REAL TIME TEXT EXTRACTION AND RECOGNITION ONLINE

A DESIGN PROJECT REPORT

Submitted by

AJANEESHWAR.S

KAMALESHWAR.A

NAVEEN KUMAR.K. R

SIDDHARTHAN.R

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENGE AND DATA SCIENCE

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM -621112

NOV 2024

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)
SAMAYAPURAM - 621112

BONAFIDE CERTIFICATE

Certified that this design project report titled **"REAL TIME TEXT EXTRACTION AND RECOGNITION ONLINE"** is the bonafide work of **AJANEESHWAR.S (REG NO: 811721243005), KAMALESHWAR.A (REG NO:811721243023), NAVEENKUMAR.K. R (REG NO:811721243038), SIDDHARTHAN.R. (REG NO: 811721243052)** who carried out the project work under my supervision.

SIGNATURE
Dr.T.Avudaiappan,M.E.,Ph.D.
HEAD OF THE DEPARTMENT
Associate Professor
Department of Artificial Intelligence
K.Ramakrishnan College of Technology(Autonomous)
Samayapuram – 621 112

SIGNATURE
Mrs. Joany Franklin,M.E.
SUPERVISOR
Assistant Professor
Department of Artificial Intelligence
K.Ramakrishnan College of Technology(Autonomous)
Samayapuram – 621 112

Submitted for the viva-voce examination held on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We jointly declare that the project report on **"REAL TIME TEXT EXTRACTION AND RECOGNITION ONLINE"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF TECHNOLOGY**. This design project report is submitted on the partial fulfilment of the requirement of the award of Degree of **BACHELOR OF TECHNOLOGY**.

SIGNATURE

_____

AJANEESHWAR.S

_____

KAMALESHWAR.A

_____

NAVEEN KUMAR.K.R

_____

SIDDHARTHAN.R

PLACE: SAMAYAPURAM

DATE:

ACKNOWLEDGEMENT

It is with great pride that we express our gratitude and in-debt to our institution **"K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)"** for providing us with the opportunity to do this project.

We are glad to credit honorable Chairman **Dr. K. RAMAKRISHNAN, B.E,** for having provided for the facilities during the course of our study in college.

We would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D,** for forwarding to our project and offering adequate duration in completing our project.

We would like to thank **Dr. N. VASUDEVAN, M.E., Ph.D**, Principal, who gave opportunity to frame the project the full satisfaction.

We whole heartily thanks to **Dr. T. AVUDAIAPPAN, M.E, Ph.D,** Head Of the Department, **ARTIFICAL INTELLIGENCE** for providing his encourage pursuing this project.

I express my deep and sincere gratitude to my project guide **Mrs. JOANY FRANKLIN, M.E.,** Assistant Professor, **ARTIFICAL INTELLIGENCE** for her incalculable suggestions, creativity, assistance and patience which motivated me to carry out the project successfully.

I render my sincere thanks to my Project Coordinator **Mr. P. B. ARAVIND PRASAD, B.E., M.Tech.,** and other staff members for providing valuable information during the course. I wish to express my special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

ABSTRACT

A Python-based computer vision application is developed using Flask and OpenCV to accurately extract characters. This system efficiently processes a wide range of visual inputs, including photos, videos, and scanned documents, extracting text characters with high precision. The extracted text supports various applications such as text recognition. The application outlines the techniques and algorithms involved in character extraction, including image pre-processing, feature extraction, and classification. Image pre-processing enhances contrast and removes noise, while segmentation methods like thresholding and contour detection isolate individual characters. These characters are then classified using machine learning models and custom deep learning model. This Flask-based system was evaluated on a dataset of images with diverse fonts and sizes, demonstrating its effectiveness in precise character extraction and classification. The web interface provides an accessible and interactive platform for real-time character recognition, making it adaptable for a variety of practical uses.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CV	Computer Vision
HCR	Handwritten Character Recognition
MLP	Multi Layered Perceptron
OCR	Optical Character Recognition

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Extracting characters from visuals and images is a crucial task in image processing and computer vision. It involves the identification and extraction of individual characters from an image, which can be used for various applications such as optical character recognition (OCR), text-to-speech conversion and natural language processing. OpenCV is a powerful open- source library that provides a wide range of functions and algorithms for image processing and analysis, making it an ideal tool for character extraction from visuals and images. The goal is to develop a system that can accurately extract the characters from a visual or image containing text in ASCII characters such that it can be recognized by a computer. OpenCV provides several techniques for character extraction from visuals and images, including contour detection, edge detection and thresholding. Contour detection is a popular technique used for extracting characters from an image or a video frame. It involves identifying the edges of the characters and creating a closed curve around them. OpenCV provides several functions for contour detection, such as find Contours and draw Contours, which can be used to locate and isolate the characters.

