IMAGE TO CARTOON CONVERTER

Final Semester Project Report

Course: Digital Image Processing (DIP)

Course Code: CSI 509

Department of Computer Science (5th Semester)

Submitted By:

Qasim Hussain (100067)

Muhammad Ali Hassan (100040)

February 2026

Abstract

This project presents the design and implementation of an Image to Cartoon Converter using classical Digital Image Processing (DIP) techniques.
The system is developed in Python using OpenCV for image processing operations and Streamlit for the graphical user interface.
The proposed system converts real-world photographs into cartoon-style images by applying edge-preserving filters, color quantization techniques,
adaptive thresholding, stylization filters, and sketch-based rendering.
Unlike deep learning-based approaches, the system does not require training data and ensures real-time performance with low computational cost.
The application produces multiple cartoon variations for a single image and allows users to download results individually or collectively.
Experimental evaluation confirms that classical techniques remain effective for artistic image transformation while maintaining structural integrity.

Keywords

Image Cartoonization, Digital Image Processing, OpenCV, Edge Detection, Bilateral Filtering, K-means Clustering, Streamlit

Acknowledgments

We would like to express our sincere gratitude to our course instructor for continuous guidance, support, and encouragement throughout the semester.
We are thankful to the faculty of the Computer Science Department for providing the necessary resources and learning environment.
We also acknowledge our classmates for their constructive feedback and discussions that contributed to the successful completion of this project.

Table of Contents

Contents

Chapter 1: Introduction

Digital Image Processing (DIP) is an important field of computer science that focuses on enhancing, analyzing, and transforming digital images using computational techniques. With the rapid growth of digital media and visual content, image processing techniques are widely used in areas such as entertainment, education, computer vision, and multimedia applications.

Image cartoonization is a non-photorealistic rendering technique that converts real-world images into simplified, artistic representations. It emphasizes strong edges, smooth color regions, and reduced visual details to produce cartoon-like effects. Cartoon-style images are commonly used in animation, gaming, mobile applications, and social media due to their visually appealing and expressive nature.

While recent cartoonization methods use deep learning and neural networks, these approaches require large datasets and high computational resources. In contrast, classical image processing techniques such as edge detection, bilateral filtering, and color quantization provide efficient and reproducible results without training data.

This project presents an Image to Cartoon Converter based on classical Digital Image Processing techniques. The system generates multiple cartoon-style outputs through a user-friendly web interface, demonstrating that traditional methods can effectively produce high-quality cartoonized images in real time.

1.1 Background

With the rapid advancement of digital technology, images have become a primary medium for communication and information sharing. The increasing use of images in social media, entertainment, education, and mobile applications has created a strong demand for creative and visually appealing image transformations.

Image cartoonization is a non-photorealistic rendering technique that simplifies real images into artistic cartoon-style representations. It reduces complex textures while preserving important edges and color regions. Traditional cartoon creation requires manual effort and artistic skills, which makes automated cartoonization an attractive solution.

Classical digital image processing techniques such as edge detection, bilateral filtering, and color quantization provide an efficient way to achieve cartoon effects without the need for large datasets or high computational resources. These techniques form the foundation of the proposed system.

1.2 Problem Statement

Manual cartoon creation is time-consuming and requires artistic expertise.
Existing automated solutions based on deep learning demand large datasets and high computational resources.
There is a need for an efficient, lightweight, and user-friendly cartoonization system using classical image processing methods.

1.3 Objectives

The main objectives of this project are:

- To design and implement an automated image cartoonization system using classical Digital Image Processing techniques.
- To convert real-world images into multiple cartoon-style representations while preserving important visual features.
- To apply edge detection, color quantization, and edge-preserving filtering methods for cartoon generation.
- To develop a user-friendly web-based interface for image upload, processing, and result download.
- To analyze and evaluate the visual quality and performance of the implemented cartoonization techniques.

Chapter 2: Literature Review

Image cartoonization has been widely studied as a part of non-photorealistic rendering in digital image processing. Early research in this area focused on simplifying images by combining edge detection techniques with region-based smoothing. These approaches aimed to preserve important structural details while reducing visual complexity.

