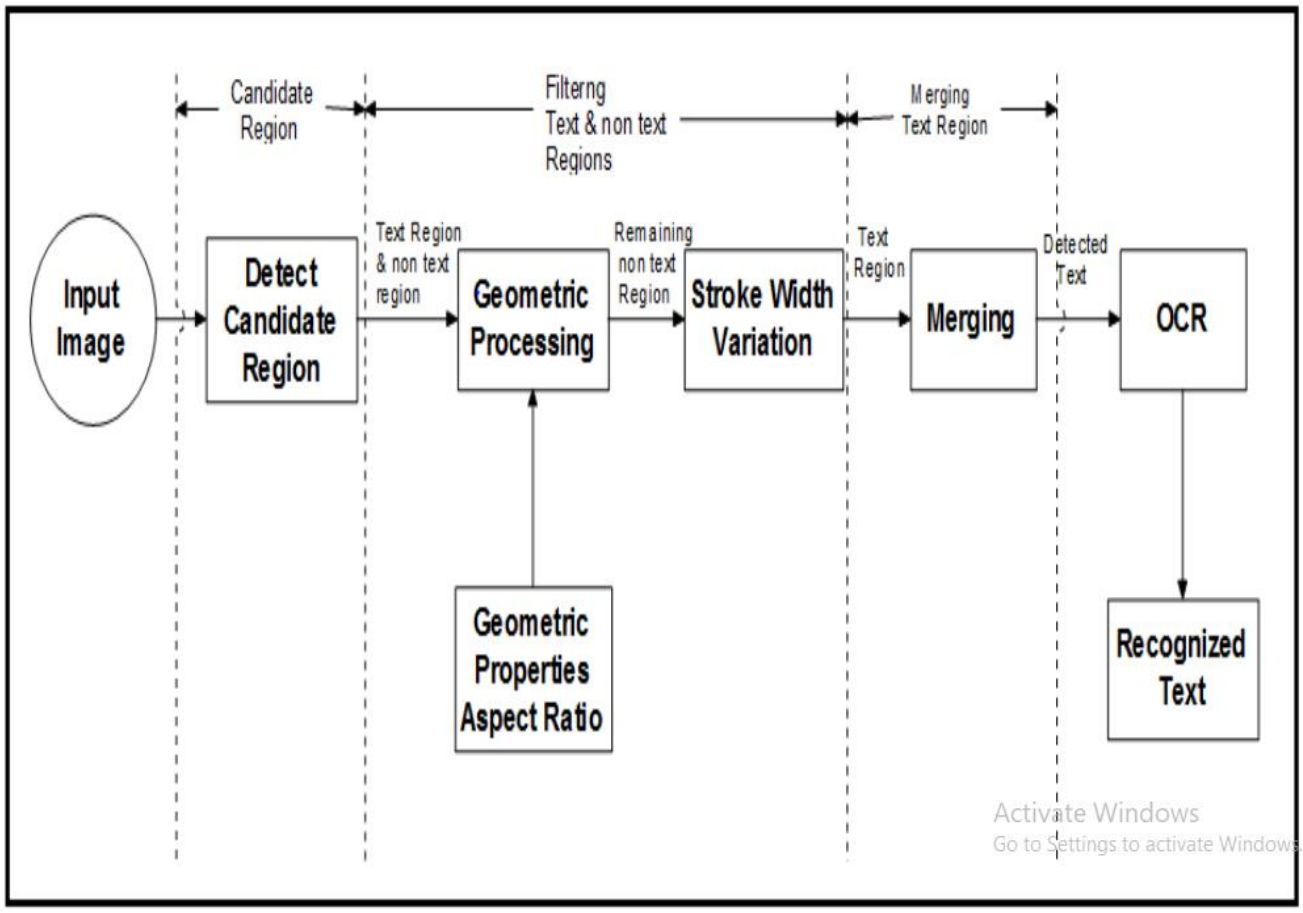

Fig.4.4.3: Over all implementation

CHAPTER-5

SOFTWARE REQUIREMENTS

5. SOFTWARE REQUIREMENTS

MATLAB 2023b

MATLAB (Matrix Laboratory) is a high-level programming language and numerical computing environment widely used in academia and industry. It offers a wide range of tools for data analysis, visualization, and algorithm development. Some of its key features include: Built-in mathematical functions: MATLAB provides a comprehensive set of mathematical functions, including linear algebra, optimization, and statistics. Interactive environment: The MATLAB environment allows users to interact with their data and algorithms in an intuitive way, making it easy to test and debug code.

