IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁙ Impact Factor 8.102 ⁙ Peer-reviewed / Refereed journal ⁙ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125285

DOCUMENT SCRUTINIZING AND IMAGING SYSTEM

Prof.Dr Ninad More[1], Aditi Roy[2], Kashmira Nagrale[3]

D. Y. Patil College of Engineering, Pune[1-3]

Abstract—Image processing means taking an image and using computer algorithms to do useful things with it. These algorithms can help us extract important information from the image. When we process images, we treat them as 2D signals and use specific methods to analyze them. OpenCV is a popular computer vision library that many people use. It has a wide range of functions for working with images and vision-related tasks. OpenCV is used in both academic and industry settings. With the increasing availability of affordable cameras and the growing demand for image-related features, OpenCV is used in many applications, including on computers and mobile devices. When we talk about "document scrutinizing and imaging," we mean closely examining a document. We use machine learning to create a model that can process the document. After processing, we convert the document into a PDF format and create an image of it.

The processing of an image includes six main stages:
1)Thresholding
2)Noise Removal
3)Edge Detection
4)Contour Extraction
5)Projection to screen
6)Saving result as PDF

Keywords: OpenCV, Thresholding, Image Smoothening, Edge Detection, Contouring.

I. INTRODUCTION

OpenCV stands for Open-Source Computer Vision Library. It is a versatile library that allows developers to create real-time computer vision applications. Its focus is on tasks such as image processing, video capturing, and analysis, including features like face detection and object detection. Originally developed in C++, OpenCV also provides bindings for Python and Java. It is compatible with various operating systems, including Windows, Linux, macOS, FreeBSD, NetBSD, and OpenBSD. This study proposes to utilize OpenCV and Python within Jupyter Notebook to create an enhanced version of a basic document processing system. The goal is to develop a model that can automatically detect edges and remove unnecessary portions of an image while improving the readability of the document. This advanced system consists of four primary stages of image processing, along with additional functions for generating PDFs after the final image processing stage.

In this project, we aim to provide a model in which an image is loaded, various algorithms and techniques are used to perform on the image and get the enhanced image. The result obtained after implementing all the steps of processing is converted in the form of Portable Document Format.

The objective of this paper is to utilize Computer Vision and Python 3.9.13 to develop a scanning application. Instead of relying on costly and bulky hardware, we propose a software solution that achieves similar results by loading an image from users directory. Our software enables users to easily convert physical documents into scanned images. It provides functionality to enhance and save these documents in the user's preferred directory. By leveraging Computer Vision techniques and Python, we aim to provide a cost-effective and accessible solution for document scanning and processing. This chapter emphasizes the importance of image processing by analysing the images to get better image in the result.

II. MOTIVATION

In technical terms, Image processing is usually related to the usage and application of mathematical functions and transformations over images. It is usually about an algorithm doing some transformations on the image such as stretching, contrasting, sharpening, or smoothening on the image. While computer vision mainly focuses on image or video given as input to a computer system, and the expected output or the desired task is to analyse/infer something about the image.

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified ⁙ Impact Factor 8.102 ⁙ Peer-reviewed / Refereed journal ⁙ Vol. 12, Issue 5, May 2023
DOI: 10.17148/IJARCCE.2023.125285

Computer Vision may/may not use image processing algorithms to perform this task. The goal here is to be able to gain knowledge from the image rather than enhancing the image. An enhanced image does make the process of gaining knowledge from the image easier though. So the main difference in Image Processing and Computer Vision lies in the goal. The field of Computer Vision has high potential for further development in the future by implementing machine learning algorithms to better analyse the images.

III. PROBLEM DEFINITION

We aim to create a user-friendly software application that utilizes OpenCV, a versatile computer vision library, to address the task of document scanning. Our objective is to develop a solution that eliminates the need for expensive scanning hardware by enabling the user to load the image in the model to convert physical documents into scanned images.

Our ultimate goal is to create an intuitive and efficient document scanning application that can convert physical documents into high-quality digital images. This solution will be user-friendly, cost-effective, and applicable in various industries and domains, simplifying document management and sharing processes.

