CCIT Report #508

November 2004

Enhancement of Color Images By Efficient Demosaicing

Liron D. Grossmann[*] and Yonina C. Eldar[†]

November 4, 2004

Abstract

We propose an efficient method for reconstructing a full-color image from its partially sampled version. The suggested algorithm is non-iterative and is based on the properties of the human visual system. While most state-of-the-art algorithms invest a great deal of computational effort in the enhancement of the reconstructed image to overcome the color artifacts, we focus on eliminating most of the color artifacts in the initial stages of the algorithm. This approach enables us to correct a specific kind of color artifact, caused by the aliasing involved in the sampling process. Our correction method is based on the relatively low sensitivity of the human eye to high transitions of the hue component of a color image. We tested our algorithm on several problematic images and found it to often be significantly superior to state-of-the-art algorithms, without consuming high computational power.

Index Terms—Demosaicing, interpolation, color spaces.

1 Introduction

It is well known that in order to display a digital color image, one needs to specify only three values for each pixel, i.e. the red, green, and blue values (often called the color channels) at the corresponding location. Due to technology limitations, however, most charge-coupled-device (CCD) cameras provide a single value for each pixel, that is, either red, green, or blue. The resulting image is often called a "mosaic". The problem of demosaicing is to reconstruct the original, full-color, image from its subsampled version, and is usually regarded as an interpolation problem.

Many algorithms have been proposed over the last decade for the demosaicing problem; see, e.g. [1], [2], [3], [4], [5], [6], [7] to mention a few. These algorithms can be broadly divided into three categories. In the first category, standard interpolation techniques, such as nearest neighbor, bilinear, or cubic interpolation are used to interpolate the missing pixels. Such methods are computationally efficient, but result in severe color artifacts. The algorithms belonging to this category completely ignore the color attributes of the image in the interpolation process, explaining their degraded visual quality.

The second category of algorithms takes into account the cross-correlation among the three color channels. Common use of this correlation is to assume that the hue changes smoothly between

[*]Department of Electrical Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel. E-mail: lirongr@tx.technion.ac.il.

[†]Department of Electrical Engineering, Technion - Israel Institute of Technology, Haifa 32000, Israel. E-mail: yonina@ee.technion.ac.il.

neighboring pixels, and therefore interpolation is performed on the color ratios instead of on the channels themselves. However, the assumption of constant hue may not hold around certain kinds of edges. As a result, most algorithms also incorporate an edge-detection mechanism to avoid interpolating over edges. Examples of algorithms belonging to this category are [2], [5], [7]. These algorithms lead to higher quality images than the algorithms in the first category, demonstrating that combining knowledge of the human visual system can improve the quality of the reconstructed image. Nevertheless, most of the above mentioned algorithms still yield poor results in certain problematic regions of the image.

The third category of algorithms includes the algorithms whose visual results are considered the best. Common to all of them is that they start with an initial estimation of the original image (such as bilinear interpolation), after which a correction procedure is applied. This procedure is intended to eliminate most of the color artifacts caused by the initial interpolation scheme. Examples of such algorithms are [1], [3], [4], [8], [9], [10]. The correction methods are based on additional properties of color images, other than the inter-channel correlation. This, in turn, improves their quality over the other two mentioned categories. Usually, the correction is performed in another color space, so that the inter-channel correlation is not exploited directly in the Red-Green-Blue (RGB) space. Typical methods of correction involve the separation of luminance and chrominance in the Fourier domain. Such separation is accomplished by a set of filters, applied to the initially estimated image. Different processing procedures are then performed on the luminance and chrominance components, after which the image is transformed back to its RGB representation. A statistical approach for processing the luminance and chrominance was suggested in [3], [11], [12]. Despite the good quality obtained by this class of algorithms, there are still several typical artifacts that cannot be completely eliminated by them, no matter how sophisticated the correction method is. The reason is that the initial estimation typically gives rise to many color artifacts, which are difficult to compensate for using a computationally efficient algorithm. Moreover, a poor initial interpolation of the original image may threaten the stability of the entire algorithm, especially the iterative algorithms. If the correction of a region depends on other regions, which were poorly estimated, then the correction of this region will also lead to degraded estimation, creating a snowball that can impair the entire reconstruction.

It is worth noting, that with the increase in the resolution of digital cameras, the problem of demosaicing becomes less and less difficult to handle, and rather simple techniques are acceptable. There may, however, still appear rapid varying regions, in which even high resolution will not eliminate the aliasing caused by the subsampling, explaining the continuing interest in finding better algorithms for demosaicing. What is lacking, therefore, is a demosaicing method, which is not too complex (both conceptually and computationally), but at the same time can reduce most of the common problems arising from the subsampling of the color channels. It is clear that a good demosaicing algorithm should be driven by the interaction between color image attributes and the human visual system. The more knowledge we can collect and exploit, the better the final outcome will be. In addition, existing algorithms demonstrate that a correction method is a useful component in the demosaicing chain. Nevertheless, the input to this stage should not contain too many visual artifacts. Thus, the computational effort should be split between the initial estimation and the correction method.

In this paper we present an algorithm which takes into account the properties of color images, and is also computationally efficient. The algorithm we develop follows the same strategy as the algorithms in the third category. Indeed, it starts with an initial estimation of the color image, and then applies a correction method to deal with the visual artifacts. The main difference in comparison with existing algorithms is that instead of investing most of the computational effort in the correction stage, we develop a good initial interpolation, thereby reducing the number of color artifacts that need to be overcome in the correction stage. Our method of initial estimation of the color image from its subsampled version classifies the possible sources of color artifacts. This enables us to keep track of these artifacts as they proceed through the algorithm and eliminate them as soon as possible. As we show, the proposed mechanism for controlling the generation of color artifacts allows for a great simplification of the enhancement process, leading to low computational demand. Furthermore, each stage in our algorithm exploits the properties of the human color vision system, explaining the success of eliminating most of the color artifacts, which are common in demosaicing algorithms.

The paper is organized as follows. In Section 2, we introduce the problem of demosaicing in a way suitable for the development of the algorithm. Section 3 briefly reviews fundamental properties of the human visual system relevant to our algorithm. The main steps of the algorithm are outlined in Section 4, and implementation details are considered in Section 5. Comparisons with existing demosaicing methods are discussed in Section 6[1].

2 Problem Formulation

The problem of demosaicing is that of reconstructing a multi-color image from a single array image, which is a subsampled version of the original image. The most popular single array pattern is the Bayer Color Filter Array (CFA), which is shown in Fig. 1, and will be the pattern considered here [13]. In this pattern, 50% of the pixels are green, 25% are red, and 25% are blue. The reason for

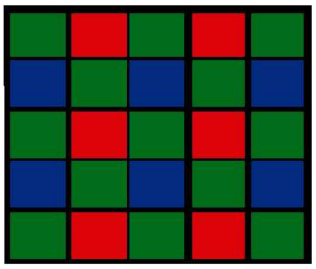


Figure 1: The Bayer array pattern

this choice of sampling rates will be explained later in Section 3.

As noted in the introduction the demosaicing problem can be regarded as an interpolation problem. The solution method should yield an outcome, that is as close as possible to the original image. By close, we mean subjectively close, as there is no universal metric for images, which is in accordance with the human visual system [14]. Therefore, throughout the paper, when we refer to a "good" quality image, we mean subjectively good as seen by the viewer of the image. If the consumer of the image is a person, then a good solution should rely on the interaction between color and the human visual system. Based on prior knowledge concerning color images, we tested

[1]The images in this paper are best viewed on a monitor. An on-line version is available at http://www.ee.technion.ac.il/Sites/People/YoninaEldar/ under Journal Publications

several hypothesis by performing many visual experiments. Some of them were verified and used in the algorithm, to obtain good visual results. For example, handling regions with rapid changes is one of the main obstacles in every demosaicing algorithms. Reasoning out its possible origins by some experiments, we developed a technique to deal with such problematic regions which we discuss in detail in Section 4. The results in Fig. 2 demonstrate the power of this technique, where we compare it with two other interpolation methods. As can be seen, our method is visually the best in the regime shown.

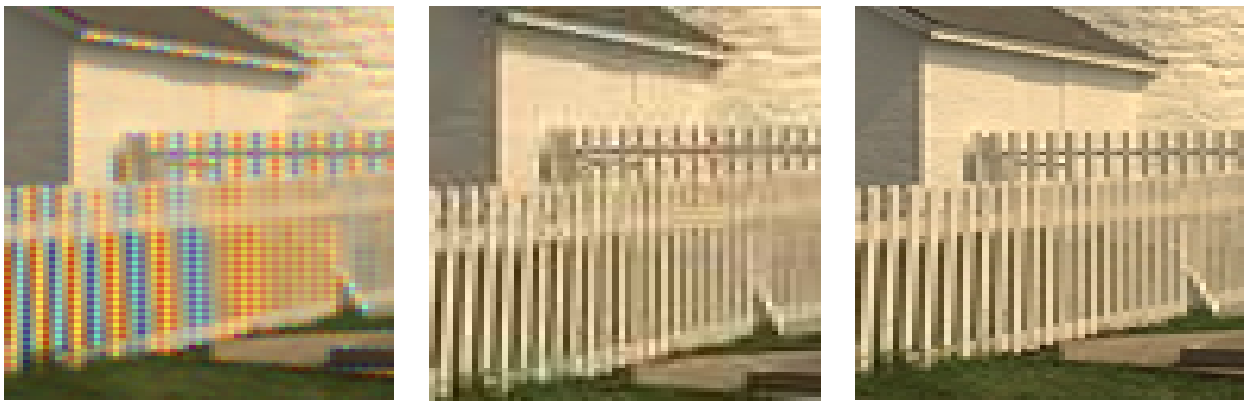


Figure 2: An example of three interpolation methods on a problematic region (a) bilinear method (b) Kimmel's method [1] (c) our method.

As mentioned in the introduction, we also aim at achieving a low complexity solution. We will shortly see, that combining known facts about the human visual system, not only results in a visually better image, but also leads to a more efficient algorithm.

The structure of the algorithm we propose consists of two main stages: an adaptive initial interpolation of the color image from the partially sampled one, followed by a method which corrects the visual artifacts that remain in the image. The initial estimation procedure first interpolates the green channel by using a classification of the missing pixels into regions. The classification allows us to apply different interpolation schemes according to the nature of the neighborhood of each missing pixel. In particular, it helps us locate the highly problematic regions in the image, caused by aliasing of the green channel itself. These regions are further classified into horizontal and vertical regions, creating two versions of the green channel: A horizontal version in which all the aliased regions are interpolated in the horizontal direction, and a vertical version, which has the corresponding regions interpolated in the vertical direction. It turns out that such a separation between horizontal and vertical direction is a key to eliminating most of the color artifacts in an efficient way, as the positions of these regions are saved and their right direction is determined in the correction stage. Adaptation schemes were previously suggested in [6], using a template matching technique, and a horizontal and vertical gradient method was proposed in [15]. We will further develop these techniques to achieve higher quality images.