Edge detection is another technique used for character extraction, which involves identifying the edges of the characters using various filters such as Sobel. OpenCV provides functions for edge detection, such as Canny and Laplacian, which can be used to locate and isolate the characters.

Thresholding is a technique used for separating the characters from the background based on a threshold value. OpenCV provides functions for thresholding, such as threshold and adaptive Threshold, which can be used to isolate the characters.

Optical Character Recognition is a technique used for recognizing the characters and extracting textual information from the image or video frame. OpenCV provides several OCR libraries, such as Tesseract and OCRopus, which can be used to recognize and extract textual information accurately.

Character extraction from visuals and images has numerous applications, including automated text recognition in images, license plate recognition, document analysis and more. Automated text recognition in images is used in various applications such as image captioning, image indexing and image search. Document analysis is used for automated text recognition, indexing and archiving. License plate recognition is used for automated toll collection, parking management and law enforcement. It involves extracting the license plate number from an image or a video frame and using it for various purposes such as toll collection, parking management and law enforcement.

Character extraction from visuals and images using OpenCV is a complex and challenging task that plays a crucial role in various applications such as automated text recognition, license plate recognition and document analysis. OpenCV provides a range of functions and algorithms for character extraction, including contour detection, edge detection and thresholding. While there are several challenges associated with character extraction, there is still a lot of scope for future research in this field, including the development of more accurate and robust algorithms, integration of machine learning techniques, exploration of new applications and development of user- friendly tools and software. Overall, character extraction from visuals and images using OpenCV has enormous potential for various applications and further research in this field can lead to significant advancements in computer vision and image processing.

1.2 OBJECTIVE

The project aims to achieve several objectives in the extraction of characters from visuals and images using OpenCV. The first objective is to accurately identify and extract individual characters from various types of visual inputs. This involves implementing techniques that can precisely locate and extract the characters from the images. To improve the accuracy of character extraction, the project focuses on optimizing the pre-processing stage. OpenCV's image processing techniques will be applied to enhance the quality of the input images. By adjusting contrast, reducing noise, and improving overall image quality, the system aims to enhance the accuracy of character extraction.

Another important objective is to refine the feature extraction process. OpenCV provides advanced feature extraction methods that can identify relevant characteristics of characters, such as edges, lines, and corners. By leveraging these methods, the project aims to ensure accurate recognition of characters.

Efficient segmentation is another key objective of the project. The system will develop algorithms and approaches that effectively divide the image into individual character regions. This segmentation step is crucial for isolating each character and enabling accurate extraction. It also explores integration with Optical Character Recognition (OCR) techniques. By leveraging OpenCV's capabilities, the system aims to interpret the extracted characters as text. This integration with OCR will enhance the functionality and usefulness of the extracted characters.

To assess the effectiveness of the system, the project includes evaluation and performance analysis. Various metrics, such as accuracy, efficiency, and robustness, will be used to measure the system's performance. This evaluation process will ensure that the system meets the desired objectives and performs accurately in character extraction from visuals and images.

CHAPTER 2
LITERATURE SURVEY

2.1 TEXT IMAGE PROCESSING USING FLASK

Authors: Triveni K.M., Thanushree R., Prof. Vinaykumar Hittalamani

Year: 2023

Abstract

Most prevalent typical applications for image processing or character recognition is from text interesting area of OCR automatically enters the pertinent data after being extracted and OCR can used many fields. Several researches may works on focusing on some new methodologies and strategies that speed up the processing while providing greater recognition precision. to digitally scan the papers and recognizes text and For further information processing, make it editable and searchable. In Todays paperless offices and digitized documents was evolving into ordinary for every kind of business and work. So it is an idea to find out easy way to creating, storing, and protecting the important documents. A better understanding of visual information for humans, in addition the handling of picture data for transmission, storage, and display, autonomous machine perception. Intention of this article is to refers to where OCR software is understand the text and then converted it into editing and reading format. They developing the computer algorithms and to identify Character recognition using optical may involves the translation of text images and into editable characters codes including an ASCII.

Keywords

Image Extraction, Text Recognition, Convert It On Audio, Pdf, Future Extraction.

