Academic Year 2024/2025

# Télécom Paris

IMA201

## Project Report

# Increasing Depth of Field by Image Fusion


Elaborated by
**Lina BELLAHMIDI**
**Atef BOUZID**

Supervised by
**Saïd LADJAL**

**Abstract**

The paper we worked on is about image fusion with guided filtering, the goal of the study was to come up with an optimized way of fusing images through taking spatial consistency into consideration. To do that, instead of using approaches like generalized random walks or Markov random fields, the paper suggests using guided filtering to construct weight maps for the fusion that take into consideration pixel saliency and spatial context. The key contribution of the paper was to introduce a fast and effective two-scale image fusion method which doesn't heavily rely on a specific image decomposition method.

Our goal was to implement the fusion algorithm as well as the guided filtering for both gray and colored images. To do that, we first implemented the fusion algorithm and guided filtering for gray images, then moved on to adapting it to colored images through performing the guided filtering on each channel separately then combining the result. For the colored images we tried two color systems, the RGB and the LAB. Finally, we compared our implementation of image fusion algorithm for gray, rgb and lab with the openCV one through metrics taking into consideration mutual information between the fused and source images, structural similarity, quality, as well as a gradient based index to evaluate the success of edge information transfer to fused images.

# Contents

# List of Figures

# Introduction

The focus of the project is to use guided filtering to construct the weight maps used for the image fusion instead of using optimization based methods like markov random fields or generalized random walks for example that require multiple iterations and tend to over-smooth the resulting weights. The advantage of guided filtering is that it allows us to take into consideration spatial consistency and pixel saliency through the filter parameters when fusing images which then makes the algorithm robust to noise and artifacts.

The remainder of the report is organized as follows, in section I the guided filtering algorithm is introduced, section II covers the image fusion algorithm and its application to different type of images, and in section III we conclude with evaluations and discussion of the experimental results.

# 1 Guided Filter

## 1.1 Theory

The guided filter assumes a local linear model between guidance $I$ and filter output $q$. We assume $q$ is a linear transform of $I$ within window $w_k$ around pixel $k$:

$$q_i = a_k I_i + b_k, \quad \forall i \in w_k \tag{2}$$

where $(a_k, b_k)$ are constants in $w_k$. Using a square window of radius $r$, this model ensures $q$ has edges only if $I$ does, since $\Delta q = a\Delta I$.

To find $(a_k, b_k)$, we minimize the difference between $q$ and input $p$ in $w_k$:

$$E(a_k, b_k) = \sum_{i \in w_k} \left( (a_k I_i + b_k - p_i)^2 + \epsilon a_k^2 \right) \tag{3}$$

where $\epsilon$ prevents $a_k$ from being too large. The solution by linear regression is:

$$a_k = \frac{1}{|w|} \sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k \frac{1}{\sigma_k^2 + \epsilon} \tag{4}$$

$$b_k = \bar{p}_k - a_k \mu_k \tag{5}$$

where $\mu_k$, $\sigma_k^2$ are the mean and variance of $I$ in $w_k$, $|w|$ is the number of pixels, and $\bar{p}_k$ is the mean of $p$ in $w_k$.

To apply to the entire image, we average all $q_i$ values across windows:

$$q_i = \frac{1}{|w|} \sum_{k: i \in w_k} (a_k I_i + b_k) = \bar{a}_i I_i + \bar{b}_i \tag{6}$$

where $\bar{a}_i$ and $\bar{b}_i$ are the averaged coefficients.

For RGB guidance images, the model becomes:

$$q_i = a_k^T I_i + b_k, \quad \forall i \in w_k \tag{7}$$

with color vectors $I_i$, $a_k$, and solution:

$$a_k = (\Sigma_k + \epsilon U)^{-1} \left( \frac{1}{|w|} \sum_{i \in w_k} I_i p_i - \mu_k \bar{p}_k \right) \tag{8}$$

$$b_k = \bar{p}_k - a_k^T \mu_k \tag{9}$$

$$q_i = \bar{a}_i^T I_i + \bar{b}_i \tag{10}$$

where $\Sigma_k$ is the covariance matrix of $I$ in $w_k$.

## 1.2 Results and Discussion

As a first step, we tried to use the predefined guided filter in the OpenCV library and then try to implement the guided filter ourselves based on the steps proposed by the paper and using the formula shown in the previous subsection.

We applied the guided filter on a grayscale image and on an RGB color image. In both cases, we used the guidance image the same as the input image.


Figure 1: Guided gray image 'Lena' with itself with different parameters.

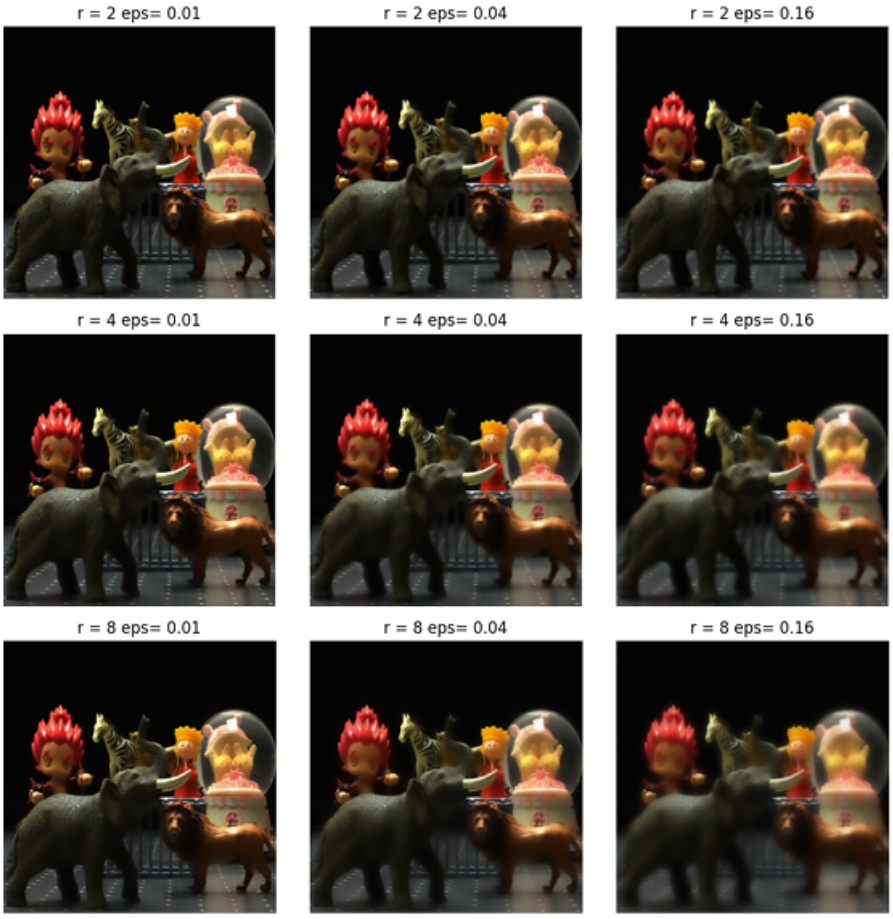


Figure 2: Guided color image with itself with different parameters.

