

## Article

# An Efficient and Effective Image Decolorization Algorithm Based on Cumulative Distribution Function

Tirui Wu <sup>1,\*</sup>, Ciaran Eising <sup>2</sup> , Martin Glavin <sup>1</sup>  and Edward Jones <sup>1</sup><sup>1</sup> School of Engineering, University of Galway, H91 TK33 Galway, Ireland;

martin.glavin@universityofgalway.ie (M.G.); edward.jones@universityofgalway.ie (E.J.)

<sup>2</sup> Department of Electronic and Computer Engineering, University of Limerick, V94 T9PX Limerick, Ireland; ciaran.eising@ul.ie

\* Correspondence: t.wu4@universityofgalway.ie

**Abstract:** Image decolorization is an image pre-processing step which is widely used in image analysis, computer vision, and printing applications. The most commonly used methods give each color channel (e.g., the R component in RGB format, or the Y component of an image in CIE-XYZ format) a constant weight without considering image content. This approach is simple and fast, but it may cause significant information loss when images contain too many isoluminant colors. In this paper, we propose a new method which is not only efficient, but also can preserve a higher level of image contrast and detail than the traditional methods. It uses the information from the cumulative distribution function (CDF) of the information in each color channel to compute a weight for each pixel in each color channel. Then, these weights are used to combine the three color channels (red, green, and blue) to obtain the final grayscale value. The algorithm works in RGB color space directly without any color conversion. In order to evaluate the proposed algorithm objectively, two new metrics are also developed. Experimental results show that the proposed algorithm can run as efficiently as the traditional methods and obtain the best overall performance across four different metrics.

**Keywords:** cumulative distribution function; edge recall ratio; gradient recall ratio; image contrast preservation; image decolorization



**Citation:** Wu, T.; Eising, C.; Glavin, M.; Jones, E. An Efficient and Effective Image Decolorization Algorithm Based on Cumulative Distribution Function. *J. Imaging* **2024**, *10*, 51. <https://doi.org/10.3390/jimaging10030051>

Academic Editor: Arslan Munir

Received: 7 January 2024

Revised: 30 January 2024

Accepted: 16 February 2024

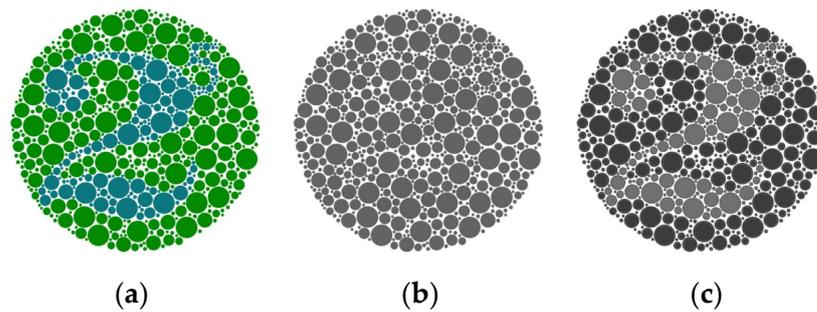
Published: 20 February 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Despite color images dominating our daily life nowadays, grayscale images are still widely used in image analysis and computer vision applications, hence the need for image decolorization (ID) algorithms. Many important computer vision algorithms are designed for grayscale images, such as Haar Cascade Classifier for face detection [1], Histogram-of-Gradients (HoG) [2], Scale-Invariant Feature Transform (SIFT) [3], Local Binary Pattern (LBP) [4], edge detection [5,6], line detection [7], corner detection [8,9], and keypoint descriptors [10–12]. These algorithms are still widely used in many visual applications, such as visual Simultaneous Localization and Mapping (vSLAM) [13–17] and Structure from Motion (SfM) [18,19]. They heavily rely on image details (e.g., edges and corners) and image contrast to achieve good performance. Also, ID is a critical step for printing applications when grayscale printing is needed. Traditional methods, for example extracting the Y component of an image in CIE-XYZ color space [20], which give each color channel a fixed coefficient to compute the weighted average of the three color channels, produce acceptable results most of the time. But sometimes, such methods may produce unacceptable output, as the example in Figure 1 below shows, where a lot of image details are lost in the image in Figure 1b.



**Figure 1.** Image decolorization. (a) The input image, (b) grayscale image generated according to [20], (c) proposed method.

In this paper, we present an efficient and effective ID algorithm. It can retain more image contrast and details than the traditional method from the original color image, as Figure 1c shows. It uses the information from the cumulative distribution function (CDF) of each channel image to compute a weight for each pixel in each color channel. Then, these weights are used to combine the three color channels (red, green, and blue) to obtain the final grayscale value. Because of its simplicity, the computational cost is nearly as low as that of the traditional methods, which makes it suitable for practical applications on computationally constrained platforms, such as mobile phones and edge computing devices. In order to measure its performance objectively, two new metrics are also proposed.

The contributions of this work are summarized as follows:

- We propose a fast, efficient, and effective ID algorithm to address the computational limitations of existing ID algorithms that limit their practical usage. The proposed algorithm has linear time complexity and can achieve comparable runtime performance to traditional methods;
- We propose two new objective metrics to measure the performance of our ID algorithm. By combining these new metrics with existing metrics, a more comprehensive evaluation of ID algorithm performance may be obtained.

The rest of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 describes the proposed method. Section 4 describes the two proposed objective metrics and demonstrates the effectiveness and efficiency of the proposed method, and Section 5 provides the conclusions.

## 2. Related Work

Several techniques have been proposed to retain more information from the original color image. They roughly are categorized into two groups: approaches based on the use of local information in the image [21–28] and based on global information [29–36]. The local approaches enhance the final grayscale output with local information, like chrominance edges and high-frequency image details, while global approaches combine both local and non-local contrast, structure, and illumination for color compression [35].

Among the first group, Bala and Eschach [21] proposed a method to preserve the local distinction between adjacent colors by integrating high-frequency chrominance information into the luminance channel. The problem with this method is that it may introduce lots of undesired artifacts around image edges due to the high frequency filter used in the algorithm. In [22], Gooch et al. take the conversion process as an optimization problem: the gray value of each pixel is iteratively adjusted to minimize an objective function, which is based on the local contrasts between all the pixel pairs. However, the high computational complexity of this method makes it impractical for high resolution images. Another optimization-based method was proposed by Rasche et al. [23] to minimize the error function based on matching the gray differences to the corresponding color differences to find an optimal conversion. The method can preserve image contrast and maintain luminance consistency, but its computational cost is high, which limits its use. Queiroz