2.2 HANDWRITTEN TEXT RECOGNITION SYSTEM

Authors: Sushmitha, sushmitha B Poojary, varshitha, vidya K C, shilpa

Year: 2021

Abstract

Handwriting detection is a computer system or capability that receives and interprets meaningful handwriting input from a variety of sources, including paper documents, touch displays, photo graphs, and so on. One type of area pattern recognition is handwritten text recognition. Pattern recognition is a technique for grouping or categorizing data or objects into one of several categories.. We represent a novel deep neural network-based approach for offline handwritten text recognition in this study. It has gotten easier to train neural networks as a result of the vast amount of data available today and the multiple algorithmic improvements that are occurring. This handwritten character recognition system based on image segmentation.

Keywords:

Handwritten text recognition, Convolutional neural network, and recurrent neural network.

2.3 A TEXT EXTRACTION FROM VIDEO USING DEEP LEARNING

Authors: Assoc. Prof. L. Rasikannan, K. Gunal, S. Sabarinathan, S. Vigneswaran

Year: 2024

Abstract

In the modern-day virtual landscape, video content has expanded unexpectedly in quantity across diverse structures, imparting a wealth of records. However, extracting textual records from these videos accurately and efficaciously remains a great assignment. This paper proposes an approach to extract textual content from video content by way of employing a combination of Optical Character Recognition (OCR) algorithms and Convolutional Recurrent Neural Network (CRNN). By leveraging the strengths of both OCR and CRNN, our approach ambitions to beautify the accuracy and performance of text extraction from video lectures, tutorials, and educational content material. The extracted textual content serves as a precious reference for college students, enriching their mastering enjoy. This study contributes to unlocking the untapped potential of textual records embedded within video content material, thereby facilitating it.

Key Words:

Optical Character Recognition (OCR), Convolutional Recurrent Neural Network (CRNN), Text Extraction, Video Content, Information Retrie

2.4 HANDWRITTEN CHARACTER RECOGNITION TO OBTAIN EDITABLE TEXT

Authors: Vaibhav. V. Mainkar, Mr. Ajinkya B. Upade, Jyoti A. Katkar, Poonam R. Pednekar

Year: 2020

Abstract

Developing an android application for character recognition and text extraction from an image is a crucial area of research in the present era. With the rising trend of storing information from handwritten documents for future use, image capturing of the handwritten document and saving it in an image format is an easy way to store the information. To transform handwritten data into electronic format the method of Optical Character Recognition is used which involves various steps like pre- processing, segmentation, feature extraction and post-processing. OCR has been used by many researchers for recognizing characters from different styles of handwriting. This system utilizes an android phone to capture the image of the document and perform further steps by OCR to recognize the characters and extract text. However, the main challenge in developing an OCR system is to recognize the characters from different styles of handwriting. Hence, a system is designed that recognizes handwritten data to obtain editable text with good accuracy rate. The output of this system depends on the data written by the writer. Our system offers an easy way to edit and share the recognized data and it can prove to be a helpful tool for people dealing with handwritten documents.

Key Words:

Optical Character Recognition(OCR) , Text Extraction , Hand written documents.

2.5 HANDWRITTEN CHARACTER RECOGNITION USING DEEP LEARNING (CONVOLUTIONAL NEURAL NETWORK)

Authors: Altaf Ramjon , Asherl Bwatiramba , Sivakumar Venkataraman

Year: 2023

Abstract

Handwriting recognition is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touch-screens and other devices. The purpose of the research is to design develops, construct, deploy, and test a convolutional neural network (CNN) for handwriting recognition. The CNN for handwriting recognition was developed with Python and MNIST dataset was used. CNN was evaluated together with Simple Neural Network (SNN) and it was found that a CNN operating on well-tuned hardware with GPU and adequate training data can recognize numbers with an accuracy of up to 98.7 percent. The accuracy and speed of the model can be improved by expanding the dataset, increasing the number of epoch runs, and executing it on parallel hardware. Using these strategies, an accuracy of up to 99.89 percent, can be achieved.

Key Words:

Simple Neural Network , Convolutional Neural Network , Handwritten Recognition.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

OpenCV is a popular computer vision library that provides a wide range of tools for image processing and computer vision tasks. The system consists of several modules, including image pre-processing, character segmentation, feature extraction and character recognition.

The image pre-processing module is responsible for cleaning up the input image, removing any noise or artifacts and enhancing the contrast and brightness of the image. The character segmentation module then divides the pre-processed image into individual characters, which are then processed by the feature extraction module to extract relevant features such as stroke width, aspect ratio and curvature.

The final step in the system is character recognition, where the extracted features are fed into a machine learning algorithm such as a support vector machine or artificial neural network to classify the characters. The system can recognize characters in a variety of fonts and sizes and has applications in fields such as document analysis and handwriting recognition. One of the advantages of this system is it0s ability to recognize characters in noisy and low- quality images.