As anticipated and through the illustrations over, able to conclude that the advantage of utilizing the same image as an input and guidance is edge-preserving smoothing by exploiting information from the guidance image to control the filtering process. Unlike the other existing methods that treat all image regions equally, guided filtering focuses on the local characteristics and structures within the image. It is clear for the case of the colored image and "Lena". In these two images, we can see that the overall image remains the same and the defining edges of the objects and Lena are still there.

Accordingly, the guided filter effectively smooths out noise and unwanted variations while preserving the sharpness of edges and boundaries. Without a doubt, this edge-preserving property can be efficient in applications where maintaining image details, such as textures, edges, or object boundaries, is crucial.

Ultimately, guided filtering improves image quality by keeping important details clear, making the image look better and more accurate.

# 2 Image Fusion

Image fusion is a technique used to combine information from several images into a single, more informative image. It aims to improve image quality and extract important features, and is applied in a number of fields.

## 2.1 State of the Art

### 2.1.1 Two-Scale Image Decomposition

We start by extracting a base layer $B_n$ and a details layer $D_n$ from the source images $I_n$, the base layer is obtained with an average filter while the details layer is the difference between the source image and its base layer.

$$B_n = I_n * Z$$

$$D_n = I_n - B_n$$

$Z$ is an average filter of size $9 \times 9$

### 2.1.2 Weight Map Construction With Guided Filtering

To fuse the two images, we need a weight map for each one to quantify the contribution of each pixel to the final fused image taking into consideration pixel saliency and space consistency. In order to construct the weight maps, we first start with the implementation of the guided filtering.

- **Guided Filtering for Gray Images**

To construct weight maps we use guided filtering. The guided filter is an edge preserving filter, its computing time is independent of the filter size as it is based on a local linear model. It assumes the filtering output is a linear transformation of the guidance image in a local window, with coefficients $a$ and $b$ estimated by minimizing the squared difference between the output image and the guidance image. As the coefficients $a$ and $b$ are calculated locally for every centered window of the guidance image, we need to average all the possible values $a$ and $b$ for each output pixel. This operation can be computationally expensive, this is why we first calculate $a$ and $b$ for every centered window once and store them in an array to which we then apply an average filter of the same parameter as the window's size to obtain the averaged coefficient $a$ and $b$. We can then compute the linear transformation that results in the output guided image.

- **Saliency Map**

To take into consideration pixel saliency when constructing the weight maps, we start with the construction of the saliency map $S_n$ of each image $I_n$. It is obtained through applying a Laplacian filter $L$ to the source image, then a low-pass Gaussian filter $g_r$, to the absolute value of the result.

$$H_n = I_n * L$$

$$S_n = |H_n| * g_r,$$

- **Weight Map**

We start constructing our first weight maps $P_n$ through comparing the saliency maps of each source image. We obtain binarized maps with 1 meaning that the saliency value of the pixel is maximal compared to the pixel's saliency value of the other source image, 0 otherwise.

$$P_{nk} = \begin{cases} 1 & \text{if } S_{nk} = \max(S_{1k}, S_{2k}, \ldots, S_{Nk}) \\ 0 & \text{if not} \end{cases}$$

However, the weight maps obtained are usually noisy and not aligned with object boundaries, which may produce artifacts in the fused image. Using spatial consistency is an effective way to solve this problem. To do this, guided image filtering is performed on each weight map with the corresponding source image serving as the guidance image.

$$W_n^B = G_{r1,1}(P_n, I_n)$$

$$W_n^D = G_{r2,2}(P_n, I_n)$$

### 2.1.3 Two-Scale Image Reconstruction

Finally, we use our refined weight maps to fuse the base layers by weighted averaging, same thing for the details layer. And then we sum the fused base layer and fused detail layer to obtain the final fused image.

$$\bar{B} = \sum_{n=1}^{N} W_n^B B_n$$

$$\bar{D} = \sum_{n=1}^{N} W_n^D D_n$$

$$F = \bar{B} + \bar{D}$$

## 2.2 Image fusion implementation for gray images

Gray images are images that have only channel with pixels taking values between 0 and 255, they are fairly simple to deal with compared to colored images that have 3 channels at least. Here is the result of applying the image fusion with guided filtering algorithm on two gray images with different field depths.

We first start by extracting the base and fusion layer.

