# COL783: Digital Image Analysis

# Assignment 1: Artistic Image Enhancement, and Style Transfer

August 28, 2023

**Collaborators:**

- Shashank Krishna Vempati (2022AIY7509)
- Kaustubh R Borgavi (2023SIY7539)

## Introduction

The report consists of 3 parts involving different stages of Artistic Image enhancement and style transfer as follows:

- Simple Image Enhancement
- Quantized Rendering
- Artistic Style Transfer

## Part 1: Simple Image Enhancement

In this part of the assignment, the steps for Simple image enhancement consist of two main steps namely, The artistic Enhancement step and the colour adjustment step.

(a) Input Image

(b) Shadow Map Output

Figure 1: Generating Shadow-map of the input image

### 1.1: Artistic Enhancement Step

The main aim of this step is to treat the light-shadow contrast. It consists of three sub-stages, namely shadow map generation, edge map generation and line draft generation. We begin this process by choosing an input image. An example of the input image is shown in the above Figure 1a.

#### 1.1.1: Shadow-Map generation

In the shadow map generation, each pixel is assigned a flag to denote whether it belongs to a light area or a shadow area (known as the shadow map).
We begin this process by converting the input image which is currently in RGB space into HSI space using the conversion formula mentioned in the Question paper. We normalise the values of H and I and compute the r-map for each pixel $(x, y)$. The HSI color model represents each color into three components: Hue (H), Saturation (S) and Intensity (I). The output image is shown in Figure 1b

| | Min. | Max. | Optimal value |
|---|---|---|---|
| **T** | 0 | 255 | 111 |
| $\lambda$ | 0 | 0.99 | 0.8 |

Table 1: Hyperparameters in Shadow-Map Generation

#### 1.1.2: Line Draft Generation

- **Grayscale Conversion:** We used OpenCV for Grayscale conversion. OpenCV has an inbuilt function to convert an image from BGR space (as OpenCV reads the image in BGR space by default) to Grayscale version. Internally, each pixel transforms from BGR to Grayscale by combining blue, green and red values in a fixed proportion. ( 0.299 * R + 0.587 * G + 0.114 * B).
- **Bilateral Filtering:** Filters like Sobel or Prewitt filters take spacial information alone to process each pixel and hence we might end up smoothing the image even at the edges. To preserve the edges in the image, the Bilateral filtering process considers spatial information as well as the range of intensity information which helps in preserving the edges. We implement Bilateral filtering using Python by referring the procedure given in [1]. We tried out various values for the Hyperparameter and selected the optimal values shown in Table 2 which showed the best results for imparting an Artistic style on the input image. We show additonal experiments with different values for the Hyperparameters in the Ablation section of this report.

| | Optimal value |
|---|---|
| **Diameter** | 5 |
| $\sigma_{space}$ | 50 |
| $\sigma_{colors}$ | 50 |

Table 2: Hyperparameters in Bilateral filtering

- **Edge Map:** Applied $Sobel_X$ and $Sobel_Y$ filter on the output image from bilateral filtering. We then computed the edge map E using the formula given in the Question Paper.

| | Optimal value |
|---|---|
| **Filter size** | $5x5$ |

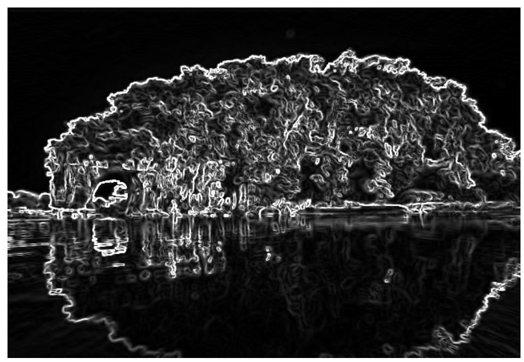
Table 3: Hyperparameters in Edge-Map Generation

