# Demosaicing with Successive Chrominance-Based Non-Local Means and Bilateral Filtering

Amy Chen
Stanford University
Stanford, CA
amyjchen@stanford.edu

## Abstract

*Demosaicing is a necessary step in processing digital images taken with color filter arrays. These arrays receive only red, green, or blue color information at each pixel, requiring us to infer missing color channels at each location. Many ways of approaching this goal can often lead to color or zipper artifacts as well as blur. We propose a solution for reducing these artifacts while retaining most details of an image.*

## 1. Introduction and Motivation

Most affordable digital cameras use sensors with color filter arrays (CFA) to separate channels of light. While there are sensors capable of capturing each channel of light at each pixel, these are more expensive. As a result, cameras with CFAs are more common. Images captured with a CFA require a process called demosaicing to infer the original image from the separate color channels. The most commonly used filter is the Bayer filter, which uses a repeated tile as seen in Figure 1. This filter has twice as many green channels than red or blue channels to coordinate with human sensitivity to green wavelengths of the visible spectrum.

The most straightforward processes to demosaicing are linear interpolation methods like Hamilton-Adams [2] and Malvar's "High Quality Linear Interpolation" [5], however, these methods often lead to color and/or zipper artifacts, as seen in Figure 2. Other suggested methods involve a linear interpolation step followed by iterative and/or non-local methods to correct these artifacts, on which this paper builds on. Many more recent methods involve machine learning techniques to demosaic images. We will describe some of these aforementioned techniques in more detail in the next section. In this paper, we explore three different techniques for demosaicing: Malvar [5], Baudes' "Self-similarity driven color demosaicking" (SSD) [1], and Li's "Demosaicing by successive approximation" (SA) [4] . We then build off of SSD [1] to retain the zipper-reducing benefits of SSD without its removal of detail. Finally, we compare performance of our new technique to the three we've explored.

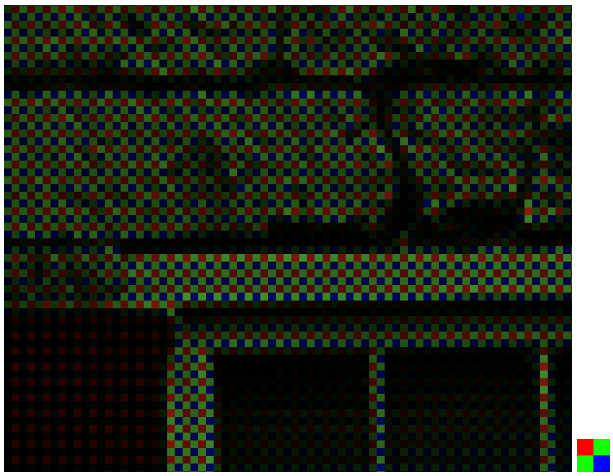

Figure 1. Left: Detail of 1st image of the Kodak dataset with Bayer Filter. Right: single tile of Bayer Filter.

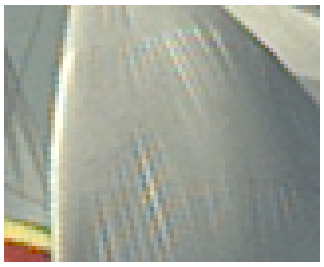

Figure 2. Detail of demosaiced image (Kodak 09) using Malvar [5]. Note the zipper patterning along the hard edge (left) as well as color artifacts on the wrinkles.