Tomasi and Manduchi introduced the bilateral filter, which became a fundamental tool for edge-preserving image smoothing. This filter allows images to be smoothed without blurring edges, making it highly suitable for cartoonization tasks. Many cartoonization methods utilize bilateral filtering to produce smooth color regions while maintaining strong object boundaries.

Color quantization is another important technique used in image cartoonization. K-means clustering is commonly applied to reduce the number of colors in an image by grouping similar pixel values. This process creates a posterized effect that resembles hand-drawn cartoons. Several studies have shown that combining color quantization with edge masks significantly improves cartoon-style outputs.

Edge detection techniques such as adaptive thresholding and Canny edge detection have also been extensively used to extract prominent outlines from images. These edges are often combined with smoothed or quantized images to enhance the cartoon appearance.

In recent years, deep learning-based approaches such as neural style transfer and generative adversarial networks (GANs) have been proposed for image cartoonization. Although these methods produce visually rich results, they require large datasets, extensive training, and high computational resources. Due to these limitations, classical image processing techniques remain popular for real-time applications and academic projects. Based on the reviewed literature, classical digital image processing methods provide an effective, efficient, and reproducible solution for image cartoonization, which forms the foundation of this project.

Chapter 3: Methodology

The methodology of this project is based on classical Digital Image Processing techniques to convert real-world images into cartoon-style representations. The overall process consists of image preprocessing, edge extraction, color simplification, and artistic rendering. Each stage contributes to enhancing the cartoon effect while preserving important visual information.

3.1 System Overview

The system takes an input image from the user and processes it through multiple cartoonization pipelines. Each pipeline applies a different combination of filters and transformations to generate distinct cartoon-style outputs. The final results are displayed through a web-based interface, allowing users to view and download the processed images.

3.2 Image Preprocessing

Before applying cartoonization techniques, the input image is preprocessed to ensure consistent results. The image is converted into an OpenCV-compatible format and resized only if necessary. Noise reduction is performed using median or Gaussian filtering to remove unwanted distortions while maintaining important image details.

3.3 Edge Detection

Edge detection is a crucial step in cartoonization, as edges define the outlines of objects in the image. In this project, adaptive thresholding is applied to grayscale images to extract prominent edges. This method adapts to local intensity variations and produces clear and continuous outlines suitable for cartoon effects.

3.4 Edge-Preserving Smoothing

To simplify color regions without losing important boundaries, bilateral filtering is applied. Unlike traditional smoothing filters, the bilateral filter preserves edges while reducing texture and noise. This step helps create smooth, uniform color regions characteristic of cartoon images.

3.5 Color Quantization

Color quantization is performed using K-means clustering to reduce the number of colors in the image. Pixels are grouped into a fixed number of clusters, and each pixel is replaced with its cluster center value. This process produces a posterized appearance that enhances the cartoon-like effect.

3.6 Advanced Cartoonization Pipeline

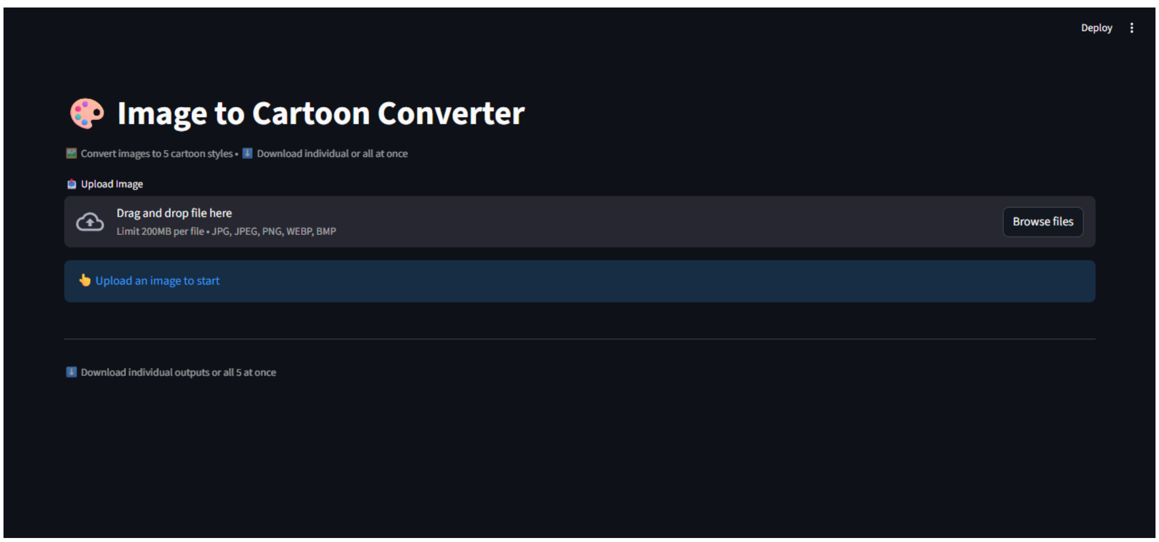
An advanced cartoonization pipeline is implemented by combining multi-pass bilateral filtering, refined edge detection, and higher-resolution color quantization. This approach generates more detailed and visually refined cartoon outputs compared to basic methods.

