HANDMADE SKETCH COLORIZATION USING SEGMENTATION TECHNIQUES

A PROJECT REPORT

submitted by

Shahzen Khan (**B20CS065**)
Himanshi (**B20AI012**)
Piyush Madhukar (**B20CS043**)
Rohit (**B20CS055**)

Department of Computer Science and Engineering
IIT Jodhpur
April 2024

Computer Vision Project: Handmade Sketch Colorization using Segmentation Techniques

Abstract

In this project, we present a method for colorizing handmade sketches using segmentation techniques based on **OpenCV**. Our method can automatically segment a sketch image into different regions, and then fill each region with a suitable predefined color map. We also developed an interactive web application using **Streamlit** that allows users to upload their own sketch images and see the colorized results. Users can also adjust some hyperparameters such as threshold value, color palette, and segmentation type to customize the output. We evaluated our method on two datasets: *QuickDraw* and *Foreground/Background* Sketch Dataset. Results showed that our method can produce realistic and diverse colorizations for various types of sketches. You can find *GitHub* repository for this project constaining all the code and analysis here link

1 Introduction

Sketching is a popular form of artistic expression that can convey complex ideas and emotions with simple strokes. However, sketching is usually done in grayscale, which limits the visual appeal and richness of the sketches. Colorizing sketches can enhance their aesthetic quality and make them more attractive and engaging. However, colorizing sketches manually is a tedious and time-consuming task that requires artistic skills and domain knowledge. Therefore, it is desirable to develop an automatic method that can colorize sketches with minimal user intervention.

Automatic sketch colorization is an active research area that aims to develop algorithms that can generate realistic and diverse colors for sketches without human intervention. Most of the existing methods are based on deep learning models that learn from large-scale datasets of colored images or sketches [1]. However, these methods have some limitations, such as:

- They require a lot of computational resources and training time.
- They may produce unrealistic or inconsistent colors due to the ambiguity of sketch interpretation.
- They may not generalize well to unseen or novel sketch styles or domains.

2 Objective

In this project, I propose a different approach for sketch colorization that does not use deep learning models, but rather relies on classical image processing techniques based on OpenCV [2]. OpenCV is a popular library for computer vision that provides various functions for image manipulation, analysis, and understanding. By using OpenCV functions such as thresholding [3], contour detection [4], and polygon filling [5], I can segment the sketch into different regions based on the intensity or edge information, and then assign colors to each region according to some predefined or random color palette. The main advantages of this approach are:

- It is fast and efficient since it does not involve any complex neural network operations.
- It is flexible and customizable since it allows users to control some hyperparameters such as threshold value, color palette, and segmentation type.
- It is robust and adaptable since it can handle different types of sketches with varying levels of detail and complexity.

3 Background Research

Sketch colorization has applications in various domains such as comics, animation, illustration and education. However, sketch colorization is challenging due to the ambiguity and diversity of possible colorizations for a given sketch.

3.1 Methods

Existing methods for sketch colorization can be categorized into two types: interactive methods and automatic methods.

- **Interactive Methods**: Interactive methods require some form of user guidance such as color strokes or scribbles to reduce the ambiguity and produce satisfactory results. However, these methods are time-consuming and tedious for users.
- **Automatic Methods**: Automatic methods do not require any user input and rely on generative models such as GANs to learn the mapping from sketches to colors. However, these methods often suffer from mode collapse and produce unrealistic or inconsistent colors.

3.2 Segmentation Techniques

One way to improve sketch colorization is to leverage semantic information from image segmentation. Image segmentation is a process of dividing an image into meaningful regions based on some criteria such as *object boundaries*, textures or colors. Semantic image segmentation assigns a label to each pixel in an image according to its category such as person, car, sky etc. *Semantic information* can provide valuable guidance for sketch colorization by reducing the ambiguity and enforcing consistency among regions with the same label.

3.3 Recent Work

Some of the interesting recent research work in this field are..