After the green channel is interpolated, the red and blue channels are interpolated exploiting the inter-channel correlation, that is using the estimate of the green channel to determine the values of the red and blue pixels. Since we have two green estimates, horizontal and vertical, we must interpolate the red and blue channels twice, once for each direction. A similar idea to that in [5] is used to interpolate the red and blue channels on the basis of the green channel. The resulting two color images are passed through the correction stage, in which color artifacts are reduced. The goal of this step is to determine the right direction of pixels for which we could not deduce any information from their neighborhood due to the aliasing effect. The exact locations of these pixels are known, thanks to the classification performed in the green estimation stage. The main key for detecting color artifacts relies on the relative smoothness of the chrominance component of each region, and therefore requires a transformation to the YIQ space. Assuming slow varying chrominance changes, each problematic region (which was originally classified in the green estimation stage) is determined to be either horizontal or vertical, depending on its relative smoothness in the chrominance (I and Q) components. The resulting color image has a smooth chrominance component, which is in accordance with the low sensitivity of the human eye to rapid color changes. Note that separating luminance and chrominance has already been suggested in several algorithms, but as mentioned in the introduction, this separation typically relies on a poor estimate of the original color image. In [3], for example, the I and Q components are low-passed filtered, while most of the computational resource is spent on the estimation of the Y component. Such a method is justified by the assumption that the eye is more sensitive to high frequencies in the luminance than in the chrominance.

Our correction technique relies on the I and Q components, and succeeds in correcting most of the color errors. The success depends on our ability to estimate the red, green and blue channels before the transformation to the YIQ space is performed. A YIQ image, whose corresponding RGB image is poorly estimated, will not yield the desired results, as can be seen in many of the state-of-the-art algorithms. In our algorithm, the YIQ space is just an error correction stage, not the main estimation stage, as will soon become clear. In fact, we use the YIQ space to correct very specific kinds of color artifacts, caused by aliasing in the green channel itself.

After developing our algorithm, we became aware of [16], in which a procedure for correcting horizontal and vertical artifacts is also proposed. The suggested method also detects the aliased regions, and compares their horizontal and vertical interpolations. However, as will be shown, our method is different in that we cluster the aliased regions to smaller groups than in [16], and our comparison criterion relies on the smoothness of the I component rather than on the variance of the color ratios. In addition, the method in [16], is only a correction method, while in our algorithm the correction stage is also based on our previous stages.

Before proceeding to the detailed description of our algorithm, in the next section we consider several basic properties of the human color vision system, which we will exploit in the development of the algorithm.

3 The Human Vision System

We now review some relevant facts about the human visual system, the consumer of the image. More details can be found in [17], [18], [19].

A digital color image is usually represented by three matrices of equal size, called its components, or channels. Each pixel is, therefore, characterized by three values, which can be viewed as the components of a vector in a three dimensional space, often called the color space. There are several reasons for this three-channel representation. One reason can be found in the trireceptor theory.

Loosely speaking, the theory claims that the light radiated from an object is translated to a visual stimuli by a set of photo-receptors located on the retina of the eye. A special class of such receptors, called cones, are responsible for our early stage color vision. In this class there exist three types of cones, having different response curves in the red, green, and blue portions of the visible spectrum, as can be seen in Fig. 3. Therefore, all the information that is relevant to the eye is described

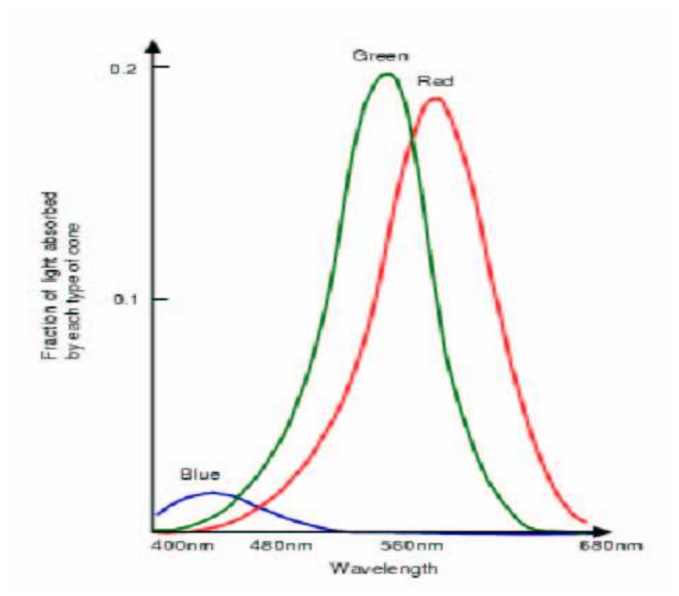


Figure 3: The response curves of the three types of cones.

by three values, corresponding to the red, green, and blue channels. Such a representation of an image is referred to as RGB (Red, Green, Blue), where each pixel is a triplet of the form (r, g, b), corresponding to the relative amount of the three colors at the position of the pixel. This model is often used to display images, and is also the model by which the mosaic image is represented. A well known fact about the RGB space is that high cross-correlation exists among the three channels. In fact, the three channels usually have the same pattern, except for possibly different intensity levels. The similarity among the three channels is easily verified in Fig. 4, which shows the red, green, and blue components, represented as grey-level images. This correlation is exploited in many algorithms in different ways, to predict values of pixels in one channel, based on those of another channel. We follow a similar approach to that taken in [5], which we review in Section 5.

Another reason for the three-channel model is based on the fact that the human eye distinguishes dominant wavelength (hue), purity (saturation) and luminance (brightness). It is therefore sufficient to describe each pixel by its luminance, hue and saturation. The HLS (Hue, Lightness, Saturation), and HSV (Hue, Saturation, Value), are examples of such color spaces.

In many color applications, however, neither the RGB nor the HSV are convenient representations. Standard methods of image processing, such as interpolation, enhancement and denoising, may lead to severe visual artifacts when applied in the RGB space. This is mainly due to the special interaction between the three color channels, which is usually misused. The HSV space, on the other hand, is a very convenient representation for color processing, although the transformation

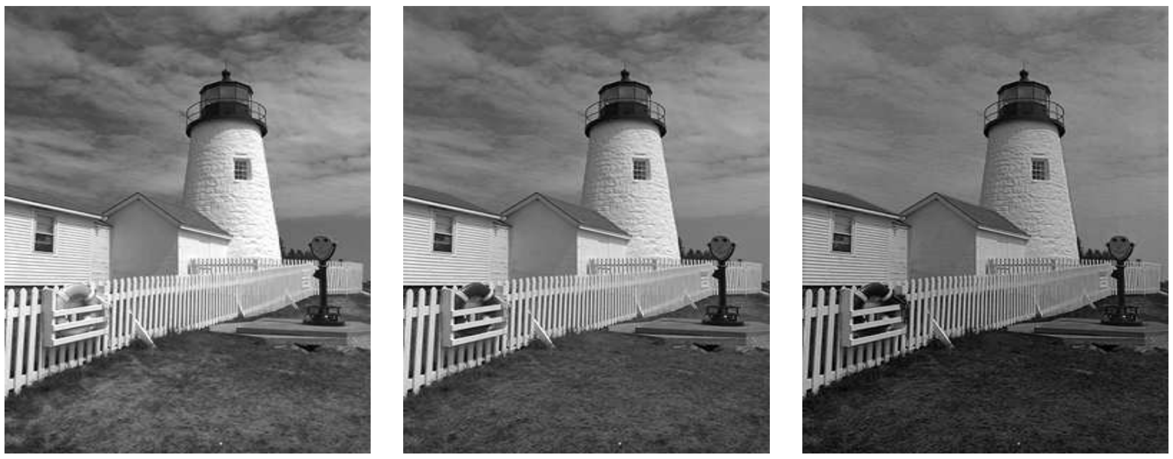

(a) (b) (c)

Figure 4: The three color channels of a color image (a) red channel (b) green channel (c) blue channel.

between the RGB and the HSV models is often computationally demanding, due the nonlinear relation between the two spaces. Since we aim at achieving good results, while maintaining low computational power, a tradeoff is required. Among the many existing color models, we choose to work with the YIQ space [19].

The history of the YIQ space goes back to the days of analog television transmission [17]. Before color television was invented, only one signal, called Y, corresponding to the luminance of the image, was transmitted to the television receivers. After the invention of color television, information concerning the chromatic attributes of the images, had to be transmitted as well. Since three components are required to fully specify a color image, three signals were used for transmission of a color image. The luminance signal, Y, remained unchanged, for the sake of compatibility with black and white receivers. The two additional signals, initially denoted by U and V, represented the color characteristics of the image. Compatibility to the black and white receivers was accomplished by simply ignoring the color related signals in those receivers. Later, the U and V were changed to I (in-phase) and Q (quadrature) using a simple rotation formula. It was found that the I and Q components are more adapted to the human visual system than the U and V components.

An appealing property of the YIQ space is that in this space we can separate between the chromatic and achromatic behavior of the image. It turns out, that the human eye is more sensitive to abrupt changes in the luminance, than to the changes in the chrominance [18]. As a consequence, we expect a smooth behavior in the color attributes of the image, namely, in the I and Q components of the image. In order to validate this assumption, the I component is shown to the left of its corresponding color image in Fig. 5. Concentrating on the fence part, it is easy to see the connection between color artifacts observed in the RGB space, and sharp transitions in the I component (the bright white patch in the fence on the dark background). The fence part in Fig. 5(a) exhibits very rapid color changes, which in turn, generates a large jump in the corresponding I component, as can be seen in Fig. 5(b). Fig. 5(c), on the other hand, does not show such a strange festival of

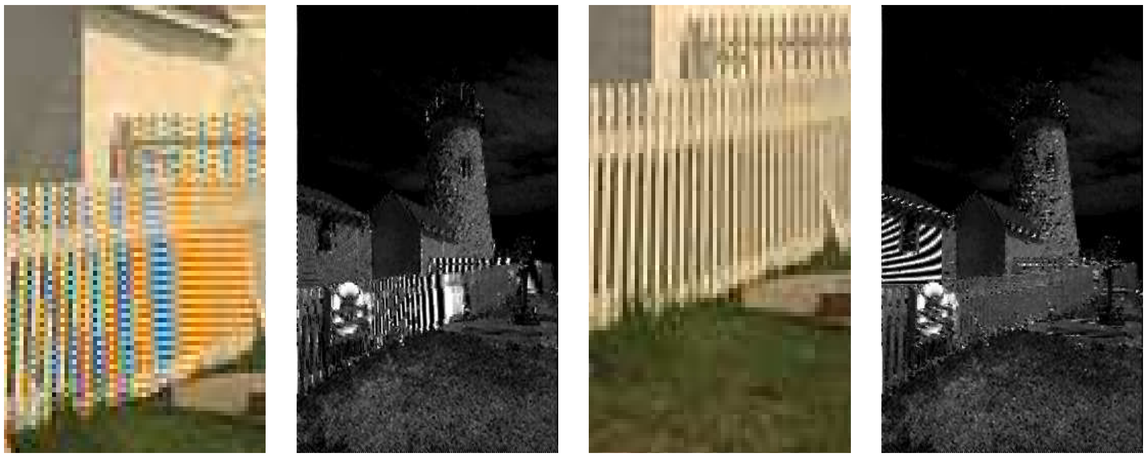


Figure 5: The I component of the color figure (a) horizontal interpolation of the red, green, and blue channels (b) the I component of (a) (c) vertical interpolation of the red, green, and blue channels (d) the I component of (c).

colors, and a regular behavior of the I component is therefore inspected in Fig. 5(d). The opposite situation occurs in the house region.

The smooth chrominance assumption will be at the heart of our correction method. Note that ignoring the Y component is useful only when detecting color artifacts, but usually the Y component should be treated more carefully than the I and Q components, as far as abrupt changes of the luminance are considered. In image coding, for example, this fact is expressed by allocating more bits to the the Y component than to the I and Q ones (see, e.g [20]). In our algorithm, however, the Y component plays a minor role, since we are mainly concerned with the color behavior. Thus, corrections in the luminance will follow from the chromatic information.