IV. SOFTWARE REQUIREMENT

Purpose and Scope of Document
The purpose of our document scanning project is to create a user-friendly and efficient solution that allows individuals and organizations to easily convert physical documents into digital images. By utilizing OpenCV and a camera , we aim to eliminate the need for costly scanning hardware. Our project aims to simplify document management by providing a convenient way to capture, enhance, and store scanned images. With this solution, users can improve accessibility, organization, and sharing of documents, ultimately streamlining workflows, and reducing paper usage.

However, it is worth noting that an enhanced image can facilitate the process of gaining knowledge from the image. Therefore, image processing techniques can support and enhance computer vision tasks by improving the quality and clarity of the image.

Overview of responsibilities of Developer
1.To have understanding of the problem statement.
2.To know what are the hardware and software
requirements of proposed system.
3.To have understanding of proposed system.
4.To do planning various activities with the help of planner.
5.Designing, programming, testing etc.

RAM : 2 GB (min)
Hard Disk : 1 GB
Hard Disk memory is required.
Processor : Intel i3/i5/i7
Jupyter , Spyder IDE that Integrated Development
Environment is to be used and data loading should be fast hence
Fast Processor is required

IDE : Jupyter, Spyder(Anaconda 3)
Best Integrated Development Environment as it gives
possible suggestions at the time of typing code
snippets that make typing feasible and fast
Coding Language : Python version 3.9.13
Highly specified Programming Language for Machine Learning
because of availability of High-Performance Libraries
Libraries : NumPy, matplotlib, cv2, img2pdf, PIL, OS,
argparse, operator, globe.
Because of availability of High-Performance Libraries
Operating System : - Windows 10/11
Latest Operating System that supports all type of installation
and development Environment.

V. SET-UP

To set up the environment for the proposed system, follow these important steps:

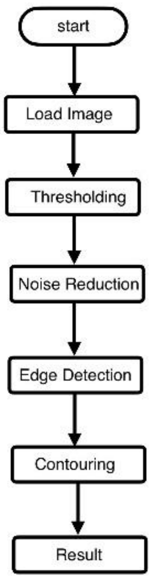
Step 1: Install Python Ensure that you have Python installed on your system. You can download the latest version of Python from the official website (https://www.python.org/) and follow the installation instructions specific to your operating system.
Step 2: Install Jupyter Notebook Jupyter Notebook is included in the Python distribution and can be installed using the package manager, pip. Open a command prompt or terminal and run the following command: pip install jupyter
This will download and install Jupyter Notebook along with its dependencies.
Step 3: Launch Jupyter Notebook Once Jupyter Notebook is installed, you can launch it by opening a command prompt or terminal and running the following command: jupyter notebook
This will start the Jupyter Notebook server and open a new tab in your web browser.
Step 4: Create a New Notebook In the Jupyter Notebook interface, click on the "New" button and select "Python 3" (or any other kernel of your choice) to create a new notebook. This will open a new notebook where you can write and execute Python code.
Step 5: Execute Code Cells Inside the notebook, you will see cells where you can write and execute code. To execute a code cell, simply click on it to select it, then press Shift + Enter or click the "Run" button in the toolbar. The output of the code will be displayed below the cell.
Step 6: Add Markdown Cells (optional) Jupyter Notebook supports Markdown cells, which allow you to write formatted text, add headings, lists, links, and more. To create a Markdown cell, click on the "+" button in the toolbar and select "Markdown". You can then enter your formatted text using Markdown syntax.
Step 7: Save and Export Notebooks You can save your Jupyter Notebook by clicking on the "Save" button or by selecting "File" > "Save and Checkpoint" from the menu. Notebooks are saved with the .ipynb extension. You can also export notebooks to different formats like HTML, PDF, or Python script using the "File" > "Download as" option

Overview of Project Modules

This research paper is a detailed study that examines document scanning techniques. The main goal is to investigate the progress and difficulties in the field of document scanning, particularly in using computer vision and image processing algorithms. The paper intends to provide a thorough understanding of the different steps involved in document scanning, such as loading the image, preparing it for processing, detecting the document, improving its quality, and storing it. The analysis includes a comparison of various approaches, highlighting their strengths, limitations, and possible uses. The findings contribute valuable knowledge to the existing methods of document scanning, offering valuable insights for researchers and professionals working in this field.