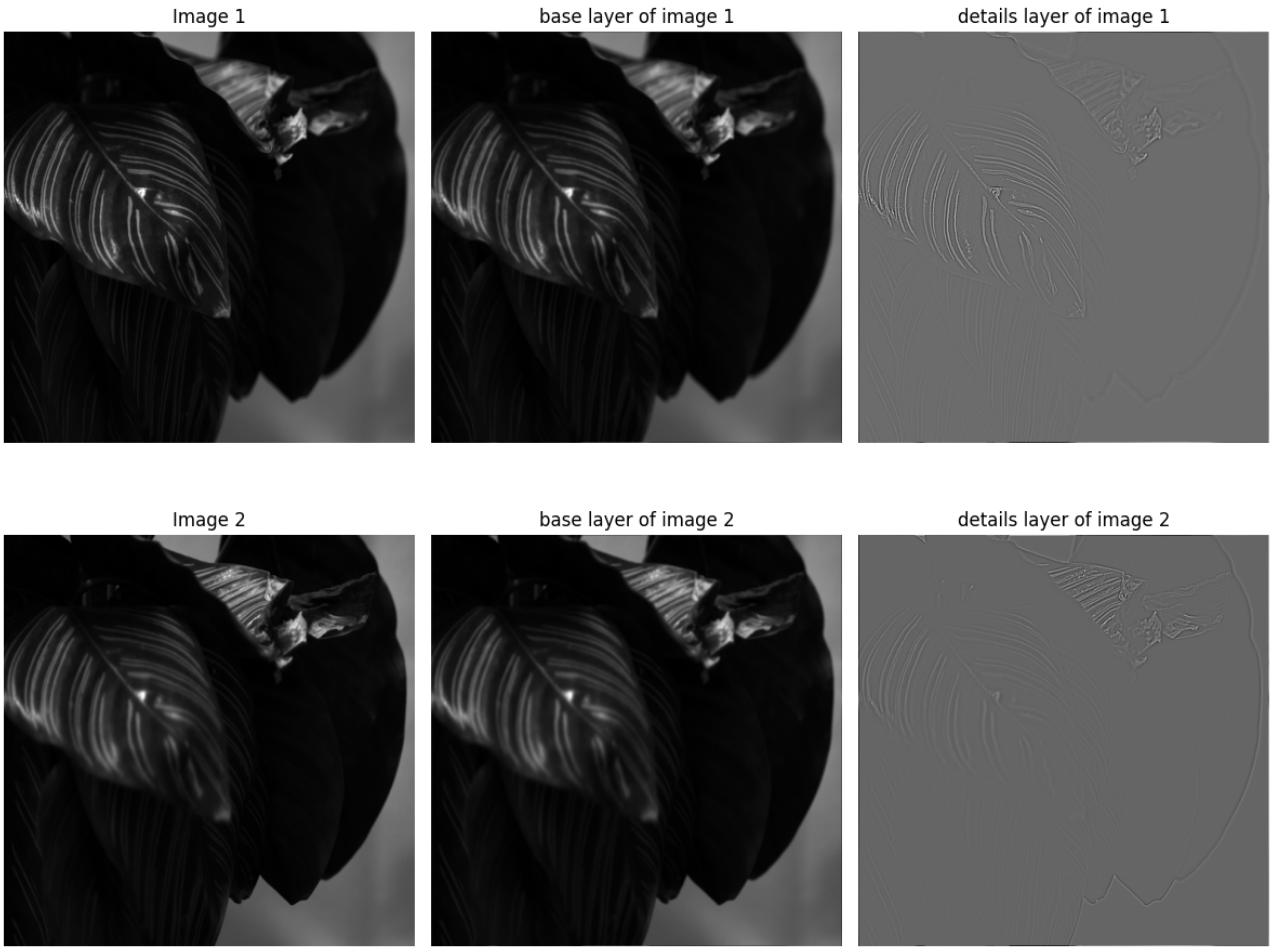


Figure 3: Two-Scale Image Decomposition.

We move on to calculating the saliency map.

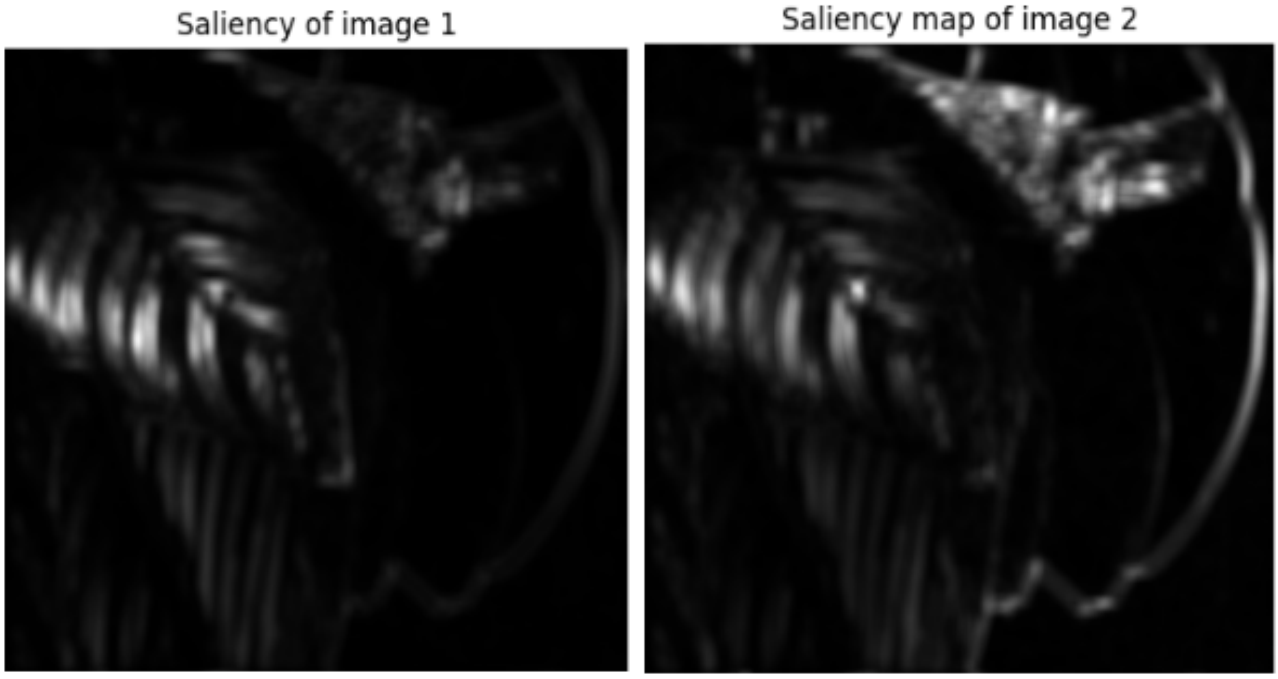


Figure 4: Saliency Map.

We then extract the first weight maps from comparing the two saliency maps according to the algorithm. Then refine them to remove the noise with guided filtering.

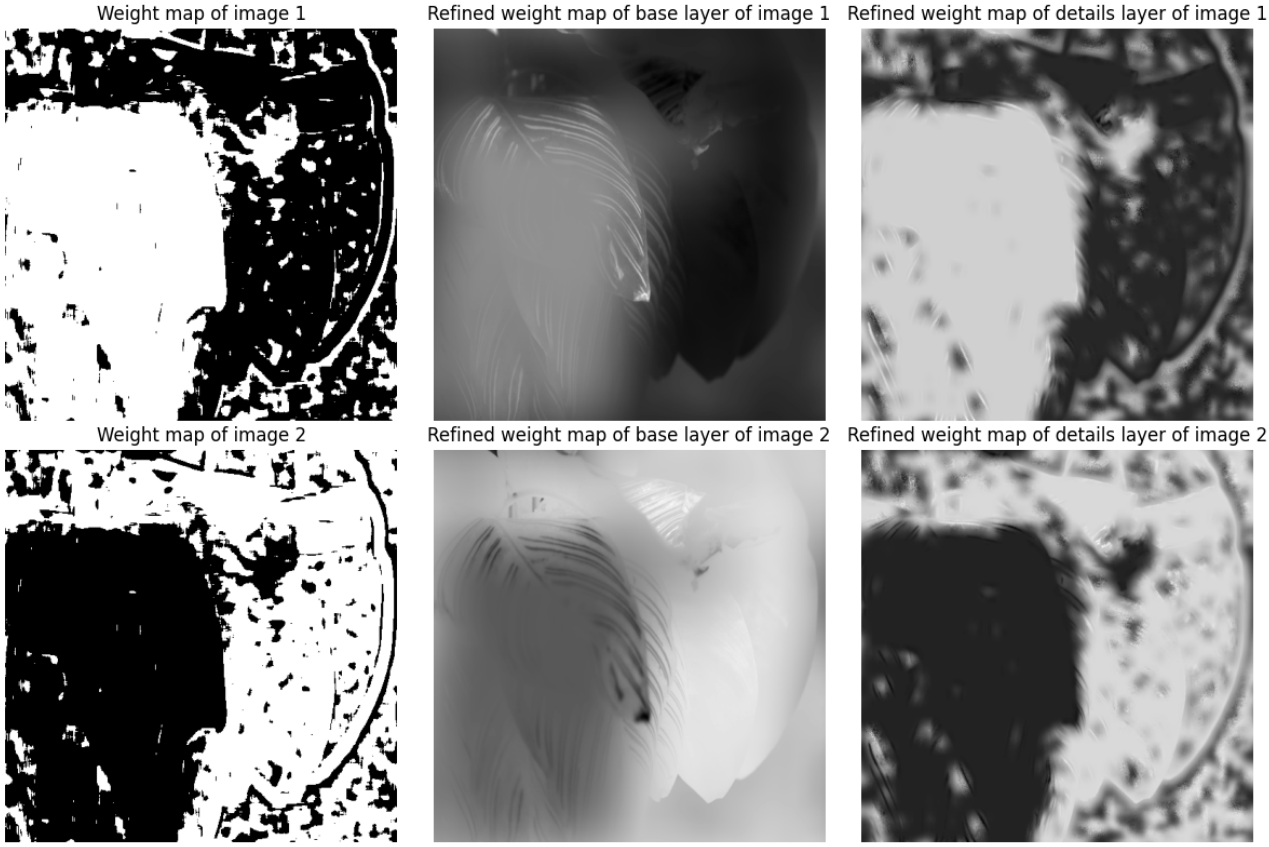


Figure 5: Weight Map & Two-Scale Image Reconstruction

Finally, we sum the weighted averaging of the base layer and details layer to obtain the fused image.