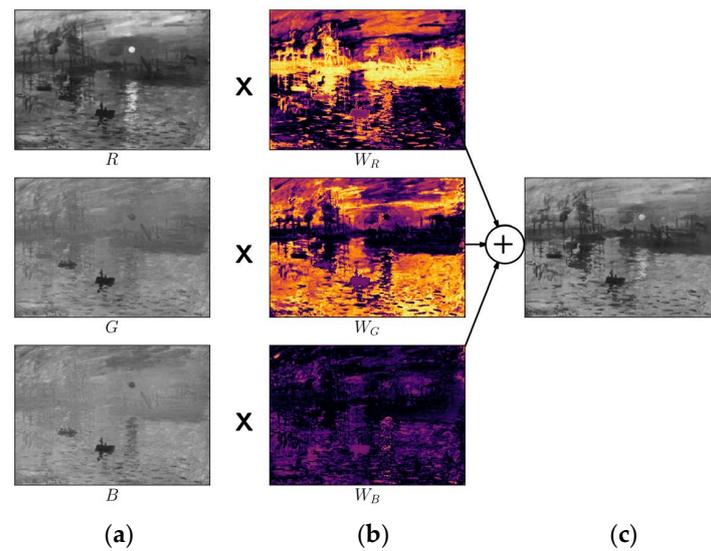
and Braun [26] developed a reversible method which is based on mapping colors to low-visibility high-frequency textures that are applied onto the gray image to achieve image decolorization. The Wavelet transform is used in the conversion process, which adds computational cost. Wu and Toet [27] proposed a method based on multiresolution channel fusion for ID. This method computes weight maps to fuse the three color channels based on several well-defined metrics. Another fusion-based algorithm was recently proposed in [28]. Such fusion-based methods are effective, but not efficient; they require a lot of computational resources for image decomposition and reconstruction when the input image is large. One recent local approach in [37] uses boundary points as optimization targets so that the output grayscale image can maximally preserve the boundary points of the color image.

As for the global approaches, Grundland and Dodgson [29] introduced a fast method to decolorize color images. It combines novel techniques for image sampling and dimensionality reduction to achieve efficient conversion. In addition to its speed and simplicity, the algorithm has the advantages of continuous mapping, global consistency, and grayscale preservation, as well as predictable luminance, saturation, and hue ordering properties. In [32], a global–local ID method was proposed. The method first globally assigns gray values and determines color ordering, then secondly locally enhances the grayscale to reproduce the original contrast. In [33,34], a convolutional neural network is used to extract high-level abstract features from the input color image to achieve effective contrast preservation. In [35], an invertible image decolorization is proposed to produce invertible grayscale images using invertible neural networks (INNs). Ref. [36] uses deep neural networks to extract content information based on the human vision system to select suitable grayscales for decolorization. The big disadvantage with this approach is that it needs to run the optimization process with each single image, which is a time-consuming process even with high performance GPU. Recently, a two-stage algorithm based on the image histogram and local variance maximization to preserve more image information was proposed in [38]. While this approach provides good performance, the use of optimization results in a relatively high computational cost and limits its practical application.

Most of the algorithms described above carry a higher computational cost than traditional methods. Therefore, it is challenging to use them in many real-time applications. While, on the one hand, modern computing devices with embedded cameras, e.g., smartphones or other edge devices, have substantial computing resources, on the other hand, there is an increasing use of high-resolution cameras, which results in a substantial pre-processing load. This is one reason why traditional algorithms are still widely used in many different applications. Our aim in this work is to propose an algorithm which can run nearly as fast as the traditional methods, preserve more image contrast and image details than the traditional methods, resulting in higher functional performance, and be implemented easily.

### 3. Proposed Algorithm

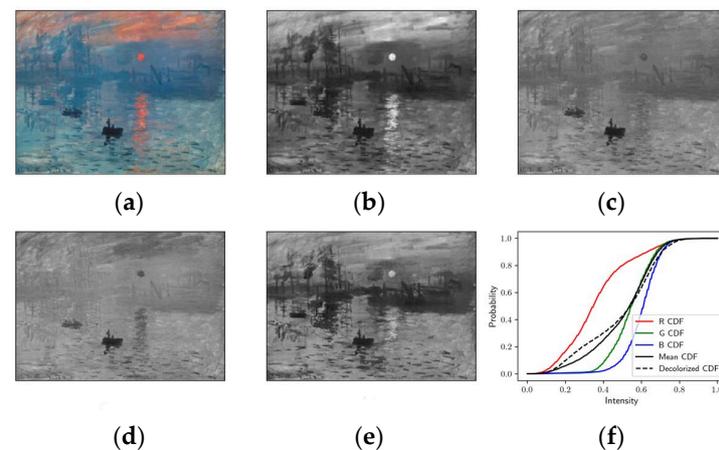
In this section, we will discuss the proposed method in detail. Figure 2 summarizes the workflow of the proposed method. In the first step, we split the input image into three separate channel images: red, green, and blue. Then, we derive weight maps based on the CDF in each channel. In the final step, we compute the weighted sum of the individual channels to obtain the final grayscale output.



**Figure 2.** Output of each stage of the proposed method. (a) Generate the individual channel images (R, G, B); (b) calculate weights for each channel image with Equations (2) and (3); (c) combine weighted channels to produce output grayscale image.

### 3.1. Algorithm Description

For a color image, each color channel (also referred to as a channel image) will have a different probability mass function (PMF, also called a histogram in image processing) which results in different cumulative distribution function (CDF), as shown in the example in Figure 3. The CDF can be computed by running a cumulative sum on the PMF ( $CDF[i] = \sum_{k=0}^i PMF[k]$ , where  $i \in [0, L - 1]$ , and  $L$  is the number of bins in the PMF). From the CDF curves for each color channel (shown in Figure 3f), we can obtain some information about each color channel image. For example, red channel images are dark because more than half of the pixel intensities are less than 100 and green/blue channel images are light because more than half of the pixels have an intensity larger than 150 (Figure 3b–d). Also, the red channel has higher image contrast than the green/blue channels, reflected in its CDF curve having a shallower slope. So, the CDF contains some useful information about image content.



**Figure 3.** Cumulative distribution function of each channel for a sample image. (a) is the input image; (b–d) are the intensities of the red, green, and blue channels, respectively; (e) the final decolorized image; (f) the corresponding CDFs for the three channels. Mean CDF is the average of the three individual channel CDFs. The decolorized CDF is also shown, and not only approximates the mean CDF but also captures some of the details of the three color channel CDFs.

Inspired by histogram equalization (HE), which uses CDF as a mapping function to redistribute pixel intensities to increase image contrast, we design a simple strategy to compute a weight map for each channel image by calculating an optimal CDF which can approximate the three channel image CDFs and then computing a weight for each pixel color component based on differences between the optimal CDF and the channel image CDFs. We build an optimal CDF by averaging the three channel CDFs so that each element in this optimal CDF has minimum total  $L_2$  distance to the corresponding elements in the three channel CDFs. This can be obtained by solving a simple minimization problem:  $\operatorname{argmin}_{CDF_{opt}} \sum_{c \in \{r,g,b\}} (CDF_{opt} - CDF_c)^2$ . The solution is:

$$CDF_{opt}[i] = \frac{\sum_{c \in \{r,g,b\}} CDF_c[i]}{3} \tag{1}$$

where  $i \in [0, L - 1]$  is the pixel intensity,  $L$  is the maximum level of pixel intensity (e.g., for 8-bit depth image,  $L = 2^8$ ), and  $CDF_c$  is the CDF of the individual channel image. After we have an optimal CDF, we can compute the weight for each level in each color channel with the following equations:

$$W_c[i] = e^{-|CDF_c[i] - CDF_{opt}[i]|} \tag{2}$$

$$W_c[i] = \frac{W_c[i]}{\sum_{i=0}^{L-1} W_c[i]} \tag{3}$$

where  $c \in \{r, g, b\}$  is the channel index. We normalize  $W_c$  into the range  $(0, 1]$  to obtain the relative importance for each grayscale level within each channel. To exclude zero denominator, 0 is not included. For pixel intensity, the closer the channel CDF is to the optimal CDF, the higher its relative importance score.