❖ Design

System Architecture



International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified ⁜ Impact Factor 8.102 ⁜ Peer-reviewed / Refereed journal ⁜ Vol. 12, Issue 5, May 2023
DOI: 10.17148/IJARCCE.2023.125285

Loading an image in OpenCV: To load an image in OpenCV, you can follow these steps:
1. Use the imread() function to read the image.
2. Display the image using the plt.imshow() function.

Gray scaling: Gray scaling is the process of converting an image from other color spaces (such as RGB, CMYK, HSV) to shades of gray. It represents the image with varying shades of black and white.

Importance of Gray scaling: Gray scaling is important for several reasons:
- Dimension reduction: Grayscale images are single-dimensional, while RGB images have three color channels and three dimensions. By converting to grayscale, we reduce the image's dimensions.
- Reducing model complexity: When training neural networks, using RGB images with dimensions like 10x10x3 would require 300 input nodes. Grayscale images, however, only need 100 input nodes for the same network.
- Compatibility with certain algorithms: Some algorithms are specifically designed to work with grayscale images. For example, the Canny edge detection function in the OpenCV library is implemented to work only on grayscale images.

To import OpenCV and read the original image using imread(), and then convert it to grayscale using cv2.cvtColor() function.

.Image thresholding is a technique used to separate an image into foreground and background by converting it into a binary image. It is a type of image segmentation that is particularly effective in images with high contrast. Thresholding is done by comparing pixel values and dividing them into two parts.

There are different techniques for image thresholding, and one of them is simple thresholding. In simple thresholding, pixels with values greater than a specified threshold are assigned a standard value.

To perform simple thresholding on an image, you can use the threshold() method from the Imgproc class. Here is the syntax of the method:

threshold(src, dst, thresh, maxval, type)

This method takes the following parameters:
- src: The source image.
- dst: The destination image where the output will be stored.
- thresh: The threshold value.
- maxval: The value assigned to pixels that are greater than the threshold.
- type: The type of thresholding technique to be used.

There are different simple thresholding techniques available:
- cv2.THRESH_BINARY: Sets the pixel value to 255 if it is greater than the threshold; otherwise, it is set to 0 (black).
- cv2.THRESH_BINARY_INV: The inverse or opposite of cv2.THRESH_BINARY.
- cv2.THRESH_TRUNC: Truncates the pixel intensity value to the threshold. Pixel values greater than the threshold are set to the threshold value.
- cv2.THRESH_TOZERO: Sets the pixel intensity to 0 for all pixels with intensity less than the threshold.
- cv2.THRESH_TOZERO_INV: The inverse or opposite of cv2.THRESH_TOZERO.

One drawback of simple thresholding is that it applies a single threshold value to the entire image, which can result in the loss of some data. Different parts of the image may have varying levels of contrast or lighting conditions, and using a single threshold may not be suitable for all areas. This can lead to the loss of important details or inaccurate segmentation.

To overcome this limitation, adaptive thresholding can be used. Adaptive thresholding is a local thresholding technique that takes into account each pixel and its neighborhood. Instead of using a global threshold for the entire image, adaptive thresholding calculates the threshold value for each pixel based on its local neighborhood.

There are two commonly used methods for calculating the threshold in adaptive thresholding:
1. Arithmetic Mean: The threshold value is determined by the average intensity of the pixels in the local neighborhood.
2. Gaussian Mean: The threshold value is calculated using the weighted average of pixel intensities in the neighborhood, with the center pixel having the highest weight and pixels farther from the center contributing less.

By considering the local characteristics of the image, adaptive thresholding can provide better results compared to simple thresholding, especially in cases where the image has variations in lighting or contrast across different regions.

In adaptive thresholding, the goal is to statistically analyze local regions of an image and determine an optimal threshold value (T) for each region. The choice of statistic used to compute the threshold value depends on the method employed. The two commonly used statistics are the arithmetic mean and the Gaussian mean.
- Arithmetic Mean: In this method, each pixel in the neighborhood contributes equally to computing the threshold value. The mean intensity of the pixels in the local sub-region is calculated.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

ISO 3297:2007 Certified ⁑ Impact Factor 8.102 ⁑ Peer-reviewed / Refereed journal ⁑ Vol. 12, Issue 5, May 2023

DOI: 10.17148/IJARCCE.2023.125285

- Gaussian Mean: In this method, the pixel values farther away from the center of the (x, y) coordinate of the region contribute less to the calculation of the threshold value. The weighted average of pixel intensities in the neighborhood is computed using a Gaussian distribution.