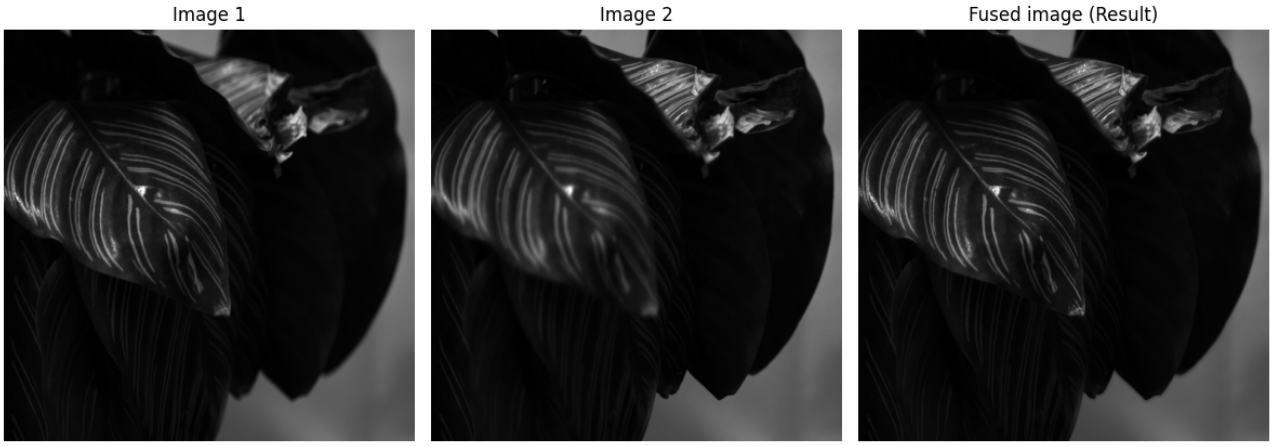


Figure 6: Fused Image

## 2.3 Image fusion implementation for colored images (RGB color system)

The image fusion concept is the same for colored and gray images, the difference is at the level of the guided filtering. RGB images are composed of three channels, one for the red, one for the blue and one for the green. When applying guided filtering we need to apply it on each channel separately which means that we apply the same algorithm for fusing gray images on each channel separately before merging the resulting 3 channels to obtain the final fused image.

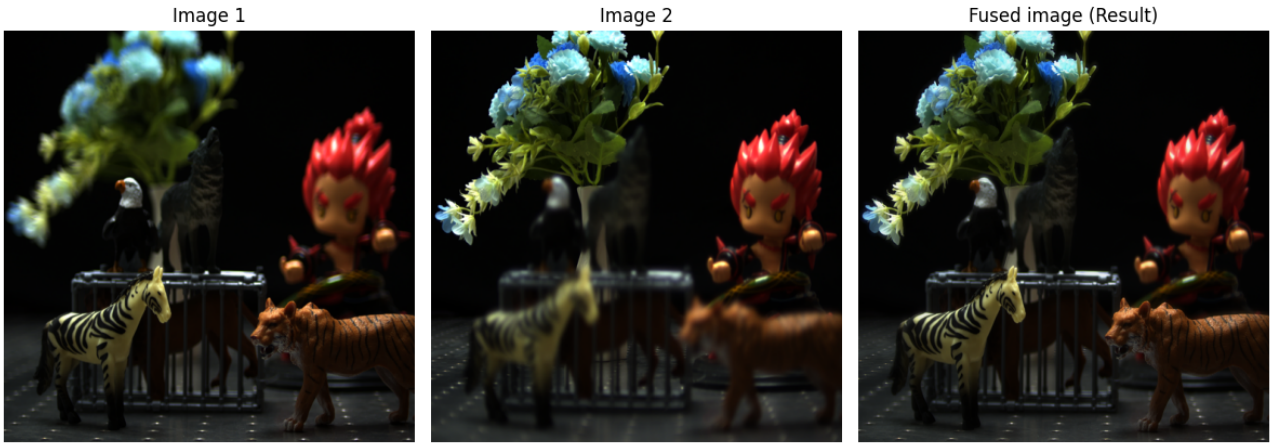


Figure 7: Image fusion Result - RGB color system.

## 2.4 Image fusion implementation for colored images (LAB color system)

We decided to try a different color system to see if the resulting fused image would be of better quality and more detailed. For that we first started by converting the RGB images to LAB images. The LAB images are composed of three channels, L channel for luminosity and details, A and B channels for the colors. We extracted the L channels from the source images and conducted the image fusion algorithm on them as they are the ones that hold information about details and depth of field. We then just average the A and B channels for the color, then merge the result to obtain the fused image in LAB color system. We convert the LAB image to RGB to better visualize the result.

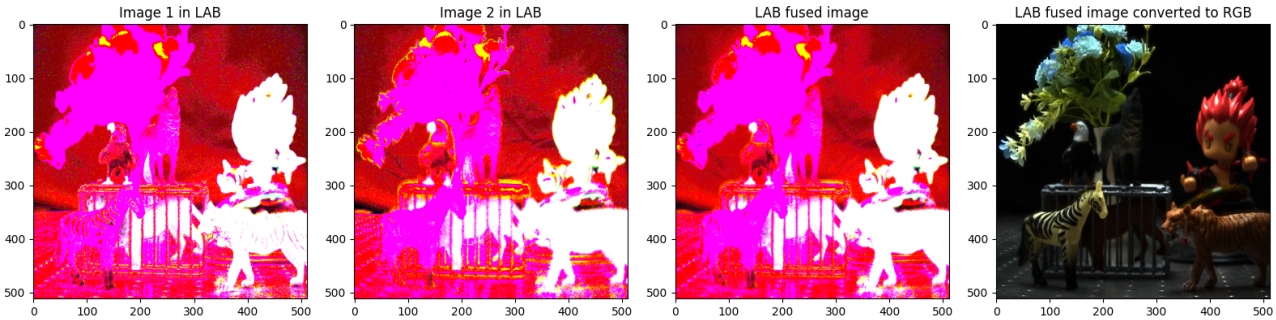


Figure 8: Image fusion Result - LAB color system.