Once we have  $W_c$ , we compute the weighted sum of each color pixel to get the grayscale output:

$$Y[x, y] = \sum_{c \in \{r,g,b\}} \frac{W_c[I_c[x, y]] \cdot I_c[x, y]}{W[I_c[x, y]]} \tag{4}$$

where  $Y$  is the output grayscale image;  $x \in [0, N - 1]$  and  $y \in [0, M - 1]$  are the column and row index;  $M$  and  $N$  are the number of image rows and columns, respectively;  $I_c$  is channel image  $c$ ,  $W[I_c[x, y]] = \sum_c W_c[I_c[x, y]]$  to ensure that the sum of the weights for a pixel color components equals 1. For the sample image in Figure 2a, the final gray scale image generated according to this method is given in Figure 2e.

In the above equation, we can use a look-up-table (LUT) strategy ( $W_c[I_c[x, y]]$ ) to obtain weights for pixel color components once the initial weights have been calculated. Also, our method works with the original RGB space directly, so no additional color space conversion is needed. The primary computational cost difference between the traditional methods and the proposed method is per-channel weight computation. For high-resolution images captured by modern devices, this may bring substantial additional computational cost, which may be a challenge for low-power CPUs or edge computing devices. A possible solution will be discussed in the next sub-section.

### 3.2. CDF Approximation

An important consideration in the design of the proposed method is that it should be as simple and fast as possible so that the computational cost will not be a concern when it is used in real applications. As we discuss above, the proposed method may bring substantial additional computational cost compared to the traditional method when the input image resolution is very high, as will commonly be the case with, e.g., high-resolution images captured by modern smartphones. So, we should consider this problem, especially for resource-constrained devices. One possible solution is to perform image sub-sampling. That is, we simply sub-sample the input image at a fixed sampling step  $s$  to approximate the true underlying CDFs. This approach works well due to inherent characteristics of natural images, whereby objects are somewhat continuous in the image.

By way of an example, in Figure 4, we sub-sample the input image with step  $s \in [1, 2, 4, 8, 16]$  and show the effect of this on the CDF. The differences among the output CDFs at different sub-sampling steps are small, even in the logarithm space (the third row). Therefore, we expect that small differences will not have a significant impact on the conversion process.

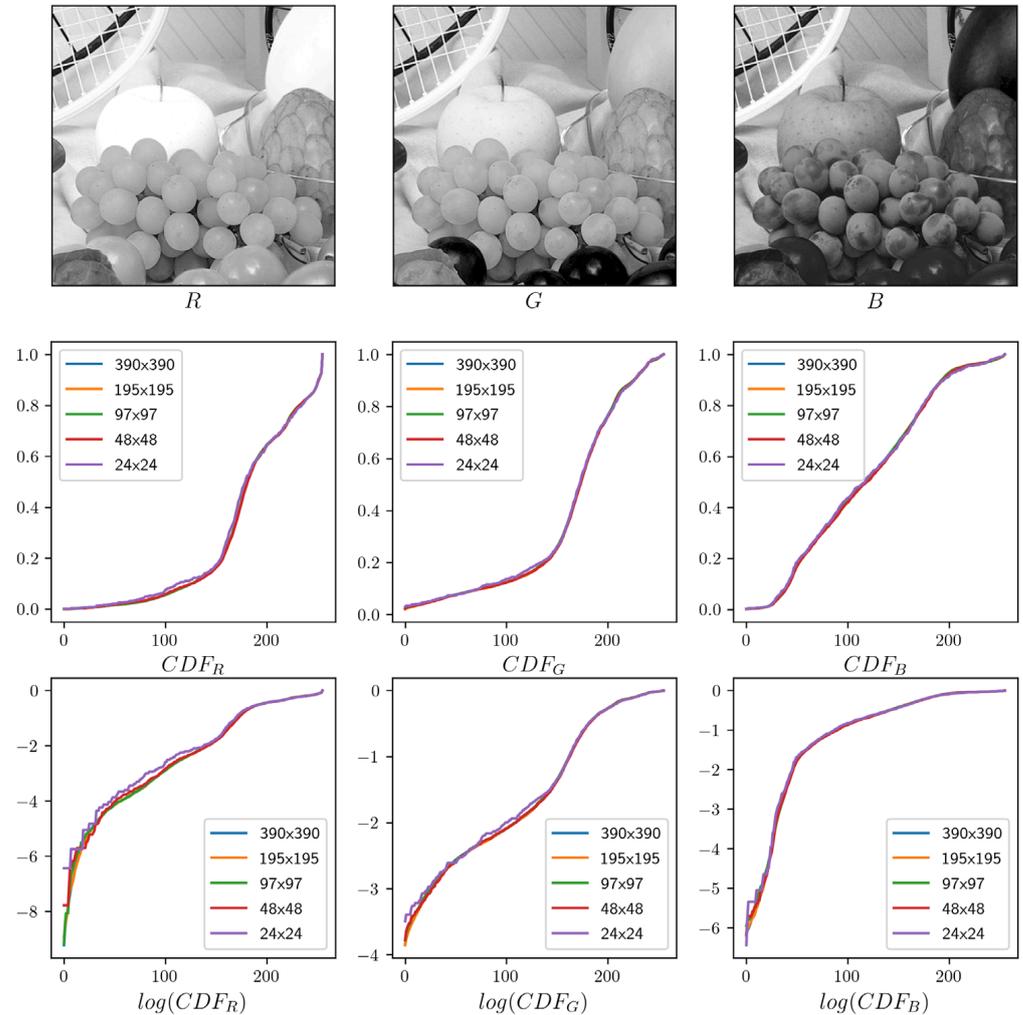


Figure 4. Effect of image sub-sampling on CDF estimation.

#### 4. Experiments

As we claim above, the proposed method can preserve more image contrast and details than the traditional methods. In this section, we compare the proposed method to several other ID algorithms with several objective metrics. We also compare the execution speed of our method with the Decolorize method of [29] as a benchmark; this algorithm is claimed to have linear computational cost as a function of image size (number of pixels).

##### 4.1. Performance Metrics

To compare performance, we require suitable objective metrics. Unfortunately, there are very few metrics specifically designed for this particular task. Reference image-based metrics that attempt to capture structural fidelity, like SSIM [39], are not suitable here because we do not have the ground-truth grayscale version of an input color image. Many of the well-known non-reference-image-based quality assessment methods, like BRISQUE [40], are not suitable here because they are designed for quality assessment of natural color images.

Based on the above discussion, we initially chose lower-level metrics such as the root mean square contrast difference (RMS) [41] to measure image contrast. RMS is appropriate for this application as it is a non-learning and non-reference-based objective metric for contrast measuring and is widely used in image quality measurement [42–44]. A larger RMS value means greater global image contrast. RMS is defined as:

$$RMS = \sqrt{\frac{1}{MN} \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} (I[x, y] - \bar{I})^2} \quad (5)$$

where  $I[x, y]$  is pixel intensity at location  $(x, y)$   $\bar{I}$  is mean intensity of the image, and  $M, N$  are the image dimensions. We used RMS to measure the contrast of the converted grayscale image of different algorithms.