The general formula to compute the threshold value (T) is: T = mean(IL) - C where mean is either the arithmetic or Gaussian mean, IL represents the local sub-region of the image, I, and C is a constant that can be adjusted to fine-tune the threshold value. Image Denoising: Image denoising, also known as image smoothing, is a technique used to remove unwanted noise from an image and enhance its quality. It involves applying filters to the image to reduce the presence of distortions.

Blurring is a commonly used technique in image denoising, where a convolution operation is performed between the image and a predefined low-pass filter kernel. This process reduces the edge content in the image, making color transitions smoother and enhancing the overall appearance. By reducing noise, the image becomes clearer and objects within it can be identified more easily.

Different types of filters can be used for image denoising, including linear (spatial) filters. Some commonly used linear filters include:

- Homogeneous filter
- Gaussian filter
- Median filter

Non-linear filters, such as the bilateral filter and adaptive bilateral filter, can also be used to blur the image while preserving its edges in specific cases.

Kernel: In image processing, a kernel (also known as a convolution matrix or mask) is a small matrix used for operations like blurring, sharpening, embossing, edge detection, and more. Convolution between an image and a kernel is performed to achieve these effects. OpenCV provides the cv2.filter2D() function to convolve a kernel with an image.

In the context of image denoising, one commonly used linear filter is the Averaging (Mean or Homogeneous) filter. The averaging filter works by taking the average of all the pixel values within a specified kernel area and replacing the central element with this average value. It is also known as a box filter or mean filter.

The properties of an averaging filter are as follows:

- The size of the filter (kernel) must be odd, such as 3x3, 5x5, etc.
- The sum of all the elements in the kernel should be 1.
- All the elements within the kernel should have the same value.

To satisfy the condition that the sum of all the elements should be equal to 1, each value in the kernel is divided by the total number of cells in the kernel. In this case, there are 9 cells, so each value in the kernel is 1/9. This ensures that 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 equals 1.

By convolving the image with the averaging filter kernel, each pixel's value is replaced with the average of its neighboring pixels within the kernel. This smoothing operation helps reduce noise and blur the image.

Gaussian filtering, also known as Gaussian blur, is a technique used in image processing to reduce noise and blur an image. Unlike the averaging filter that uses equal filter coefficients, the Gaussian filter employs a Gaussian distribution for weighting the neighboring pixels in both the x and y directions. It requires the specification of a standard deviation (sigma) parameter.

The equation for a Gaussian filter kernel of size (2k+1) x (2k+1) is given by:

Gaussian filter equation

A 5x5 Gaussian filter kernel would look like this:

$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

The values in the kernel are calculated based on the Gaussian distribution formula. Each value represents the weight or contribution of the corresponding pixel to the blurred output. The center pixel has the highest weight, and the weights decrease as we move away from the center.

By convolving the image with the Gaussian filter kernel, the image is smoothed, and noise is reduced. The extent of blurring depends on the size of the kernel and the standard deviation value provided. Higher standard deviation values result in more significant blurring.

Median filtering is a non-linear filtering technique used for image denoising. In median filtering, the central pixel value within a kernel window is replaced with the median value of all the pixels within that window. This method is particularly effective in removing salt-and-pepper noise, which appears as random white and black pixels scattered throughout the image.

DOI: 10.17148/IJARCCE.2023.125285

Unlike other filters such as Gaussian or averaging filters, median filtering ensures that the filtered value for the central element is always an actual pixel value present in the original image. This characteristic makes median filtering highly effective in reducing noise while preserving the sharpness and details of the image.
The process of median filtering involves the following steps:

1. Place a kernel window over the image.
2. Collect the pixel values within the window.
3. Sort the pixel values in ascending order.
4. Replace the central pixel with the median value, which is the middle value in the sorted list of pixel values.
5. Repeat this process for every pixel in the image.