## 2.5 Applying the image fusion algorithm to images we took ourselves

To test our implementation even further, we decided to take pictures with different depth of field and try fusing them using our algorithm.

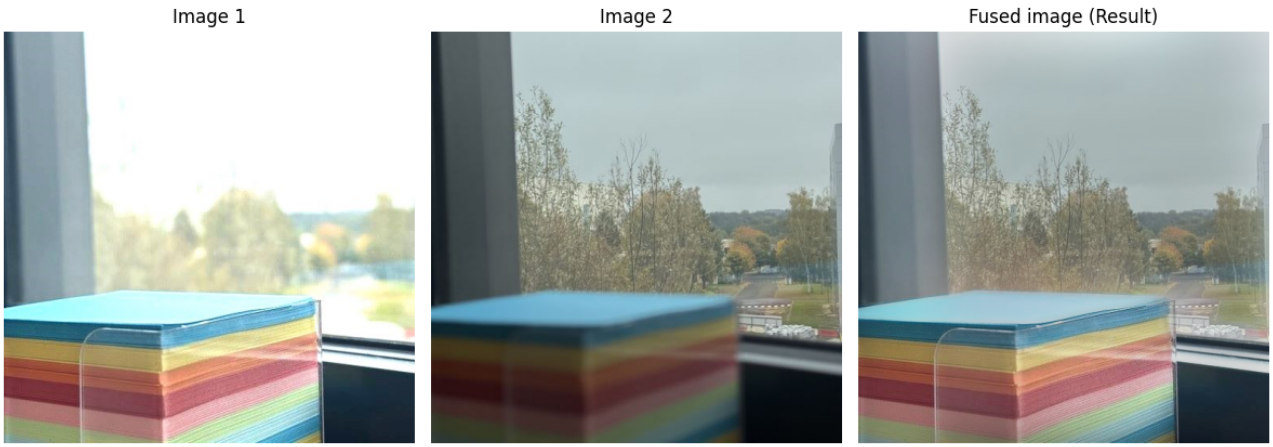


Figure 9: Multi-exposure fusion

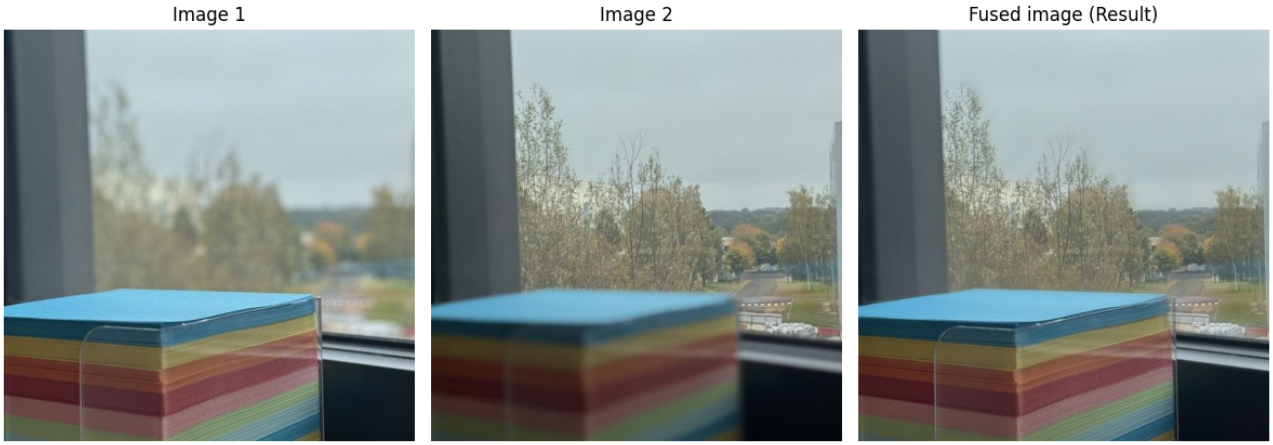


Figure 10: Multi-focus image fusion - Result 1


Figure 11: Multi-focus image fusion - Result 2

## 2.6 Improving Image Fusion - Using Iterative Approach

As seen in the figure 11, we can see the blur in the board of the blue note which is the result of the big gap in the depth of field and the focus, so we came up with an idea.

We decided to retake pictures with different depths of field but this time we tried to we fuse 3 images and minimize the gap on the depth between each image instead of 2 images with a big gap using our algorithm in an iterative way, like fusing the two first images then fuse their result with the third one and see the result.