We also extended the normalized root mean squared contrast difference (NRMS) to color images so that we can measure deviations between the converted grayscale image and the color channel images. This is an important metric to ensure that the final output does not have very different appearance from the input image:

$$NRMS = \left( \sum_c \frac{\|I_c - Y\|}{I_c} \right) / 3 \quad (6)$$

where  $\|\cdot\|$  denotes the Frobenius norm and  $Y$  is the final output image. For this metric, the smaller the better.

To measure image detail preservation, two new metrics were proposed. These two new measures are used to capture image details which are important for image analysis and computer vision tasks.

Gradient Recall Ratio (GRR): Gradient information is a very important component of image detail and is used in many popular computer vision algorithms, such as feature extraction for HoG [2] and Seam Carving [45]. So, we propose the *GRR* metric to measure how much gradient information has been retained in the grayscale output image.

For an input color and its corresponding grayscale image, we computed the gradient image for each channel image in the original RGB image and the corresponding grayscale image. Then, we evaluated the *GRR* as follows:

$$GRR = \frac{\sum_{x,y} \sqrt{Y_x[x, y]^2 + Y_y[x, y]^2}}{\sum_c \sum_{x,y} \sqrt{I_x^c[x, y]^2 + I_y^c[x, y]^2}} \quad (7)$$

where  $I_x^c, I_y^c, Y_x,$  and  $Y_y$  are channel and grayscale image derivatives along  $x$  and  $y$ . A higher score indicates better retention of gradient information.

Edge Recall Ratio (ERR): Image edges are one of the fundamental sources of information for many computer vision tasks, for example line detection [7] and shape recognition [46]. This metric is used to measure how faithfully edge information is preserved in the image decolorization. We used the Canny operator [5] on each color image channel to obtain the whole edge mask:

$$E_{rgb} = (E_r \vee E_g \vee E_b) \quad (8)$$

where  $E_{rgb}, E_r, E_g,$  and  $E_b$  are edge masks of the original color image and the  $R, G, B$  color channels, respectively;  $\vee$  is the logical OR operator. The edge mask values are 1 if a point is on an edge; otherwise, they are 0.

We also ran the Canny operator on the corresponding grayscale image to obtain  $E_Y$ . Then, we defined the *ERR* as follows:

$$ERR = \frac{\sum_{x=0}^{N-1} \sum_{y=0}^{M-1} \left( (E_{rgb}[x, y] \oplus SE) \wedge E_Y \right) [x, y]}{\sum_{x=0}^{N-1} \sum_{y=0}^{M-1} E_{rgb}[x, y]} \quad (9)$$

where  $\wedge$  is logical AND operator,  $\oplus$  is morphological dilation operator, and  $SE$  is a morphological structural element. The reason we used morphological dilation operator is because there may be some degree of edge shifting among color channels caused by chromatic aberration. This may be brought into the grayscale image during the conversion. By using dilation, shifting in the range  $[-\lfloor \frac{r}{2} \rfloor, \lfloor \frac{r}{2} \rfloor]$  can be tolerated, where  $r$  is the radius of structural element. In our experiment, we used a square structural element of size 5. For this metric, the higher the score, the more edge information has been preserved. Due to the dilation operation, this score may be larger than 1.

4.2. Results

We compared the proposed method to several classical algorithms and two modern deep learning-based algorithms from the literature. Figure 5 shows some example results. Based on visual inspection, the proposed method can retain more image details and image contrast than the traditional method (CIE-Y). In comparison with the other benchmark methods, our method consistently produces visually appealing output. For example, in the second row, many salient features and texture of surfaces are retained well in the grayscale image with the proposed algorithms.

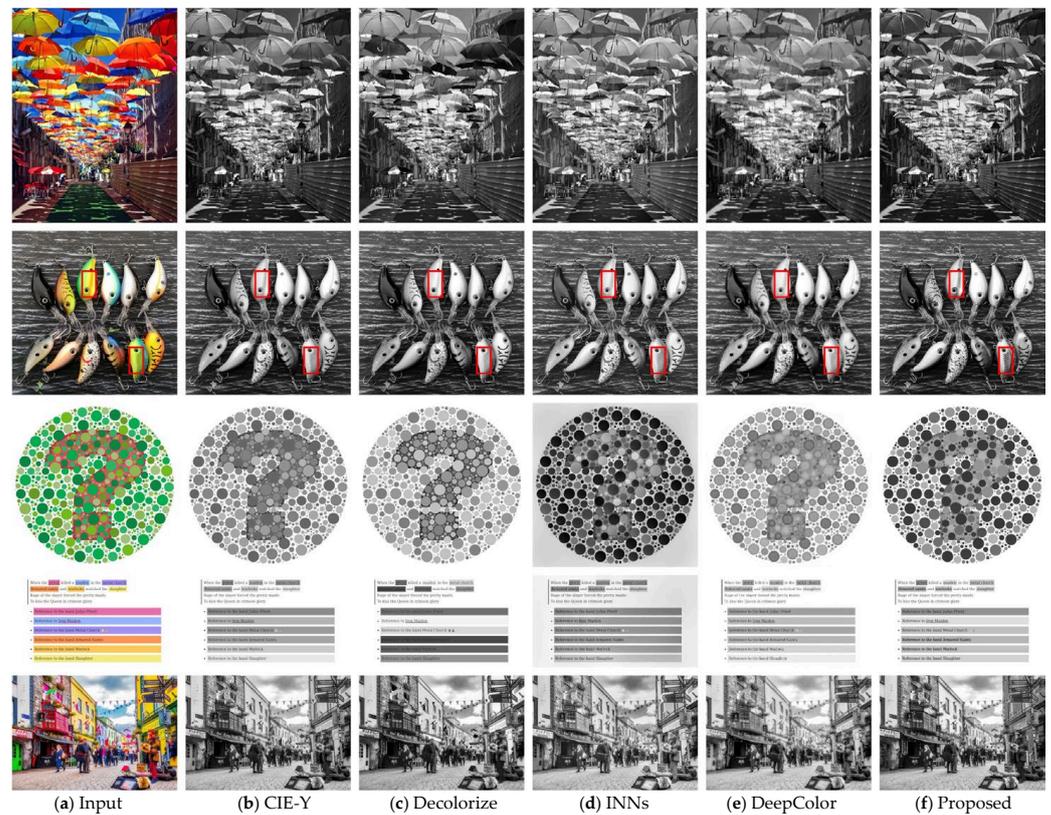


Figure 5. Some results from different ID algorithms.

To compare the algorithms objectively, we used the standard dataset built by Cadik [47] in our experiments. This dataset was designed specifically for the evaluation of ID algorithms and contains a carefully selected set of 25 images, including both natural and artificial images. Table 1 gives the RMS scores of each algorithm for each test image in the dataset presented in [47]. The best score for each test image (each row) is highlighted in bold text. There is some variation in that different algorithms perform better with different test images, but the proposed algorithm achieves the best overall score, which is much higher than the traditional CIE-Y method. This shows that the adaptive weights which are computed based on CDF can combine every color pixel effectively to produce a grayscale

output with high image contrast. This improves visual appearance because the human vision system is more sensitive to image contrast than absolute luminance.

