Ramakrishna Mission Vivekananda Educational and Research Institute

Computer Vision

Project Report

Image Filtering and Hybrid Images

Submitted By

Sayan Das
B2430035

Raihan Uddin
B2430070

February 10, 2025

CONTENTS

I Introduction **3**
I.1 Motivation 3

II Methodology **5**
II.1 Image Filtering 5
II.2 Hybrid Image Generation 6

III Implementation **7**
III.1 Tools and Libraries 7
III.2 Implementation Details 7
III.2.1 `my_imfilter` function 7
III.2.2 `create_hybrid_image` function 8
III.2.3 Creating the filter 9

IV Results **10**
IV.0.1 Cat-Dog Hybrid Image 10

V Task Split **11**

VI Conclusion **12**

Acknowledgement **13**

References **14**

SECTION I

INTRODUCTION

Hybrid images, as introduced by Oliva, Torralba, and Schyns in their SIGGRAPH 2006 paper [1], are static images that change interpretation based on viewing distance. This phenomenon leverages the human visual system's multi-scale processing, where high-frequency details dominate perception at close range, while low-frequency components become prominent from afar. The goal of this project is to implement image filtering and hybrid image generation, aligning with the methodology described in the paper.

I.1 Motivation

The primary challenge in creating compelling hybrid images lies in the precise separation and recombination of these frequency components. This requires:

1. **Accurate Filtering:** Implementing an image filtering algorithm that can effectively extract low and high-frequency components without introducing artifacts.
2. **Perceptual Alignment:** Ensuring that the two images are aligned in a way that their frequency components blend seamlessly, avoiding perceptual conflicts.
3. **Parameter Tuning:** Selecting appropriate cutoff frequencies for the filters to achieve the desired perceptual effect.

This project aims to address these challenges by implementing a custom image filtering function and using it to generate hybrid images. The goal is to replicate the results described in the paper [1] and explore the perceptual effects of hybrid images. By doing so, we aim to gain a deeper understanding of multi-scale image processing and its applications in computer vision.

The problem can be summarized as follows:

1. **Input:** Two aligned images - `Image 1` and `Image 2` (e.g., `dog.bmp` and `cat.bmp`).
2. **Output:** A hybrid image that changes interpretation based on viewing distance, along with intermediate results (low-frequency and high-frequency components).

SECTION II

METHODOLOGY

II.1 Image Filtering

The core of hybrid image creation lies in filtering operations. The custom function `my_imfilter` implements 2D convolution with the following steps:

1. **Padding:** he input image is padded using zero-padding to preserve spatial dimensions post-convolution. The padding size is determined by the filter dimensions (k, l) with

$$\begin{aligned} pad_k &= \left\lfloor \frac{k}{2} \right\rfloor \\ pad_l &= \left\lfloor \frac{l}{2} \right\rfloor \end{aligned} \tag{II.1}$$

2. **Convolution:** For each color channel (RGB), a sliding window extracts image patches, which are element-wise multiplied with the filter and summed to compute the output pixel. Mathematically,

$$O(x, y, c) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \sum_{d=0}^{D-1} I(x+i, y+j, d) \cdot K(i, j, d, c)$$

where:

- $O(x, y, c)$ is the output pixel value at position (x, y) for channel c.
- $I(x + i, y + j, d)$ is the input image pixel value at position $(x + i, y + j)$ for channel d.
- $K(i, j, d, c)$ is the convolution kernel (filter) value at position (i, j), mapping from input channel d to output channel c.
- $m \times n$ is the size of the filter (height and width).
- D is the number of input channels (for RGB, $D = 3$).

II.2 Hybrid Image Generation

The function `create_hybrid_image` combines two images:

1. **Low-Frequency Component:** Obtained by applying a Gaussian low-pass filter to `Image 1`.

$$I_1 \cdot G_1$$

2. **High-Frequency Component:** Derived by subtracting the low-pass filtered version of `Image 2` from itself.

$$I_2 \cdot (1 - G_2)$$

3. **Hybrid Image:** The sum of low and high frequencies, clipped to [0, 1] to maintain valid pixel intensities.

$$H = I_1 \cdot G_1 + I_2 \cdot (1 - G_2)$$

SECTION III

IMPLEMENTATION

III.1 Tools and Libraries

The tools and libraries used for this project are:

1. **Python:** The primary programming language for this project.
2. **Jupyter Notebook:** For interactive code development.
3. **Libraries:**
    (a) **Numpy:** For numerical operations like array sums, and for faster mathematical computations.
    (b) **Matplotlib:** For displaying images and visualizing filters.
    (c) **CV2:** For image processing tasks such as reading, writing, and manipulating images.

