Assignment 3

Paolo Deidda, Raffaele Perri

March 11, 2025

Problem 1 [5 points]

We aim to verify that applying histogram equalization twice to a grayscale image A yields the same result as applying it once. Mathematically, let T be the transformation defined by:

$$\tilde{s}_k = L \sum_{j=0}^{k-1} p_j + \frac{L}{2} p_k - \frac{1}{2}, \quad T(k) = \mathrm{round}(\tilde{s}_k)$$

where $L = 256$, p_j is the probability of intensity j. Then, $B = T(A)$ and $C = T(B)$, and we test if $C = B$.

Figure 1: Original (A), B, C

These are the images 1: A (before), B (after first equalization), and C (after second equalization).

We can visually see that B and C images look the same but let's compute the mean average to check it better. In the code I implemented the following check:

```
difference = np.abs(np.mean(B) - np.mean(C))
mean = np.mean(difference)
print("Average difference:", mean)
```

We get that the average difference is equal 0, confirming that B = C.

```
Sum of absolute differences between B and C: 0
B and C are identical.
Average difference: 0.0
```

Problem 2 [5 points]

We aim at modifying the histogram matching algorithm to incorporate a pre-processing step.
Instead of directly using the pixel intensity k at position (i, j), we compute an average intensity $\bar{k}$ based on its neighbors.
The neighbors are selected such that their intensity differs by at most two levels $|k - k'| \leq 2$.
This modification helps to smooth local variations while preserving the original image structure.

Pre-processing step.

For each pixel in the source image:

- Identify neighboring pixels within an 8-connected neighborhood.
- Include only those neighbors whose intensity differs by at most 2 levels.
- Compute the average intensity of the valid neighbors.

This produces a new intensity value $\bar{k}$ for each pixel, which is used in histogram matching instead of the original intensity k.

Histogram matching.

Histogram matching is performed as follows:

- Compute normalized histograms and cumulative distribution functions (CDFs) for both the source (pre-processed) and target images.
- Compute the transformation function T_p by matching the CDFs of the source and target images.
- Apply the transformation to each pixel in the source image to generate the final result.

Results.

By Figure 2, we can see that before the matching, the dog image has a distinct intensity distribution compared to the forest image.

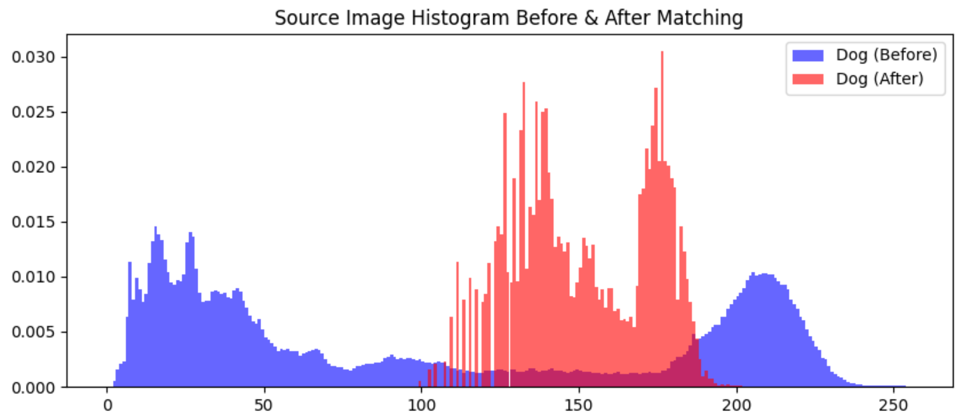

Figure 2: Dog histogram before and after the matching.

By looking instead at the histograms after the matching (Figure 2 red histogram), the histogram of the processed dog image aligns more closely with the forest image (Figure 3, representing histogram for the forest image), indicating successful transformation.