**Table 1.** RMS scores (higher is better, the best score is in bold text).

Image Name	Smith [32]	CIE-Y [20]	Decolorize [29]	Color2Gray [22]	Neumann [24]	INNs [35]	DeepColor [36]	Proposed
butterfly	0.74	0.72	0.72	0.48	<b>0.77</b>	0.61	0.70	0.73
155_5572	0.40	0.40	0.42	0.45	0.53	0.40	0.39	<b>0.53</b>
dscn9952	0.53	0.49	0.44	0.54	0.74	0.47	0.45	<b>0.91</b>
im2-color	0.06	0.05	0.15	<b>0.29</b>	0.16	0.04	0.15	0.20
colorspastel	0.06	0.06	0.13	<b>0.35</b>	0.13	0.05	0.10	0.06
25_color	0.40	0.38	0.67	0.52	0.45	0.36	<b>0.84</b>	0.56
balls0_color	0.40	0.39	0.38	0.46	0.44	0.39	0.47	<b>0.59</b>
impatient	0.44	0.40	0.50	0.36	0.41	0.39	0.41	<b>0.53</b>
c8tz7768	0.30	0.29	<b>0.48</b>	<b>0.48</b>	0.33	0.28	0.28	0.29
sunrise312	0.18	0.16	<b>0.33</b>	0.26	0.20	0.15	0.19	0.31
monarch	0.42	0.37	0.38	0.38	0.39	0.36	0.35	<b>0.48</b>
serrano	0.49	0.47	0.51	0.46	<b>0.53</b>	0.41	0.46	0.48
girl	0.76	0.72	0.63	0.65	<b>0.78</b>	0.69	0.62	0.73
ski_tc8	0.66	0.62	0.60	0.43	<b>0.71</b>	0.56	0.62	0.70
watch	<b>0.60</b>	0.50	0.53	0.29	0.55	0.46	0.47	0.57
arctichare	0.21	0.21	<b>0.26</b>	0.22	0.22	0.19	0.24	0.23
ramp	0.03	0.03	0.19	<b>0.52</b>	0.16	0.02	0.06	0.08
text	<b>0.28</b>	0.25	0.27	0.21	0.26	0.19	0.27	0.26
colorwheel	0.08	0.03	0.33	<b>0.42</b>	0.22	0.04	0.30	<b>0.42</b>
tulips	0.60	0.53	<b>0.71</b>	0.46	0.57	0.53	0.66	0.64
fruits	0.28	0.28	0.35	0.26	0.31	0.25	0.28	<b>0.42</b>
kodim03	0.41	0.38	0.41	<b>0.42</b>	0.40	0.39	0.34	0.39
portrait	0.53	0.50	0.51	0.42	0.52	0.46	0.43	<b>0.54</b>
34445	0.63	0.58	0.61	0.37	0.62	0.49	0.58	<b>0.64</b>
tree_color	0.79	0.73	0.75	0.65	0.77	0.74	0.69	<b>0.80</b>
Overall average	0.41	0.38	0.45	0.41	0.45	0.36	0.41	<b>0.48</b>

Table 2 presents the detailed NRMS scores. Our algorithm achieved the best overall performance along with two other methods [20,32]. This is because the proposed algorithm is still a linear combination of channel images, but different from traditional methods [20] in that different color pixels will have different weights to combine them. No low-level image information (e.g., image edges and textures) is used in the conversion process. As a result, the conversion process will not cause large deviation from the original input image. This shows that the proposed algorithm can retain the original illuminance appearance very well.

Table 2. NRMS scores (lower is better, the best score is in bold text).

Image Name	Smith [32]	CIE-Y [20]	Decolorize [29]	Color2Gray [22]	Neumann [24]	INNs [35]	DeepColor [36]	Proposed
butterfly	0.18	0.17	0.17	0.32	0.19	0.34	0.17	<b>0.16</b>
155_5572	0.63	0.62	0.72	0.69	0.61	<b>0.58</b>	0.59	0.62
dscn9952	0.63	0.62	0.81	0.72	0.62	<b>0.61</b>	0.70	0.69
im2-color	<b>0.25</b>	<b>0.25</b>	0.28	0.43	0.33	<b>0.25</b>	<b>0.25</b>	<b>0.25</b>
colorspastel	<b>0.13</b>	<b>0.13</b>	0.15	0.36	0.21	<b>0.13</b>	0.15	<b>0.13</b>
25_color	0.35	0.37	<b>0.27</b>	0.32	0.33	0.37	0.32	0.28
balls0_color	0.42	0.42	0.56	0.43	0.40	0.43	<b>0.38</b>	0.41
impatient	0.39	0.39	0.47	0.46	0.38	0.39	0.39	<b>0.37</b>
c8tz7768	0.30	0.30	0.36	0.50	0.32	0.35	0.32	<b>0.29</b>
sunrise312	<b>0.18</b>	<b>0.18</b>	0.27	0.23	0.19	0.19	0.19	0.23
monarch	<b>0.30</b>	<b>0.30</b>	0.36	0.42	0.31	0.31	0.36	0.32
serrano	<b>0.34</b>	<b>0.34</b>	0.38	0.45	0.36	<b>0.34</b>	0.35	0.35
girl	<b>0.19</b>	<b>0.19</b>	0.25	0.22	<b>0.19</b>	<b>0.19</b>	0.22	<b>0.19</b>
ski_tc8	0.31	<b>0.30</b>	0.37	0.42	0.31	0.32	0.33	<b>0.30</b>
watch	<b>0.13</b>	<b>0.13</b>	0.16	0.32	0.14	0.21	0.25	<b>0.13</b>
arctichare	0.08	0.06	0.10	0.10	0.07	0.07	0.10	0.07
ramp	0.21	<b>0.20</b>	0.26	0.53	0.27	<b>0.20</b>	0.22	0.22
text	0.09	0.08	0.08	0.11	0.08	0.16	0.09	<b>0.07</b>
colorwheel	0.54	0.54	0.53	0.78	0.58	0.55	<b>0.50</b>	0.51
tulips	<b>0.31</b>	<b>0.31</b>	0.42	0.47	<b>0.31</b>	0.33	0.32	0.32
fruits	<b>0.24</b>	0.25	0.26	0.30	0.28	0.26	0.25	0.26
kodim03	0.31	0.31	0.33	0.38	0.32	0.30	0.35	<b>0.29</b>
portrait	<b>0.12</b>	<b>0.12</b>	0.14	0.19	0.13	0.17	0.16	<b>0.12</b>
34445	<b>0.13</b>	<b>0.13</b>	0.15	0.25	0.13	0.20	0.14	<b>0.13</b>
tree_color	0.07	0.07	0.07	0.15	0.07	0.09	0.11	<b>0.06</b>
Overall average	<b>0.27</b>	<b>0.27</b>	0.32	0.38	0.29	0.29	0.29	<b>0.27</b>

Table 3 captures the *GRR* scores of the different algorithms. Again, our algorithm achieves the best performance with this metric. This result coincides with the *RMS* score: high image contrast output helps to retain more image gradient information, which is vital information for many computer vision algorithms, as we mentioned before. These experimental results also show that different ID algorithms will cause different degrees of image detail information loss, which is important in many computer vision tasks. Table 4 presents the *ERR* scores. As expected, *ERR* scores are consistent with *GRR* scores because gradient information is key information for edge detection. We propose that this metric is not a simple duplicate of the *GRR* because sometimes, high image gradient information does not mean that an image contains lots of edges; for example, in an image with dark background which has many isolated white pixels.

