Assignment 2. report

Computer vision - Fall 2024
Guillaume Drui, Tymofii Voitekh, Jakub Zika

Introduction

The assignment focused on three key tasks related to frequency domain image processing and multi-scale representations. These tasks build upon fundamental techniques in image analysis, such as Fourier transforms, Gaussian smoothing, and image pyramids, as well as advanced methods for aligning and reconstructing images from their color channels.

The tasks demonstrate practical applications of these techniques, ranging from the creation of hybrid images to building Gaussian pyramids and reconstructing colorized images. Each section of this report provides a detailed description of the methods used, results obtained, and the challenges encountered during implementation.

Folder structure

The implementation is located in *solution_1.ipynb*, the other 2 ipynb files can be ignored.

output folder contains following folders

- *1st-part* - results of the overlayed images from the original data folder
- *1st-part-custom-output.png* - results of the overlayed images on our own data in the *my_data* folder

- *3rd-part* - results for the 3rd part using the image pyramids from the original data folder
- *3rd-part-custom_output.jpg* - result for the 3rd part using the image pyramids from our own data in the *my_data* folder

- *3rd-part-phase-correlation* - result for the 3rd part using the alternative phase correlation approach from the original data folder

my_data folder contains our custom data

Task 1. - Hybrid image

In this task, we aimed to create a hybrid image by blending two images using frequency domain manipulation. The approach leverages Gaussian kernels and Fourier transforms to control which parts of each image are combined based on their frequency content. Below, we outline the methodology and key steps performed in this process.

Both input images (A and B) were converted into the frequency domain using the Fast Fourier Transform (FFT). This allowed us to manipulate the frequency components of each image individually.

To center the low frequencies in the Fourier-transformed images, we applied an fftshift operation. This step is essential as it shifts the zero-frequency component (DC component) to the center of the spectrum. This step helped simplify further processing by aligning the frequency components spatially around the center.

Once the images were transformed and shifted, we generated and applied the Gaussian kernel to image A to isolate its low-frequency components and to image B for its high-frequency content. The filtering was done as follows:

$$A_{\text{low}} = \mathcal{F}_s(A) \cdot K_1$$

$$B_{\text{high}} = \mathcal{F}_s(B) \cdot K_2$$

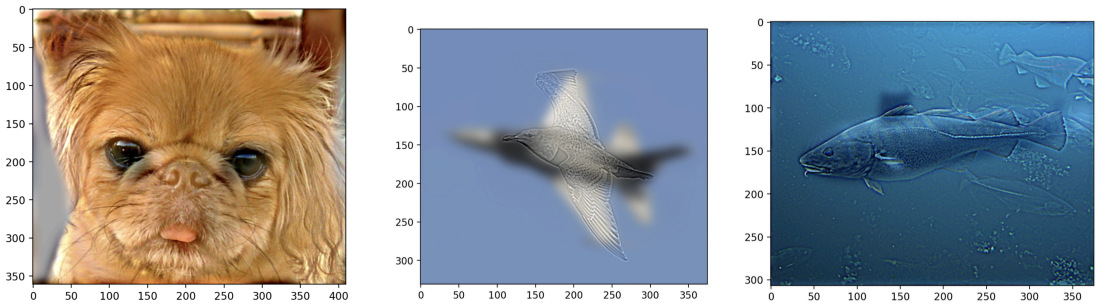
Where Fs represents the shifted Fourier transform and K1 and K2 are the Gaussian kernels applied to images A and B, which were manually tuned for optimal results with different σ values.

After applying the filters, we recombined the low-frequency content from image A and the high-frequency content from image B in the frequency domain. Then, we shifted and applied the inverse Fourier transform to return to the spatial domain:

$$\text{ifftshift}(\mathcal{F}_s^{-1}(A_{\text{low}} + B_{\text{high}})) = \text{Hybrid Image}$$

This yielded the final hybrid image, which visually combined the characteristics of both images.

results:



The success of the hybrid image heavily relied on manually tuning the σ of the Gaussian kernel. Different values of σ affected how much of the low and high frequencies were retained, requiring experimentation to balance the visual outcome.