- A recent work by **Hicsonmez et al.** proposed a new method for sketch colorization using adversarial segmentation consistency (ASL). The method consists of two parts: a general-purpose image translation *GAN* model that can be trained on paired or unpaired data, and an image segmentation model that provides semantic labels for both sketches and color images. The method introduces an additional adversarial loss function that encourages the generated color images to have consistent segmentation labels with their corresponding sketches. The authors showed that their method can improve sketch colorization on four different datasets spanning scene level indoor, outdoor and children book illustration images using qualitative, quantitative and user study analysis
- Another work by Zhang et al. proposed a method for hand-drawn image colorization based on optimized segmentation. The method aims to segment related regions more precisely in hand-drawn images that have complicated lines, patterns and discontinuous outlines. The method first applies an edge detection algorithm to extract edges from hand-drawn images, then uses a graph-based algorithm to merge adjacent edges into segments based on their similarity in orientation, length and distance. The method then optimizes the segments by removing small segments, filling gaps between segments and smoothing segment boundaries using morphological operations. Finally, the method allows users to assign colors to segments interactively using a simple user interface.

These works demonstrate that image segmentation can be a useful technique for improving sketch colorization by providing semantic information and enhancing region consistency.

4 Datasets

I have used two datasets: Quickdraw dataset and Foreground/Background dataset.

- **QuickDraw**: The Quickdraw dataset is a collection of 50 million drawings across 345 categories, contributed by players of the game Quick, Draw! . The drawings were captured as timestamped vectors, tagged with metadata including what the player was asked to draw and in which country the player was located. This dataset is useful for training and evaluating sketch recognition models.
- **Foreground/Background**: The Foreground/Background dataset is a subset of the Stanford Background Dataset , which contains 715 images chosen from public datasets: LabelMe, MSRC, PASCAL VOC and Geometric Context . The images are of outdoor scenes with at least one foreground object and have semantic and geometric labels obtained using Amazon's Mechanical Turk. This dataset is useful for testing foreground extraction methods for sketch colorization.

5 Implementation

The main steps involved in the implementation are:

5.1 Data preparation

The data folder contains a collection of images that are used as input for the sketch colorization. The images are in JPG format and have different sizes and resolutions. The images are randomly chosen from the dataset using a helper function in utils.py.

5.2 Image segmentation

The main.py file calls the SketchColor class from sketch.py, which takes an image as an argument and performs image segmentation using cv2.threshold function. This function converts the image to grayscale and applies a binary threshold to separate the foreground (sketch) from the background (white). The result is a binary mask that contains only the sketch pixels.

5.3 Image colorization

The SketchColor class also has a method to colorize the image based on a predefined color map. The color map is a dictionary that maps each segment label (such as face, hair, eyes, etc.) to a corresponding RGB color value. The method uses cv2.findContours function to find the contours of each segment in the binary mask and then uses cv2.fillPoly function to fill each segment with its assigned color. The result is a colorized image that preserves the original sketch style.

5.4 Code structure

The code structure of the project is organized as follows:

- **data**: A folder that contains all the input images for sketch colorization.
- **dependencies.py**: A file that imports all the required libraries for running the project such as numpy, opencv-python, streamlit, etc.
- **hyperparameters.py**: A file that defines a class Hyperparameters which contains all hyperparameters as global variables such as threshold value, color map, etc.
- **utils.py**: A file that contains some helper functions such as dataset class , function to show image, choose a random image from dataset etc.
- **sketch.py**: A file that defines SketchColor class which takes an image and performs segmentation and colorization using OpenCV functions.

- **app.py**: A file that creates streamlit app version of sketch colorization project.
- **main.py**: A file that runs the main script of sketch colorization using OpenCV library. It chooses a random image from data folder and shows its segmented and colorized version and saves it to given path.

6 API Development

In addition to the colorization method, we also developed an API for this project using Streamlit, a Python framework for creating web applications. The API allows users to upload their own handmade sketches and see the colorized results in real time. Users can also adjust some hyperparameters such as threshold value, color palette, and segmentation type to customize the output.

