IMAGE TO PENCIL SKETCH PROJECT REPORT

Project Title:

Image Dataset Conversion to Pencil Sketch using OpenCV

Course:

Digital Image Processing

Tool / Language

Python (Google Colab)

Institution:

Government Graduate College Multan Road Burewala.

Submitted By:

Iman Zahid (100049)

Ayesha Bibi (100042)

Submitted To:

Prof Rimsha Kiran

Department:

Computer Science

Abstract

This project presents a simple and effective approach for converting a dataset of color images into pencil sketch representations using Python and OpenCV. The main objective of the project is to apply fundamental digital image processing techniques, including grayscale conversion, Gaussian blurring, edge detection using the Canny algorithm, and image inversion.

The dataset images are first extracted and loaded automatically, after which each image is processed step by step to generate a pencil sketch effect. The results are visualized using the Matplotlib library in both individual and grid formats. This project helps in understanding basic image preprocessing, edge detection, and artistic image transformation techniques, which are widely used in computer vision and image editing applications.

1.Introduction

Image Processing is a core area of computer science that deals with analyzing and manipulating images to enhance their quality or extract meaningful information. One creative application of image processing is image stylization, such as converting an image into a pencil sketch.

The goal of this project is to design a simple yet effective pipeline that transforms color images into pencil sketches using edge detection techniques. This project is suitable as a mini project for Digital Image Processing and helps build a foundation for advanced topics like image inpainting, restoration, and computer vision.

2.Objectives

The main objectives of this project are:

1. To understand basic image preprocessing techniques
2. To convert RGB images into grayscale images
3. To apply Gaussian blur for noise reduction
4. To detect edges using the Canny Edge Detection algorithm
5. To generate a pencil sketch effect by inverting detected edges
6. To visualize results using Matplotlib

3.Tools and Technologies Used

- **Programming Language:** Python
- **Libraries:**
  OpenCV (cv2)
  NumPy
  Matplotlib
  OS and Zipfile modules
- **Platform:**
  Google Colab / Local Python Environment

4.Dataset Description

The dataset consists of multiple image files stored in a ZIP folder. The images can be of any object, person, or scene.

Supported formats include:

1. .jpg
2. .jpeg
3. .png

After extracting the ZIP file, the program recursively scans all subfolders to collect image paths.

6.Methodology

The project follows a step-by-step image processing pipeline:

5.1: Dataset Extraction

The ZIP file containing images is extracted using Python's zipfile module. This makes the images accessible for further processing.

5.2: Image Loading

All image paths are collected recursively using the **os.walk() function**. Each image is read using OpenCV.

5.3: RGB to Grayscale Conversion

Color images are converted into grayscale using:

```
cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

This reduces computational complexity and focuses on intensity information.

5.4: Gaussian Blur

Gaussian blur is applied to smooth the image and remove noise:

```
cv2.GaussianBlur(gray, (5,5), 0)
```

This step improves edge detection accuracy.

5.5: Edge Detection (Canny)

Canny edge detection is used to identify strong edges in the image:

```
cv2.Canny(blur, 60, 100)
```

Edges form the base of the pencil sketch effect.

6.6: Pencil Sketch Generation

The detected edges are inverted using bitwise NOT operation:

```
cv2.bitwise_not(edges)
```

This creates a sketch-like appearance similar to pencil strokes.

6.Results and Output

The system successfully converts all input images into pencil sketch versions. Each output image highlights object boundaries and fine details, resembling hand-drawn pencil sketches.

Additionally, a **sketch weight** (number of non-zero pixels) is calculated for each image to provide a simple quantitative measure of sketch intensity.

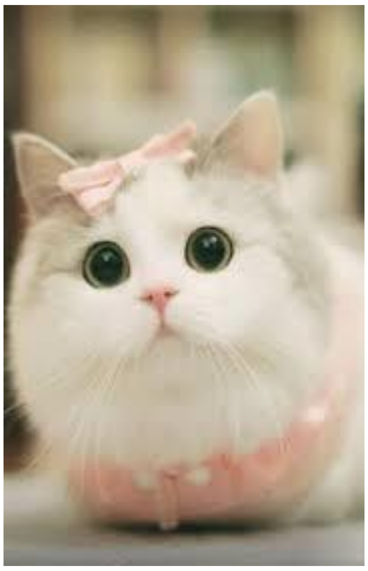

Figure 1: Original Image

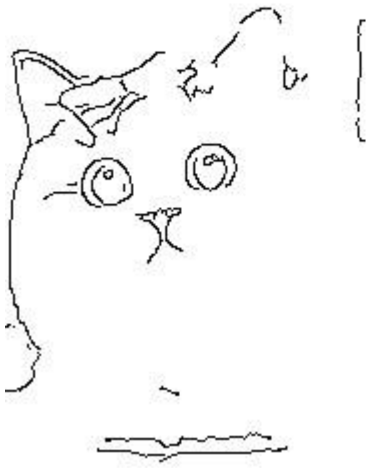

Figure 2: Pencil Sketch Output

7.Advantages

- Simple and easy to implement
- No need for deep learning or large datasets
- Fast execution

- Suitable for beginners in image processing

8.Limitations

- Results depend on lighting conditions
- Not suitable for highly textured or noisy images
- No color sketch generation

9.Applications

- Photo editing software
- Digital art and illustration
- Image stylization
- Preprocessing step for artistic filters

10.Conclusion

This project successfully demonstrates how basic digital image processing techniques can be combined to convert color images into visually appealing pencil sketch representations. By applying grayscale conversion, Gaussian blur, edge detection, and image inversion, the system produces sketch-like images that closely resemble hand-drawn pencil sketches.
The implementation using Python and OpenCV proves to be simple, efficient, and effective for processing multiple images from a dataset. The results show that even without using complex deep learning models, meaningful and artistic image transformations can be achieved. Overall, this project provides a strong foundation for understanding core image processing concepts and can be further

extended to develop advanced image processing and computer vision applications.

11.References

[1] Gonzalez, R. C., & Woods, R. E.,
Digital Image Processing, 4th Edition, Pearson Education, 2018.

[2] OpenCV Documentation,
OpenCV Library – Image Processing Functions.

[3] Canny, J.,
“A Computational Approach to Edge Detection,”
IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986.