Visualization tools: MATLAB provides powerful tools for data visualization, including 2D and 3D plotting, and animations. Toolboxes and add-ons: MATLAB offers a variety of toolboxes and add-ons for specific domains such as signal processing, control systems, and image processing. Integrations: MATLAB can integrate with other programming languages, such as C, C++, and Java, allowing users to combine the best of both worlds. In conclusion, MATLAB is a versatile tool for numerical computing and data analysis that can be applied to a wide range of applications. Whether you are working in academia or industry MATLAB offers a comprehensive and user-friendly environment for developing and testing your algorithms.

USES OF MATLAB

• Performing numerical linear algebra.

• Numerical computation of Matrices.

• Data analysis and visualization.

• Plotting larger data sets.

• Developing algorithms.

• Creating interfaces for the user that is the GUI- Graphical User Interface and other applications that is the API – Application Programming Interface.

CHAPTER-6

RESULTS AND DISCUSIONS

6. RESULTS AND DISCUSIONS

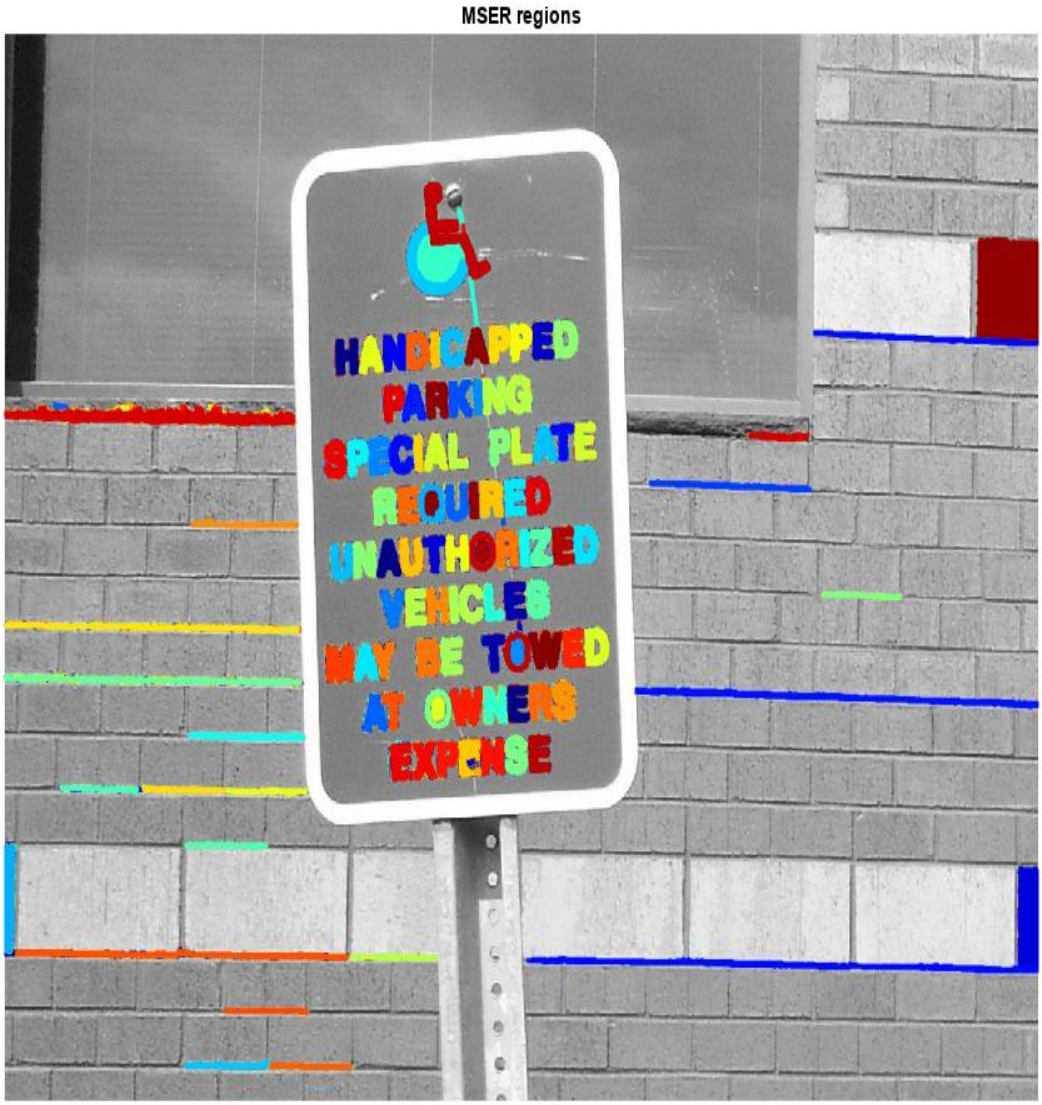

Fig 6.1 :MSER REGION DETECTION

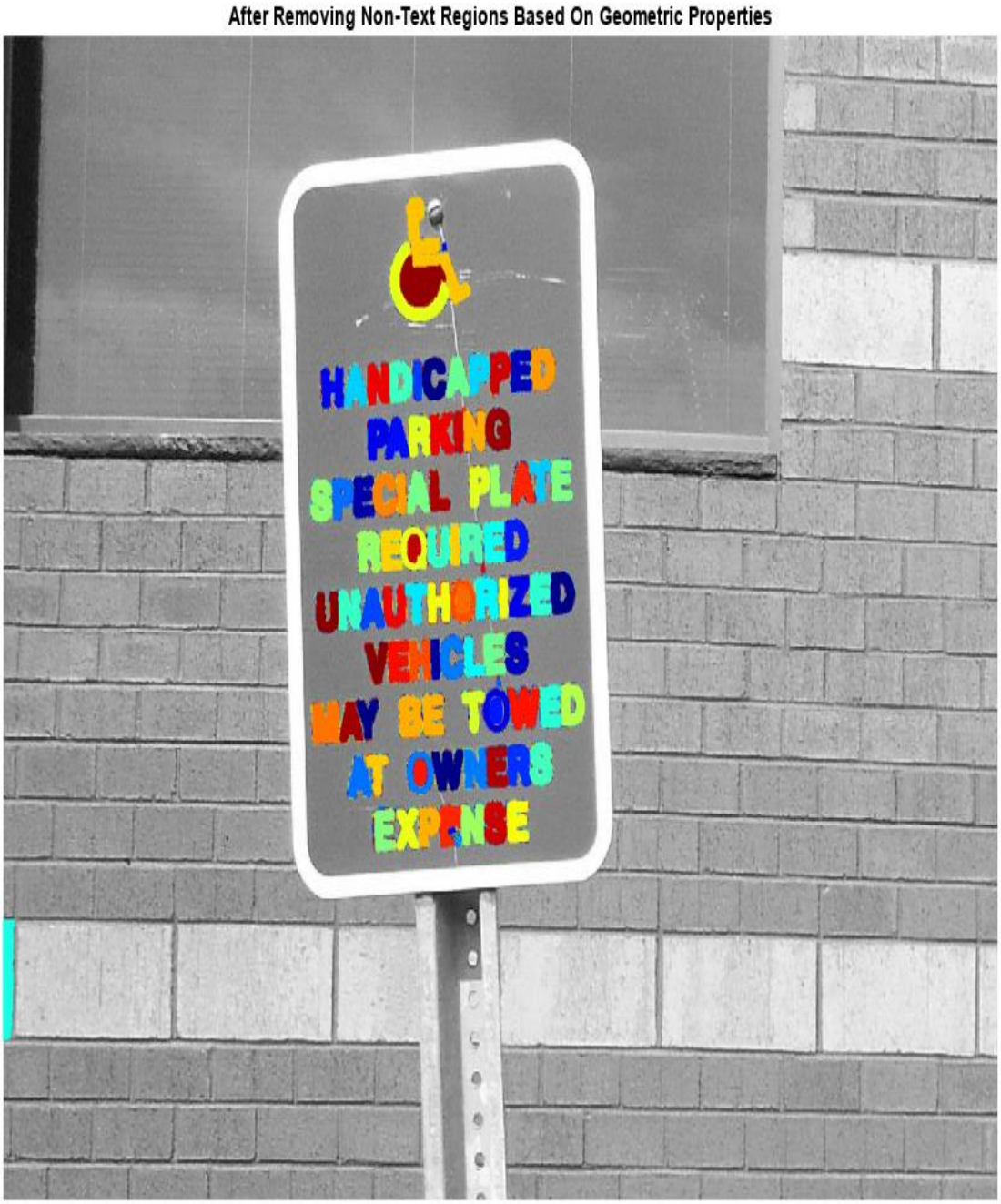

Fig 6.2: After Removing Non-Text Regions Based on Geometric Properties