Task 2. - Image pyramid

The implementation focuses on creating a Gaussian pyramid representation of images using frequency domain operations. The key components of the implementation include:

- Gaussian Kernel Generation
    - A custom Gaussian kernel is generated using normalized distance from the center
    - The kernel is parameterized by σ (sigma) to control the amount of smoothing
    - Implementation uses a normalized coordinate system (-1 to 1) to ensure consistent behavior across different image sizes

- Pyramid Construction Process
    - Each level of the pyramid is created through the following steps: a. Apply Gaussian smoothing in the frequency domain b. Downsample the smoothed image by taking every second pixel c. Store the scale factors between consecutive levels
    - The process continues until the desired number of pyramid levels is reached

- Frequency Domain Operations
    - Images are transformed to frequency domain using FFT
    - Convolution with Gaussian kernel is performed as multiplication in frequency domain
    - Inverse FFT is used to return to spatial domain
    - This approach is more efficient than spatial domain convolution for larger kernels

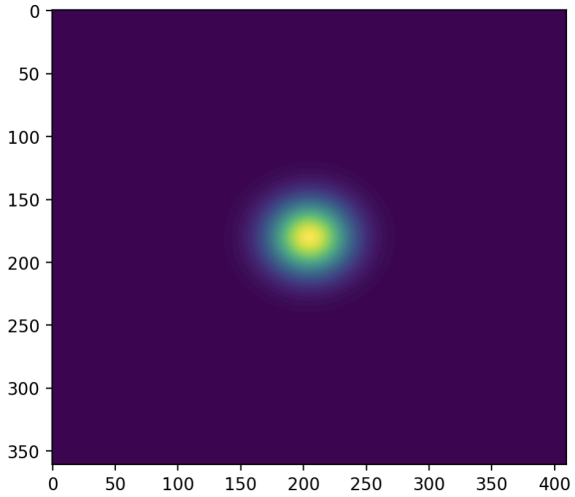


Figure 1: Gaussian Kernel

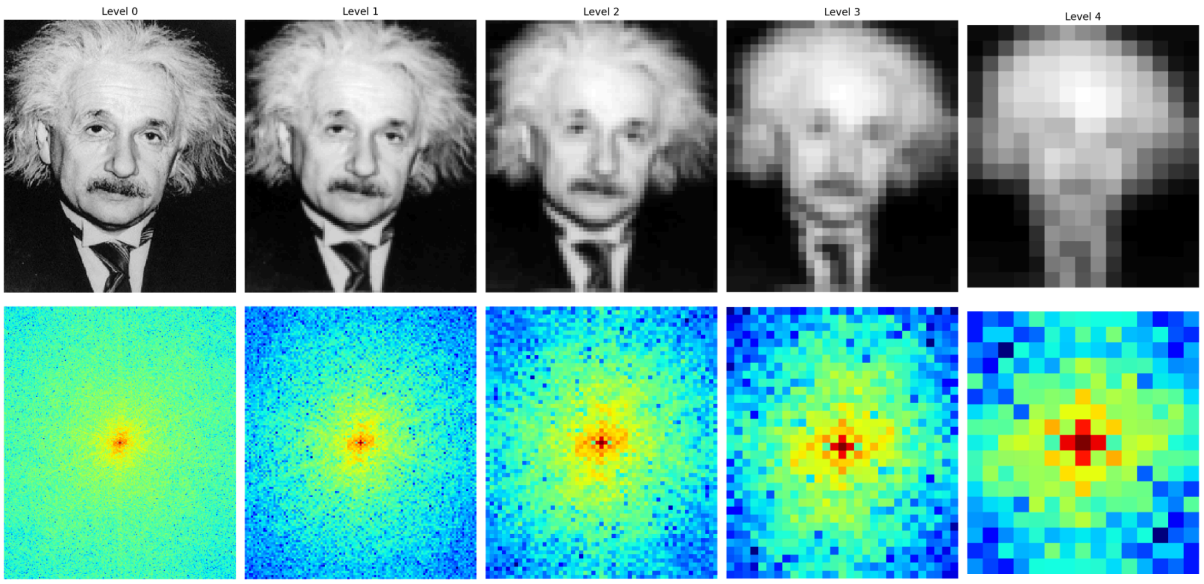


Figure 2: Pyramid Levels

- Top row: Spatial domain representation of each pyramid level
- Bottom row: Corresponding frequency domain representation using jet colormap

Key observations:

1. Each level shows approximately half the spatial frequencies of the previous level
2. The Fourier transform visualizations clearly show the band-limited nature of each level
3. The Gaussian smoothing prevents aliasing artifacts during downsampling

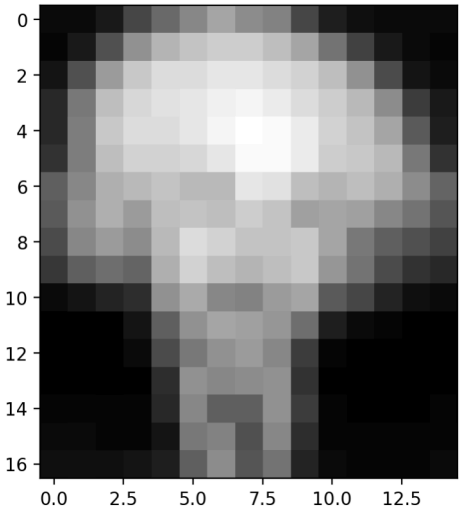


Figure 3: Final Level Comparison

In our implementation, Several important technical aspects were addressed:

- Proper normalization of images to maintain consistent intensity ranges
- Careful handling of boundary conditions during frequency domain operations
- Scale tracking between pyramid levels for potential reconstruction
- Memory efficiency through progressive downsampling

The implementation successfully generates multi-scale representations that:

- Preserve important image structures at different scales
- Show smooth transitions between pyramid levels
- Maintain proper frequency content at each level
- Enable efficient multi-scale analysis

Task 3. - Colorizing the Russian Empire

The implementation focuses on image matching techniques to align and reconstruct colored images from their separated RGB channels. The process begins by decomposing the input image into its constituent Red, Green, and Blue channels, followed by generating Gaussian pyramids for each channel to create a hierarchical representation at different scales. The matching process utilizes an L1 norm distance metric for comparing channel similarities, as initial experiments with L0 norm proved less effective. Several optimization techniques were implemented, including a limited search space around previously determined offsets at each pyramid level and dynamic adjustment of the search radius as the algorithm progresses through pyramid levels. Image cropping was also applied to eliminate false positives caused by edge artifacts. The search space is systematically explored to find the offset that minimizes the distance metric between channels, with each subsequent pyramid level refining the previous estimation. This approach successfully determines the optimal offset values for both the Green and Blue channels relative to the Red channel, enabling accurate reconstruction of the original colored image.

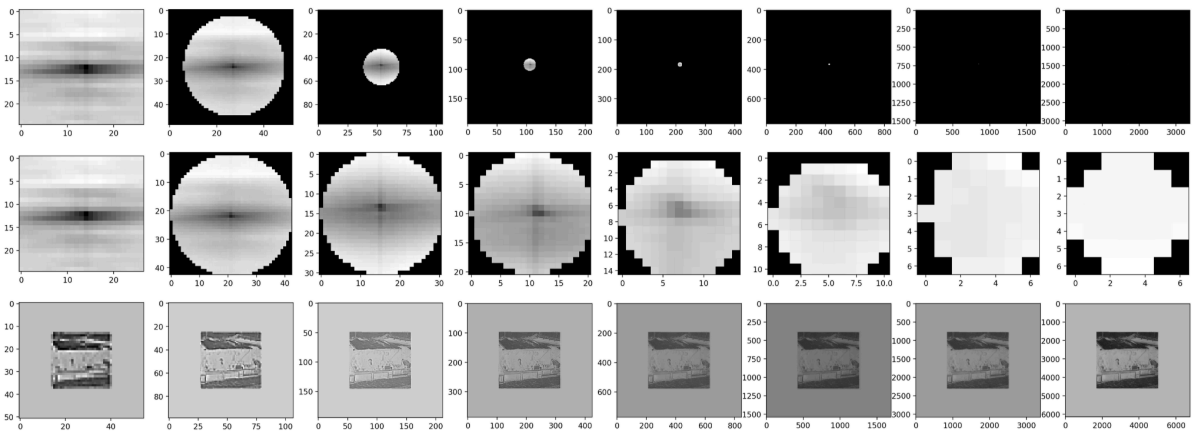


Image shows how the algorithm starts with a lower resolution image and continues to refine its search for offset (from left to right) while also narrowing the search radius to avoid costly computation.