- **Thresholding:** Here, we are asked to perform various ablations and choose an optimal threshold value based on the input image. The Line Draft threshold is adjusted to this optimal value, and then the output image is generated using the formula provided in the Question Paper.
  The output images from the above steps is shown in Fig. 2
  We show the effect of various values of the Threshold in the Ablation section for generating the Line Draft image from the input.

| | Min. | Max. | Optimal range |
|---|---|---|---|
| **T** | 0 | 255 | $[130 \pm 30]$ |

Table 4: Hyperparameter in Line-Draft Generation

(a) Bilateral Filtering Output



(b) Edge Map Output

(c) Line Draft Output (Inverted Version) with T=130

Figure 2: Steps in Line Draft Generation

### 1.2: Color Adjustment Step:

We begin this step of enhacement by transforming the input image from RGB to LAB space to separate the light and chromatic channels. We modify the L channel of image in LAB space to contain the mid-value of the entire light channel and we combine this channel further with the A and B channels to obtain chromatic map as depicted in Figure 3a.

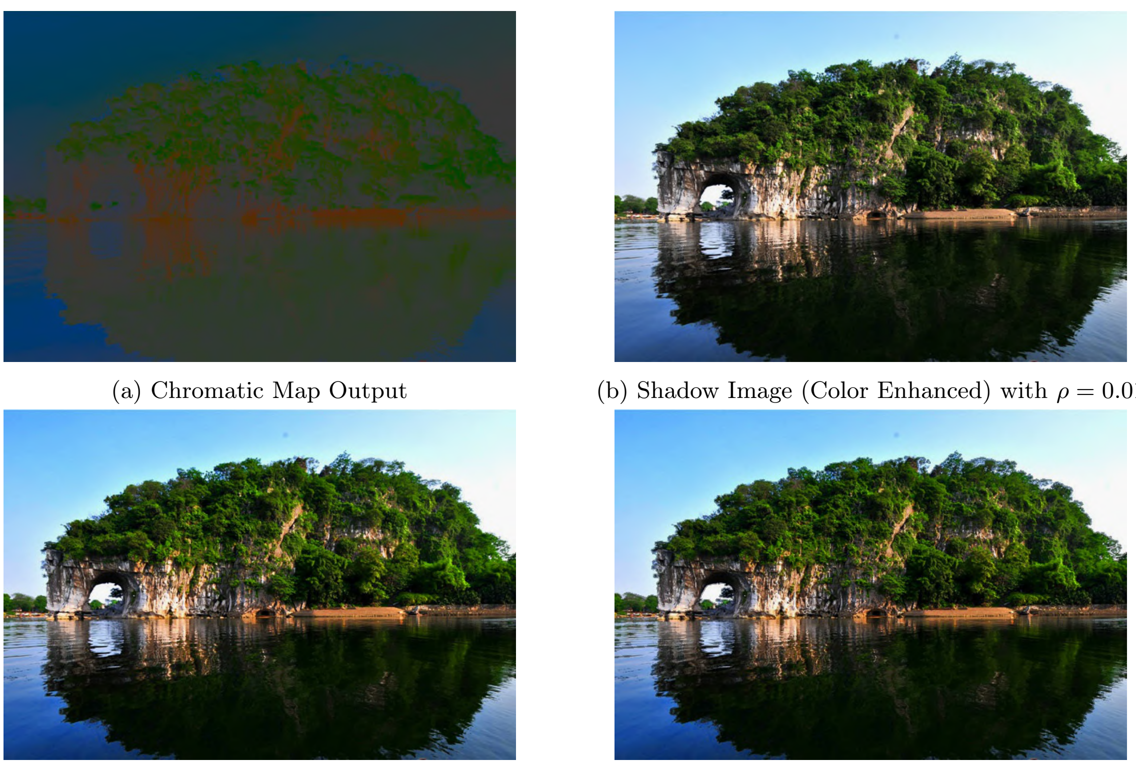
Using the obtained chromatic map, the color components (R, G, and B) of each pixel have been enhanced using a specific formula provided in the assignment question paper. The formula applies a tanh function to the difference between the chromatic map and 128, which is then scaled by a hyper-parameter $\rho$. This operation has enhanced the color components while retaining the shadow information, as illustrated in Figure 3b.