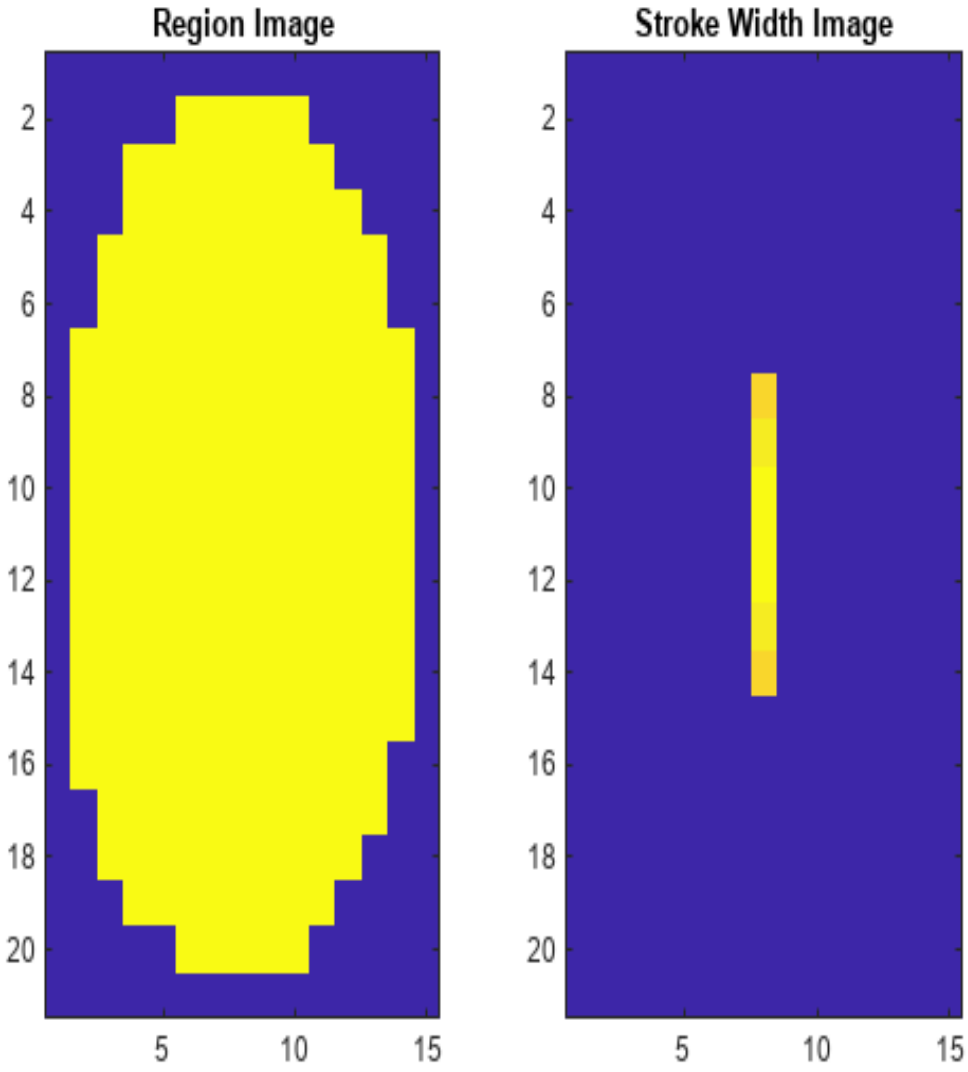

Fig 6.3 : Non-Text Regions Based on Stroke Width Variation

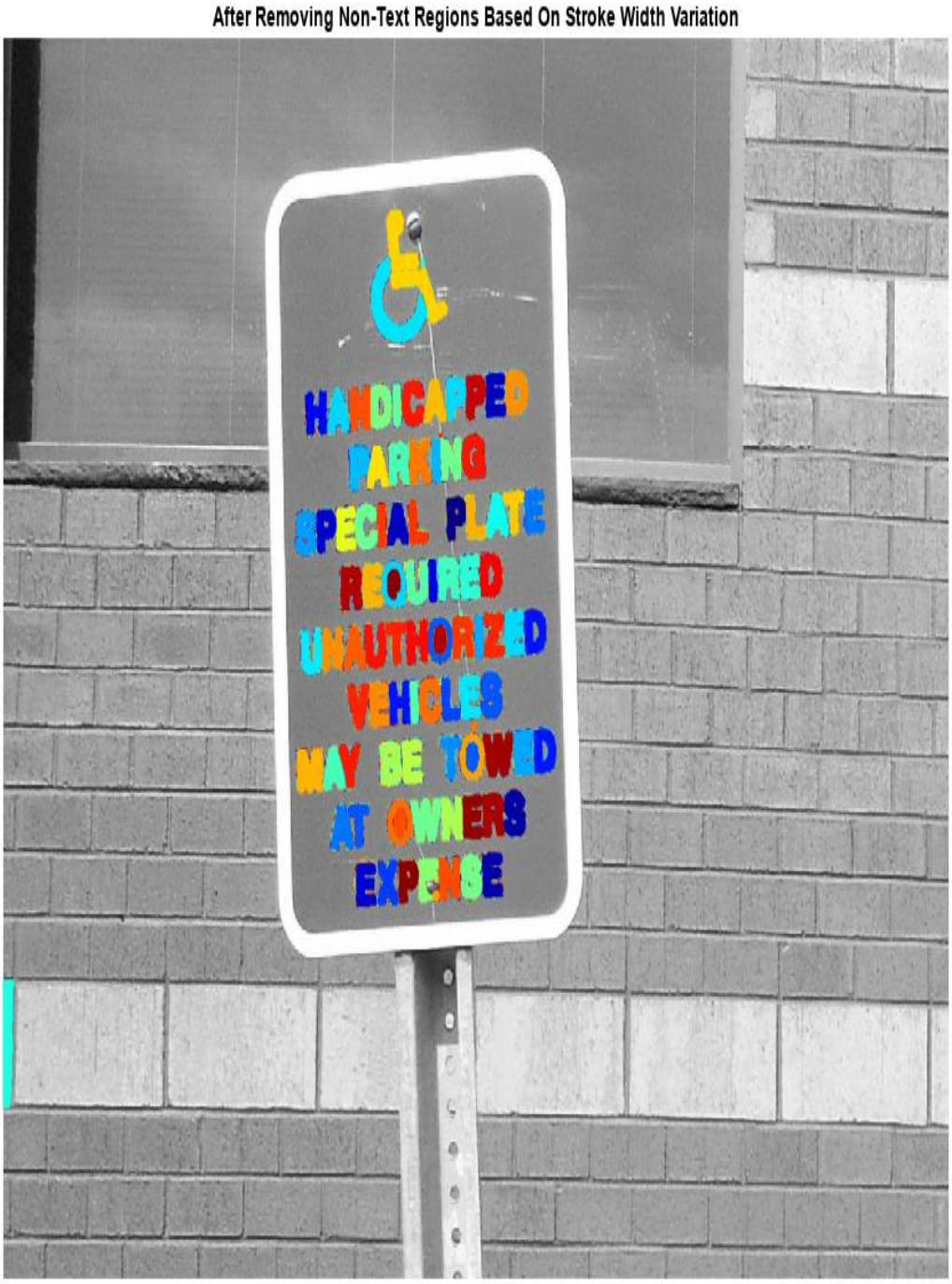

Fig 6.4 : After Removing Non-Text Regions Based on Stroke Width Variation

Fig 6.5 : Merge Text Regions For Final Detection Results

Fig 6.6 : detected text using OCR

CHAPTER-7

CONCLUSION

7. CONCLUSION

In this project , we have recognized a text from image using MSER algorithm. In this project we have used five steps to recognize text. For recognizing the text we use two important properties first is Geometric Properties and second is Stroke Width Variation Properties. First MSER algorithm detects the text regions and then OCR identifies the characters. The MSER algorithm has good text recognition performance.

Certainly! MSER (Maximally Stable Extremal Regions) is a robust method for real-time image text extraction. It detects stable regions in an image, making it suitable for text localization. By integrating MSER with speech synthesis technology, real-time image text can be extracted and converted into speech, enabling seamless interaction with visually impaired individuals or applications requiring text-to-speech functionality. This integration enhances accessibility and usability, making information more readily available to diverse user groups.

APPEX:

Final Code

```
colorImage = imread("handicapsign.jpg");
I = im2gray(colorImage);

% Detect MSER regions.
[mserRegions, mserConnComp] = detectMSERFeatures(I, ...
"RegionAreaRange",[200 8000],"ThresholdDelta",4);

figure
imshow(I)
hold on
plot(mserRegions, "showPixelList", true,"showEllipses",false)
title("MSER regions")
hold off
% Use regionprops to measure MSER properties
mserStats = regionprops(mserConnComp, "BoundingBox", "Eccentricity", ...
"Solidity", "Extent", "Euler", "Image");

% Compute the aspect ratio using bounding box data.
bbox = vertcat(mserStats.BoundingBox);
w = bbox(:,3);
h = bbox(:,4);
aspectRatio = w./h;

% Threshold the data to determine which regions to remove. These thresholds
% may need to be tuned for other images.
filterIdx = aspectRatio' > 3;
```

```
filterIdx = filterIdx | [mserStats.Eccentricity] > .995 ;
filterIdx = filterIdx | [mserStats.Solidity] < .3;
filterIdx = filterIdx | [mserStats.Extent] < 0.2 | [mserStats.Extent] > 0.9;
filterIdx = filterIdx | [mserStats.EulerNumber] < -4;

% Remove regions
mserStats(filterIdx) = [];
mserRegions(filterIdx) = [];

% Show remaining regions
figure
imshow(I)
hold on
plot(mserRegions, "showPixelList", true,"showEllipses",false)
title("After Removing Non-Text Regions Based On Geometric Properties")
hold off
% Get a binary image of the a region, and pad it to avoid boundary effects
% during the stroke width computation.
regionImage = mserStats.Image;
regionImage = padarray(regionImage, [1 1]);

% Compute the stroke width image.
distanceImage = bwdist(~regionImage);
skeletonImage = bwmorph(regionImage, "thin", inf);

strokeWidthImage = distanceImage;
strokeWidthImage(~skeletonImage) = 0;

% Show the region image alongside the stroke width image.
figure
```

```
subplot(1,2,1)
imagesc(regionImage)
title("Region Image")

subplot(1,2,2)
imagesc(strokeWidthImage)
title("Stroke Width Image")
% Compute the stroke width variation metric
strokeWidthValues = distanceImage(skeletonImage);
strokeWidthMetric = std(strokeWidthValues)/mean(strokeWidthValues);
% Threshold the stroke width variation metric
strokeWidthThreshold = 0.4;
strokeWidthFilterIdx = strokeWidthMetric > strokeWidthThreshold;

% Process the remaining regions
for j = 1:numel(mserStats)

regionImage = mserStats(j).Image;
regionImage = padarray(regionImage, [1 1], 0);

distanceImage = bwdist(~regionImage);
skeletonImage = bwmorph(regionImage, "thin", inf);

strokeWidthValues = distanceImage(skeletonImage);

strokeWidthMetric = std(strokeWidthValues)/mean(strokeWidthValues);

strokeWidthFilterIdx(j) = strokeWidthMetric > strokeWidthThreshold;

end
```

```