Table 3. GRR scores (higher is better, the best score is in bold text).

Image Name	Smith [32]	CIE-Y [20]	Decolorize [29]	Color2Gray [22]	Neumann [24]	INNs [35]	DeepColor [36]	Proposed
butterfly	<b>0.35</b>	0.32	0.33	0.23	0.32	0.19	<b>0.35</b>	0.33
155_5572	0.31	0.28	0.30	0.25	0.29	0.21	0.33	<b>0.45</b>
dscn9952	0.39	0.30	0.40	0.33	0.29	0.17	0.33	<b>0.57</b>
im2-color	0.12	0.03	0.20	<b>0.45</b>	0.30	0.10	0.29	0.31
colorspastel	0.23	0.21	0.38	<b>0.47</b>	0.33	0.19	0.33	0.22
25_color	0.26	0.24	<b>0.33</b>	0.28	0.27	0.21	0.32	0.31
balls0_color	0.34	0.3	0.31	0.29	0.35	0.26	0.34	<b>0.46</b>
impatient	0.31	0.27	0.32	0.27	0.29	0.23	0.31	<b>0.40</b>
c8tz7768	0.25	0.20	0.33	<b>0.40</b>	0.21	0.23	0.28	0.36
sunrise312	0.34	0.28	0.37	0.41	0.26	0.23	0.31	<b>0.46</b>
monarch	0.35	0.31	0.35	0.32	0.31	0.29	0.34	<b>0.39</b>
serrano	<b>0.35</b>	0.31	0.31	0.26	0.32	0.24	<b>0.35</b>	0.32
girl	<b>0.38</b>	0.32	0.35	0.31	0.33	0.27	0.34	<b>0.38</b>
ski_tc8	<b>0.37</b>	0.31	0.33	0.25	0.32	0.25	0.35	0.34
watch	<b>0.38</b>	0.33	0.36	0.24	0.35	0.27	0.35	0.35
artichare	0.34	0.33	<b>0.37</b>	<b>0.37</b>	0.33	0.26	0.36	0.34
ramp	0.15	0.06	0.33	<b>0.53</b>	0.27	0.08	0.30	0.25
text	<b>0.36</b>	0.33	0.33	0.26	0.34	0.21	0.34	0.34
colorwheel	0.12	0.04	0.21	0.24	0.15	0.12	0.28	<b>0.30</b>
tulips	0.34	0.31	0.26	0.24	0.3	0.26	0.34	<b>0.37</b>
fruits	0.33	0.30	0.33	0.25	0.31	0.23	0.33	<b>0.41</b>
kodim03	<b>0.38</b>	0.33	0.32	0.36	0.33	0.26	0.36	<b>0.38</b>
portrait	<b>0.36</b>	0.32	0.35	0.31	0.34	0.26	0.34	0.34
34445	<b>0.36</b>	0.32	0.32	0.24	0.34	0.22	0.33	0.35
tree_color	0.36	0.31	0.32	0.34	0.33	0.24	0.31	0.33
Overall average	0.31	0.27	0.32	0.32	0.30	0.22	0.33	<b>0.36</b>

Table 4. ERR scores (higher is better, the best score is in bold text).

Image Name	Smith [32]	CIE-Y [20]	Decolorize [29]	Color2Gray [22]	Neumann [24]	INNs [35]	DeepColor [36]	Proposed
butterfly	0.83	0.82	0.81	0.79	0.83	0.78	0.83	<b>0.87</b>
155_5572	0.52	0.48	0.50	0.44	0.54	0.37	0.60	<b>0.75</b>
dscn9952	<b>0.67</b>	0.61	0.66	0.61	0.49	0.38	0.59	0.57
im2-color	0.50	0.00	0.81	0.78	0.86	0.00	<b>0.91</b>	0.89
colorspastel	0.57	0.58	0.76	0.63	0.58	0.50	<b>0.71</b>	0.61

Table 4. Cont.

Image Name	Smith [32]	CIE-Y [20]	Decolorize [29]	Color2Gray [22]	Neumann [24]	INNs [35]	DeepColor [36]	Proposed
25_color	0.97	0.97	0.97	0.97	0.96	0.99	0.97	<b>1.17</b>
balls0_color	0.61	0.59	0.60	0.59	0.65	0.55	0.61	<b>0.85</b>
impatient	0.39	0.35	0.45	0.39	0.38	0.29	0.40	<b>0.63</b>
c8tz7768	0.61	0.45	0.70	<b>0.79</b>	0.50	0.50	<b>0.79</b>	0.75
sunrise312	0.62	0.51	0.64	<b>0.74</b>	0.41	0.34	0.56	0.70
monarch	0.66	0.62	<b>0.67</b>	0.64	0.63	0.59	0.66	0.67
serrano	<b>0.76</b>	0.74	0.71	0.70	0.73	0.65	<b>0.76</b>	<b>0.76</b>
girl	<b>0.80</b>	0.65	0.77	0.63	0.60	0.43	0.68	0.76
ski_tc8	<b>0.73</b>	0.66	0.71	0.59	0.65	0.57	0.67	0.70
watch	0.85	0.82	<b>0.89</b>	0.79	0.84	0.76	0.84	0.86
arctichare	0.64	0.62	0.64	<b>0.68</b>	0.62	0.43	0.63	0.65
ramp	0.25	0.00	0.69	0.31	<b>0.80</b>	0.00	0.22	0.51
text	0.92	0.93	0.93	0.92	0.93	<b>0.96</b>	0.93	0.93
colorwheel	0.57	0.00	0.75	0.79	0.68	0.30	<b>0.86</b>	<b>0.86</b>
tulips	0.70	0.66	0.52	0.59	0.66	0.53	0.67	<b>0.73</b>
fruits	0.56	0.53	0.59	0.42	0.53	0.36	0.58	<b>0.73</b>
kodim03	0.77	0.72	0.69	0.73	0.68	0.53	0.73	<b>0.78</b>
portrait	<b>0.73</b>	0.70	<b>0.73</b>	0.72	<b>0.73</b>	0.63	0.72	0.72
34445	<b>0.77</b>	0.76	0.76	0.75	<b>0.77</b>	0.71	0.76	0.76
tree_color	0.61	0.52	0.59	<b>0.62</b>	0.58	0.38	0.53	0.57
Overall average	0.66	0.57	0.70	0.66	0.67	0.50	0.69	<b>0.75</b>

#### 4.3. Speed Assessment

As noted above, the proposed method is nearly as fast as the traditional simple methods yet provides superior performance on average across all the metrics considered. In this sub-section, we ran a simple speed test for three methods, CIE-Y, Decolorize [29], and the proposed method, to illustrate this point. The experiment was simple: we executed the three algorithms with a high-resolution image (e.g.,  $4928 \times 3280$ ) hundreds of times. At each step, we down-sampled the input image by a factor of 2. We implemented the proposed method in Python and in C++ (single thread) without using any specific optimizations. Except the last column in Table 5, all were implemented in Python.