Furthermore, we apply linear saturation correction to address any alterations in the image's overall brightness. Consequently, we observe that the image now exhibits vibrancy with various colors and appropriate lighting, as demonstrated in Figure 3c.

This step is followed by the generation of the Artistic Image which involves blending the Line Draft generated in the previous steps with the color enhanced version of the Shadow Image. This creates a visually pleasing image with an artistic touch, as illustrated in Figure 3d. We observed that the Line Draft information enhances the image by incorporating sharp and crisp edges, providing a vivid display of colors with distinct color transitions. The enhanced shadows in the image strengthen them which in turn helps the image to look artistic in nature or painting-like.

| | Min. | Max. | Optimal value |
|---|---|---|---|
| $\rho$ | 0.005 | 0.2 | 0.01 |
| **Saturation Scale** | – | – | $1.3 \pm 0.1$ |
| $\beta$ | 0.0 | 0.99 | 0.7 |

Table 5: Hyperparameters in Color Adjustment Step



(a) Chromatic Map Output

(b) Shadow Image (Color Enhanced) with $\rho = 0.01$

(c) Shadow Image with Saturation Correction

(d) Final Artistic Rendered Image

Figure 3: Steps in Color Adjustment and Artistic Rendered Image

## Part 2: Quantized Rendering

We perform Quantization to the artistically enhanced image obtained from Part 1. We will be applying "Median Cut" as a color quantization method and further apply Floyd-Steinberg's dithering as asked in the assignment question. To apply Median cut, we first begin by flattening the RGB image and then splitting the color space into subsequent bins. The median value of the color space in the flattened image is where we apply splitting. Each split divides the color space into two levels and we recursively split the color space based on the number of quantization levels we want the image to be represented in. In figure 4a, we show an example of splitting the color space 5 times to represent our artistic image using 32 quantization levels. We observe in 4a that the image has uneven flow of color visible in the sky and the water portion as the total colors are now reduced after applying Median Cut to the image.

Now, to apply Floyd Steinberg's dithering process, we consider the original artistic image and the quantization levels obtained from Median Cut in the previous step. For each pixel in the original image, we find the nearest possible quantization level obtained from Median Cut and update the pixel with this nearest value. Further, the Floyd Steinberg's algorithm encourages to propagate the quantization error between the original pixel value and the updated value to all the neighbouring pixels. We do so by applying the Error-propagation formula and updating the neighboring pixels. We carry out these steps for each pixel in the image and continue propogating the error. We notice in Figure 4b that applying Floyd Steinberg's algorithm significantly improves the quantized image with 32 colors and makes it visually resemble the original 24-bit Artistic image. The quantization error computed between the input artistic image and the outputs produced with both methods is given in Table 6.

We use OpenCV to read and write images and the numpy library for flattening the image. We have implemented the rest of the procedure as explained in [2] using Python.

| | Quantization Error( Mean ) |
|---|---|
| Artistic image + Median Cut | 126.72 |
| Artistic image + Median Cut + FS Dithering | 121.47 |

Table 6: Quantitative Evaluation

(a) Median Cut Quantized Image (32 colors)

(b) Floyd Steinberg Dithering (32 colors)

Figure 4: Quantization and Floyd Steinberg Dithering

## Part 3: Artistic Style Transfer

In this part of Artistic Style Transfer, our goal is to apply the artistic effect obtained from the procedure outlined in Part 1 to another similar target image. The paper [3] presents two methods for transferring colors from RGB images to Grayscale images, with and without the use of swatches. The source and target images used for our experiment are depicted in Fig. 5a and Fig. 6a respectively.

