Image Processing - Exercise 3

Areen Mansour, areen0507, 212900211

(1) Introduction

This code focuses on the creation of visually compelling images through two distinct techniques: blending with mask and hybrid image generation. The primary goal is to seamlessly merge two images using advanced methods, specifically Laplacian and Gaussian pyramids. The blend function combines images based on a specified mask, emphasizing specific regions from each source. Meanwhile, the hybrid function constructs a hybrid image by intelligently blending the Laplacian pyramids of two input images.

(2) Algorithm

i. Description:

Blending with Mask:

1. **Gaussian Pyramid Construction:**
   - Create Gaussian pyramids for both input images and the mask using the specified number of levels.
   - Apply Gaussian blur and downsampling for each level.
   - Calculate Laplacian pyramids for both images.

2. **Blending:**
   - Combine Laplacian pyramids based on the mask's pixel values, emphasizing contributions from each image.
   - Upsample and accumulate the blended pyramid to reconstruct the final blended image.

Hybrid Image:

1. **Gaussian Pyramid Construction:**
   - Generate Gaussian pyramids for both input images using the specified number of levels.
   - Apply Gaussian blur and downsampling for each level.

2. **Hybrid Pyramid Creation:**
   - Combine the low-frequency components of the first image's Laplacian pyramid with the high-frequency components of the second image's Laplacian pyramid.
   - Reverse the combined pyramid to obtain the hybrid image.

ii. Implementation Details:

Blending with Mask:
- Utilizes OpenCV for image processing and Matplotlib for visualization.
- Key hyperparameter: Number of pyramid levels.
- Challenges addressed include maintaining image dimension consistency during blending.

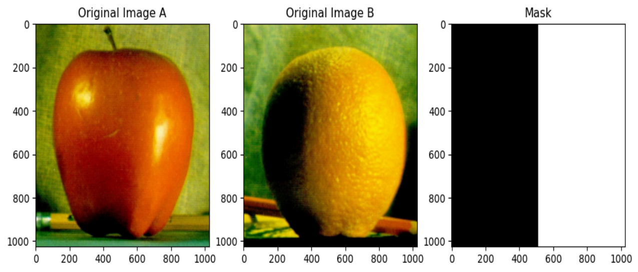
Hybrid Image:
- Employed OpenCV and Matplotlib for image manipulation and visualization.
- Key hyperparameter: Number of pyramid levels.
- Challenges involved managing the integration of low and high-frequency components.

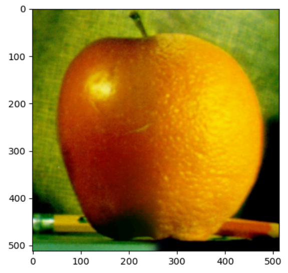
Difference Between Algorithms:

The first algorithm focuses on blending two images seamlessly using a mask, allowing controlled integration of image details. In contrast, the second algorithm creates a hybrid image by combining the low-frequency details from one image with the high-frequency details from another. While both algorithms leverage Gaussian and Laplacian pyramids, their objectives differ – one emphasizing controlled blending, and the other producing a visually intriguing hybrid composition.

(3) Results

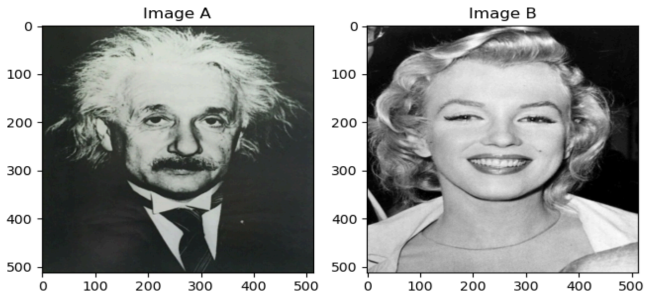
(a)

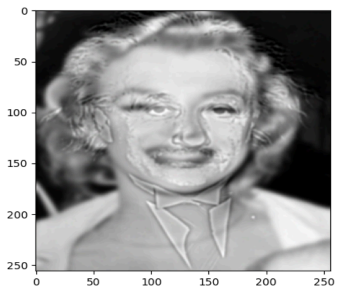




The original images A, B and Mask

Blending images



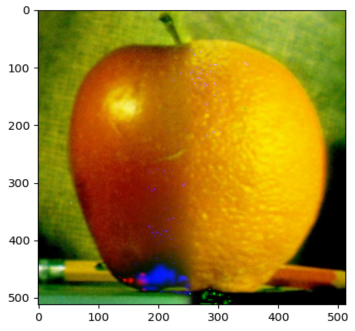


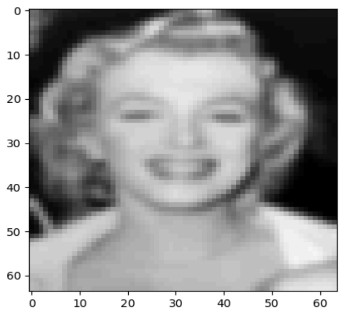
The original images A, B

Hybrid Image

(b)

unsuccessful blendings





(1)

(2)