Table 5. Runtime comparison (seconds).

Scale	CIE-Y	Decolorize	Proposed	Proposed (CDF Approx.)	Proposed (C++)
1	0.1447	8.5286	1.1775	0.7790	0.1204
1/2	0.0409	1.5657	0.2969	0.1982	0.0280
1/4	0.0103	0.3394	0.0828	0.0510	0.0068
1/8	0.0022	0.0698	0.0204	0.0121	0.0015
1/16	0.0009	0.1492	0.0052	0.0030	0.0009
1/32	0.0007	0.0043	0.0019	0.0012	0.0007

From Table 5, we can see that the computational load of all methods is approximately linear with image size (reflected in the down-sampling rate, denoted by “Scale” in the table): for each scale value, the execution time is reduced to approximately 1/4 of the previous value. For C++ implementation (without CDF approximation), our method can achieve 8 FPS with the original 15 MP image with mobile Intel i7 CPU without any optimization techniques applied. We believe that the execution speed of the proposed method can be boosted significantly by using, e.g., Single Instruction Multiple Data (SIMD) instructions such as Streaming SIMD Extensions (SSE) and Advanced Vector Extensions (AVX). There is also scope for performance boosting by implementing GPUs.

The logarithm of the execution time of each algorithm as a function of scale is plotted in Figure 6. Both linear and logarithmic plots are presented in order to both illustrate large differences in execution time at low-scale values as well as show the detail at higher-scale values. The unoptimized C++ implementation is slightly faster than the simple CIE-Y method, which is implemented in Python with NumPy [48]. NumPy is supported by the highly optimized Intel MKL library, which provides performance optimization. So, while it is difficult to directly compare implementations on different software platforms, the run-time figures in Table 5 and Figure 6 suggest that the computational cost of the proposed method can be comparable to the CIE-Y traditional method with appropriate optimization.

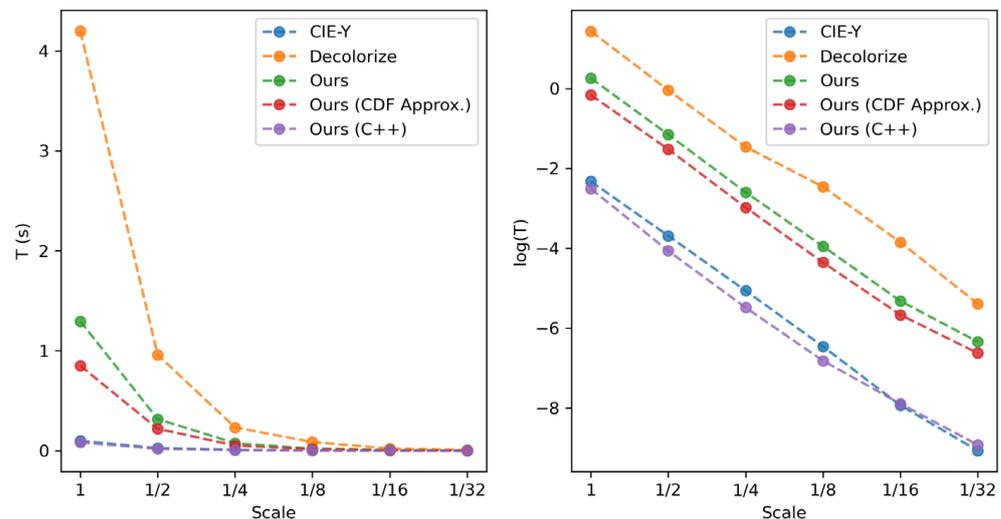


Figure 6. Runtime of the algorithms.

### 5. Conclusions

In this paper, we propose an efficient and effective image decolorization algorithm. It not only preserves image contrast and detail from the input color image, but also has nearly the same computational complexity as the simple CIE-Y method used as a benchmark, while providing a performance advantage in terms of structural (edge) information preservation. To evaluate its performance objectively, two new metrics are proposed. Experimental results show our method can achieve the best overall performance across all metrics. The proposed method has potential to be a good alternative to the traditional methods in terms of balancing the need to preserve image contrast and detail without significant additional computational cost.

While the proposed algorithm exhibits a number of advantages, there is scope for further investigation and improvement. For example, instead of using the CDF, alternative measures for image content may be used, as well as a more effective distance metric to compare the difference between the optimal measure and each channel image measure. Also, the impact of the image decolorization process on the performance of feature extraction or other computer vision algorithms (e.g., HoG [2], SIFT [3]) is of interest. These are areas for future research.

**Author Contributions:** Conceptualization, T.W.; methodology, T.W.; software, T.W.; validation, T.W.; formal analysis, T.W.; investigation, T.W.; resources, T.W.; data curation, T.W.; writing—original draft preparation, T.W.; writing—review and editing, T.W., C.E., M.G. and E.J.; visualization, T.W.; supervision, E.J.; project administration, E.J.; funding acquisition: not applicable. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is contained within the article.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

- Viola, P.; Jones, M. Rapid Object Detection using a Boosted Cascade of Simple Features. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, HI, USA, 8–14 December 2001; IEEE: Piscataway, NJ, USA, 2001.
- Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; IEEE: Piscataway, NJ, USA, 2005.
- Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [[CrossRef](#)]
- Ojala, T.; Pietikainen, M.; Harwood, D. Performance Evaluation of Texture Measures with Classification based on Kullback Discrimination of Distributions. In Proceedings of the 12th International Conference on Pattern Recognition, Jerusalem, Israel, 9–13 October 1994; IEEE: Piscataway, NJ, USA, 1994.
- Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *PAMI-8*, 679–698. [[CrossRef](#)]
- Qu, Y.-D.; Cui, C.-S.; Chen, S.-B.; Li, J.-Q. A Fast Subpixel Edge Detection Method using Sobel–Zernike Moments Operator. *Image Vis. Comput.* **2005**, *23*, 11–17.
- Duda, R.O.; Hart, P.E. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Commun. ACM* **1972**, *15*, 11–15. [[CrossRef](#)]
- Rosten, E.; Drummond, T. Machine Learning for High-speed Corner Detection. In Proceedings of the 9th European Conference on Computer Vision, Graz, Austria, 7–13 May 2006; Springer: Berlin/Heidelberg, Germany, 2006.
- Trajković, M.; Hedley, M. Fast Corner Detection. *Image Vis. Comput.* **1998**, *16*, 75–87. [[CrossRef](#)]
- Calonder, M.; Lepetit, V.; Ozuysal, M.; Trzcinski, T.; Strecha, C.; Fua, P. BRIEF: Computing a Local Binary Descriptor Very Fast. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 1281–1298. [[CrossRef](#)] [[PubMed](#)]
- Leutenegger, S.; Chli, M.; Siegwart, R.Y. BRISK: Binary Robust Invariant Scalable Keypoints. In Proceedings of the International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; IEEE: Piscataway, NJ, USA, 2011.
- Alahi, A.; Ortiz, R.; Vanderghyest, P. Freak: Fast Retina Keypoint. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; IEEE: Piscataway, NJ, USA, 2012.
- Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. In Proceedings of the Proceedings Ninth IEEE International Conference on Computer Vision, Nice, France, 13–16 October 2003; IEEE: Piscataway, NJ, USA, 2003.
- Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [[CrossRef](#)]
- Engel, J.; Sturm, J.; Cremers, D. Semi-Dense vVisual Odometry for a Monocular Camera. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; IEEE: Piscataway, NJ, USA, 2013.
- Forster, C.; Pizzoli, M.; Scaramuzza, D. SVO: Fast Semi-Direct Monocular Visual Odometry. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; IEEE: Piscataway, NJ, USA, 2014.
- Hirose, K.; Saito, H. Fast Line Description for Line-based SLAM. In Proceedings of the BMVC, Surrey, UK, 3–7 September 2012.
- Iglhaut, J.; Cabo, C.; Puliti, S.; Piermattei, L.; O'Connor, J.; Rosette, J. Structure From Motion Photogrammetry in Forestry: A Review. *Curr. For. Rep.* **2019**, *5*, 155–168. [[CrossRef](#)]
- Schonberger, J.L.; Frahm, J.-M. Structure-from-Motion Revisited. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
- Fairchild, M.D. *Color Appearance Models*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
- Bala, R.; Eschbach, R. Spatial Color-to-Grayscale Transform Preserving Chrominance Edge Information. In Proceedings of the Color Imaging Conference, Scottsdale, AZ, USA, 9–12 November 2004; pp. 82–86.
- Gooch, A.A.; Olsen, S.C.; Tumblin, J.; Gooch, B. Color2Gray: Saliency-Preserving Color Removal. *ACM Trans. Graph. (TOG)* **2005**, *24*, 634–639. [[CrossRef](#)]
- Rasche, K.; Geist, R.; Westall, J. *Re-Coloring Images for Gamuts of Lower Dimension*; Eurographics: Dublin, Ireland, 2005.
- Neumann, L.; Čadik, M.; Nemcsics, A. An Efficient Perception-based Adaptive Color to Gray Transformation. In Proceedings of the Third Eurographics conference on Computational Aesthetics in Graphics, Visualization and Imaging, Banff, AB, Canada, 20–22 June 2007.