3.7 Output Generation

The final cartoon images are generated by combining edge masks with smoothed or quantized color images. Multiple cartoon-style outputs are produced for a single input image, and the results are presented to the user through the graphical interface.

Chapter 4: System Implementation

The system implementation of the Image to Cartoon Converter focuses on translating the classical digital image processing methodology into a working software application. The system is designed to be interactive, efficient, and user-friendly, allowing users to convert images into multiple cartoon-style outputs with minimal effort. The implementation leverages Python as the core programming language, OpenCV for image processing, and Streamlit for a web-based interface.



4.1 Tools and Technologies

The development of the system utilized the following tools and technologies:

Component	**Details**
Programming Language	Python 3.10
Image Processing Libraries	OpenCV, NumPy, Pillow
User Interface Framework	Streamlit
File Handling	OS, Zipfile libraries for batch downloads
IDE / Environment	Visual Studio Code, Jupyter Notebook for testing
Platform	Cross-platform (Windows/Linux/Mac)

- **Python:** Chosen for its simplicity, extensive libraries, and strong support for image processing tasks.
- **OpenCV:** Provides robust methods for filtering, edge detection, stylization, and image manipulation.
- **Streamlit:** Used to create an interactive web-based interface, allowing easy image upload, preview, and download without complex front-end programming.
- **NumPy & Pillow:** Used for numerical operations and image file handling.

4.2 Application Architecture

The system architecture follows a modular design, dividing the workflow into discrete processing components. This ensures scalability and maintainability.

Figure 4.1 – System Architecture Diagram Placeholder

Description of Workflow:

1. **Image Upload Module:**
    - Users can upload a single image or multiple images in standard formats (JPEG, PNG).
    - Input validation ensures only valid images are processed.
2. **Preprocessing Module:**
    - Converts images into OpenCV-compatible format.
    - Optional resizing to maintain resolution constraints.
    - Noise reduction using Gaussian or median filtering.
3. **Cartoonization Pipelines:**
   The system supports multiple pipelines:
    - **Edge-Preserving Method:** Bilateral filtering + adaptive thresholding.
    - **Color Quantization:** K-means clustering + edge mask overlay.
    - **Artistic Stylization:** OpenCV's stylization function.
    - **Pencil Sketch Extraction:** Grayscale edge extraction + tone mapping.
    - **Advanced Cartoon Pipeline:** Multi-pass bilateral filtering + refined edge detection + high-resolution color quantization.
4. **Output Generation Module:**
    - Combines edge masks with processed color images.
    - Generates multiple outputs per input image.
    - Displays results in the Streamlit interface alongside the original image.
5. **Download Module:**
    - Users can download individual cartoonized images.
    - Batch download option compresses all outputs into a ZIP file.

Chapter 5: Experimental Results and Analysis

The experimental phase of the Image to Cartoon Converter project focuses on evaluating the effectiveness, quality, and performance of the implemented cartoonization techniques. Experiments were conducted on a variety of sample images, including portraits, landscapes, and mixed-content scenes. The main objectives of this chapter are to demonstrate the visual results, analyze the differences among cartoonization methods, and assess the system's computational efficiency.