To implement the global algorithm without the use of swatches, we start by taking Grayscale target image as shown in Fig. 6a. The source image is then transformed into the $LAB$ color space, followed by linearly shifting the $L$ values of the image. We subsequently clip the pixel values within the range of 0 to 255. We compute the standard deviation for each pixel in the target image, considering a neighbourhood patch of $5 \times 5$ neighboring pixels for each pixel. We precompute this standard deviation matrix for the target image to facilitate further processing. For computational efficiency, the authors of [3] suggest selecting 200 colors randomly from the source image for transfer to the target image. To achieve this, we divide the source image into 200 grids, choosing one color randomly from each grid and storing them in a list. The color transfer process involves computing the absolute difference of $L$ values for each of the selected 200 colors from the source image with the weighted sum of the standard deviation and the $L$ value for each pixel in the target image. We map the color for which the absolute difference of source color's $L$ value and the weighted sum is minimum. This process is carried out for each pixel of the target image. The results of this method are shown in Figure 6b.

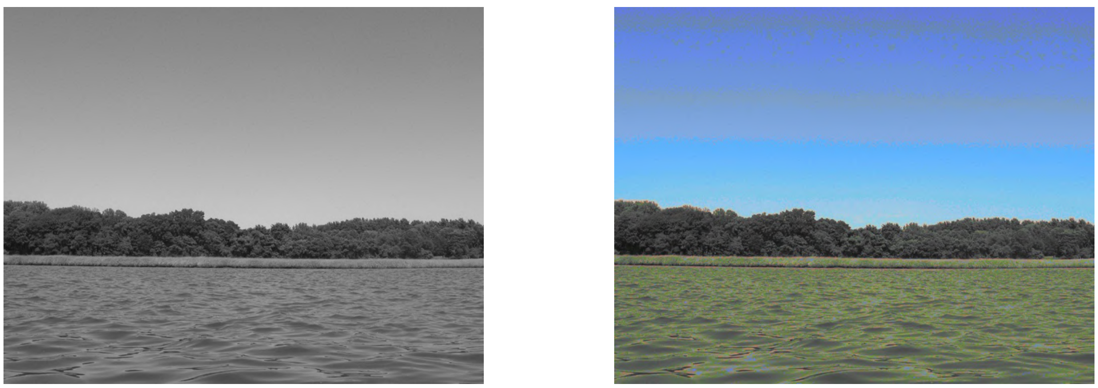
Now, we implement the Artistic Style Transfer algorithm using swatches. We first identify potential swatches by observing regions/objects in the source and target images that contain distinctive colors. For this case, we choose swatches in the source and target image as shown in Fig. 5b and Fig. 6c respectively. Following the procedure in [3], we randomly choose 50 colors from each swatch in the source image and store them in a list. Next, we compare the intensity values from each target swatch region with the $L$ channel values of the colors chosen from each source swatch. We map colors to the pixels in the target swatch for which the computed $L2$ distance between the intensity values is minimal. After coloring all target swatch regions, we apply the same procedure to the remaining pixels in the target image.

The results of Artistic Style Transfer using swatches are shown in Figure 6d. We observe that the style transfer is enhanced when using swatches. Without swatches, color mapping between source and target images is solely based on underlying intensity, potentially leading to regions being inaccurately colored due to shared similar intensities but in the case of swatches, we choose them in such a way that we include the possible colors from each region to appropriately color the target image.