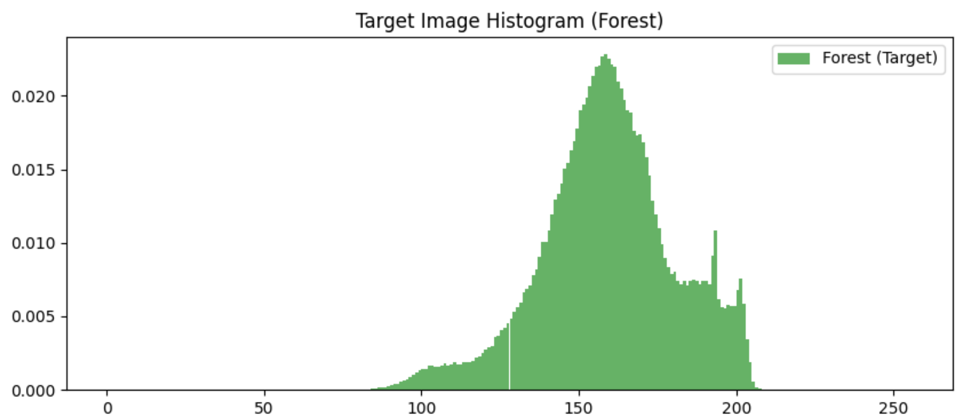

Figure 3: Forest histogram (target).

Figure 4 plots of the initial dog image before and after the matching.
We can easy observe how the pre-processing step smooths intensity variations, leading to a more gradual transition in the final matched image.

Figure 4: Dog image before and after the matching.

Finally, Figure 5 shows the histogram and the discrete cumulative distribution function.

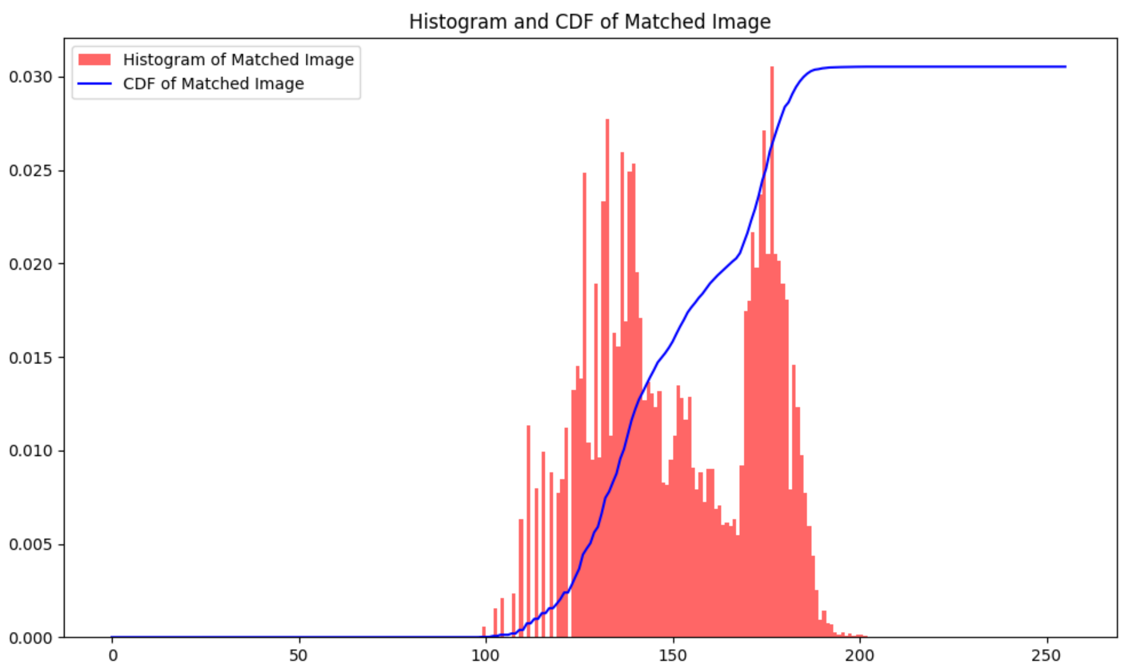

Figure 5: histogram and the discrete cumulative distribution function

To conclude, this modified histogram matching method effectively adjusts the source image's intensity distribution while preserving local intensity consistency through neighborhood averaging.
The results demonstrate improved visual consistency between the source and target images, clearly visible from Figure 6 that shows the difference between the original function and the improved ones.

Figure 6: Matched image in old histograms matching version vs matched image in new histograms matching version