5.1 Test Setup

- **Hardware:** Intel Core i5, 16 GB RAM
- **Software:** Python 3.10, OpenCV, Streamlit
- **Sample Images:** 10–15 images of varying content
- **Resolution:** 1024×768 pixels (average)

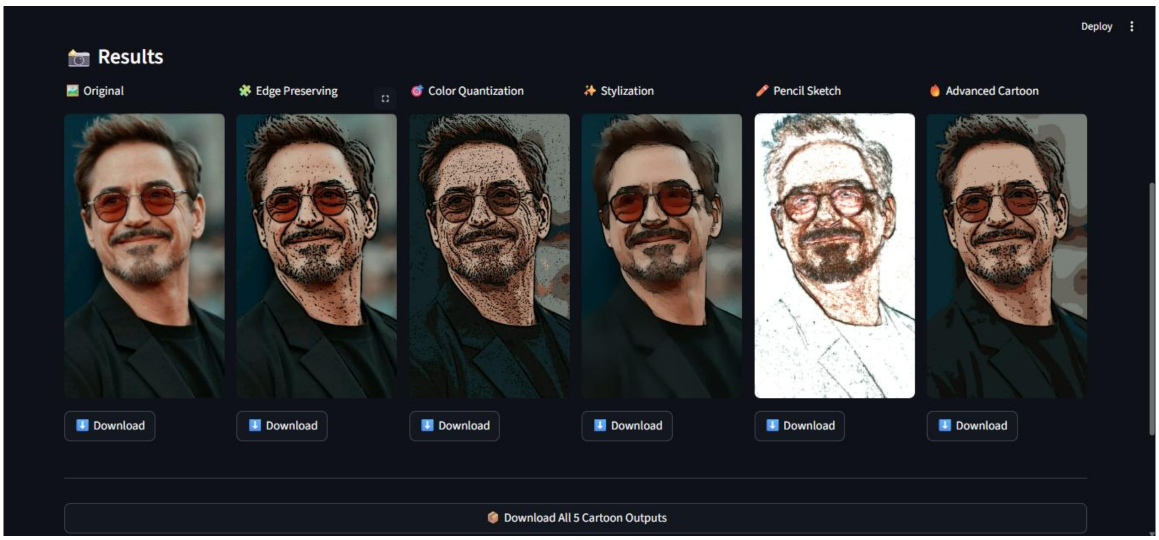
5.2 Qualitative Analysis

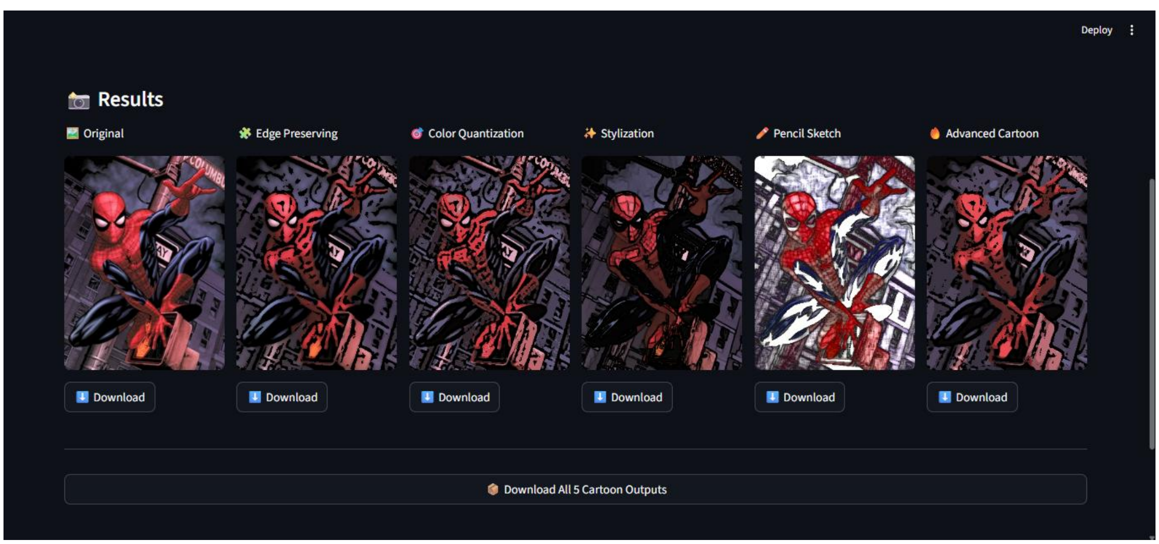
Five cartoonization methods were tested:

Method	**Edge Clarity**	**Color Smoothness**	**Visual Appeal**
Edge-Preserving	High	Medium	Moderate
Color Quantization	Medium	High	Moderate
Artistic Stylization	Low	High	High
Pencil Sketch	Very High	Low	Moderate
Advanced Cartoon	High	High	Very High

Observations:

- Edge-Preserving: Clear outlines, simple cartoon effect
- Color Quantization: Posterized colors, some detail loss
- Artistic Stylization: Soft and painterly, less defined edges
- Pencil Sketch: Monochrome, good artistic look
- Advanced Cartoon: Best overall balance of edges and colors





5.3 Performance Analysis

Average processing time per image:

Method	**Avg. Time (*s*)**
Edge-Preserving	1.2
Color Quantization	1.0
Artistic Stylization	1.5
Pencil Sketch	0.8
Advanced Cartoon	2.5

- Pencil Sketch is fastest, Advanced Cartoon takes longest due to multi-pass filtering.
- All methods are suitable for real-time processing at standard resolutions.

Chapter 6: Discussion

The experimental results demonstrate that classical digital image processing techniques can effectively generate cartoon-style images. Each method exhibits unique characteristics suitable for different use cases:

1. **Edge-Preserving Method:** Produces sharp edges and smooth color regions, making it ideal for portraits and images with well-defined structures.
2. **Color Quantization (K-means):** Creates a posterized effect that emphasizes dominant colors. Minor details may be lost, but the result is visually appealing for stylized images.
3. **Artistic Stylization:** Generates a soft, painterly look with smooth color transitions. Edges are less pronounced, giving a more artistic rather than cartoon-like effect.
4. **Pencil Sketch:** Provides a monochrome sketch representation, highlighting edges and outlines without color information. Useful for artistic illustrations.
5. **Advanced Cartoon Pipeline:** Combines multiple techniques to produce balanced outputs with clear edges, smooth colors, and high visual appeal, making it the most versatile method.

Chapter 7: Conclusion and Future Work

7.1 Conclusion

This project successfully developed an **Image to Cartoon Converter** using classical Digital Image Processing techniques. The system converts real-world images into multiple cartoon-style representations while preserving essential visual details.

Key achievements of the project:

- Implemented five distinct cartoonization methods, including edge-preserving smoothing, color quantization, artistic stylization, pencil sketch, and an advanced multi-pass pipeline.
- Developed a **user-friendly web interface** using Streamlit, enabling easy image upload, preview, and download.
- Demonstrated that classical image processing methods are **computationally efficient, reproducible, and capable of producing visually appealing results** without requiring deep learning or large datasets.
- Conducted experiments on diverse images, analyzing both visual quality and processing performance, confirming the effectiveness of the implemented techniques.

The project highlights that **classical approaches remain highly relevant** for real-time applications, educational purposes, and lightweight image stylization tasks.

7.2 Future Work

Potential enhancements for the system include:

1. **Deep Learning Integration:** Incorporating neural network-based stylization for more advanced and flexible cartoon effects.
2. **Real-Time Video Cartoonization:** Extending the system to process video streams for animated or live cartoon effects.
3. **User-Adjustable Parameters:** Allowing users to control filter strength, number of color clusters, and edge intensity to create personalized cartoon styles.
4. **Mobile Application Deployment:** Developing a mobile version of the system for on-the-go image cartoonization.
5. **Performance Optimization:** Implementing parallel processing or GPU acceleration to improve processing time for high-resolution images.

By incorporating these improvements, the system can evolve into a more versatile, real-time, and customizable image cartoonization tool suitable for a wide range of applications.

References

1. Tomasi, C., Manduchi, R. Bilateral Filtering for Gray and Color Images.

2. Gonzalez, R. C., Woods, R. E. Digital Image Processing.

3. OpenCV Official Documentation.