III.2 Implementation Details

III.2.1 `my_imfilter` function

Before constructing a hybrid image, we need a function to apply a filter to an image. The function `my_imfilter(image, filter)` does this by:

1. **Padding the Image:** Since filtering requires accessing pixels around each center pixel, we pad the input image using `np.pad`. The padding size is determined following the process mentioned in methodology 1.
   Unlike how it was mentioned in the documentation, we used $k \times l$ filter, to support rectangular kernels.

2. **Applying the Filter:** For each pixel in the image, a window of size $(k \times l)$ is extracted.This window is element-wise multiplied with the filter and summed to get the new pixel value. This operation is repeated across all three color channels (R, G, B). 2. We used `np.sum` for faster computation.

The function returns the filtered image, where each pixel is computed using the convolution operation.

III.2.2 `create_hybrid_image` function

The **hybrid image** is constructed by combining the low-frequency content of one image and the high-frequency content of another.

1. Step 1: Extract Low-Frequency Content We obtain the **low-frequency** content of `image1` by filtering it with a low-pass filter:

$$\text{low\_frequencies} = \text{my\_imfilter}(\text{image1}, \text{filter})$$

   A low-pass filter (e.g., a Gaussian blur) removes high-frequency details, leaving only smooth variations.

2. Step 2: Extract High-Frequency Content The high-frequency content of `image2` is obtained by subtracting its low-frequency component:

$$\text{high\_frequencies} = \text{image2} - \text{my\_imfilter}(\text{image2}, \text{filter})$$

   This step removes smooth regions and enhances sharp edges and fine details.

3. Step 3: Combine Both Components Finally, the **hybrid image** is formed by adding the low-frequency and high-frequency components:

$$\text{hybrid\_image} = \text{low\_frequencies} + \text{high\_frequencies}$$

   Since pixel values must be between 0 and 1 (for correct image representation), we apply **clipping**:

$$\text{hybrid\_image} = \text{np.clip}(\text{hybrid\_image}, 0, 1)$$

The hybrid image contains **smooth variations** from `image1` and **sharp details** from `image2`. When viewed **up close**, the high-frequency details dominate (showing `image2`). When viewed **from a distance**, the low-frequency content dominates (showing `image1`).

III.2.3 Creating the filter

We used **`cv2.getGaussianKernel(ksize, sigma)`** to generate a 1D Gaussian kernel (column vector). We muliplied it with its transpose (filter @ filter.T) to convert it into a 2D Gaussian kernel. This method leverages separability, making Gaussian blurring more efficient than computing a full 2D kernel directly.

SECTION IV

RESULTS

IV.0.1 Cat-Dog Hybrid Image

SECTION V

TASK SPLIT

Raihan Uddin was in charge of most of the implementation (coding) part.
Sayan Das was in charge of making this report in LaTeX. He also contributed in identifying and fixing a bug in my_imfilter function where it was not working as intended when applying a rectangular filter.
We Both worked on reading the paper [1] and helped each other understand it.

SECTION VI
CONCLUSION

This project successfully implemented hybrid images using principles from Oliva et al.'s [1] work. The results demonstrate the interplay of spatial frequencies in human perception, validating the paper's claims. Future work could explore separable filters for speed.

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our supervisor, **Br. Bhaswarachaitanya** (Tamal Maharaj), Assistant Professor in the Department of Computer Science at *Ramakrishna Mission Vivekananda Educational and Research Institute (RKMVERI)*, for his invaluable guidance, encouragement, and support throughout the course of this project. His expertise in computer vision and his insightful suggestions were instrumental in shaping this project.

We would also like to thank *Ramakrishna Mission Vivekananda Educational and Research Institute* for providing the necessary resources and facilities that enabled the successful completion of this project.

Lastly, we extend our heartfelt thanks to our batch mates for their constant encouragement and understanding during the challenging phases of this project.

REFERENCES

[1] Oliva, A., Torralba, A., & Schyns, P. G. "Hybrid Images." *ACM Transactions on Graphics (TOG)*, 2006.

[2] Szeliski, R. "Computer Vision: Algorithms and Applications." *Springer*, 2010.