Additionally, OpenCV is an open-source library, which makes it easily accessible to developers and researchers. However, the system may not perform well on complex images with multiple fonts and sizes and the accuracy of the system is highly dependent on the quality of the input image.

The existing system has shown promising results in recognizing characters from images and visuals with high accuracy. However, it has limitations in recognizing handwritten characters or characters in noisy environments, where additional pre-processing techniques and feature extraction algorithms may be required.

Furthermore, the system is limited to recognizing characters in a single language or script and may require language-specific training data and models for recognition in other languages or scripts.

3.1.1 Demerits

• **Variability in handwriting** Handwriting varies significantly between individuals, making it challenging to create a universal recognition system that can accurately recognize all handwriting styles.

• **Limited language support** Most handwritten character recognition systems are developed for a limited set of languages, which means cannot be used for languages with different scripts.

• **Poor accuracy** Despite significant progress, the accuracy of handwritten character recognition systems is still not perfect and errors can occur frequently. This is especially true for cursive or irregular handwriting.

• **Complex characters** Some characters in certain languages can be very complex, making them difficult to recognize accurately by a computer system.

• **Resource-intensive** Handwritten character recognition systems require significant computational resources and large amounts of training data to achieve high accuracy.

• **Time-consuming** It can take a significant amount of time to train a handwriting recognition system and even after training, the system may require significant time to process large amounts of data.

• **Limited applications** Handwritten character recognition systems are typically limited to specific applications, such as OCR or signature recognition and cannot be easily adapted for other applications.

3.2 PROPOSED SYSTEM

The proposed system for automatic text recognition and extraction from images, videos, and live feeds utilizes web technologies like Flask (a lightweight web framework in Python) and HTML (for front-end display). The system enables users to upload images or stream video content to a web server for processing. On the backend, Flask will handle requests and integrate with image processing and OCR models to detect and extract text from the media. HTML and JavaScript will be used to create an interactive interface where users can upload images or videos, view extracted text in real time, or download results. The system will leverage Optical Character Recognition (OCR) tools like Tesseract or OCR to recognize text and OpenCV for image/video preprocessing. The backend Flask server will interact with the OCR engines, process the input, and return extracted text to the user. The live feed functionality can be built using Flask along with WebSocket or WebRTC for streaming video from the user's webcam to the server. On the front end, users can view processed frames or text directly on the web page, providing a seamless, interactive experience. This system allows users to automatically extract text from static images, video content, or even real-time live video streams through a simple and intuitive web interface.

3.2.1 Merits

- **Accuracy** OpenCV has high accuracy in character recognition, which means that the system can correctly identify the characters in the image.
- **Speed** OpenCV is designed to be highly efficient and can perform character recognition in real-time.
- **Accessibility** OpenCV is an open-source library that is freely available, making it easy to access and use for character recognition tasks.
- **Integration** OpenCV can be easily integrated with other technologies such as machine learning and artificial intelligence to further enhance its capabilities.

CHAPTER 4

SYSTEM SPECIFICATION

4.1 HARDWARE SPECIFICATION

Processor: Dual core processor

RAM: 4GB

Hard disk: 256 GB

Monitor: 15-inch colour monitor

Webcam: 5 MP

4.2 SOFTWARE SPECIFICATION

Operating system: Any

IDLE: Python 3

Web Application: VS code

Browser: Any

CHAPTER 5

ARCHITECTURAL DESIGN

5.1 SYSTEM ARCHITECHTURE

A system architecture is the conceptual model that defines the structure, behavior and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

The system architecture for "Real Time Text Extraction and Recognition online " involves several stages and components working together to achieve the desired outcome. The first stage is image acquisition, where the input image is obtained from a source such as a camera or file. The acquired image is then pre-processed to enhance its quality and reduce noise. This stage involves image resizing, conversion to grayscale and smoothing using filters such as Gaussian or median filters.

The next stage involves segmentation, where the characters in the preprocessed image are isolated from the background. This stage can be performed using techniques such as thresholding, contour detection or edge detection. Once the characters are isolated, then subjected to feature extraction.