By replacing each central pixel with the median value of its neighborhood, median filtering effectively reduces noise while preserving the overall structure of the image. This makes it a popular choice for denoising images affected by salt-and-pepper noise.
Edge detection is an image processing technique used to identify the boundaries of objects or regions within an image. It plays a crucial role in computer vision and various image analysis tasks.
To perform edge detection, we commonly employ two methods: Sobel Edge Detection and Canny Edge Detection.
Image Gradient Image gradient is a fundamental concept in edge detection. It refers to the directional change in color or intensity within an image. By analyzing these changes, we can detect the presence of edges.

There are different methods for finding the image gradient, including the Sobel operation and the Laplacian.
1. Sobel Operation: The Sobel operation is a widely used method for computing the image gradient. It calculates the gradient magnitude and direction at each pixel by convolving the image with a set of filters. The Sobel operator approximates the first derivative of the image intensity in the x and y directions. The magnitude of the gradient represents the strength of the edge, and the direction indicates the orientation of the edge.
2. Laplacian: The Laplacian operator is another method for finding the image gradient. It calculates the second derivative of the image intensity to identify regions of rapid intensity changes. The Laplacian operator is more sensitive to noise compared to the Sobel operator, but it can capture finer details in the image.
Both the Sobel operation and the Laplacian are widely used in edge detection algorithms to identify edges based on the intensity changes in the image. These methods form the foundation for many edge detection techniques and are essential for further image analysis and processing tasks.

Contours play a vital role in computer vision as they enable computers to detect and analyze shapes and objects within an image. OpenCV provides functions to find and draw contours, and we can follow these steps to accomplish that:
1. Convert the image into a binary image: Before finding contours, we need to convert the image into a binary format. This can be achieved by applying thresholding or edge detection techniques. In this case, we have already converted the image into a binary image using the Canny edge detector.
2. Find the contours: We can use the cv2.findContours function in OpenCV to find the contours in the binary image. This function takes the binary image as input and returns a list of contours present in the image.
3. Draw the contours on the image: Once we have obtained the contours, we can use the cv2.drawContours function to draw the contours on the original image. This function takes the original image, the list of contours, the contour index (or use -1 to draw all contours), the color, and other optional parameters. By drawing the contours on the image, we can visualize and analyze the detected shapes.
By following these steps and using the appropriate OpenCV functions, we can find contours in an image and draw them to facilitate object classification, segmentation, and identification tasks in computer vision applications.

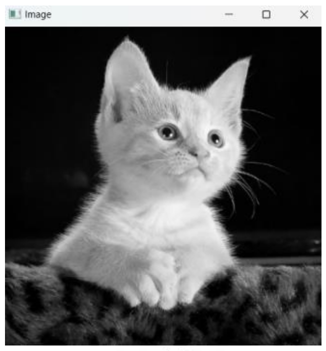
To convert an image to PDF in Python, we can use the **img2pdf** library. First, we need to install the library using the following command: pip install img2pdf
Once installed, we can use the **img2pdf.convert()** function provided

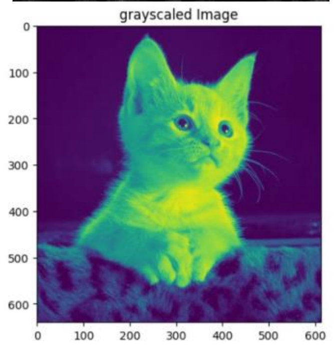
International Journal of Advanced Research in Computer and Communication Engineering

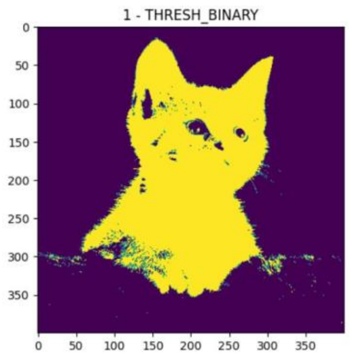
ISO 3297:2007 Certified ⁙ Impact Factor 8.102 ⁙ Peer-reviewed / Refereed journal ⁙ Vol. 12, Issue 5, May 2023