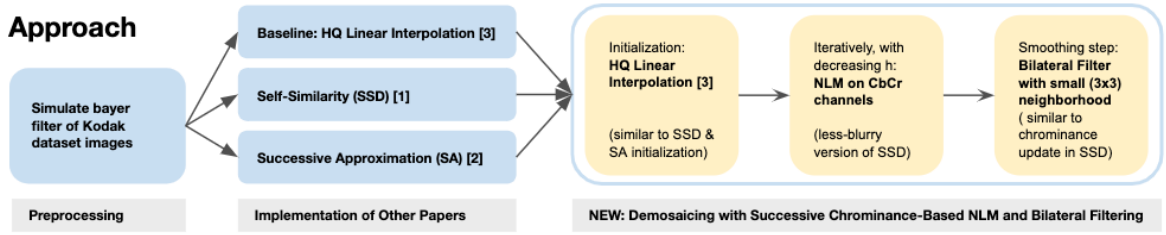


Figure 3. Flowchart of our research process

## 2. Related Work

Early work on demosaicing focuses on linear interpolation, such as the aforementioned Hamilton-Adams [2] and Malvar [5]. These methods both focus on 5x5 neighborhoods, however, take different approaches into how colors are averaged between pixels. Both approaches are susceptible to zipper artifacts, but the latter is better at maintaining detail.

More modern work is well summarized in Menon's "Color image demosaicking: an overview" [6], which describes several techniques up to 2011: (1) heuristic methods, such a adaptive interpolation, pattern matching, and weighted sums, (2) directional interpolations, (3) frequency domain approaches, (4) wavelet-based methods, (4) reconstruction approaches, and (5) joint approaches with denoising, zooming, and super-resolution. We go in depth into two of these methods.

Buades et al. 2009, "Self-similarity driven color demosaicking," presents an algorithm that first fills in missing colors through assessing similarity of non-local neighborhoods. They then perform a chromatic regularization step. This process is repeated multiple times with a resolution parameter. This method was found to work reasonably well at reducing color artifacts through relying on self-similarity [1].

"Demosaicing by successive approximation" by Li et al. 2005, suggests a method of iteratively updating the red blue channels and then the green channel, only stopping when a criterion has been satisfied. The update is determined by a 3x3 grid, while the criterion is based on correction color misregistration and zipper artifacts. The criterion adapts to areas with low and high aliasing, focusing on classifying pixels as high or low aliased regions, calculating the difference between pixel colors between iterations, and terminating if the threshold reaches a specific threshold [4].

Finally, many more recent work in demosaicing focuses on machine learning, such as Khashabi's "Joint demosaicing and denoising via learned nonparametric random fields" [3], Syu's "Learning deep convolutional networks for demosaicing" [7], and Zhou's "Deep residual network for joint demosaicing and super-resolution" [8]. While these methods are intriguing, we focus on the previous non-local and iterative methods in this work.

## 3. Method

A diagram of our research pipeline can be found in Figure 3. We start with an exploratory approach, and build off of one of our explored methods (SSD) to reach our own technique.

### 3.1. Simulating Bayer Filtering

We use the 768x512 pixel images of the Kodak Dataset to test each of these techniques. To simulate Bayer filtering in each of these images, we convolve each image's color with a kernel with the corresponding the red, green, or blue channel. We then combine these convolved images into a single image, in which each pixel contains only the value of it's corresponding color channel according to the Bayer pattern.

### 3.2. Exploration of Other Methods

We explore three methods. The first is a simple baseline of Malvar's High Quality Linear Interpolation (HQLI). The the others are non-local and/or iterative methods, both of which are initialized with a linear interpolation step. We describe their implementations in further detail below.

#### 3.2.1 High-Quality Linear Interpolation (HQLI)

High-Quality Linear Interpolation (HQLI) is fairly trivial to implement, as Matlab's demosaic function implements it. We use it as a baseline.

#### 3.2.2 Self-Similarity Driven Demosaicing (SSD)

Self-Similarity Driven Demosaicing (SSD) relies on successive steps of non-local means using decreasing smoothing parameters, followed by a chromatic regularization step to

remove zippering and color artifacts in an image. Sudocode for the algorithm is as follows:

**Algorithm 1** SSD

I = image
$u_0 \leftarrow initialInterpolation(I)$
**for all** h in $\{16, 4, 1\}$ **do**
  $u \leftarrow NL_h(u_0, h)$
  $u \leftarrow CR(u, I)$
  $u_0 \leftarrow u$
**end for**
**return** $u_0$

In this algorithm, non-local means is done using the combined euclidean distance of all three color channels. The chrominance step involves:

1. decomposing the RGB image into YUV components.
2. median filtering the U and V components on a 3x3 neighborhood, resulting in U' and V'.
3. recomposing the RGB image from YU'V'.
4. re-assigning the original channels for observed pixels on the Bayer filter to their locations on the updated RGB image.