Feature extraction involves extracting relevant features from the segmented characters that can be used for recognition.These features can include aspects such as shape, size, curvature or texture. Principal Component Analysis or Linear Discriminant Analysis can be used to extract these features

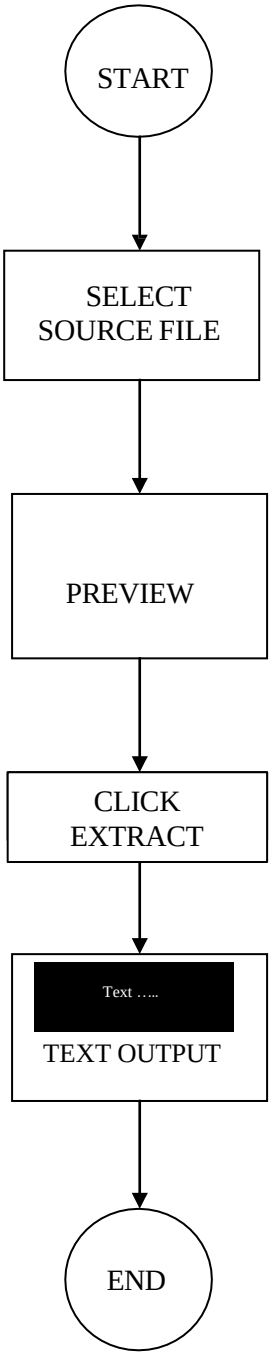

Figure.No. 5.1 System Architecture

Finally, the recognized characters are post-processed to refine the output and correct any errors that may have occurred during recognition. The postprocessing stage involves techniques such as spell-checking, error-correction and character grouping. the segmentation module, the feature extraction module, the character recognition module and the post-processing module.

.

CHAPTER 6

MODULE DESCRIPTION

6.1 IMAGE CHARACTER EXTRACTION

This module focuses on building a sophisticated computer vision system to accurately extract characters from documents and images. Leveraging tools like OpenCV, Pytesseract, Matplotlib, and Flask, the system ensures efficient and precise text extraction. The workflow includes image preprocessing to optimize input quality by reducing noise and enhancing contrast, followed by character segmentation to isolate individual characters. Features such as stroke width and aspect ratio are extracted and analyzed to enable accurate recognition by the character recognition module.

The system is evaluated on a diverse dataset, showcasing its high accuracy and efficiency in character extraction. Its potential applications include document digitization, text recognition, and information retrieval, making it a valuable solution for processing and analyzing visual data with reliability and precision.

6.2 VIDEO CHARACTER EXTRACTION

This project represents a robust integration of computer vision and OCR technologies, designed to accurately extract textual information from both images and video content. By leveraging OpenCV for image and frame processing, the system efficiently manages tasks such as noise reduction, contrast enhancement, and character segmentation. Tesseract OCR, accessed via the pytesseract wrapper, serves as the core text recognition engine, capable of detecting and interpreting complex textual content with high precision. These tools work in unison to provide a reliable solution for extracting text from diverse visual inputs.

The system is optimized for efficiency, with strategies such as selective frame processing in videos to balance resource utilization and thorough analysis. Bounding boxes visually highlight recognized text, enhancing user comprehension. This adaptable solution finds applications in document digitization, video surveillance, real-time captioning, and content analysis. Combining real-time processing capabilities, user-friendly design, and versatile applications, it offers a powerful tool for unlocking textual data in various domains.

6.3 VISUALS CHARACTER EXTRACTION

This module is designed to perform real-time character extraction from a live video stream using OpenCV and Tesseract OCR. It begins by initializing the video capture using the default camera (index 0). If the camera fails to open, an IOError is raised to alert the user.Within a continuous loop, each frame from the video stream is read using cap.read(). If a frame is successfully captured (ret == True), Tesseract OCR is utilized to extract text from the frame using pytesseract.image_to_string(frame)

The extracted text is then printed for debugging purposes. Additionally, bounding boxes are drawn around the detected characters using pytesseract. image_to_boxes(frame). These boxes visually represent the positions of the recognized characters within the frame.

To provide visual feedback to the user, the extracted text is overlaid onto the frame using cv2.putText(). This allows the user to see the recognized text displayed alongside the video feed.

The processed frame, with the overlaid text and bounding boxes, is displayed in a window titled 'Character Extraction from Visuals' using cv2.imshow().

User interaction is supported through keyboard commands. Pressing 'q' terminates the program and closes the video capture. Pressing 'e' prints the recognized text to the console along with an additional developer message.

In essence, this module encapsulates the synergy between computer vision and OCR technologies, offering a powerful and adaptable solution for unlocking textual information embedded within visual content.

Overall, this module demonstrates the seamless integration of OpenCV and Tesseract OCR for real-time character extraction from live video streams, providing users with valuable insights into the textual content present within visual data.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The proposed system for Real time text extraction recognition online is a robust and efficient solution for extracting characters from images and visual media. The system is designed to preprocess the images, segment the characters, extract the features, classify them using machine learning algorithms and post-process the results to generate accurate output.