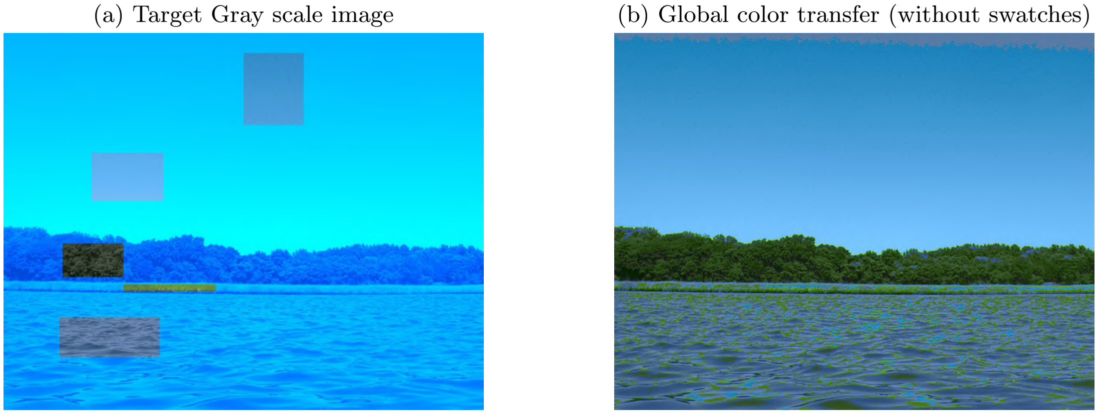
In this section, we utilize OpenCV for reading, writing, and image conversion between $RGB$ and $LAB$ color spaces. We use Numpy for array operations. We have implemented the rest of the functions using Python as per the steps referred from the paper [3].

(a) Source image (Artistic image from Part 1) (b) Source image with selected Swatches

Figure 5: Source image and corresponding swatch selection



(a) Target Gray scale image (b) Global color transfer (without swatches)



(c) Swatches selected in Target image (d) Color transfer (with swatches)

Figure 6: Artistic Style Transfer with and without the use of swatches

## Results and Discussion

We performed Artistic image enhancement on the input image by first locating the shadowed regions and enhancing them using the shadow-map. The next procedure of Line draft generation helps us to locate all the boundaries in the images. The Color adjustment followed by Saturation correction, enhances the colors by maintaining the darker shadows. Lastly, we merge the Line

Draft by enhancing the edges to obtain the final artistic image. Here, we observed that the Artistic enhancement of an image highly depends on the choice of Hyperparameters involved at various stages. As per our experiments, we noticed that the optimal set of Hyperparameters found by us work for most images involving Nature/scenery subjects but for images involving human subjects, we noticed that the hyperparameters would require more tweaking to achieve the desired Artistic enhancement. One of such examples is present in the Ablation section of this report.

Coming to Quantized rendering, "Median Cut" is a color quantization technique that groups colors into clusters by their median values, reducing the palette size for efficient storage and display. It handles color reduction well but may lack detail in complex images. It displays images as a combination of some solid colors where dithering is handy to solve this issue. "Floyd-Steinberg Dithering" helps in error diffusion that distributes color errors to neighboring pixels during conversion from continuous to discrete color spaces, preserving more details. Both methods a dapt to color images by treating each channel separately. While Median Cut simplifies color distribution, Floyd-Steinberg maintains smoother gradients to make humans perceive the image better.

The third part focused on color transfer from colored to grayscale images using two approaches: global transfer and swatch-based transfer. In global transfer, errors arose due to texture similarities causing color inaccuracies. The primary metric, intensity, proved inadequate for precise matching. Swatch-based transfer involved annotating source and target swatches to reduce errors present in global transfer. This improved results, particularly in complex scenes where intensities aligned. However, imperfections persisted. To enhance accuracy, a proposed approach suggests utilizing $k$-nearest neighbors. Colors are chosen based on the top $k/2 + 1$ matches, ultimately selecting the color with the nearest match among them. This method builds upon existing distance measurement between pixel intensities while offering a simple yet effective extension. While more advanced techniques exist, this approach presents a practical solution for addressing color transfer inaccuracies, proving advantageous over global transfer and providing improved results compared to swatch-based methods in challenging scenarios.

## Note:-

- The code was developed using Python version 3.8.15
- Optimal values for hyperparameters are chosen based on the results for the given input image.

## References

[1] Pierre Kornprobst, Jack Tumblin, and Frédo Durand. Bilateral filtering: Theory and applications. *Found. Trends Comput. Graph. Vis.*, 4(1):1–74, 2009.

[2] Paul S. Heckbert. Color image quantization for frame buffer display. In *SIGGRAPH*, pages 297–307. ACM, 1982.

