Computer Vision Project Report

Shalin Jain (B21CS070), Shashwat Roy (B21CS071)
Tarun Raj Singh (B21CS076), Diya Fursule (B21CS075)

The code can be found here: GitHub Repository

1 Problem Statement

Image blending is a widely used technique in computer vision and image processing, enabling the seamless combination of two or more images to create visually appealing results. It is a technique used in various fields like panorama creation and special effects in movies.

This project is an exploration and analysis of methods for image blending. We aim to come up with customizable code implementations for image blending algorithms. The algorithms that are implemented will then be evaluated based on the quality of outputs they produce for various input images.

2 Methodology

2.1 Methods explored

Cut-paste Blending

Alpha Blending

Additive Weighted Blending

Multi-Band Blending

Poisson Blending

Gradient Domain Blending

3 Implementation & Results

3.1 Cut-Paste Blending

Cut-Paste Blending is a naive approach that combines two images directly. A mask is defined to crop out the region of interest from the source image. This is then directly pasted into the target image to yield the blended image.

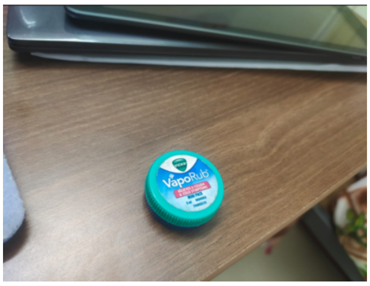

Fig. 1: Source

Fig. 2: Target

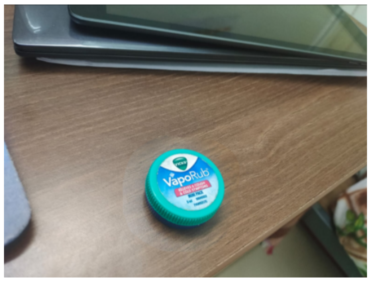

Fig. 3: Blended Image

3.2 Alpha Blending

Alpha blending combines two images or objects based on their opacity levels. At every pixel of the image, we need to combine the foreground image color (F) and the background image color (B) using the alpha mask (α). This is done using the equation described below:

$$I = \alpha F + (1 - \alpha)B$$

Fig. 4: Source

Fig. 5: Target

Fig. 6: Alpha Blend output

3.3 Additive Weighted Blending

It is an extension of the Alpha Blending algorithm. This algorithm allows manipulation of weights of individual mask pixels. It can also blend multiple images with their respective masks into one image.

Fig. 7: Source

Fig. 8: Target

3.4 Multi-Band Blending

Multi-Band Blending is an image blending technique in which involves creating the Laplacian pyramids of the images, combining them using a mask and reconstructing the blended image. This is achieved as follows :

1. Given the image and the number of levels of the pyramid, construct the Gaussian pyramids for both images.
2. Construct the corresponding Laplacian pyramids. Every level of the Laplacian pyramid comes from taking the same level of the Gaussian pyramid and subtracting the expanded version of the next level, with the exception of the last level, which is the same as the last level of the Gaussian pyramid.
3. Combine the Laplacian pyramids at each level using the alpha mask of the corresponding level from the Gaussian pyramid of the original mask.
4. Recover the blended image output from the resulting Laplacian pyramid by collapsing the levels from the bottom-up.

Fig. 9: Source

Fig. 10: Target

Fig. 11: Multi Banded output

3.5 Poisson Blending

The objective of this algorithm is to compose a source image and a target image in the gradient domain. A good blend should preserve gradients of source region without changing the background. This algorithm treats pixels as variables to be solved. It minimizes the squared difference between gradients of foreground region and gradients of target region keeping the background pixels constant according to the equation :

$$v = \underset{v}{\operatorname{argmin}} \left(\sum_{i \in S, j \in N_i \cap S} ((v_i - v_j) - (s_i - s_j))^2 + \sum_{i \in S, j \in N_i \cap S^c} ((v_i - t_j) - (s_i - s_j))^2 \right) \quad (1)$$

Given a source image s, and a target image t, we want to find new intensity values v within source region S (as defined by our masking) that minimize the difference in gradients between the new values and the old source image values. We split up this problem into 2 portions, the first summation regards all pixel values inside of S and the second summation deals with all boundary values of S. All values v inside of S should should have the same gradients as that of S.

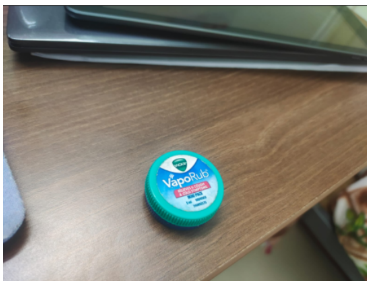

Fig. 12: Source

Fig. 13: Target

Fig. 14: Poisson Blended output

3.6 Gradient Domain Blending

Poisson blending always aims to minimize the error from the target gradient. But if the source image contains very little detail and the background is detail-rich, some areas in the source mask can cause the background to look blurry. Mixed gradient blending fixes this by taking the gradient from either the source or target image with the largest magnitude to compute each pixel-to-pixel neighbor constraint. The new function to minimize is as follows:

$$v = \underset{v}{\operatorname{argmin}} \left(\sum_{i \in S, j \in N_i \cap S} ((v_i - v_j) - d_{ij}^2 + \sum_{i \in S, j \in N_i \cap S^c} ((v_i - t_j) - d_{ij}^2 \right) \tag{2}$$

$$where, d_{ij} = \max(\operatorname{abs}(s_i - s_j), \operatorname{abs}(t_i - t_j))$$

Fig. 15: Source

Fig. 16: Target

Fig. 17: Target

4 Observations & Conclusions

In this report, we have compared the performances of various types of image blending algorithms. Each algorithm has its specific strengths and weaknesses. Alpha blending, while easy to implement, leads to noticeable brightness variations at the edges of the mask. Poisson blending tends to preserve the gradients of the intensity values of the target image after blending. Pyramid blending is computationally expensive and tends to perform not as well as Poisson due to the information loss that accompanies the downsampling of images when creating the levels of the pyramid.

5 References

CS 194-26: Image Manipulation and Computational Photography, William Tait, Berkeley. `https://inst.eecs.berkeley.edu/~cs194-26/fa17/upload/files/proj3/cs194-26-abw`

CS 194-26: LaplacianPyramids. `https://inst.eecs.berkeley.edu/~cs194-26/fa17/Lectures/LaplacianPyramids.pdf`

Gradient Domain Fusion Using Poisson Blending, hingxyu. `https://hingxyu.medium.com/gradient-domain-fusion-using-poisson-blending-8a7dc1bbaa7b`

Image Blending Using Laplacian Pyramids, Michelle Zhao. `https://becominghuman.ai/image-blending-using-laplacian-pyramids-2f8e9982077f`