The user interface is designed to be user-friendly and intuitive, allowing users to easily upload images and extract the characters. The system's pre-processing module is responsible for enhancing the image quality and reducing noise, thereby improving the overall character recognition accuracy. The segmentation module uses various techniques to separate the characters from the image's background and isolate them for feature extraction.

The feature extraction module extracts relevant features from the segmented characters. The postprocessing module eliminates false positives and generates the final output. Overall, the system is expected to deliver accurate and efficient results in character extraction, making it suitable for applications like document digitization, handwriting recognition and optical character recognition. With further research and development, the system can be improved to handle complex image processing tasks and deliver better performance.

7.2 FUTURE ENHANCEMENT

The future scope is a wide open in research aspect for all applications. Various other feature extraction methods can be applied to test the accuracy of the system. There are several potential future enhancements that could be made to the "Real time text extraction recognition online" system.

- **Multi-language support** Currently, the system is designed to recognize characters in a specific language. Expanding the system to recognize multiple languages would greatly increase its utility.
- **Deep learning-based approaches** While the current system uses traditional machine learning techniques, incorporating deep learning-based approaches such as Convolutional Neural Networks and Recurrent Neural Networks could potentially improve the accuracy and speed of character recognition.
- **Improved segmentation techniques** The segmentation module is a critical component of the system and further research could be done to develop more advanced techniques for identifying and separating individual characters from an image.
- **Real-time translation** as technology advances, it may become possible to implement the system in real-time applications such as video surveillance or document scanning.
- **Mobile application** Developing a mobile application that utilizes the character recognition system would allow users to easily extract text from images taken on their mobile devices.

Overall, there is great potential for the "Real time text extraction recognition online" system to be improved and expanded upon in the future, making it an even more powerful tool for character recognition.

APPENDIX 1 SAMPLE CODE

```
from flask import Flask, render_template, request, jsonify, redirect, url_for
import pytesseract
import cv2
import os
import time
import threading

app = Flask(_name_)
pytesseract.pytesseract.tesseract_cmd = (file location)

os.makedirs('static/uploads', exist_ok=True)
os.makedirs('static/outputs', exist_ok=True)

detected_text = ""
detected_text_lock = threading.Lock()
camera_running = False

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/extract', methods=['POST'])
def extract_text():
    if 'file' not in request.files:
        return "No file uploaded", 400

    file = request.files['file']
    if file.filename == '':
        return "No file selected", 400


    image_path = os.path.join('static', 'uploads', file.filename)
    file.save(image_path)

    img = cv2.imread(image_path)


    frames_text = []
    cntr = 0
```

```
    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break

        cntr += 1
        if cntr % 8 == 0:  # Process every 8th frame for text extraction
            imgchar = pytesseract.image_to_string(frame)
            print(f"Extracted text from frame {cntr}: {imgchar}")
            cv2.imshow('Camera', frame)

            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
    finally:
        cap.release()
        cv2.destroyAllWindows()
        camera_running = False

@app.route('/start_camera', methods=['POST'])
def start_camera():
    if not camera_running:
        threading.Thread(target=capture_text_from_camera).start()
    return redirect(url_for('camera_page'))

@app.route('/camera', methods=['GET'])
def camera_page():
    return render_template('camera.html')

@app.route('/stop_camera', methods=['POST'])
def stop_camera():
    global detected_text
    with detected_text_lock:
        text_to_return = detected_text.strip()  # Strip to remove any trailing newlines
        detected_text = ""  # Reset detected text
    print("Detected text from camera:", text_to_return)  # Debugging line
    return render_template('result_camera.html', text=text_to_return)

@app.route('/get_detected_text', methods=['GET'])
def get_detected_text():
    with detected_text_lock:
        text_to_return = detected_text
```