In our implementation, we made a few changes that should be noted. First, we use Malvar's HQLI [5] technique rather than Hamilton-Adams [2] to do the initial linear interpolation step. Additionally, we strayed away from the recommended parameters of h, using h = $\{0.1, 0.075, 0.05\}$ instead of h = $\{16, 4, 1\}$. We found that the recommended set of parameters led to extreme blurring in every step, that was subsequently followed by an odd, faded appearance in the final image. We decreased these smoothing parameters until our resulting images were more reasonable and aligned with the results shown in the paper.

#### 3.2.3 Successive Apprroximation (SA)

Li's Successive Approximation approach begins with a simple linear interpolation for red and blue channels and bilinear interpolation of green channels. This is followed by Laplacian convolution of color differences between the red and green and red and blue channels, of which are used to threshold pixels. This convolution categorizes pixels as areas of high and low zippering. These categories designate the thresholds used in the iterative interpolation step of the algorithm. At every iteration, we only update a pixel if the change in pixels values from the previous step is above the pixel's threshold. This change in pixel values necessarily converges to 0, and is guaranteed to as low-threshold pixels stop updating early on. This update step is based on the

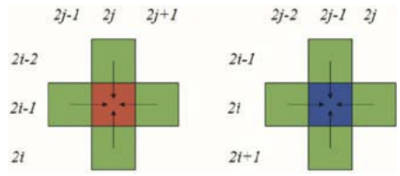


Fig. 4. Updating green channel. (Left) Interpolate $D_R$ at the location of red sublattice. (Right) Interpolate $D_B$ at the location of blue sublattice.

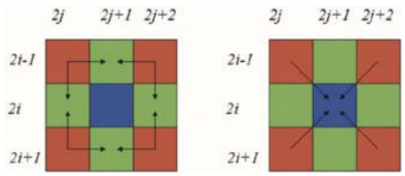


Fig. 5. Updating red channel. (Left) Interpolate $D_R$ at the location of green sublattice. (Right) Interpolate $D_R$ at the location of blue sublattice.

Figure 4. Description of interpolation step from Li's "Demosaicing by successive approximation" [4]

color differences of pixels as well, but follows a fairly simple interpolation scheme as shown in Figure 4. Sudocode for the algorithm is as follows:

**Algorithm 2** SA

R, G, B = image
$I_0 \leftarrow initialInterpolation(I)$
$T \leftarrow laplacianThresholding(I_0)$
$dR, dG, dB = \{\infty, \infty, \infty\}$
**while** $dR > 0$ **or** $dG > 0$ **or** $dB > 0$ **do**
  $R_n \leftarrow updateBasedOnThresholds(R, T.R, dR)$
  $G_n \leftarrow updateBasedOnThresholds(G, T.G, dG)$
  $B_n \leftarrow updateBasedOnThresholds(B, T.B, dB)$
  $dR \leftarrow recalculateDifference(R, R_n)$
  $dG \leftarrow recalculateDifference(G, G_n)$
  $dB \leftarrow recalculateDifference(B, B_n)$
  $R \leftarrow R_n$
  $G \leftarrow G_n$
  $B \leftarrow B_n$
**end while**
output $\leftarrow \{R, G, B\}$

Our own implementation uses masking matrices to track the thresholds of pixels as well as which pixels should be updated on each iteration.