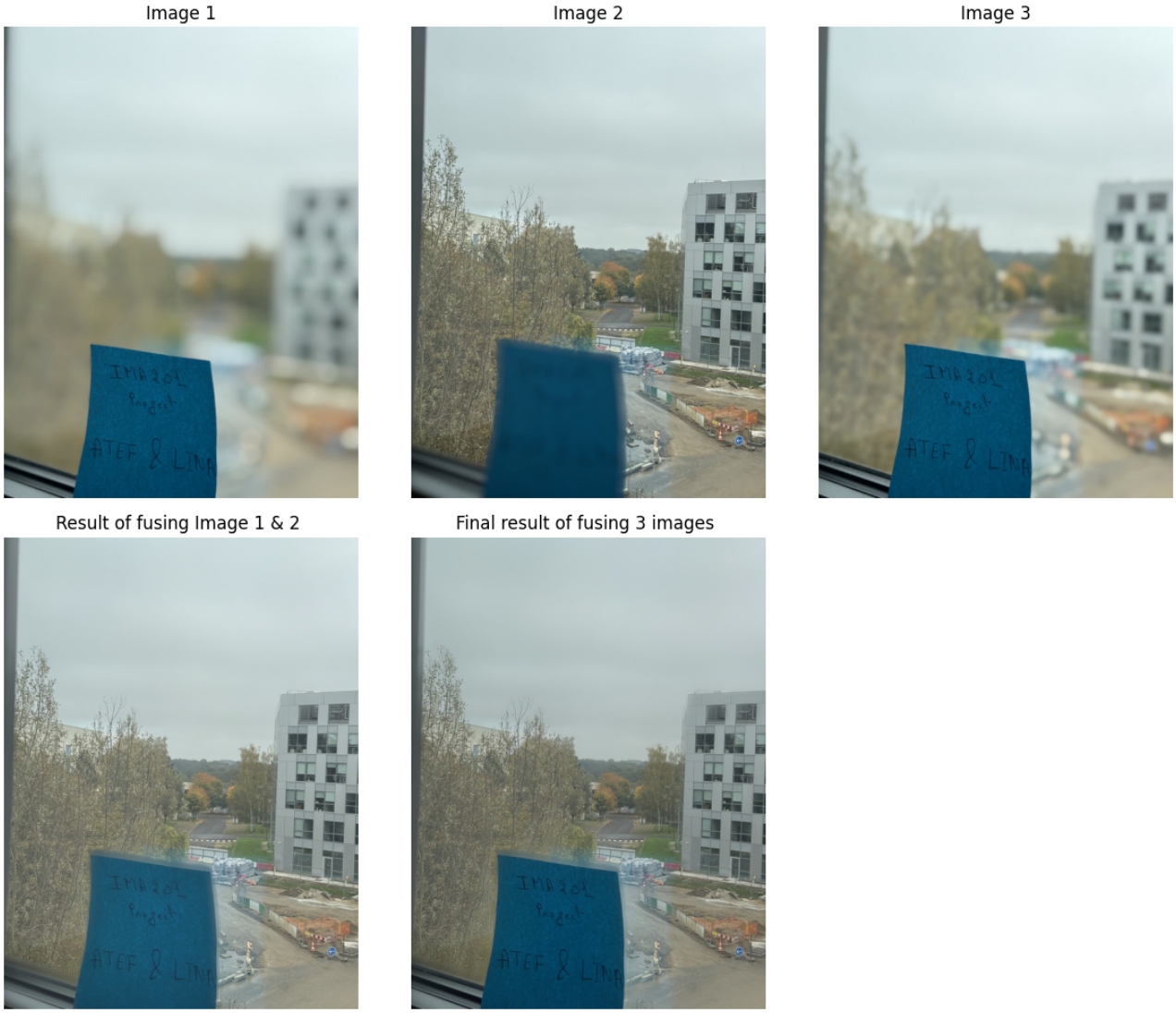


Figure 12: Iterative Approach - Result 1

While fusing three images doesn't show a big difference compared to fusing two images globally, focusing on details like text reveals a clear improvement.

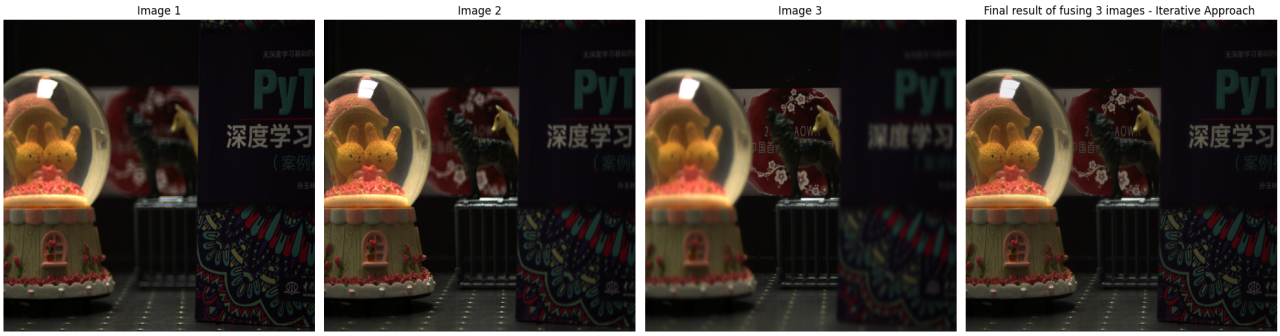


Figure 13: Iterative Approach - Result 2

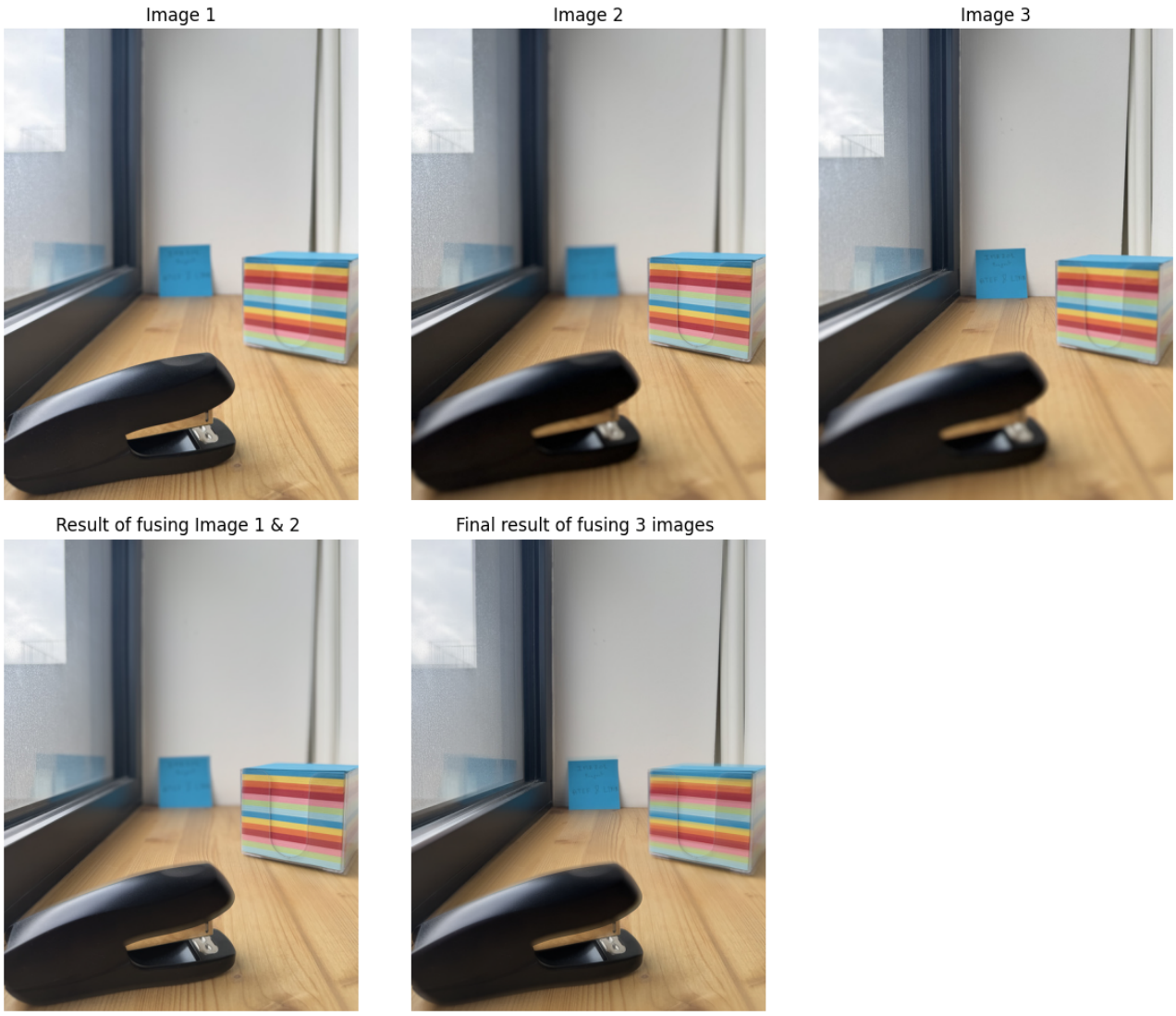


Figure 14: Iterative Approach - Result 3

# 3 Evaluation and Potential Improvements

## 3.1 Evaluation Metric Used

To better assess the quality of our implementation, we implemented specific metrics to evaluate the obtained quality of the images obtained through the different image fusion algorithms.