Another noticeable advantage of the YIQ space over other spaces, is its simple relationship with the RGB space. An RGB image is linearly transformed to its YIQ representation by

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.2755 & 0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \qquad (1)$$

The inverse transformation has a similar structure. As a result, we do not loose much in terms of complexity, when we convert the image to the YIQ space, while at the same time, we can gain a lot in terms of quality.

From (1) we can infer that there is a high correlation between the green channel and the luminance component of the image. We can therefore combine several properties of human achromatic vision into the green estimation process, that is we estimate the green as if it were a grey-level image, i.e. a luminance image. One known fact is that the human eye distinguishes different intensity levels in a logarithmic manner, as suggest by the Weber law [19]. Weber law describes a general relationship between stimulus intensity and its perceived magnitude, and is applicable in many physiological phenomena, not only in vision [19]. Denoting the initial magnitude of the stimulus (the intensity of light in this case) by $\mathbf{L}$ and the change required to create a sensation by $\mathbf{\Delta L}$, Weber law states that the relation $\frac{\mathbf{\Delta L}}{\mathbf{L}}$ is constant. This law provides us with a way of discriminating between two intensity levels of adjacent pixels, and will be our classifying mechanism in the green estimation stage. A more accurate version of the law was developed by Fechner [21], which showed a logarithmic relationship between the two changes. Nevertheless, we will use the Weber law as is, but will adapt the constant. The adaptation of the constant is easier to implement and gives very accurate results.

4 A Short Overview of the Algorithm

We are now ready to take a first glance at the algorithm. A block diagram of our algorithm is shown in Fig. 6. This section provides an overview of each of the boxes in the synopsis. Implementation details will be given in Section 5.

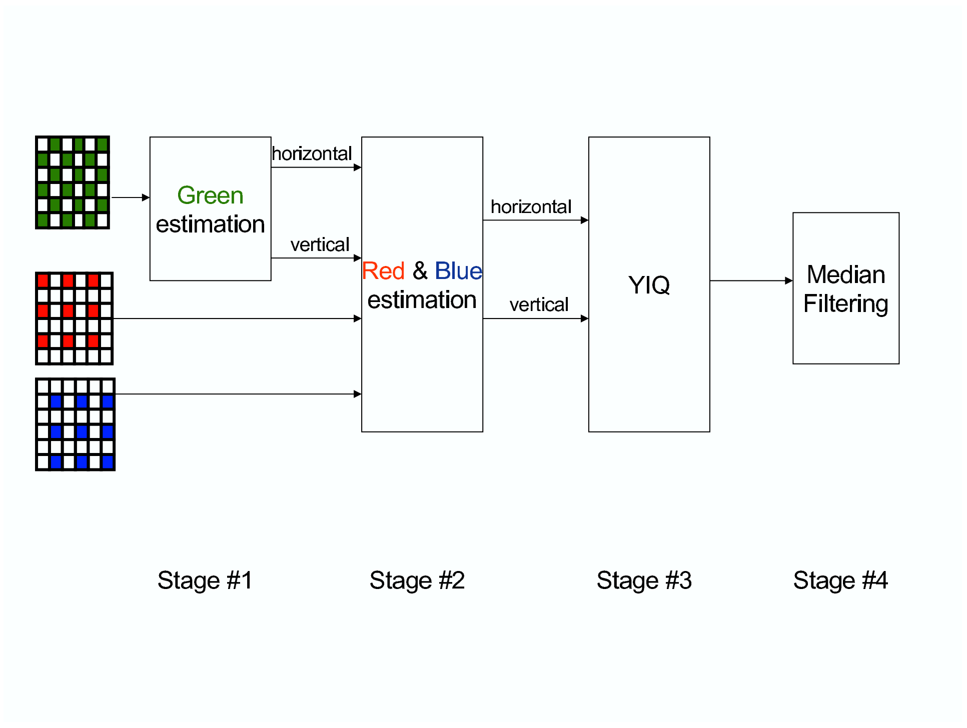


Figure 6: A block diagram of the algorithm.

In order to facilitate the description of the algorithm, figures corresponding to the inputs and outputs of each stage will accompany its description. Moreover, to emphasize the power of our algorithm, we chose the Lighthouse image, which is considered to be a very difficult image to reconstruct from its mosaic.

We now outline the basic steps of the algorithm.

4.1 Green Estimation

The input to this stage is the subsampled version of the green channel, as shown in Fig. 7. In

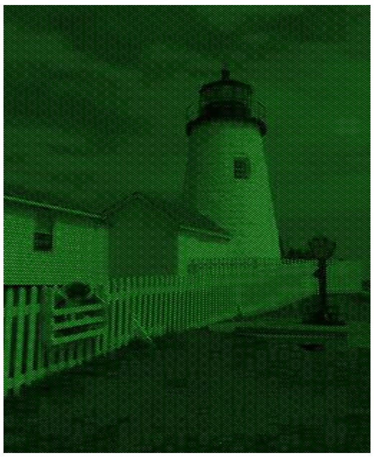


Figure 7: The mosaic green channel.

order to "fill in" the missing pixels, an adaptive interpolation scheme is performed on the input. Adaptation is achieved by classifying the regions to which each missing pixel belongs to, where with each class, we associate a different interpolation method. A region or a neighborhood of a missing green pixel is shown in Fig. 8.

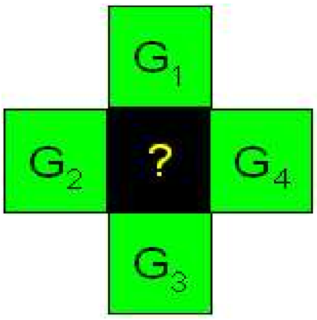


Figure 8: The neighborhood of a missing pixel.

In our development, we consider three possible classes. The classification criterion is the degree of smoothness of the missing pixel's neighborhood. The exact definition of the degree of smoothness is deferred to the next section. Here, we only explain how each one of the classes is handled.

A "smooth" neighborhood of a missing pixel results in averaging the values of its neighbors. Pixels with "smooth" neighborhood are said to be class A pixels.

In a "less smooth" neighborhood the missing pixel is interpolated using a larger neighborhood. The enlarged environment consists of the neighbors of the missing pixel in addition to pixels from the other two channels, which are spatially close to it. Fig. 9 shows the pixels, which are included

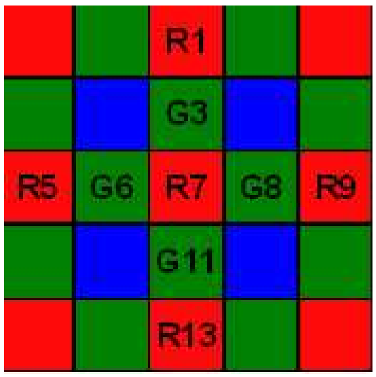


Figure 9: The mosaic image, where the marked pixels are used in the interpolation of class B.

in the interpolation of this class (The numbering will be used in the detailed explanation of the interpolation in Section 5). Note that it is here, where we exploit the inter-channel correlation for the first time. Pixels whose neighborhood is “less smooth” constitute class B.

Class C is characterized by the property, that neither the neighboring pixels, nor those of the other channels, can help in estimating the value of the desired pixel. Such a situation is the result of the aliasing created in the subsampling of the green channel itself. The aliasing phenomenon is even more disastrous in the red and blue channels, since their subsampling is lower than that of the green. We are therefore led to a situation, where the inter-channel correlation can no longer be exploited in the interpolation process. This is why we treat pixels whose neighborhood is neither smooth nor “less smooth” as a special class. Figure 10 shows the effect of aliasing in the green channel. It is

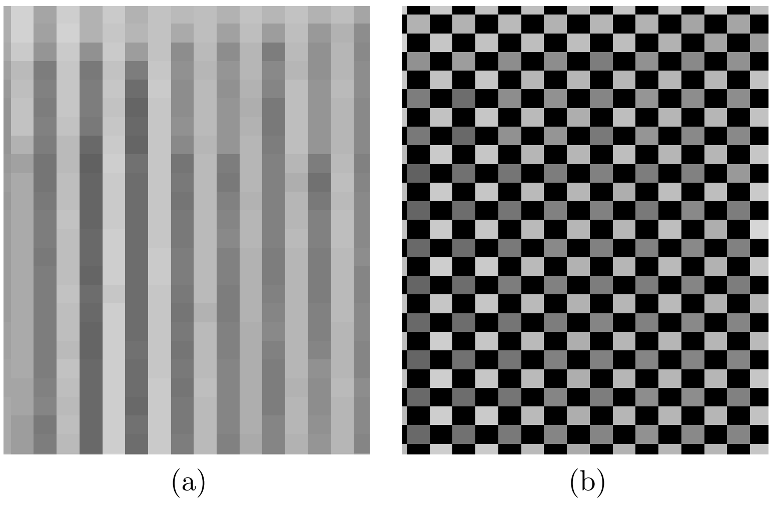


Figure 10: A region in the green channel that is aliased after subsampling (a) original part of the fence (b)the corresponding region in the mosaic image.

clear that the right direction of the fence can no longer be inferred from the subsapmled green.

In our algorithm, we try to recover only horizontal and vertical frequencies that were aliased.

This choice of reconstruction stems from two reasons: the first is driven by the simplicity of implementation, while the other is a psychophysical one. Using only two possible directions for interpolation reduces the complexity of the comparison and selection process in the correction stage. In addition, it is a known fact that the human eye is most sensitive to edges in the horizontal and vertical directions, and less sensitive to diagonal edges [22]. Practically, this means that in our interpolation scheme pixels in class C will either be an average of their left and right neighbors, or the average of their upper and lower neighbors. Since no useful information exists at this stage as to the real orientation of the pixels in class C, we postpone their interpolation to a later stage. In the meantime, we allow the pixels of this class to take on two values. One value corresponds to the average of the horizontal neighbors of the missing pixel, while the other one to the average of the vertical ones. As a consequence, we end up with two versions of the reconstructed green. One, called the horizontal image, has all its aliased regions interpolated in the horizontal direction. The second image, the vertical one, has them all vertically interpolated. The pixels belonging to the other two classes remain the same in both images.

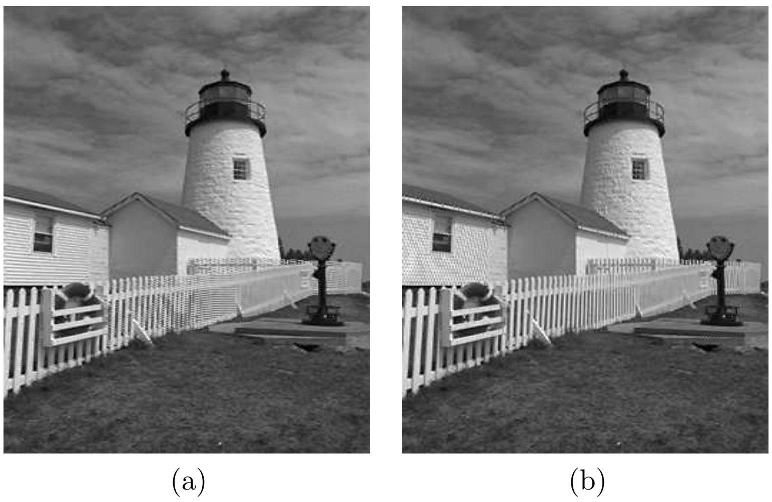


Figure 11: The outputs of the green estimation stage (a) horizontal green channel (b) vertical green channel.

In summary, two green channels are the output of this stage. Figure 11 shows the outputs for the input given in Fig. 7.