### 3.3. Development of Own Method

The goal of our method, Chrominance-Based Non-Local Means with Bilateral Filtering (CNLM-BF) is to reduce color and zipper artifacts while retaining fine details in our images. We base this method on SSD, but instead of iteratively performing non-local means on the color channels —

of which can lead to a loss of these fine details— we iteratively perform non-local means on the chrominance channels. After these iterations are done, we then perform a small-neighborhood bilateral filter (degreeOfSmoothing = 0.001, 3x3 neighborhood) on all channels of the YCbCr image. This bilateral filter is intended to remove any zippering remaining in the luminance channel of the image. Sudocode for our algorithm is as below:

**Algorithm 3** CNLM-BF

I = image
$I \leftarrow initialInterpolation(I)$
$Y, Cb, Cr \leftarrow rbg2ycbcr(I)$
**for all** h in {0.02, 0.01, 0.005} **do**
  $Cb \leftarrow NLM_h(Cb, h)$
  $Cr \leftarrow NLM_h(Cr, h)$
**end for**
$Y \leftarrow bilateralFilter(Y, dos = 0.001, Nsz = 3)$
$Cb \leftarrow bilateralFilter(Cb, dos = 0.001, Nsz = 3)$
$Cr \leftarrow bilateralFilter(Cr, dos = 0.001, Nsz = 3)$
$I \leftarrow ycbcr2rgb(Y, Cb, Cr)$
**return** $I$

## 4. Results and Analysis

Our method proved to be largely effective, having improved performance over HQLI, SSD, and SA both quantitatively and qualitatively.

As seen in Table 1, our technique resulted in better results than the other three results on 21 of the 24 images in the Kodak dataset. We included a version of our process without the bilateral filter as well, called CNLM. Each method's parameters were tuned separately to achieve the best results. The parameters on each of the algorithms are as below:

1. HQLI: N/A
2. SSD: h = { 0.1, 0.075, 0.05 }, searchWindow: 15x15, neighborhood: 3x3
3. SA: Classification threshold: 0.05, low update threshold: 4, high update threshold: 0.05
4. CNLM-BF: h = { 0.02, 0.02, 0.005 }, searchWindow: 15x15, neighborhood: 3x3, bilateral smoothing: 0.001.
5. CNLM: h = { 0.03, 0.02, 0.01 }, searchWindow: 15x15, neighborhood: 3x3.

We additionally note that when other methods achieved higher PNSRs than our method, it was usually by less than 1 dB. Even in circumstances where other methods achieved high PSNRs, however, there are visual issues with the higher PSNR result that our technique does not have. For example, Figure 5 shows that the SSD result is qualitatively less desirable — the parrots' feathers lose most of their definition. In the next three subsections, we talk about the qualitative benefits and drawbacks of SSD, SA, and CNLM-BF, respectively.

| | HQLI[5] | SSD [1] | SA [4] | CNLM-BF | CNLM |
|---|---|---|---|---|---|
| Kodak 1 | 28.38 | 28.4 | 26.83 | 30.57 | 28.77 |
| Kodak 2 | 28.38 | 29.62 | 30.56 | 30.17 | 27.28 |
| Kodak 3 | 32 | 32.81 | 32.17 | 32.49 | 30.73 |
| Kodak 4 | 30.94 | 31.04 | 30.98 | 32.13 | 29.3 |
| Kodak 5 | 27.83 | 26.85 | 25.91 | 29.35 | 28.05 |
| Kodak 6 | 29.28 | 29.42 | 28.29 | 30.91 | 29.3 |
| Kodak 7 | 32.32 | 30.89 | 30.15 | 32.63 | 30.36 |
| Kodak 8 | 25.55 | 27.14 | 23.97 | 28.16 | 27.38 |
| Kodak 9 | 34.32 | 33.57 | 31.39 | 35.71 | 34.63 |
| Kodak 10 | 34.48 | 31.94 | 30.23 | 35.34 | 34.12 |
| Kodak 11 | 29.95 | 29.55 | 28.67 | 31.57 | 30.22 |
| Kodak 12 | 34.76 | 32.96 | 31.72 | 35.79 | 34.1 |
| Kodak 13 | 25.69 | 26.76 | 24.56 | 27.74 | 26.57 |
| Kodak 14 | 28.73 | 28.1 | 29.19 | 29.84 | 28.2 |
| Kodak 15 | 29.74 | 30.66 | 29.21 | 31.2 | 29.19 |
| Kodak 16 | 32.27 | 30.57 | 31.39 | 33.26 | 32.95 |
| Kodak 17 | 31.08 | 30.81 | 30.59 | 34.04 | 32.57 |
| Kodak 18 | 28.41 | 27.71 | 27.7 | 29.55 | 27.96 |
| Kodak 19 | 29.75 | 30.13 | 28.73 | 31.33 | 29.74 |
| Kodak 20 | 31.06 | 31.51 | 30.35 | 32.36 | 31.18 |
| Kodak 21 | 29.92 | 29.66 | 28.64 | 31.84 | 30.3 |
| Kodak 22 | 30.76 | 30.06 | 29.67 | 31.4 | 29.37 |
| Kodak 23 | 32.49 | 33.18 | 31.66 | 32.45 | 30.32 |
| Kodak 24 | 28.95 | 29.14 | 27.34 | 30.65 | 29.3 |