- First, the normalized mutual information $Q_{MI}$ between the fused and source images to measure how well the original information from source images is preserved in the fused image.

$$Q_{\mathrm{MI}} = \frac{2\,MI(A,F)}{H(A)+H(F)} + \frac{MI(B,F)}{H(B)+H(F)}$$

  where $H(A)$, $H(B)$, and $H(F)$ are the marginal entropies of $A$, $B$, and $F$, and $MI(A,F) = H(A)+H(F)-H(A,F)$ is the mutual information between the source image $A$ and fused image $F$.

- Second, a structural similarity based index $Q_Y$ to measure how well the structural information of source images is preserved.

$$Q_{\mathrm{Y}} = \begin{cases} \lambda_w\, SSIM(A_w,F_w) + (1-\lambda_w)\, SSIM(B_w,F_w), & \text{if } SSIM(A_w,B_w|w) \geq 0.75 \\ \max\{SSIM(A_w,F_w), SSIM(B_w,F_w)\}, & \text{if } SSIM(A_w,B_w|w) < 0.75 \end{cases}$$

  where $w$ is a $7 \times 7$ window, $A$ and $B$ are input images, $F$ is the fused image, and $\lambda_w = \frac{s(A_w)}{s(A_w)+s(B_w)}$, with $s(A_w)$ and $s(B_w)$ as local variances.

- Third, a $Q_C$ quality metric to estimate how well the important information in the source images is preserved in the fused image, while minimizing the amount of distortion that could interfere with interpretation.

$$Q_{\mathrm{C}} = \mu(A_w,B_w,F_w)\, UIQI(A_w,F_w) + (1-\mu(A_w,B_w,F_w))\, UIQI(B_w,F_w)$$

  with $\mu(A_w,B_w,F_w) = \frac{\sigma_{AF}}{\sigma_{AF}+\sigma_{BF}}$, constrained between 0 and 1, where $\sigma_{AF}$ and $\sigma_{BF}$ are covariances, and $UIQI$ is the universal image quality index.

- Fourth, a gradient based index $Q_G$ to evaluate the success of edge information transferred from the source images to the fused image

$$Q_{\mathrm{G}} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\left(Q_{AF}(i,j)\,\tau_A(i,j) + Q_{BF}(i,j)\,\tau_B(i,j)\right)}{\sum_{i=1}^{N}\sum_{j=1}^{M}\left(\tau_A(i,j)+\tau_B(i,j)\right)}$$

  where $Q_{AF} = Q^g_{AF}Q^o_{AF}$, representing edge strength and orientation preservation, with $\tau_A(i,j)$ and $\tau_B(i,j)$ reflecting the importance of each source image.

## 3.2 Results

We used the pre-built guided filtering function available in OpenCV library to implement the image fusion algorithm so as to assess how good our implementation is performing compared to the pre-built one.

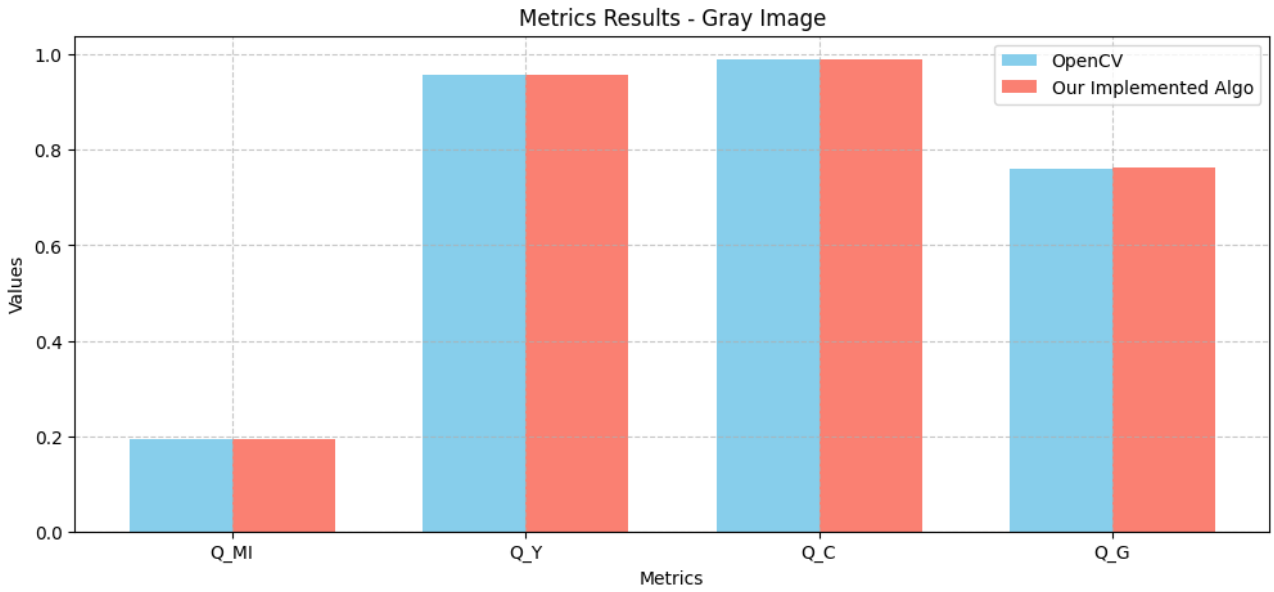


Figure 15: Results - Gray image

| Metric | $Q_{MI}$ | $Q_Y$ | $Q_C$ | $Q_G$ |
|---|---|---|---|---|
| OpenCV | 0.194 | 0.9563 | 0.9895 | 0.7599 |
| Our Implemented Algo | 0.1939 | 0.958 | 0.9898 | 0.7639 |

Table 1: Results - Gray Images

Based on the plots and the data tables, we can see that for gray images, our implementation is as good as OpenCV even outperforming it in certain aspects like information conservation as seen through $Q_G$ and $Q_C$ as well as structural information through $Q_Y$.