25. Yu, J.; Li, F.; Lv, X. Contrast Preserving Decolorization based on the Weighted Normalized L1 Norm. *Multimed. Tools Appl.* **2021**, *80*, 31753–31782. [[CrossRef](#)]
26. de Queiroz, R.L.; Braun, K.M. Color to Gray and Back: Color Embedding into Textured Gray Images. *IEEE Trans. Image Process.* **2006**, *15*, 1464–1470. [[CrossRef](#)]
27. Wu, T.; Toet, A. Color-to-grayscale conversion through weighted multiresolution channel fusion. *J. Electron. Imaging* **2014**, *23*, 043004. [[CrossRef](#)]
28. Yu, N.; Li, J.; Hua, Z. Decolorization Algorithm based on Contrast Pyramid Transform Fusion. *Multimed. Tools Appl.* **2022**, *81*, 15017–15039. [[CrossRef](#)]
29. Grundland, M.; Dodgson, N.A. Decolorize: Fast, Contrast Enhancing, Color to Grayscale Conversion. *Pattern Recognit.* **2007**, *40*, 2891–2896. [[CrossRef](#)]
30. Kim, Y.; Jang, C.; Demouth, J.; Lee, S. Robust Color-to-Gray via Nonlinear Global Mapping. In Proceedings of the ACM Siggraph Asia, Yokohama, Japan, 16–19 December 2009; pp. 1–4.
31. Alsam, A.; Kolås, Ø. Grey Colour Sharpening. In Proceedings of the Color Imaging Conference, Scottsdale, AZ, USA, 6–10 November 2006.
32. Smith, K.; Landes, P.E.; Thollot, J.; Myszkowski, K. Apparent Greyscale: A Simple and Fast Conversion to Perceptually Accurate Images and video. In *Computer Graphics Forum*; Wiley Online Library: Hoboken, NJ, USA, 2008.
33. Zhang, X.; Liu, S. Contrast Preserving Image Decolorization Combining Global Features and Local Semantic Features. *Vis. Comput.* **2018**, *34*, 1099–1108. [[CrossRef](#)]
34. Liu, S.; Zhang, X. Image Decolorization Combining Local Features and Exposure Features. *IEEE Trans. Multimed.* **2019**, *21*, 2461–2472. [[CrossRef](#)]
35. Zhao, R.; Liu, T.; Xiao, J.; Lun, D.P.; Lam, K.-M. Invertible Image Decolorization. *IEEE Trans. Image Process.* **2021**, *30*, 6081–6095. [[CrossRef](#)]
36. Cai, B.; Xu, X.; Xing, X. Perception Preserving Decolorization. In Proceedings of the 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018; IEEE: Piscataway, NJ, USA, 2018.
37. Zhang, L.; Wan, Y. Color-to-Gray Conversion based on Boundary Points. In Proceedings of the 2022 11th International Conference on Communications, Circuits and Systems (ICCCAS), Singapore, 13–15 May 2022; IEEE: Piscataway, NJ, USA, 2022.
38. Yu, J.; Li, F.; Hu, X. Two-Stage Decolorization Based on Histogram Equalization and Local Variance Maximization. *SIAM J. Imaging Sci.* **2023**, *16*, 740–769. [[CrossRef](#)]
39. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [[CrossRef](#)]
40. Mittal, A.; Moorthy, A.K.; Bovik, A.C. No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Process.* **2012**, *21*, 4695–4708. [[CrossRef](#)]
41. Peli, E. Contrast in Complex Images. *JOSA A* **1990**, *7*, 2032–2040. [[CrossRef](#)]
42. Lin, W.; Kuo, C.-C.J. Perceptual Visual Quality Metrics: A Survey. *J. Vis. Commun. Image Represent.* **2011**, *22*, 297–312. [[CrossRef](#)]
43. Damera-Venkata, N.; Kite, T.D.; Geisler, W.S.; Evans, B.L.; Bovik, A.C. Image Quality Assessment Based on a Degradation Model. *IEEE Trans. Image Process.* **2000**, *9*, 636–650. [[CrossRef](#)] [[PubMed](#)]
44. Gu, K.; Zhai, G.; Lin, W.; Yang, X.; Zhang, W. No-Reference Image Sharpness Assessment in Autoregressive Parameter Space. *IEEE Trans. Image Process.* **2015**, *24*, 3218–3231. [[PubMed](#)]
45. Avidan, S.; Shamir, A. Seam Carving for Content-Aware Image Resizing. In Proceedings of the ACM Siggraph, San Diego, CA, USA, 4–5 August 2007.
46. Mikolajczyk, K.; Zisserman, A.; Schmid, C. Shape Recognition with Edge-based Features. In Proceedings of the British Machine Vision Conference (BMVC'03), Norwich, UK, 9–11 September 2003.
47. Čadík, M. Perceptual Evaluation of Color-to-Grayscale Image Conversions. In *Computer Graphics Forum*; Wiley Online Library: Hoboken, NJ, USA, 2008.
48. Harris, C.R.; Millman, K.J.; Van Der Walt, S.J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N.J. Array Programming with NumPy. *Nature* **2020**, *585*, 357–362. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.