4.2 Red and Blue Estimation

The input to this stage are the two green channel images from the first stage, shown in Fig. 11, and two subsampled versions of the red and blue components, shown in Figure 12. Using the cross correlation among the channels, the differences between the red and green are interpolated, after which they are added to the partially sampled red channel to yield its estimate. A similar procedure is carried out for the blue channel. This process is performed twice for each channel, resulting in two color images, shown in Fig. 13. Note that one image has all its aliased regions interpolated horizontality, while the other has them all in the vertical direction. Thus, at the end of this stage, we have a full-color image in the aliased-free regions, and all that is left is to determine the right

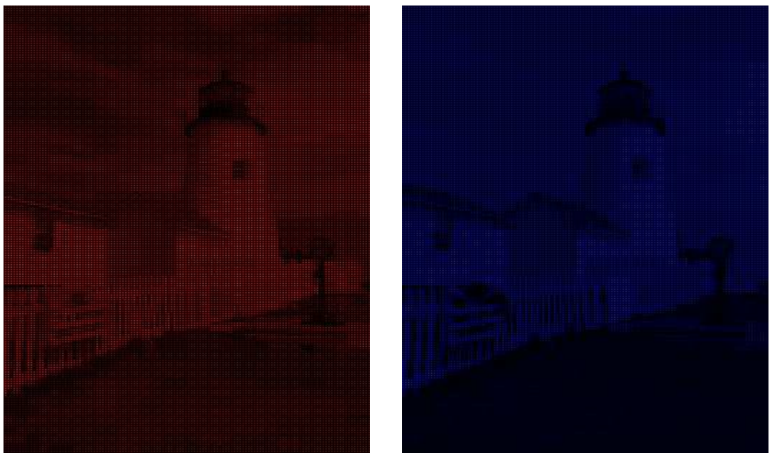


Figure 12: The inputs to the second stage (a) mosaic red (b) mosaic blue.

interpolation method in the aliased regions.

4.3 YIQ transformation

To combine the two color images from the previous stage, we now use the YIQ transformation. Specifically, we use the YIQ space as a tool to select for each region, which interpolation scheme, i.e horizontal or vertical averaging, results in a smoother chrominance component. Testing the chrominance smoothness is easily accomplished in the YIQ space, due to the separation between luminance and chrominance, as opposed to the RGB space. Considering an aliased region in the I component, we compare the relative smoothness of its horizontal and vertical versions, and finally select the smoother one. By selecting, we mean replacing the corresponding triplet of Y, I and Q components (horizontal or vertical) in the final output. The reason for comparing only the I component stems from the fact that, empirically, the Q component was found to be less helpful in the determination of the smoothness.

In Fig. 14, we present the output Lighthouse image of this stage in its RGB representation.

4.4 Median Filtering

The resulting image from last stage may still contain splotches, that are very annoying to the human eye. To eliminate these false color points, we perform median filtering on the I and Q components of the image. The final Lighthouse image is shown in Fig. 15.

5 Implementation Details

In this section, we will focus on the parts, which were deliberately omitted in the previous section for the sake of simplicity. Specifically, in Section 5.1 we describe the exact method for discriminating between pixels, as well as the nature of the three classes, used in the classification of the green channel. In Section 5.2 we consider the three interpolation schemes, which are applied to each

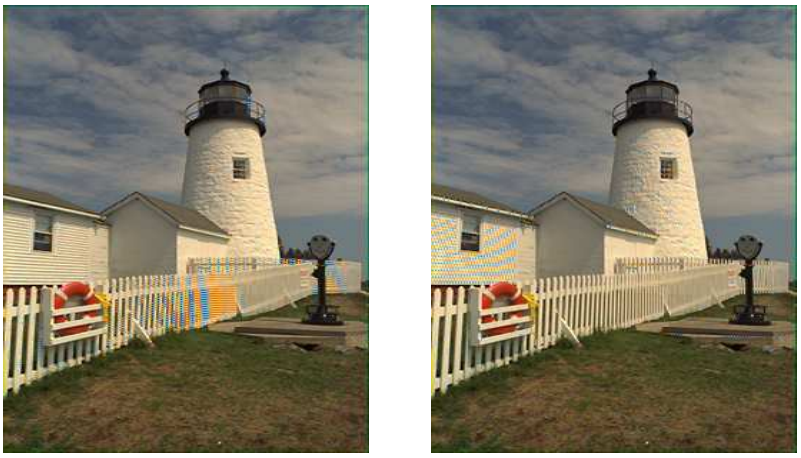

(a) (b)

Figure 13: The output of the red and blue estimation stage (a) horizontal interpolation (b) vertical interpolation.

class. Section 5.3 provides a detailed description of the red and blue interpolation, using the differences between the channels. Finally, the exact method for comparing and selecting an aliased region in the YIQ domain is described in Section 5.4.

5.1 Discrimination and Classification

Recall that our classification of pixels in the green estimation stage was based on the smoothness of the neighborhood of each green pixel. Each missing green pixel has four neighbors as depicted in Fig. 8.

In order to define a smoothness criterion, we first need a way to distinguish between two adjacent pixels. Since the green channel is highly correlated with the luminance, its seems natural to use a mechanism that discriminates two grey-level pixels. The Weber law, as described in Section 3, models the human eye's mechanism for discriminating two intensity levels, which are closely located. Thus, we shall adapt the Weber law as our discriminating tool. Formally, given any two pixels, we denote their intensity values (in our case it is their green values) by G_1 and G_2, and compute the ratio

$$\frac{G_2 - G_1}{G_1}. \tag{2}$$

If this ratio is above a predetermined threshold (which depends on the intensity levels), then we conclude that there is a noticeable difference between the pixels. That is, we can distinguish between the two pixels. Depending on intensity G_1 we decide which constant to choose. The adaptivity of the constant is due to empirical results that show the non-linear relationship between the background G_1 and the difference $G_2 - G_1$.

By performing several visual experiments we divided the dynamic range of the intensity levels into a number of groups, to which we associated different constants.

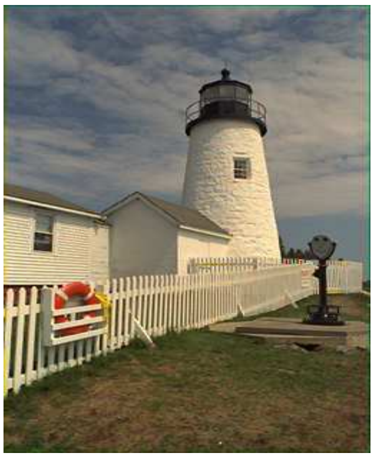


Figure 14: The output of the YIQ stage.

5.2 Three different interpolation methods

Now that we have a way to distinguish between two pixels, we can introduce the three classes. Figure 16 shows the three types of classes we consider. In this figure, pixels of the same brightness are considered to be indistinguishable, except for the gray pixels which may assume any value. Note that in each class there may be more than one possible configuration of a neighborhood. The same interpolation is used for all neighborhoods belonging the same class. The classes are characterized by the interpolation method that is performed on each one.

Depending on its class, each missing pixel is estimated by one of the following methods. As in Fig. 8 we denote by G_1, G_2, G_3, G_4 the intensity levels of the pixels in the neighborhood. We denote the value of the estimated pixel by $\hat{G}$.

- In class A there is a high spatial correlation between adjacent pixels, therefore the interpolation method is designed to preserve the smoothness. Each missing pixel is replaced by average of its neighbors. If the neighborhood is completely smooth, that is, all of its pixels are indistinguishable (a test for similarity is performed using the Weber law), then

  $$\hat{G} = \frac{G_1 + G_2 + G_3 + G_4}{4}. \tag{3}$$

  If the neighborhood is partially smooth (such as the right one in class A of Fig. 16), then we use the interpolation

  $$\hat{G} = \frac{G_1 + G_2 + G_3}{3}. \tag{4}$$

- Class B contains all the pixels that are neither in class A nor in C. It is in this class that the inter-channel correlation can be exploited to obtain better visual results. Note that we could also use it in class A, but it is not necessary there, since the intra-channel correlation suffices. In fact, using the inter-correlation in class A, even decreases the visual performance. In class C the cross-correlation among channels is not helpful, since in this class all three channels are aliased. A simple approach to exploit the inter-channel correlation was proposed

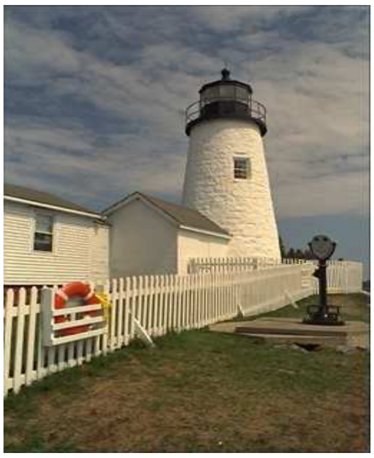
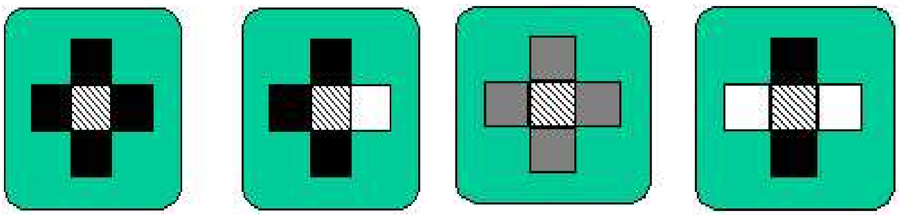


Figure 15: The final output of the algorithm.



Class A Class B Class C

Figure 16: The three classes used in the classification process.

in [5], and will also be used in the interpolation of the red and blue channels. In order to interpolate a missing green pixel, pixels of its neighborhood and the pixels from the other channels enter into the interpolation formula. Figure 9 shows the extended neighborhood of a missing green pixel which incorporates pixels from the other channels. The value of the estimate is computed by adding R_7 to the estimated difference $G_7 - R_7$, leading to the interpolation formula

$$\hat{G_7} - \hat{R_7} = \frac{(G_3 - \frac{(R_1+R_7)}{2}) + (G_6 - \frac{(R_5+R_7)}{2}) + (G_8 - \frac{(R_9+R_7)}{2}) + (G_{11} - \frac{(R_{13}+R_7)}{2})}{4}. \quad (5)$$

Each summand in the numerator is an estimate of the difference $G_7 - R_7$, and the final estimate is their average. A similar procedure is applied when the neighborhood contains pixels of the blue channel.

- The determination of class C pixels is deferred to later stages as explained in Section 4. Meanwhile, we create two images, both have the same pixels in class A and B, but differ in the orientation of pixels belonging to class C, interpolated as explained in Section 4.

5.3 Interpolation of the Red and Blue channels

The basic idea, which was used to interpolate pixels of class B in the green channel, is used to interpolate the red and blue channels. However, several modifications are required, because of the sparser sampling of the red and blue channels. In addition, using directly the method of [5], may create undesirable effects, such as smoothing edges, misalignment of edges due to phase shift between the given samples, and a zippering effect. In Section 6, all of these effects will be clearly seen.

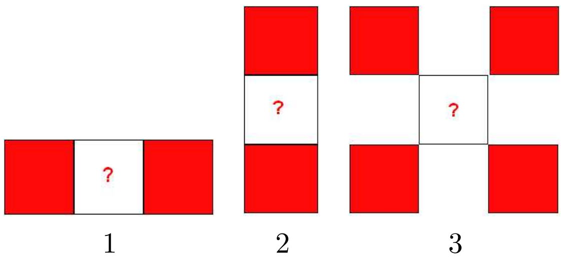


Figure 17: The red possible environments.