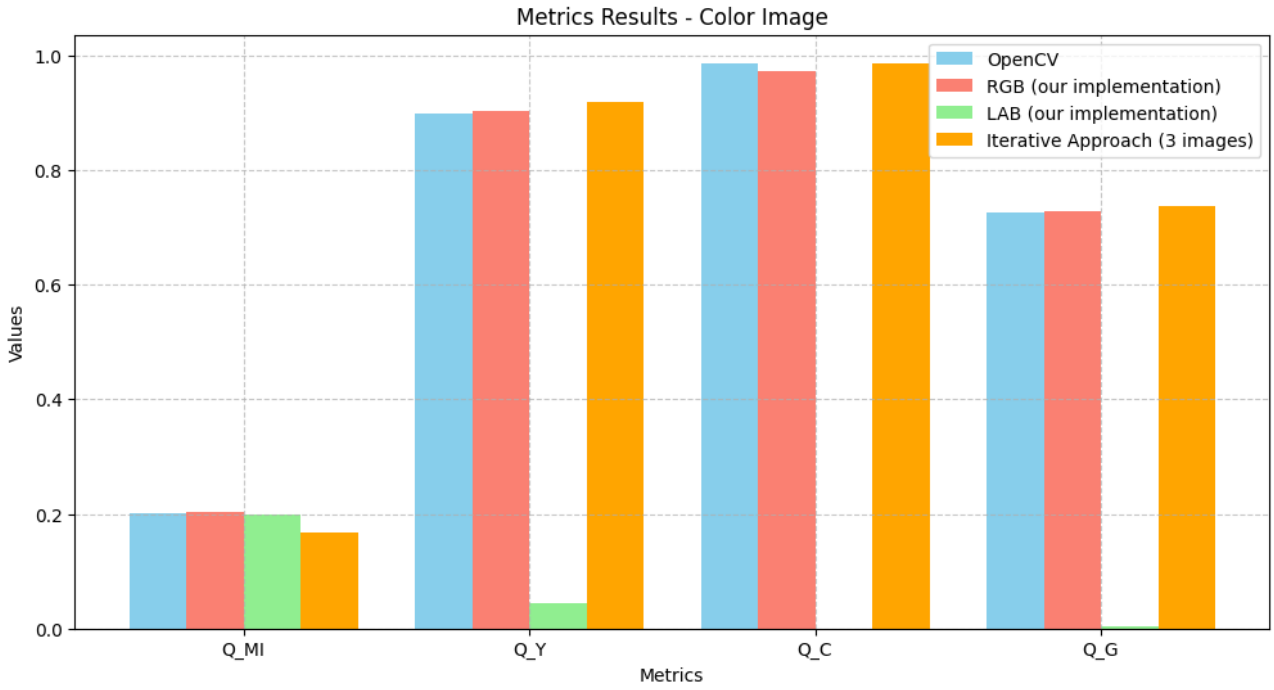


Figure 16: Results - Color images

| Metric | $Q_{\mathbf{MI}}$ | $Q_{\mathbf{Y}}$ | $Q_{\mathbf{C}}$ | $Q_{\mathbf{G}}$ | Runtime |
|---|---|---|---|---|---|
| OpenCV | 0.2012 | 0.8978 | 0.9858 | 0.7259 | 43.677 sec |
| Our Implemented Algo (RGB) | 0.2045 | 0.904 | 0.9725 | 0.7289 | 46.213 sec |
| Our Implemented Algo (LAB) | 0.1987 | 0.0442 | 0.0001 | 0.0041 | 8.903 sec |
| Iterative Approach (3 images) | 0.1685 | 0.9190 | 0.9868 | 0.7379 | 91.194 sec |

Table 2: Results - Color Image

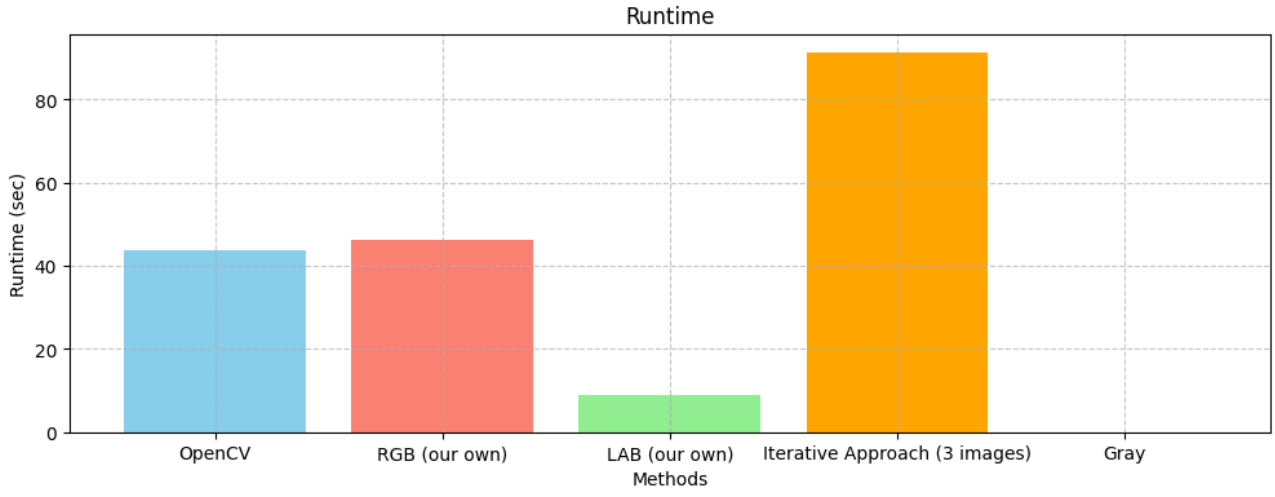


Figure 17: Runtime Results

Concerning the colored images, we can see that the most efficient computationally is the LAB color system as the algorithm is only applied on the L channel. However the metrics seem to indicate poor results for information preservation even though visually the results appear as good as the rest of the color systems. This is a point that could be further researched.

To summarize, finding an algorithm that outputs the best solution quickly is difficult. In fact, if we want to drive optimal results, we often need to rely on sophisticated and computationally expensive processes. This means that the algorithm will take much longer to run. However, for quickly running the algorithm it is often at the cost of some accuracy. Therefore, the most appropriate algorithm we consider is one that runs as accurately with respect to speed.

## 3.3 Potential Improvements

While we have successfully implemented the algorithms in the paper and have even tested with some variations, there is still territory to explore.

- **Algorithm Complexity:** Our current algorithm for color images works well, but we believe that improving the way the guided filtering is applied to color images could reduce processing time without sacrificing accuracy, or even improve accuracy.
- **Alternative Methods:** Such as deep learning-based algorithms for image fusion could give us better results or introduce new challenges.
- **Data and Metrics:** Trying new metrics and get into the image quality assessments more might need a dataset containing correctly merged images so we can easily compare our outputs.

All in all, these areas open the doors for future research that has the potential to yield better results or helpful insights.

# Conclusion

We have worked on implementing the image fusion algorithm using a guided filtering to construct weight maps. Experiments show that the proposed method can well preserve the original and complementary information of multiple input images. We have also explored the effect of different color systems on the image fusion outcome, and found out the LAB color system for colored images was the most computationally efficient. Encouragingly, the result of the image fusion applied to pictures we took ourselves has also proved very efficient and resulted in clear fused images. The key takeaway is that we managed to implement a guided filtering that outperforms the pre-defined one on the openCV library when applied in the context of image fusion. However, the computational efficiency of our implementation of certain color systems can be improved, and this would be a great step to be further researched.

# References

[1] S. Li, X. Kang, and J. Hu, *Image Fusion with Guided Filtering*, IEEE Transactions on Image Processing, vol. 22, no. 7, July 2013.

[2] K. He, J. Sun, and X. Tang, *Guided Image Filtering*, Proceedings of the European Conference on Computer Vision, Heraklion, Greece, Sep. 2010.

[3] Used Images: `https://docs.opencv.org/4.x/index.html`

[4] Used Images: `https://drive.google.com/drive/u/0/folders/1PMS5Rr4SUL1ptnC8qA0aEdGKgcBFvj5i`