[3] Tomihisa Welsh, Michael Ashikhmin, and Klaus Mueller. Transferring color to greyscale images. *ACM Trans. Graph.*, 21(3):277–280, 2002.

# I. Ablation Study

## Part 1: Simple Image Enhancement

### 1.1: Hyperparameter selection in Shadow-Map Generation

| | Threshold (T=81) | Threshold (T=111) | Threshold (T=141) |
|---|---|---|---|
| Shadow Map |  |  |  |

Table 7: Shadow Map at different T values

| | $\lambda = 0$ | $\lambda = 0.8$ | $\lambda = 0.99$ |
|---|---|---|---|
| Shadow Map |  |  |  |

Table 8: Shadow Image at different $\lambda$ values

### 1.2: Hyperparameter selection in Bilateral Filtering and Line Draft

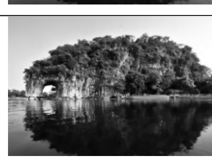
| $\sigma_{space} \backslash \sigma_{color}$ | 10 | 50 | 100 | 200 |
|---|---|---|---|---|
| 10 |  |  |  |  |
| 50 |  |  |  |  |
| 100 |  |  |  |  |
| 200 |  |  |  |  |

Table 9: Bilateral Filtering at different $\sigma_{space}$ and $\sigma_{color}$ values with fixed diameter=5

| | $T = 100$ | $T = 130$ | $T = 160$ |
|---|---|---|---|
| Line Draft |  |  |  |

Table 10: Line Draft at different T values

### 1.3: Hyperparameter selection in Color Adjustment

| | $\rho = 0.005$ | $\rho = 0.01$ | $\rho = 0.2$ |
|---|---|---|---|
| Shadow Image (Enhanced) |  |  |  |

Table 11: Shadow Image at different $\rho$ values

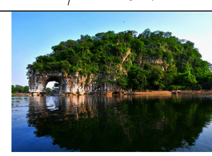
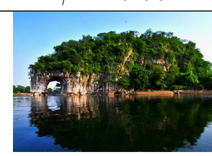
| | $\beta = 0$ | $\beta = 0.7$ | $\beta = 0.99$ |
|---|---|---|---|
| Artistic Image |  |  |  |

Table 12: Artistic Image at different $\beta$ values

## Part 2: Quantized Rendering

### 2.1: Median Cut Images

| | Colors=4 | Colors=32 | Colors=128 |
|---|---|---|---|
| Median Cut Image |  |  |  |

Table 13: Median Cut Images at different quantization levels

### 2.2: Floyd Steinberg Dithered Images

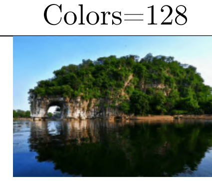
| | Colors=4 | Colors=32 | Colors=128 |
|---|---|---|---|
| Dithered Image |  |  |  |

Table 14: Floyd Steinberg's Dithering at different quantization levels

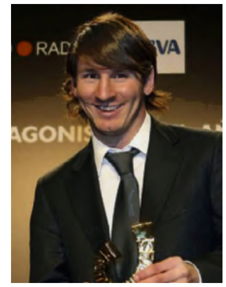
# II. Other Experiments

## 1: Alternate Image Analysis (same set of hyperparameters as above)


(a) Source Image


(b) Edge Map Output


(c) Line Draft Output




(d) Artistic Image


(e) Median Cut Quantized Output (32 colors)


(f) Dithered Output (32 colors)

Figure 7: Outcomes for an alternate image

## 2: Color Transfer using Global match

(a) Source Image

(b) Target Image

(c) Colored Image without swatch

(d) Source Image

(e) Target Image

(f) Colored Image without swatch

(g) Source Image

(h) Target Image

(i) Colored Image without swatch

Figure 8: Part 3 experiment on various other images

### 3: Color Transfer using Global match and Swatches

(a) Source Image

(b) Target Image

(c) Colored Image without swatch

(d) Colored Image with swatch

Figure 9: Part 3 experiment with and without swatches on other images