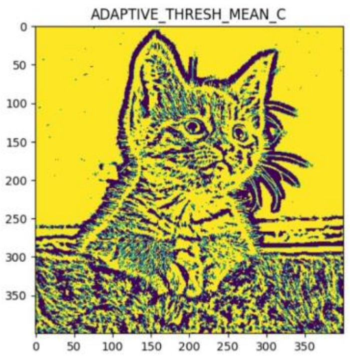
DOI: 10.17148/IJARCCE.2023.125285

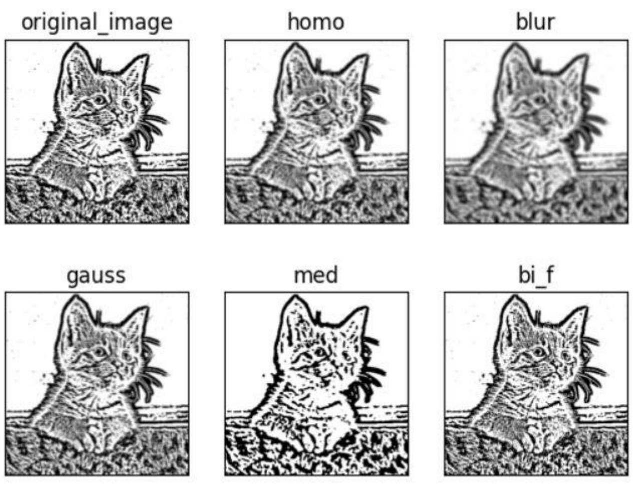
VI. OUTPUT



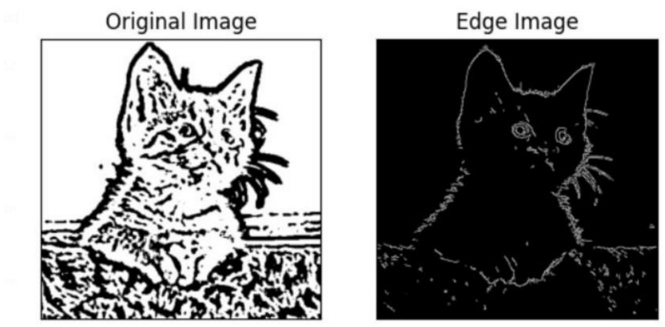


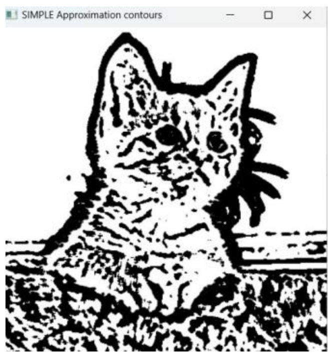






DOI: 10.17148/IJARCCE.2023.125285





REFERENCES

[1] Boris Rewald [1], of University of Natural Resources and Life Sciences
[2] Ayushe Gangal, Peeyush Kumar and Sunita Kumari ayushe17@gmail.com, peeyushstark@gmail.com, sunitakumari@gbpec.edu.in *Dept. of CSE, GB Pant* Govt. Engineering College New Delhi-110020, India
[3] Ruslan Briklenkov's paper "Document Scanner from Scratch with Python"
[4] M. Sridevi, National Institute of Technology, Tiruchirappalli, India, "Image Segmentation based on Multilevel Thresholding using Firefly Algorithm", Proceedings of the International Conference on Inventive Computing and Informatics (ICICI 2017), IEEE Xplore Compliant. (2016)
[5] Image Segmentation based on Multilevel Thresholding using Firefly Algorithm M. Sridevi Department of Computer Science and Engineering, National Institute of Technology, Tiruchirappalli - 620 015, India
[6] Gloria Bueno Garcia, Oscar Deniz Suarez, José Luis Espinosa Aranda, Jesus Salido Tercero, Ismael Serrano Gracia, Noelia Vallez Enano-Learning image processing with OpenCV

CONCLUSION

The proposed system was tested on number of documents/images to find their scanned images as outputs. We proposed a computer vision-based scanning method.by using OpenCV we computed the shape and resized the original image for better enhancement.The original copy of the image was uploaded and by using canny edge detection we removed the noise from the image and blurred the image. Contouring was applied on the obtained image and the result was projected to the screen with saving the image into portable document form.