



Palette-based Photo Recoloring

Team Name : Kanyaraasi

Team Members : Anudeep Chaluvadi (20171042, ECE)
Pavan Kalyan Reddy (20171205, ECE)
Sushanth Reddy (20171172, ECE)
Rupa Reddy (20171111, ECE)
Pavan Chaitanya Reddy (20171109, CSE)

Mentor TA : Surendra Kumar Reddy

Repo Link : <https://github.com/Digital-Image-Processing-IIITH/project-kanyaraasi>



Introduction

- Colour Manipulation is a key process in photo enhancement.
- Image editing applications offer a wide array of tools for color manipulation.
- Some of these tools are easy to understand but offer a limited range of expressiveness. Other more powerful tools are time consuming for experts and inscrutable to novices.
- Researchers have described a variety of more sophisticated methods but these are typically not interactive, which is crucial for creative exploration.



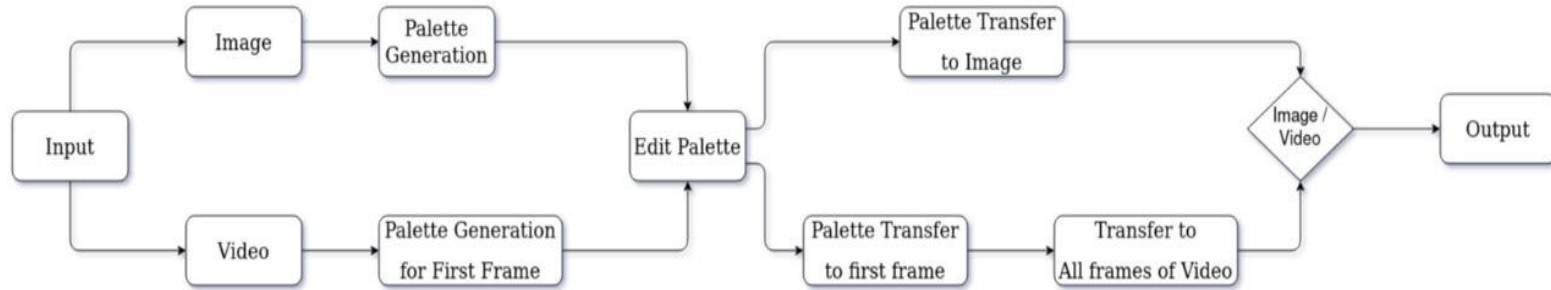
Project Description

The project is to implement a simple, intuitive and interactive tool that allows non-experts to recolor an image by editing a color palette.

This system is comprised of several components:

- A GUI that is easy to learn and understand.
- An efficient algorithm for creating a color palette from an image
- And a novel color transfer algorithm that recolors the image based on a user-modified palette.

Pipeline of the Algorithm





Palette Generation

Our goal is to select a set of k colours (Palette) which can be used as “controls” during editing. For this, we used **K-means** approach where we segregate N colors into K different clusters.

What is the value of K ?

What are the issues with the naive k -means ?

What can be done to make it efficient ?



Acceleration

- Exploiting the property that our data are colors restricted to $R, G, B \in [0, 1]$, we assign them to bins in a $b \times b \times b$ histogram (we use $b = 16$ in RGB).
- We have reduced the data used in the computations by at least a couple orders of magnitude.
- Data Independent of Image size

Are the computations Done differently now ?

Yes, we use a weighted mean (no. of pixels) associated with the bin when finding the k-means.



Efficient Implementation

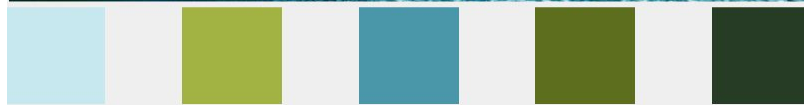
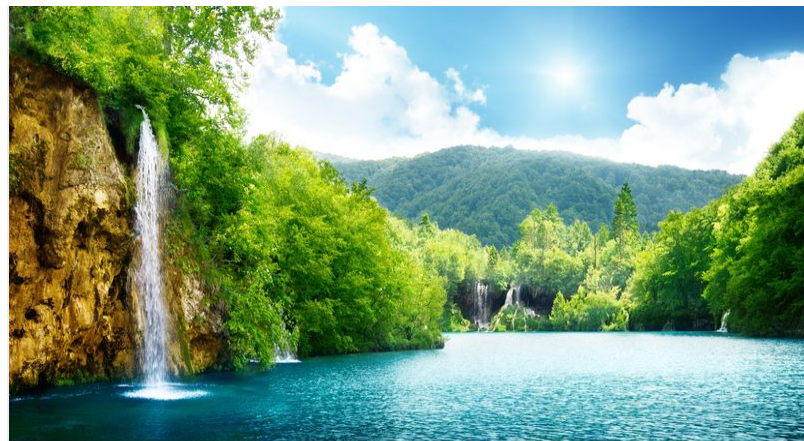
- To present the most helpful user friendly GUI to the user, we prefer the algorithm to be deterministic and also the palette colors be far from one another.