APPENDIX 2 SCREENSHOTS

Figure. No. A.2.1 Image Character Extraction

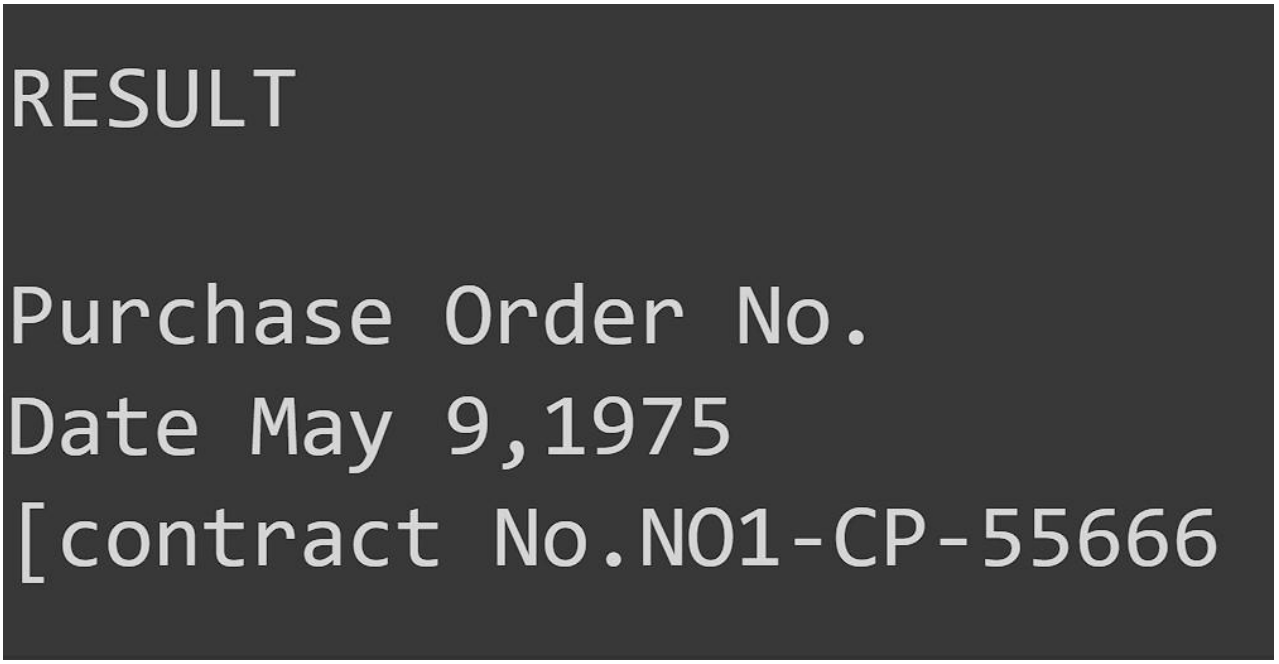

Figure. No. A.2.2 Image Character Extraction

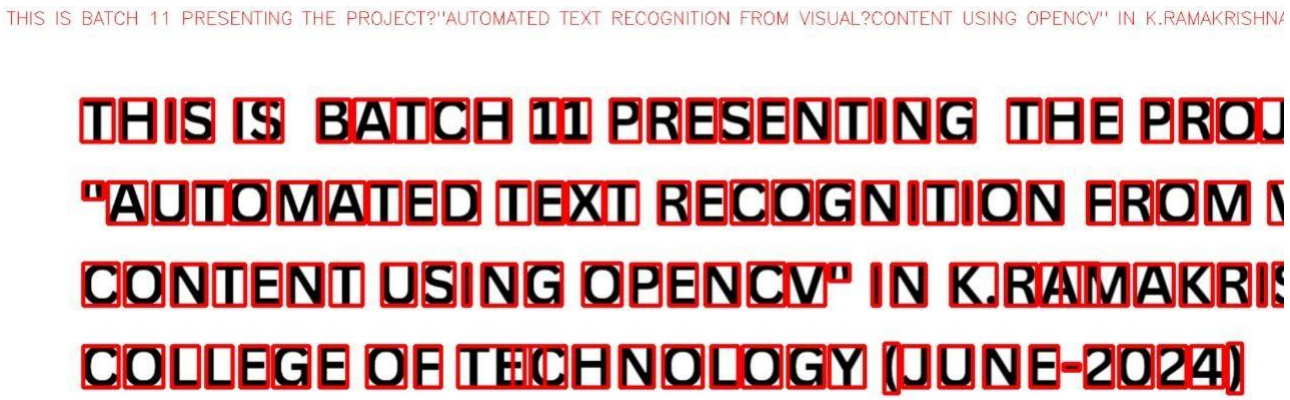

Figure. No. A.2.3 Video Character Extraction

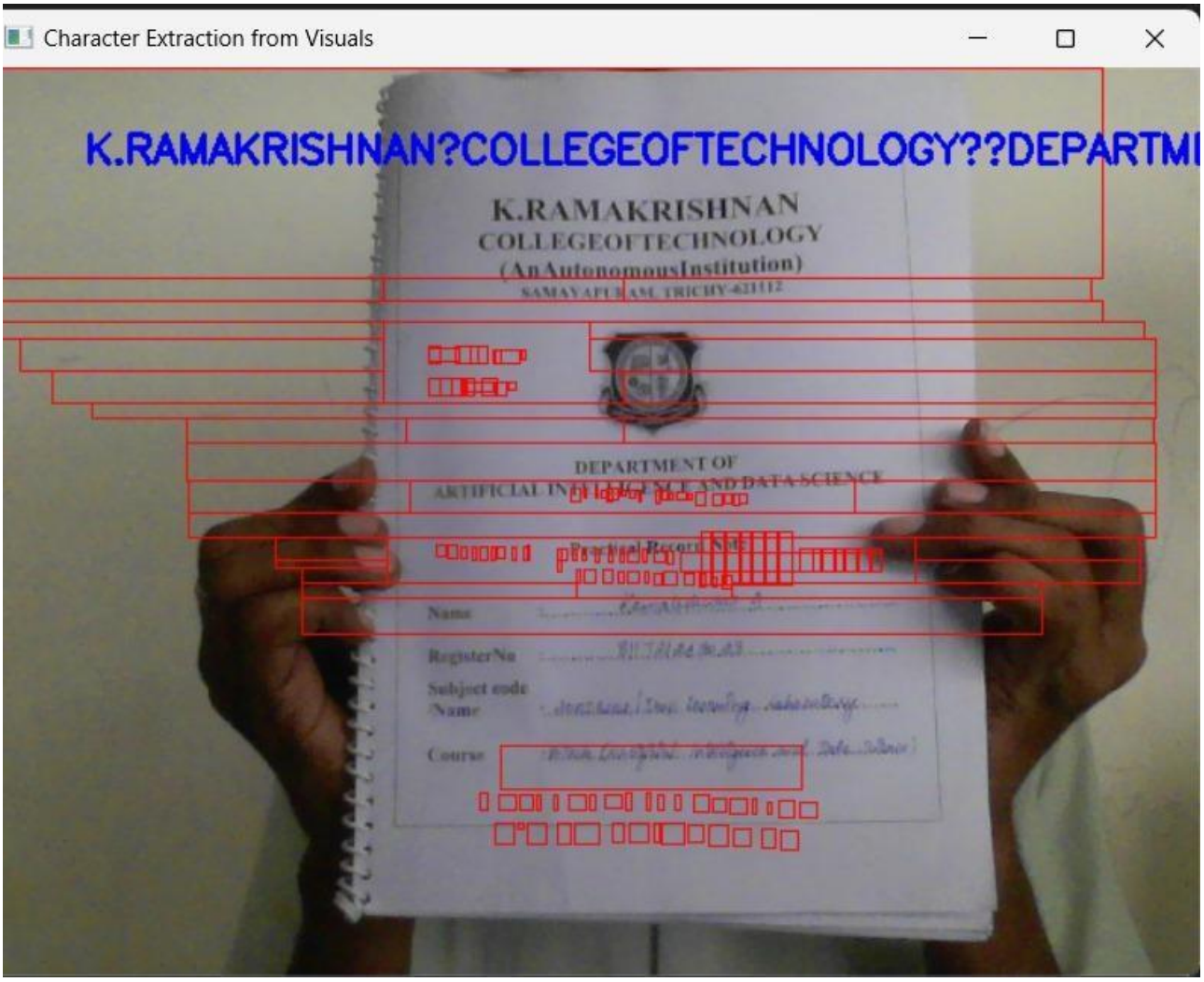

Figure. No. A.2.4 Visual Character Extraction

REFERENCES

1. Altaf Ramjon1Asherl Bwatiramba1* Sivakumar Venkataraman2 (2023) Department of Computer Science, University of East London, London, UK 1*,2 Botho University, Botho Education Park, Gaborone, Botswana

2. Assoc. Prof. L. Rasikannan1, K. Gunal2, S. Sabarinathan3, S. Vigneswaran (Apr 2024) Dept. of Computer Science and Engineering, Government College of Engineering Srirangam, Tamilnadu, India Volume: 11 Issue: 04

3. Sushmita B Poojary, 3 varshitha, 4 vidya K C, 5shilpa Assistant Professor “Handwritten Text Recognition System” (2021) Department of Computer Science and Engineering 1, 2,3,4,5 Alva’s Institute of Engineering and Technology, Mijar, India.

.

4. Triveni K.M.**1, Thanushree R. 2, Prof. Vinaykumar Hittalamani*3 *1,(2022)Dept. Of MCA VTU University Belagavi Karnataka, India. *3Assistant Professor, Dept. Of MCA VTU University Belagavi Karnataka, India..

5. Nafiz Arica, and Fatos T. Yarman-Vural, (2024``) "Optical Character Recognition for Cursive Handwriting", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.24, no.6, pp. 801-113.