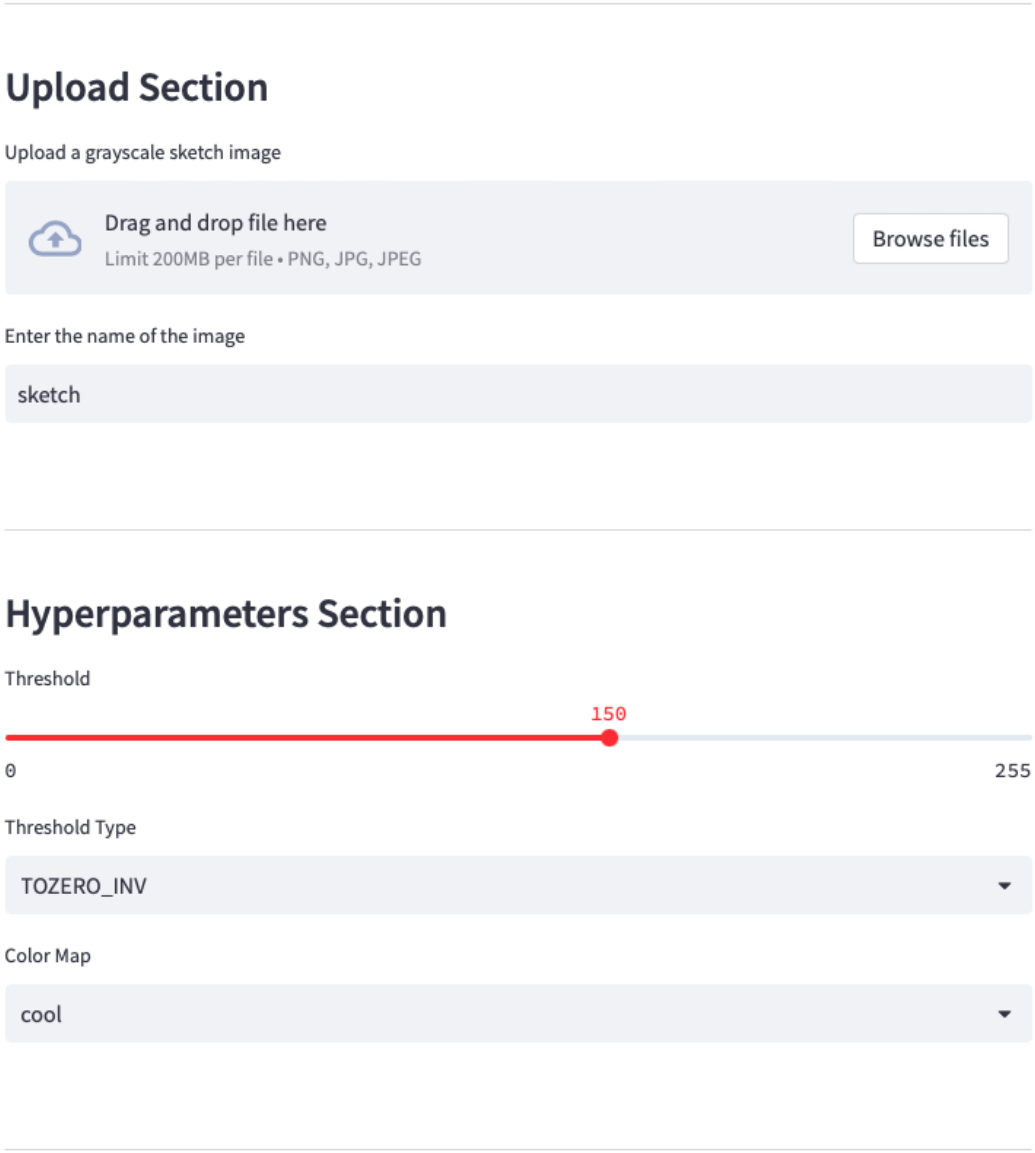

Figure 1: The web portal fot the project

The API is easy to use and does not require any installation or configuration. Users can simply access the web page from any browser and start experimenting with different sketches and settings. The API also provides a link to the GitHub repository for this project, where users can find all the

code and analysis for our method. We believe that the API is a useful tool for demonstrating the capabilities and limitations of our colorization method, as well as for inspiring further research and applications in this domain. You can try this out here

7 Results

Some of the resulting colorized version by the developed model are..