What Can we Do ? --- Initialisation.

- Another observation was that when choosing a palette based on pixel color clustering often leads to a very dark (near black) palette entry, because typically a significant number of image pixels are dark.

What Can we Do here now ? --- Compute $(k+1)$ means , where one of them is initialized and perpetually locked to black. At last, we discard the black entry to leave k remaining palette colors

Some Palettes





Color-Transfer Goals

Now that we have a set of palette colors $\{C_i\}$, and our GUI allows the user to modify the palette colors to $\{C'_i\}$. We now define a transfer function f that maps colors in the original image to colors in the edited image. These are some of the key properties we would like in f that address:

- Interpolation
- In - gamut
- Pixel Continuity
- Palette Continuity
- One-to-one
- Dynamic Range



Color-Mapping

- All the colour mappings are performed in LAB color space to provide perceptual uniformity in the falloff.
- We design a transfer function that has two orthogonal components – f_L (which modifies pixel luminance based on the palette luminance) and f_{ab} (which modifies the corresponding ab values).
- The luminance transfer function simply takes a weighted combination of the two nearest palette entries (or one nearest entry and either black or white if the pixel is darker or brighter than all the palette entries).

Does this follows the colours transfer goals that we have listed above ?

Yes and No.



Color-Mapping (ab)

- First we device a function f_1 for the simple case where the original palette contains a single color C , and the user modifies it to be color C' .
- For any color x we would like to know $x' = f'(x)$. In general we want to translate colors in the same direction, and a naive strategy might simply add exactly the same offset vector $(C' - C)$ to every x .

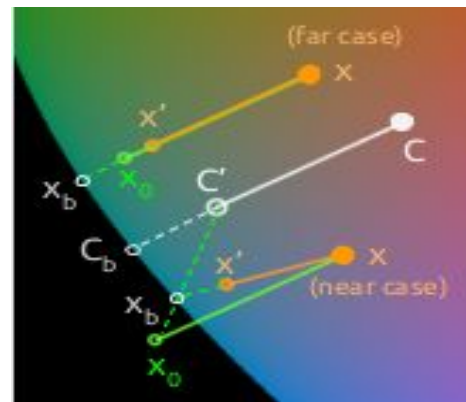
What can be the problem with this ?

How Do we Solve this ?

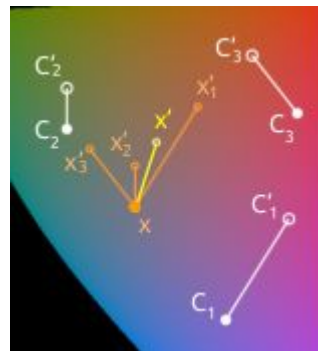
Color-Mapping (ab) (Cntd.)

- First we find C_b , the point where the ray from C towards C' intersects the gamut boundary.
- Next, we determine if $x' = x + C' - C$ is in gamut. If so (the “far” case) we find x_b the location where the parallel ray from x intersects the gamut boundary. If not (the “near” case) we take x_b to be the point where the ray from C' towards x' intersects the boundary.
- These intersections are found by binary search. Finally, we take $f_1(x) = x'$, the point on the ray from x to x_b such that:

$$\frac{\|x' - x\|}{\|C' - C\|} = \min\left(1, \frac{\|x_b - x\|}{\|C_b - C\|}\right)$$



RBF Blending



- Now that we have a transfer function $f_1(x)$, we should generalize it to handle the case of larger palettes containing $k > 1$ entries.
- Our strategy is to define k transfer functions $f_i(x)$, each equivalent to the $f_1(x)$ map described above as if it were the only palette entry, and then blend them, weighted by proximity:

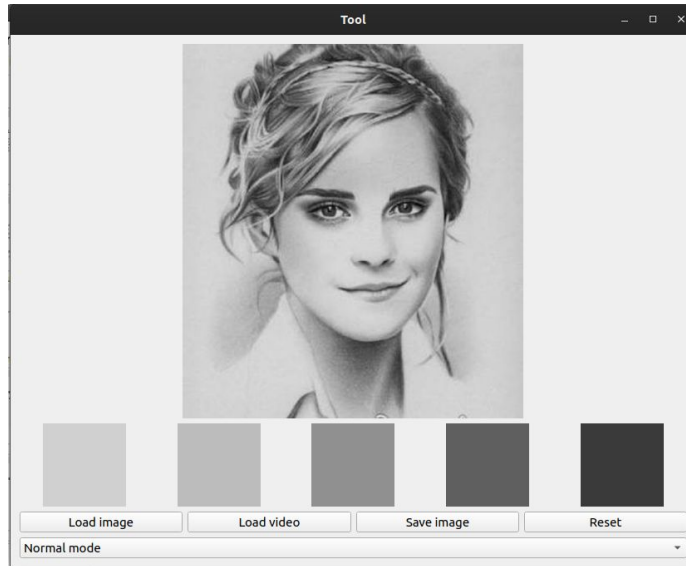
$$f(x) = \sum_i^k w_i(x) f_i(x) \quad \text{and} \quad \sum_i^k w_i(x) = 1$$