Figure 17 shows all three possible configurations of neighborhoods a missing red pixel can have. A similar situation occurs in the blue channel. In order to illustrate the idea suppose, for the moment, that we wish to interpolate the missing red, R_2, between R_1 and R_3 of Fig. 18. The situation corresponds to the environment 1 in Fig. 17.

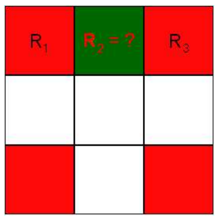


Figure 18: An example of the red interpolation.

At this stage, we have all the values of the corresponding green pixels, G_1, G_2 and G_3. A naive interpolation formula for the missing red is,

$$R_2 \;=\; G_2 + (R_1 - G_1). \tag{6}$$

Now, suppose further that there is an increase in the green level, i.e $G_2 > G_1$, while at the same time, the red pixel (which is missing) R_2 has its intensity decreases compared to R_1. Using the difference $R_1 - G_1$ instead of $R_3 - G_3$ in (6), causes an increase in the value of R_2, instead of a desired decrease. To infer the correct difference to use, we replace (6) by

$$R_2 = \begin{cases} G_2 + (R_1 - G_1) & \text{if } R_1 < R_3 \\ G_2 + (R_3 - G_3) & \text{otherwise.} \end{cases} \tag{7}$$

Environment 2 is treated in a similar way, while in Environment 3 we compare the two diagonal directions. The interpolation of the missing red pixels is performed twice - once based on the horizontal green image, and once based on the vertical green image. The same process is repeated for the blue channel.

5.4 The correction method

In this stage we are mainly concerned with correcting the aliased regions. These regions contain pixels whose green component belongs to class C, and since their positions are not changed in their YIQ representation, we continue to refer to them as class C pixels in the YIQ domain. Pixels of class C in the green channel may also affect the estimation of the red and blue channels in their 3×3 neighborhood. We therefore need to take an enlarged environment for each color pixel when we try to compare smoothness between horizontal and vertical estimation. Grouping neighboring class C pixels, using a 3×3 neighborhood, the aliased regions are formed into clusters. Each cluster has two versions, the horizontal and vertical one. Our next step is to compute the relative smoothness of each version and then to select the smoother one. The relative smoothness of each version is taken to be the ratio between the mean value of the pixels inside the cluster and the mean value of the pixels, which are outside the cluster, but are close to its boundary. Specifically, denote by C_h and by C_v the horizontal and vertical version of a cluster, respectively, and denote the average value of their I components by AvC_h and AvC_v respectively. We denote the averages of the horizontal and vertical environments by AvE_h and AvE_v, respectively. The selected cluster, denoted by C_s, is determined by the following rule:

$$C_s = \begin{cases} C_h & \text{if } \frac{AvC_h}{AvE_h} < \frac{AvC_v}{AvE_v} \\ C_v & \text{otherwise.} \end{cases} \tag{8}$$

The selection of a cluster involves inserting its pixels' values (that is the corresponding Y, I and Q values) into the final image. After the selection of each cluster is performed, we obtain one color image, which is a mixture of clusters, either horizontal or vertical, plus pixels belonging to class A and B (whose values were already determined in the previous stage).

6 Examples

We now compare our algorithm with several state-of-the-art algorithms. In our comparison, we consider the following reconstruction methods: bilinear [2] interpolation, a constant hue based algorithm [7], a color correction algorithm [9], a color gradient based algorithms [10], Kimmel's method [1] and ours.

6.1 Lighthouse

The Lighthouse image is a very difficult image to demosaic, due to the aliasing that occurs during the subampling of the green channel. The house and fence are excellent examples where this aliasing occurs. Figure 19 shows the original Lighthouse. Figures 20 -25 show the result of the above mentioned algorithms, and Fig. 26 shows our algorithm.

The bilinear interpolation method creates many color artifacts, especially in the aliased regions, i.e. the fence and the house. Better results are achieved by all of the human visual system based algorithms. Assuming smooth hue transitions improves the visual quality of the image in certain regions, but still results in sever color artifacts in the fence and house regions, where the aliasing of the three channels occurs. In all of the above images, we can see that the fence and house suffer from false color artifacts. In Figs. 23 and 25 the artifacts are less severe. In our outcome, Fig. 26 both the fence and the house have no color artifacts, and our result is the closest to the original.

6.2 Sails

The Sails image is also a benchmark. A problematic region is the number "14255". Figure 27 shows our outcome, which is very close to the original Sails. Kimmel's method and the variable gradients method are shown in Fig. 28. It can be seen that false colors are created by these methods, in particular across the edges of the digits.

6.3 Window

The Window image is another benchmark. A rapid transition region appears in the stem of the flower. An enlarged part of this image containing that region is shown in Fig. 29. The variable gradients method are shown in Fig. 30. It can be seen that splotches of various colors are created by these methods across the stem. In addition false colors appear in the shutters of window in the variable gradients method.

7 Conclusions

We have proposed a computationally efficient solution to the demosaicing algorithm. The solution is based on the interaction between color attributes and the human visual system. We have shown that our algorithm outperforms many of the existing algorithm. Our algorithm follows a similar pattern to most of the state-of-the-art algorithms, that is, it consists of two basic steps, an initial interpolation followed by an enhancement stage. Nevertheless, our approach differs from the existing methods in that it focuses on accomplishing a better initial reconstruction of the image, rather than a better enhancement method. This strategy has proven to be successful in both preventing most of the color artifacts, and in controlling the inevitable ones. Since the basic operations in the suggested initial estimation are not complex, our algorithm can be implemented directly as a part of the camera chip. The complexity of the initial interpolation procedure can be further reduced by considering several eliminations of its steps, while still maintaining a good visual performance. For example, the number of classes in the green interpolation can be reduced from three to two, by dropping the smoothest class, turning the classification into a binary decision process. The reason for not using two classes in the original algorithm is that in the smooth region, the average of the green pixels results in better performances than using pixels from the other channels.

In addition, we introduced a simple method for detecting and correcting color artifacts, relying on the orientational sensitivity of the human eye and on the smooth color transition assumption. The method may be extended to other color image applications, such as image enhancement and denoising. Further simplifications in this stage may also be considered. The reason for this is that a large number of clusters may contain only one pixel, resulting in many redundant comparisons made during the clustering, followed by recomputing the environment. Instead, we can improve the clustering process, by avoiding the comparisons in single pixel clusters, and interpolating them later. The interpolation of single pixels clusters is performed using the Y component of the partial image. In the Y component, we interpolate the missing pixel with the aid of its enlarged 4×4

neighborhood.

8 Acknowledgment

The authors wish to thank Gil Rosenfeld and Noam Lipcer for implementing the code, and for their contribution to the algorithm, and the SIPL lab at the Technion for supporting this research. Finally, we wish to thank Dr. Yacov Hel-Or for fruitful discussions.

References

[1] R. Kimmel, "Demosaicing: Image reconstruction from color CCD samples," *IEEE Trans. on Image Processing*, vol. 8(9), pp. 1221–8, Sept. 1999.

[2] J. E. Adams, "Interactions between color plane interpolation and other image processing functions in electronic photography," *Proceedings of SPIE*, vol. 2416, pp. 144–151.

[3] Y. Hel-Or and D. Keren, "Demosaicing of color images using steerable wavelets," Tech. Rep. HPL-2002-206R1 20020830, HP.

[4] S. Susstrunk, D. Alleysson and J. Herault, "Color demosaicing by estimating luminance and opponent chromatic signals in the fourier domain," *Proc. IS&T/SID 10th Color Imaging Conference*, pp. 331–336, 2002.

[5] S.C. Pei and I.K. Tam, "Effective color interpolation in CCD color filter array using signal correlation," *Proc. ICIP*, pp. 488–491, Sept. 2000.

[6] X. Wu et.al., "Color restoration from digital camera data by pattern matching," *Proceedings of SPIE*, pp. 12–17.

[7] D. R. Cok, "Reconstruction of CCD images using template matching," *Proc. of IS & Ts Anual Conference/ICPS*, pp. 380–385, 1994.

[8] Y. Altunbasak, B. Gunturk and R. Mersereau, "Color plane interpolation using alternating projections," *IEEE Transactions on Image Processing*, vol. 11, pp. 997–1013, 2002.

[9] J. E. Adams et.al., "Design of practical color filter array interpolation algorithms for digital cameras," *Proceedings of SPIE*, vol. 3028, pp. 117–125.

[10] Ed. Chang et.al., "Color filter array recovery using a threshold-based variable number of gradients," *Proceedings of SPIE*, January 1999.

[11] C. Wu et. al., "Reconstruction of color images from a single-chip CCD sensor based on markov random field models," *Proceedings of SPIE*, vol. 2564, pp. 282–288.

[12] D. H. Brainard et. al., "Bayesian method for reconstructing color images from trichromatic samples," *IS & T 47th Annual Conference*, pp. 375–380, 1994.

[13] B. E. Bayer, "Color imaging array," U.S. Patent, 3,971,065.

[14] J. L. Manos and D. J. Sakrison, “The effects of a visual fidelity criterion on the encoding of images,” *IEEE Trans. on Information Theory*, pp. 525–536, 1974.

[15] C. A. Laroche and M. A. Prescott, “Apparatus and method for adaptively interpolating a full color image utilizing chrominance gradients,” U.S. Patent, 1994, 5,373,322.

[16] I. Omer and M. Werman, “Using natural image properties as demosaicing hints,” to appear in ICIP 2004.

[17] D.H. Pritchard, “US color television fundementals - A review,” *IEEE Transaction on Consumer Electronics*, vol. 23.

[18] M. D. Fairchild, *Color Appearance Models*, ADDISON-WESLEY, 1988.

[19] S. K. Feiner J. D. Foley, A. van Dam and J. F. Hughes, *Computer Graphics: Principles and Practice*, Addison-Wesley, 1996.

[20] B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand reinhold, New York, 1993.

[21] T. Cornsweet, *Visual Perception*, Academic Press, 1970.

[22] M. D. Levine, *Vision in Man and Machine*, McGraw-Hill, 1985.