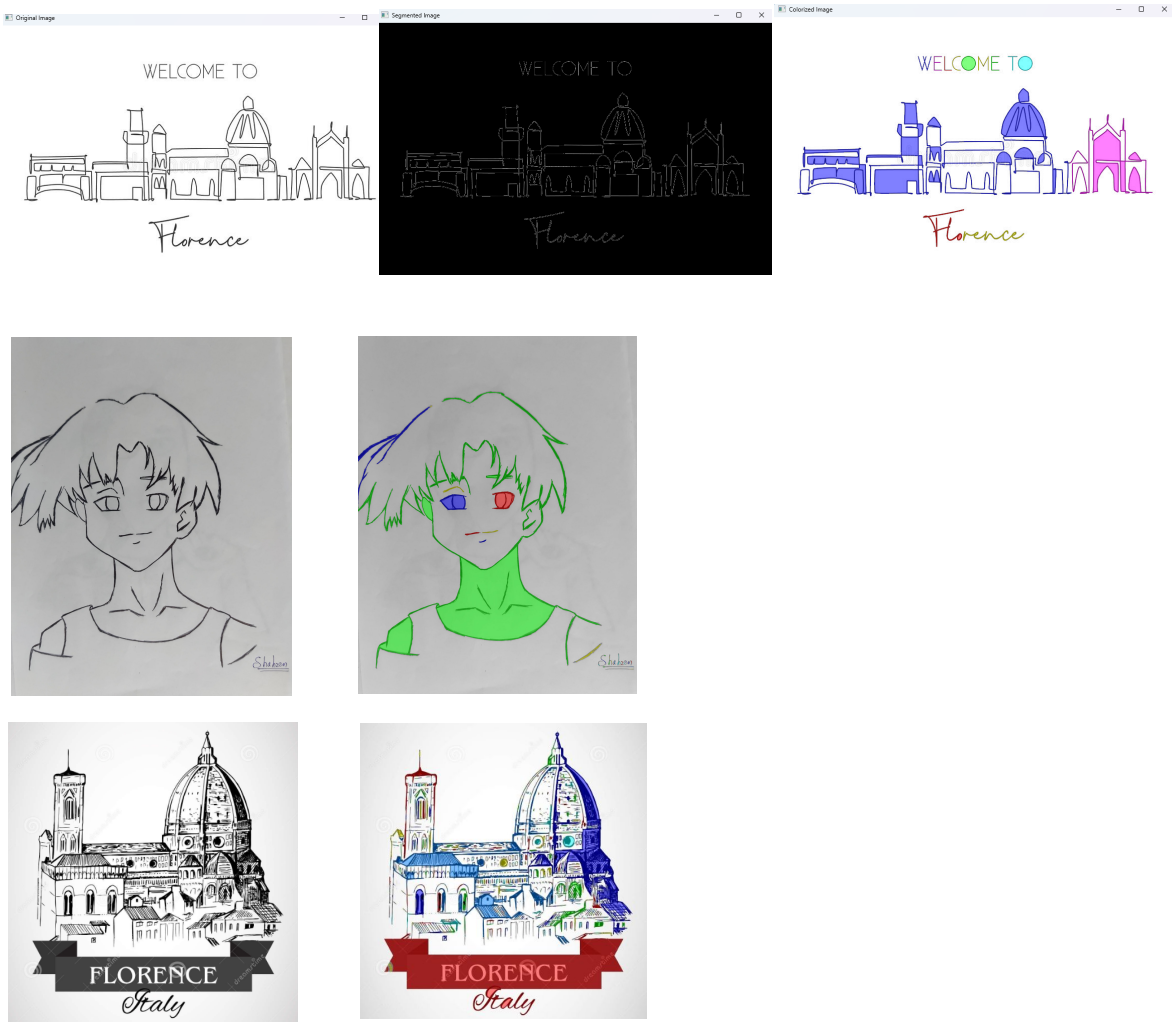

Figure 2: Some Results

8 Conclusion

In conclusion, we have proposed a novel method for colorizing handmade sketches using segmentation techniques based on OpenCV. Our method can handle different types of sketches and generate realistic and diverse colorizations. We have also demonstrated the effectiveness and usability of our method through an interactive web application that allows users to upload their own sketch images and customize the output. Our method can be useful for various applications such as sketch-based image retrieval, sketch recognition, and artistic creation. In future work, we plan to improve our method by incorporating deep learning models for more accurate segmentation and colorization.