- For the weights we use radial basis functions (RBFs): $w_i(x) = \sum_j^k \lambda_{ij} \phi(\|x - C_j\|)$
- This approach leads to smooth interpolation of the individual transfer functions f_i at C_i .

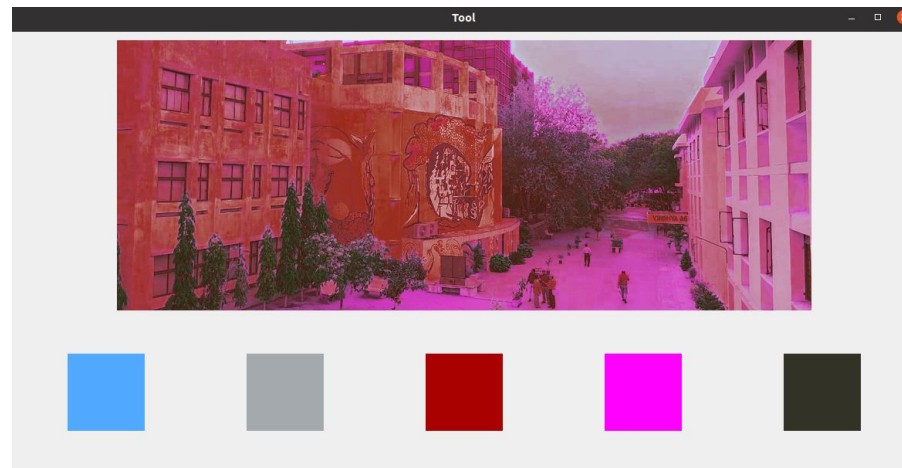
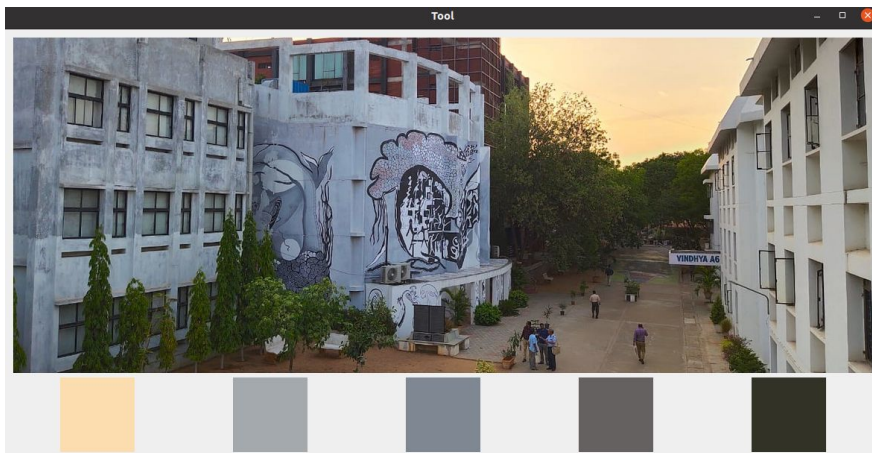
GUI

Some of the criteria important for a good color manipulation user interface are:

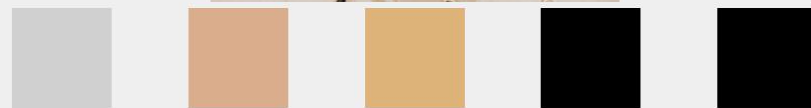
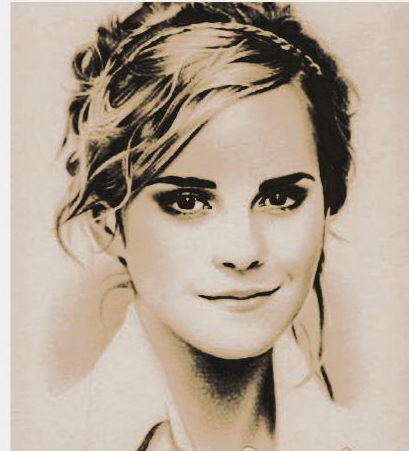
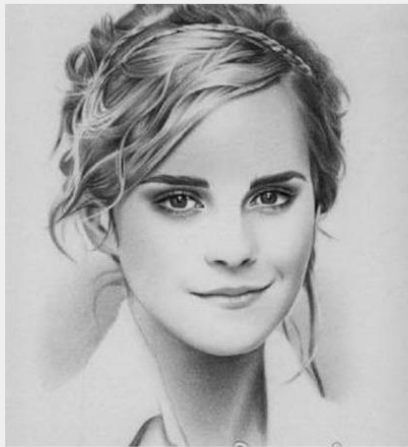
- Simple
- Expressive
- Intuitive
- Responsive