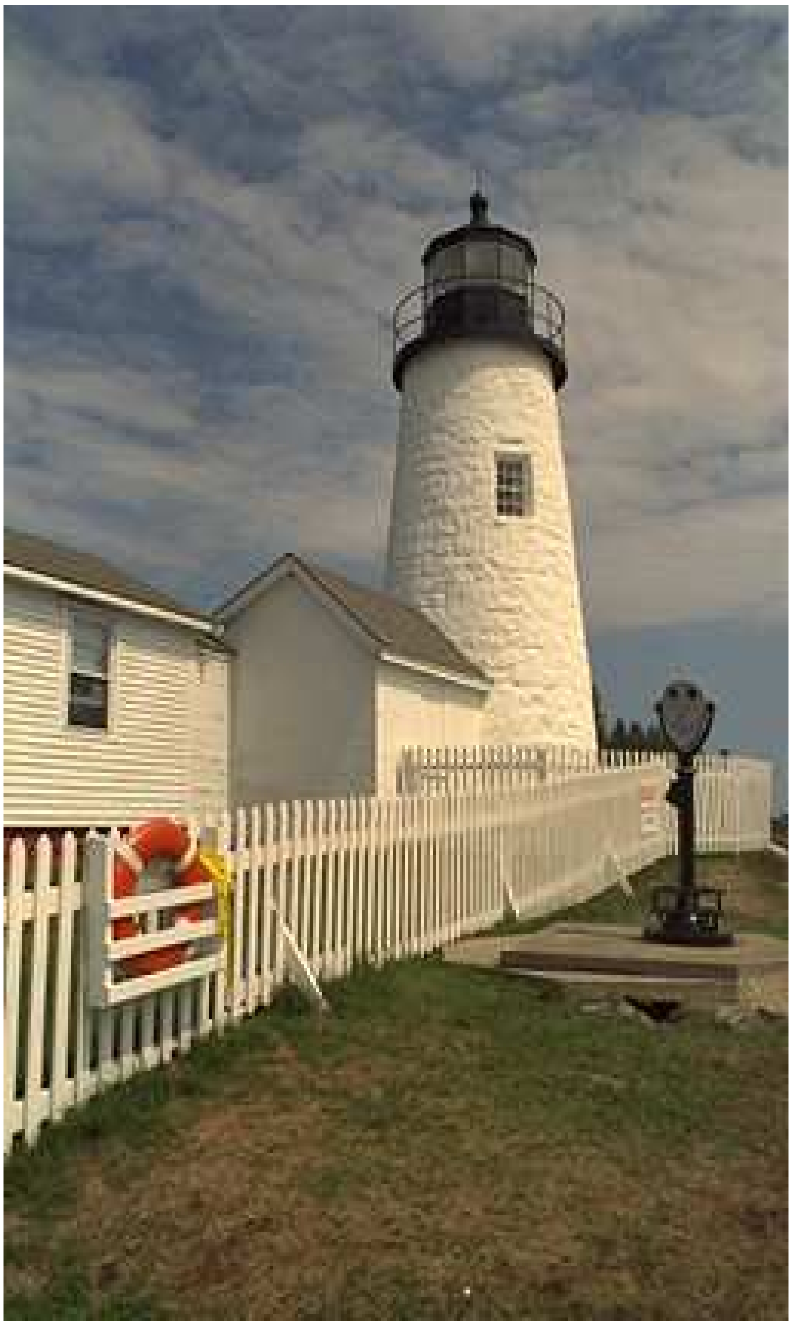


Figure 19: The original Lighthouse.

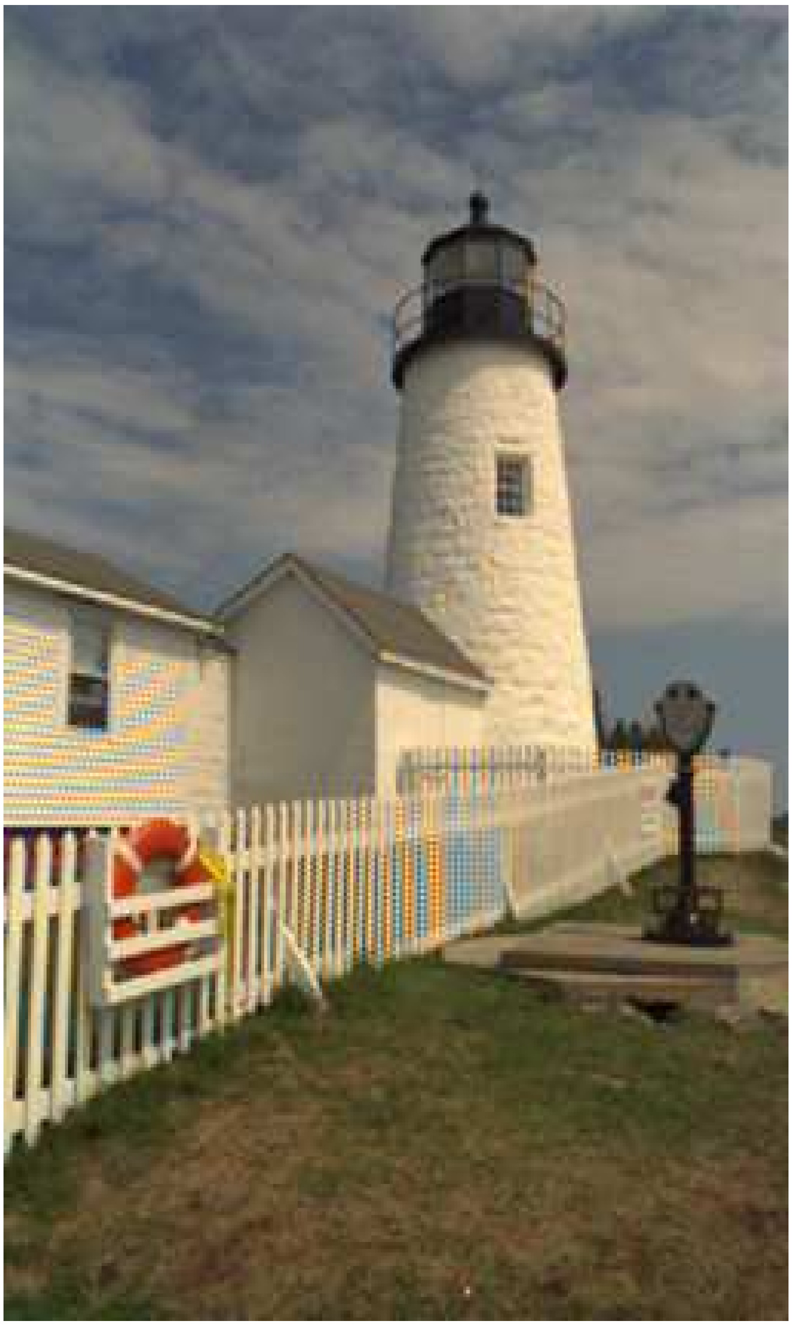


Figure 20: Bilinear interpolation.

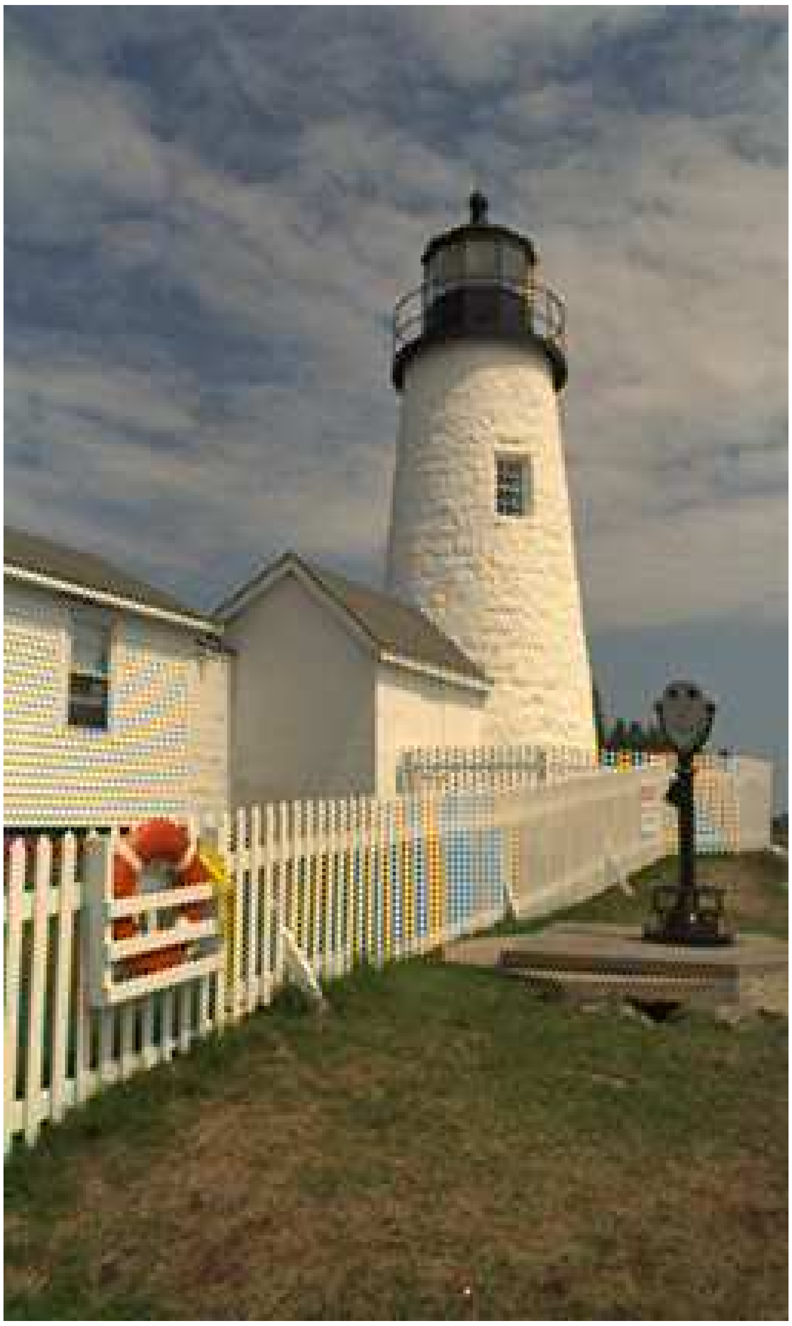


Figure 21: Smooth hue transition.

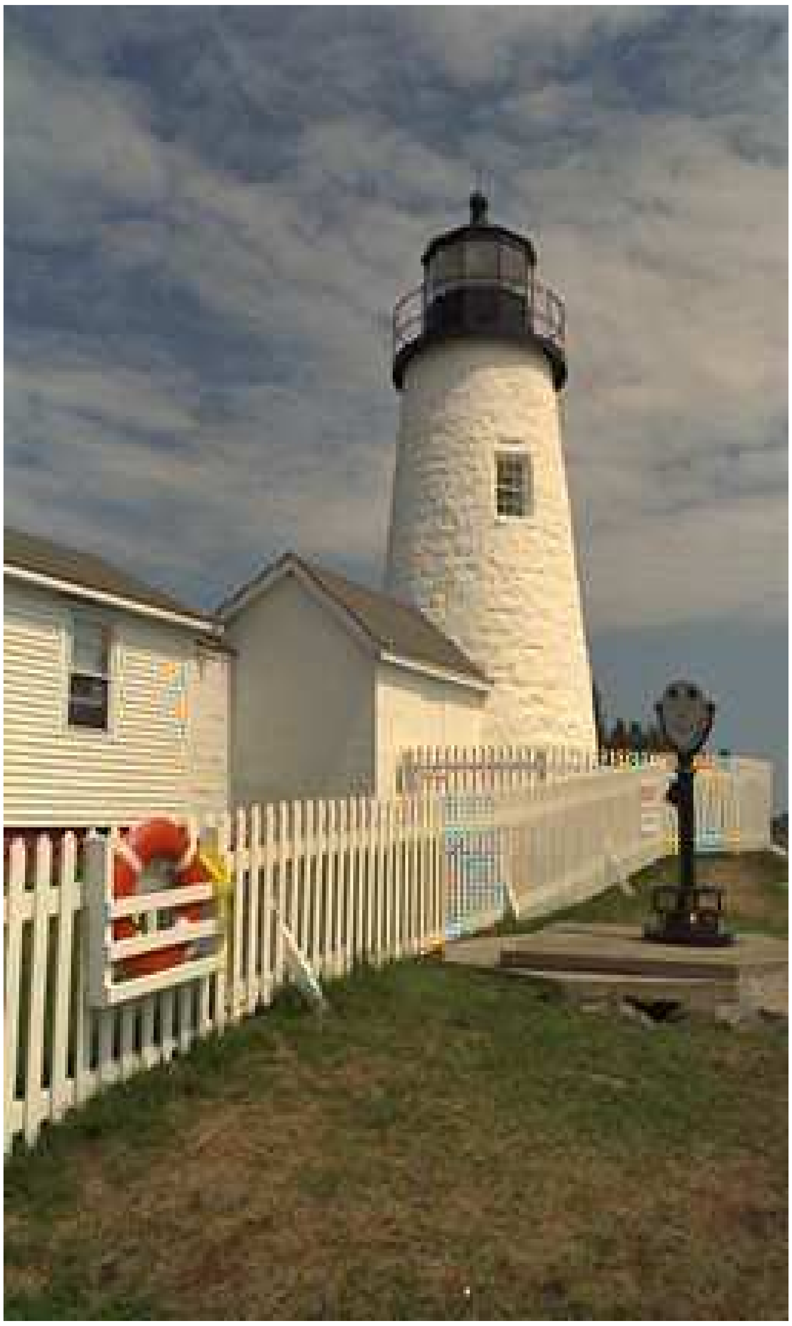


Figure 22: Edge detection algorithm.