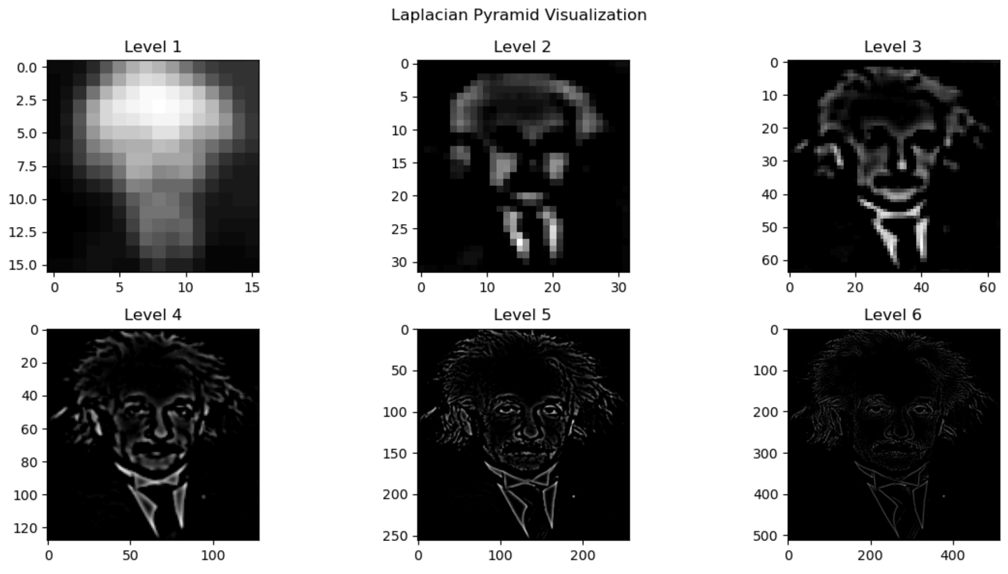
(1) The code exhibits two key issues: Firstly, during the construction of the Gaussian pyramid, an unnecessary blur is introduced using `cv2.GaussianBlur`, deviating from the typical purpose of downsampling without additional smoothing. This deviation may yield unexpected results in the pyramid construction. Secondly, in the blending process, there is a data type mismatch between the arrays `laplacian_pyr1` and `laplacian_pyr2`, resulting in a TypeError. To ensure seamless blending, it is essential to maintain consistent data types for these arrays, avoiding arithmetic errors during the blending operation.

(2) In the context of the provided code, the number of levels plays a crucial role in the construction and blending of Laplacian pyramids. Laplacian pyramids are used for image processing tasks such as blending and representation of images at multiple scales. The bug in the original code occurred in the `hybrid` function, where the levels were incorrectly chosen for blending the Laplacian pyramids.

(4) Pyramids

a - **Gaussian** pyramid visualization - for hybrid images

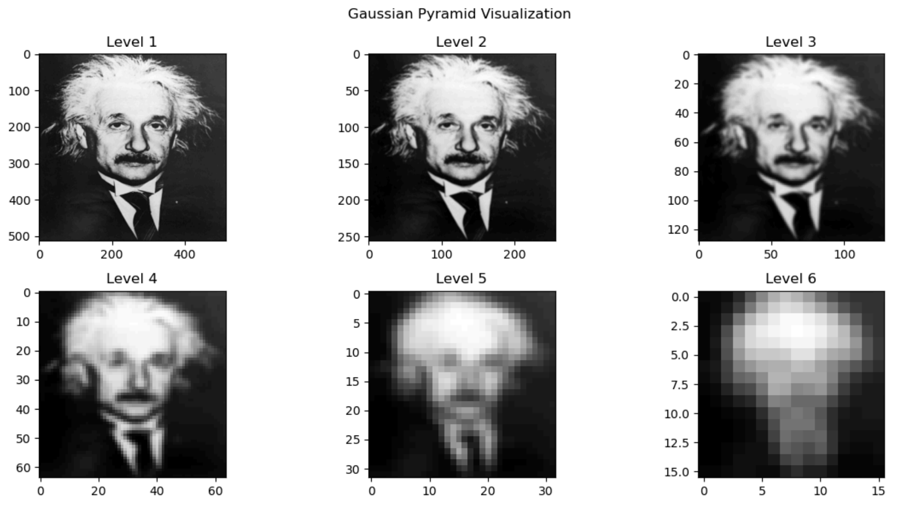
1. **Detail Levels:** Each pyramid level captures details at different scales, with lower levels emphasizing fine details and edges.
2. **Smoothing Effect:** Higher levels exhibit more blurring, representing larger structures and low-frequency features.
3. **Edge Emphasis:** Lower levels highlight areas with rapid intensity changes, focusing on edge information.
4. **Multi-Scale Representation:** The pyramid provides a multi-scale view, aiding in tasks like image blending, compression, and feature extraction.
5. **Reconstruction Role:** Used in image blending, the pyramid levels contribute to seamless reconstruction for a visually appealing result.



b - **Laplacian** pyramid visualization - for hybrid images.

- **Level of Detail:** Each level emphasizes different details, with lower levels focusing on finer details and higher levels on broader structures.
- **Edge Information:** Lower levels highlight edges and rapid intensity changes in the image.
- **Smoothing Effect:** Moving up the pyramid results in more smoothed and blurred images, representing lower-frequency components.
- **Multi-Scale Decomposition:** The Laplacian pyramid offers a multi-scale decomposition, beneficial for tasks like blending.

In summary, the Laplacian pyramid visually conveys the contribution of different detail scales to the overall image structure, playing a vital role in processes like blending



(5) Conclusion

In conclusion, the provided code implements key functions for image blending using Gaussian and Laplacian pyramids. The 'build_gaussian_pyramid' function is responsible for generating these pyramids, employing OpenCV functions for blurring and down/up-sampling. The 'blend_images_with_mask' function blends two input images using Laplacian pyramids and a specified mask, while the 'blend' function independently blends each color channel and stacks them to form the final image. The 'hypred' function adopts a unique approach, combining the top and bottom halves of Laplacian pyramids from two input images. An example usage showcases the 'hypred' function, and the code includes commented-out sections for potential visualization adjustments.