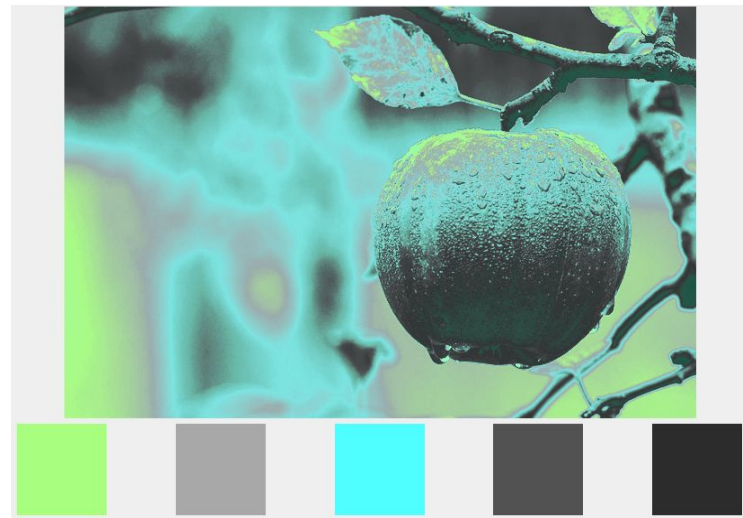
Some Outputs for Images



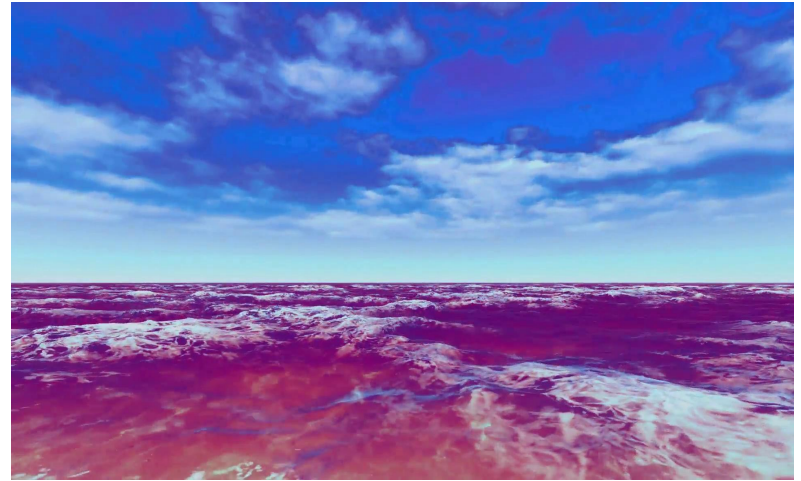
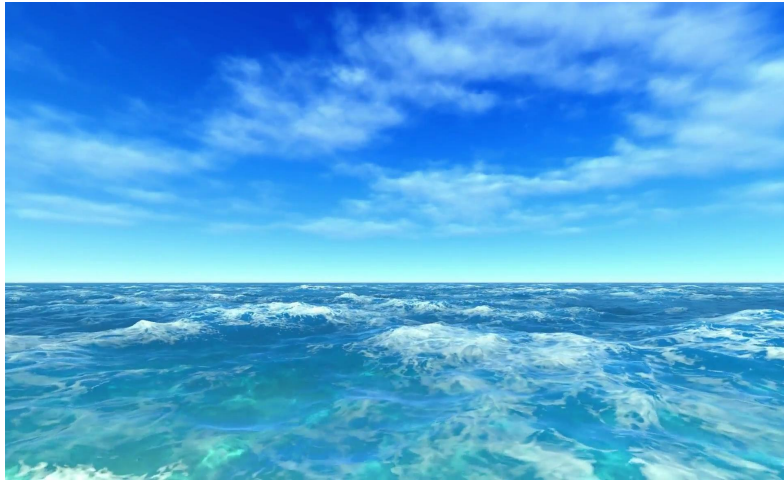
Some Outputs for Images



Some Outputs for Images



Some Outputs for Videos




Some Outputs for Videos





Individual Contribution

Team Members	Contributed Part
Anudeep and Sushanth	GUI
Pavan Kalyan and Rupa	Palette Generation
Pavan Chaitanya and Anudeep	Single Colour Transfer
Sushanth and Pavan Kalyan	Video Editing
Rupa and Pavan Chaitanya	RBF Blending



Thank You!
Wishing u good health!!!