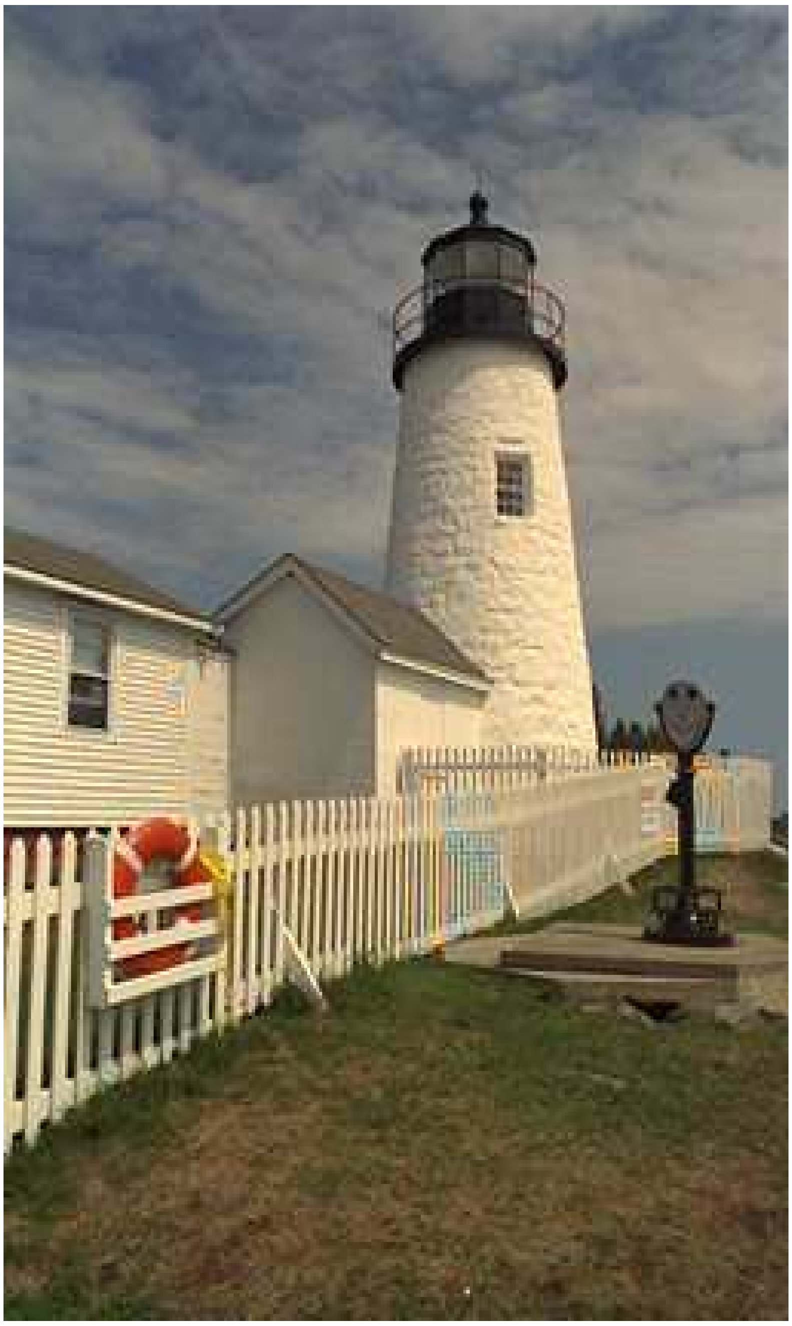


Figure 23: Color correction algorithm.

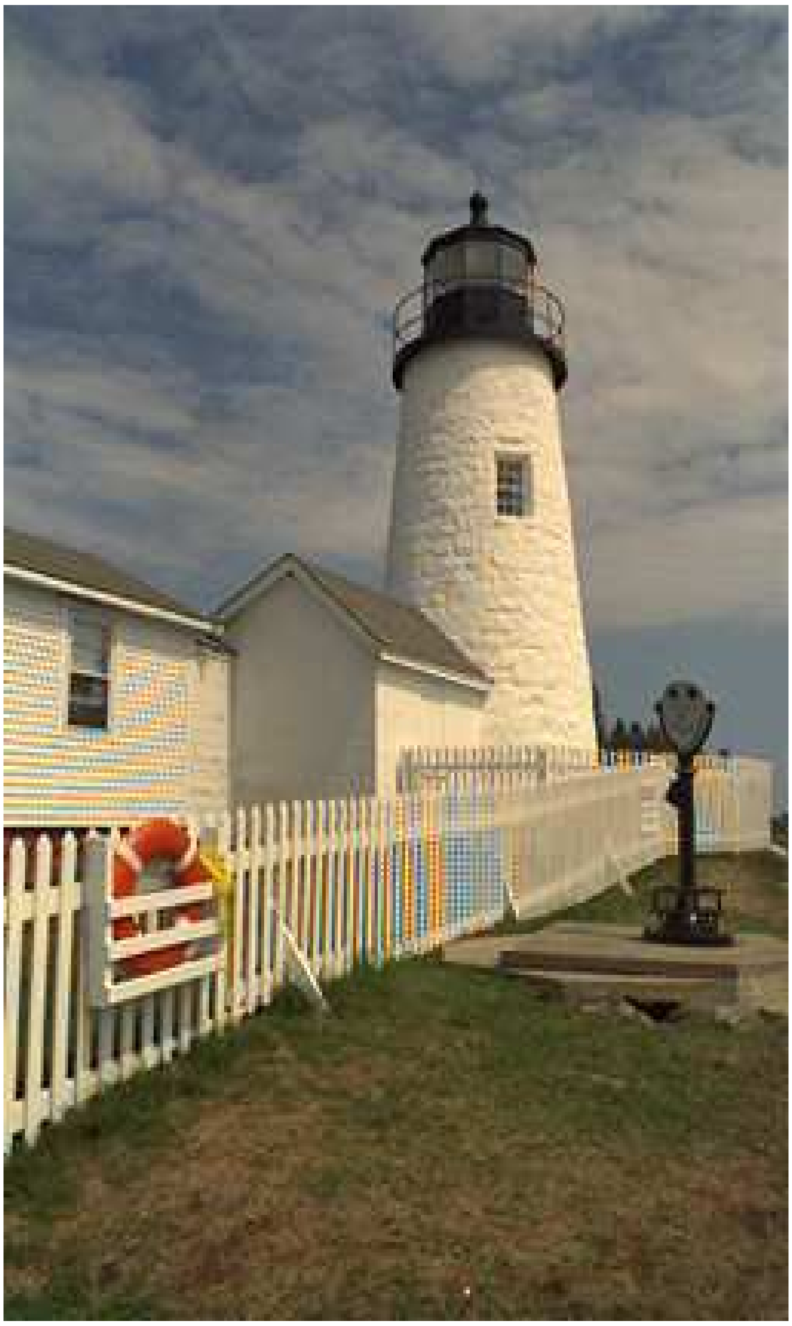


Figure 24: Gradient based algorithm.

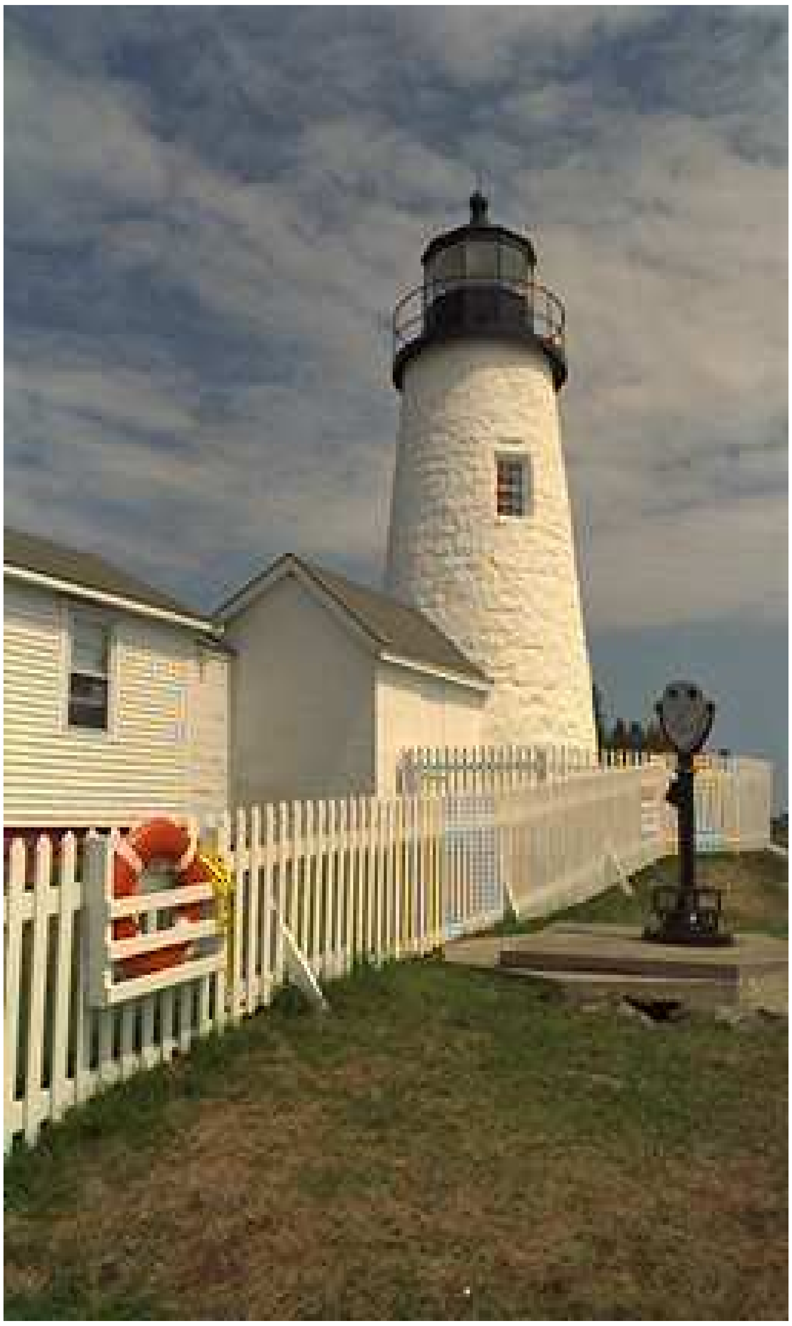
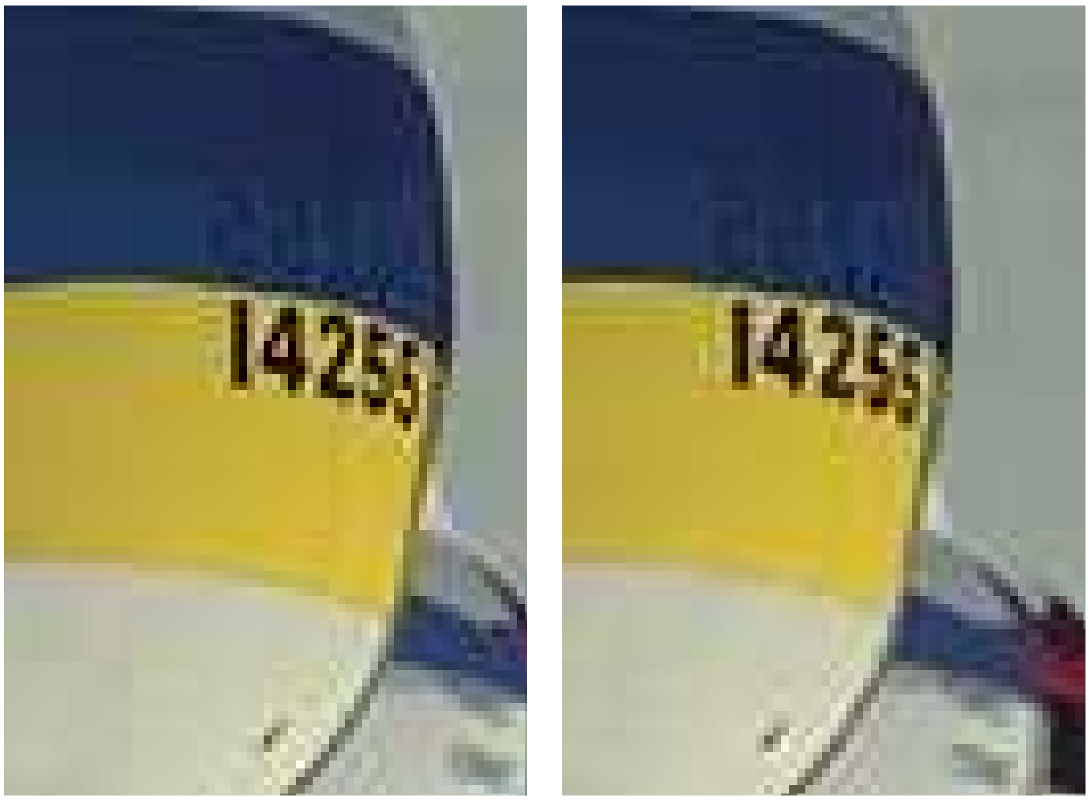