First row is the resulting image distance between two selected channels for every offset. It can be seen that in the first column it does the search for all offsets. But at further levels of the pyramid it starts to limit the offsets of which it computes the image similarity. Black color around the circle of search means that those offsets have been excluded from search.

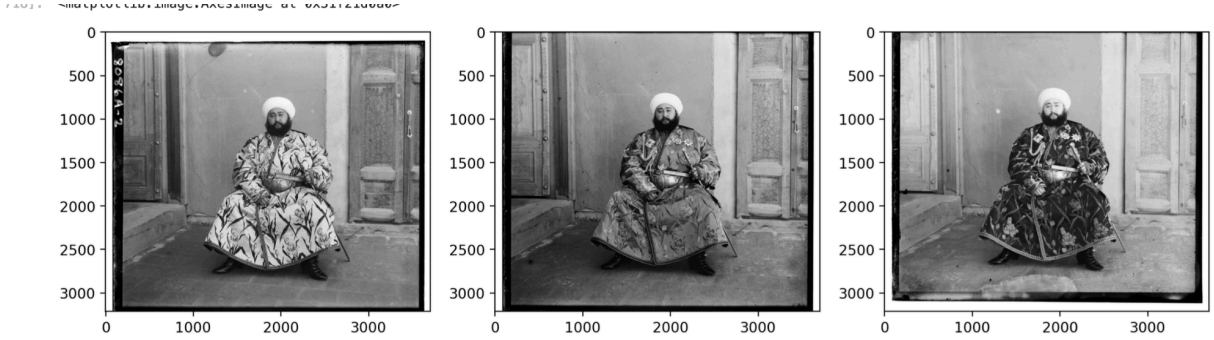
The second row shows just the circle of search.

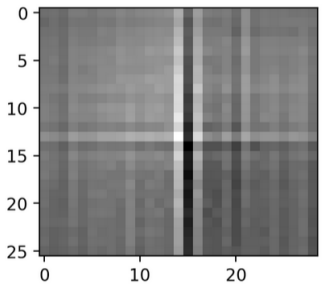
Third row displays what the A image looks like at that particular level

Better matching with Laplacian-Gaussian kernel

To enhance the matching process, a Gaussian-Laplacian filter is applied to the image channels. This filtering operation emphasizes edge features in the images, resulting in more robust and accurate matching between the color channels. The filter is effective at identifying corresponding structural elements across different color components.

Removing borders from images

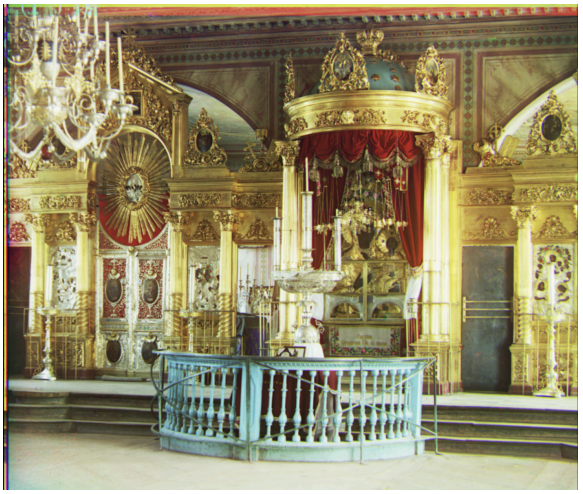




The images were at first cut in a naive way by just cutting the original image into three. The resulting images thus contained the black borders. This confused the algorithm because it made most sense to align the image by the borders instead of by the contents of the image. This confusion can be noticed in the attached second image where a black vertical bar can be seen. Thus in the image cutting phase the borders are thrown out.

Matching in spatial domain (main approach) results:



Phase correlation matching in frequency domain results (alternative approach):


Ав 132

Discussion

The implemented algorithm works well but as has been mentioned has issues with image borders which first need to be cut to achieve good image match.This process could be automated by finding the offset on a smaller subset of the image and then applying the offset to the full scale image

Although the matching algorithm runs reasonably fast the construction of the pyramid takes more time. But the pyramid can be later used for other purposes.

Team member contribution

Guillaume Drui:

- Solution ideation
- Code review
- Report writing

Jakub Zíka:

- Code writing
- Code review
- Report writing

Tymofii Voitekh:

- Solution ideation
- Code review
- Report writing