% Remove regions based on the stroke width variation
mserRegions(strokeWidthFilterIdx) = [];
mserStats(strokeWidthFilterIdx) = [];

% Show remaining regions
figure
imshow(I)
hold on
plot(mserRegions, "showPixelList", true,"showEllipses",false)
title("After Removing Non-Text Regions Based On Stroke Width Variation")
hold off

% Get bounding boxes for all the regions
bboxes = vertcat(mserStats.BoundingBox);

% Convert from the [x y width height] bounding box format to the [xmin ymin
% xmax ymax] format for convenience.
xmin = bboxes(:,1);
ymin = bboxes(:,2);
xmax = xmin + bboxes(:,3) - 1;
ymax = ymin + bboxes(:,4) - 1;

% Expand the bounding boxes by a small amount.
expansionAmount = 0.02;
xmin = (1-expansionAmount) * xmin;
ymin = (1-expansionAmount) * ymin;
xmax = (1+expansionAmount) * xmax;
ymax = (1+expansionAmount) * ymax;
```

```
% Clip the bounding boxes to be within the image bounds
xmin = max(xmin, 1);
ymin = max(ymin, 1);
xmax = min(xmax, size(I,2));
ymax = min(ymax, size(I,1));

% Show the expanded bounding boxes
expandedBBoxes = [xmin ymin xmax-xmin+1 ymax-ymin+1];
IExpandedBBoxes = insertShape(colorImage,"rectangle",expandedBBoxes,"LineWidth",3);

figure
imshow(IExpandedBBoxes)
title("Expanded Bounding Boxes Text")
% Compute the overlap ratio
overlapRatio = bboxOverlapRatio(expandedBBoxes, expandedBBoxes);

% Set the overlap ratio between a bounding box and itself to zero to
% simplify the graph representation.
n = size(overlapRatio,1);
overlapRatio(1:n+1:n^2) = 0;

% Create the graph
g = graph(overlapRatio);

% Find the connected text regions within the graph
componentIndices = conncomp(g);
% Merge the boxes based on the minimum and maximum dimensions.
xmin = accumarray(componentIndices', xmin, [], @min);
ymin = accumarray(componentIndices', ymin, [], @min);
xmax = accumarray(componentIndices', xmax, [], @max);
```

```
ymax = accumarray(componentIndices', ymax, [], @max);

% Compose the merged bounding boxes using the [x y width height] format.
textBBoxes = [xmin ymin xmax-xmin+1 ymax-ymin+1];
% Remove bounding boxes that only contain one text region
numRegionsInGroup = histcounts(componentIndices);
textBBoxes(numRegionsInGroup == 1, :) = [];

% Show the final text detection result.
ITextRegion = insertShape(colorImage, "rectangle", textBBoxes,"LineWidth",3);

figure
imshow(ITextRegion)
title("Detected Text")
ocrtxt = ocr(ITextRegion);
textext=ocrtxt.Text
%% TEXT TO SPEECH USING WINDOWS SPEECH RECOGINATION
caUserInput = char(textext); % Convert from cell to string.
NET.addAssembly('System.Speech');
obj = System.Speech.Synthesis.SpeechSynthesizer;
obj.Volume = 100;
Speak(obj, caUserInput);
```