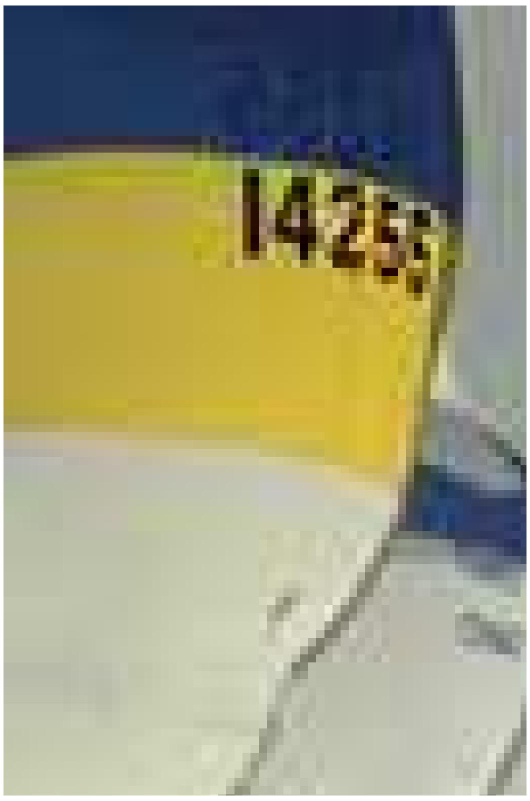
Figure 25: Kimmel's algorithm.

Figure 26: Our Lighthouse.



(a) (b)

Figure 27: (a) The original Sails (b) Our Sails.


(a)

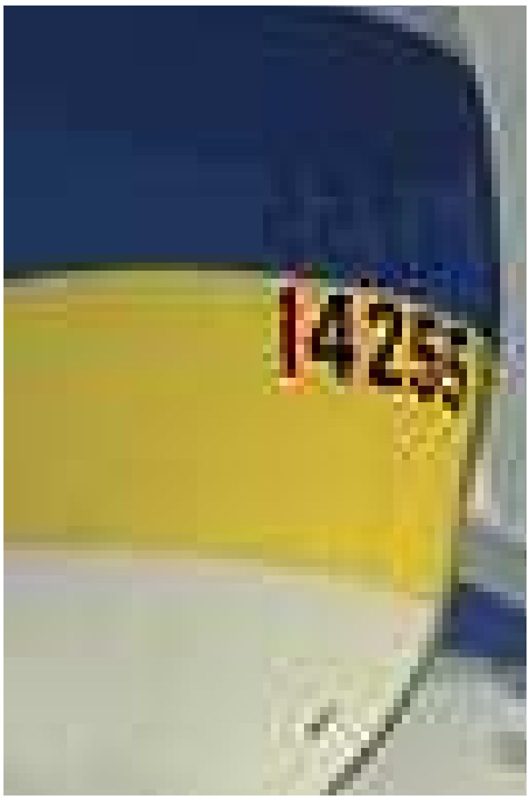
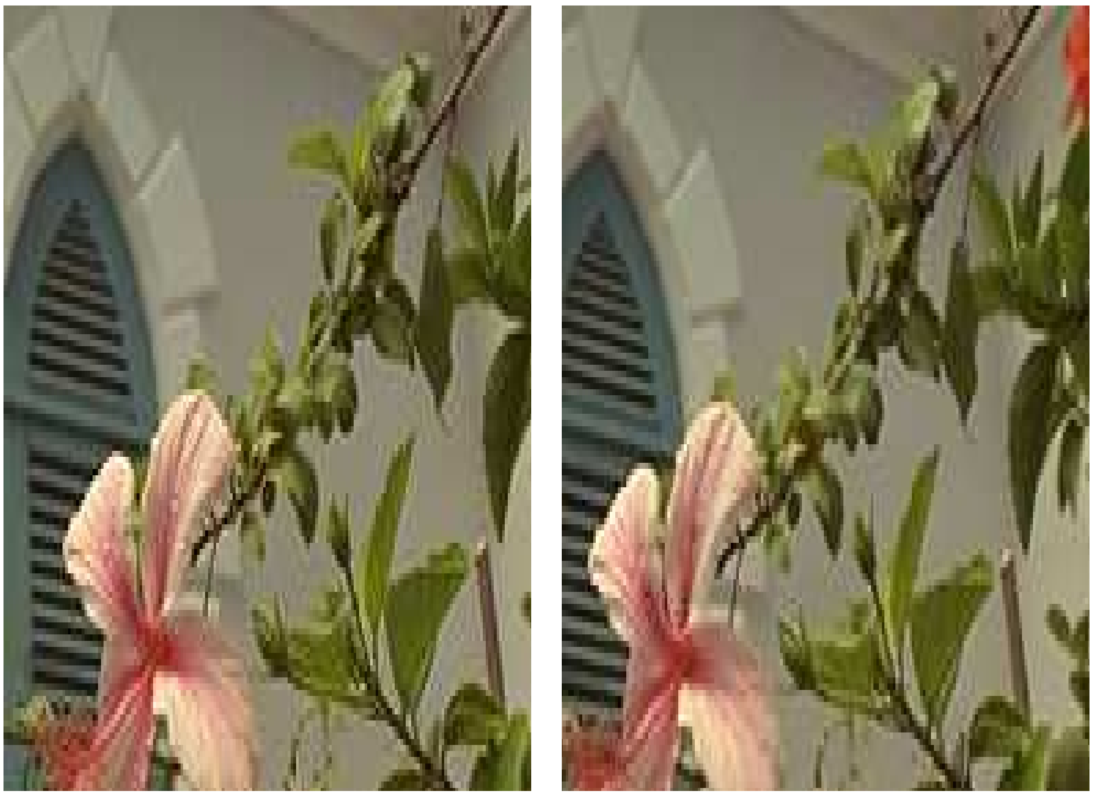
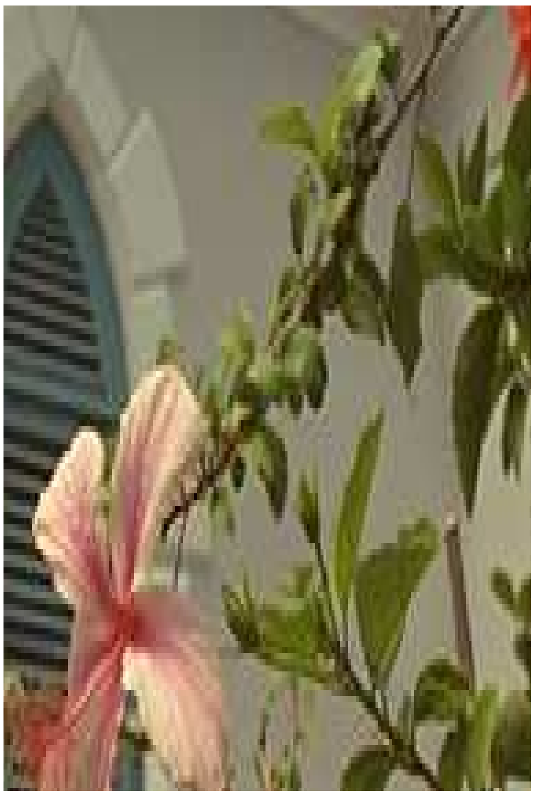

(b)

Figure 28: (a) Gradient based Sails (b) Kimmel's Sails.

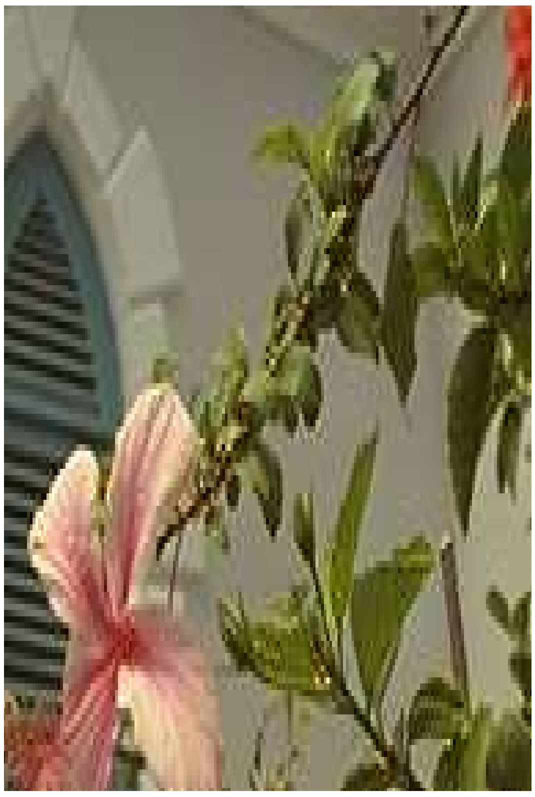


(a) (b)

Figure 29: (a) The original Window (b) Our Window.



(a)



(b)

Figure 30: (a) Gradient based Window (b) Kimmel's Window.