Table 1. Resulting PNSRs on the Kodak Dataset. Highest PSNR for each image is highlighted in yellow.

### 4.1. Self-Similarity Driven Demosaicking (SSD)

SSD was fairly effective at removing zipper artifacts in our images. This is apparent in 6, where we see that the zipper and color artifacts found in the baseline have largely disappeared. In the same image, however, we see one main downside of this approach: the detail in the panels on either side of the window are nearly lost, which further shows that SSD leads to oversmoothing. Aside from the smoothing found in our resulting images, we found that there were some inconsistencies in how SSD works as detailed in the paper [1] — mainly, that the final image must result in some odd Bayer-patterned artifacts, despite the method being geared towards removing zippering. Given that the chrominance step involves median filtering two channels, and the last step is always to reassign the original pixels, the process inevitably leads to remaining Bayer patterning

Figure 5. Kodak 23. From top to bottom: original, SSD, CNLM-BF. We note that even though SSD has a higher PSNR, the resulting image retains significantly less detail in the parrots' feathers.

in the resulting images. Figure 7 demonstrates this leftover Bayer patterning, which can be seen in the black portion of the windows of the lighthouse. Additionally, as noted before, we had some issues with the parameters of the original paper. In Figure 8, we show the before and after of our parameter re-tuning process. The only discrepancy between the paper's description and our implementation is the

Figure 6. Window of Kodak 01. From top to bottom, left to right: original, HQLI, SSD, SA, CNLM-BF, CNLM.

use of HQLI instead of Hamilton-Adams, but it is unlikely that this change made such a considerable difference in our result.

### 4.2. Successive Approximation (SA)

SA proved to be a step above our HQLI baseline in terms of zippering and color artifacts, but was the least effective of our non-baseline methods. This is especially apparent in Figure 7. In it, we see the zippering in our image decreases — however, we also see odd white speckling artifacts around the edges. We note that these speckled artifacts were found in the original paper as well. We additionally see artifacts that lead to slightly faded coloration, as seen in 9.

### 4.3. CNLM-BF and CNLM

Our new method is also almost as effective as SSD when it comes to zippering and color artifacts, but is notably better at retaining detail than SSD. This is clear in Figures 5, 6, and 7. We note that it is also effective to use our method

Figure 7. Window in lighthouse of Kodak image 19. From top to bottom, left to right: original, HQLI, SSD, SA, CNLM-BF, CNLM. Note the loss of detail on the wall and Bayer patterning in the window of SSD. Also note the zippering on SA.

without the bilateral filtering step and with different parameters for decent results. However, doing so still leads to odd remnants of the Bayer filter, like we see in SSD and along edges in SA.

One main detractor from both CNLM-BF and CNLM is the resulting dulling of colors. In small details of images, the colors are faded and duller, such as in Figure 9. This is likely because we apply NLM over the chrominance of the image, which leads to color mixing. We also see some loss of detail in these images due to the biilateral filtering step.

## 5. Discussion

Through our exploration of HQLI, SSD, SA, and CNLM-BF, we have explored several benefits and trade-offs of each approach. It's widely known that our baseline, HQLI, results in undesirable zipper and color artifacts. SSD reduces zipper effects and color artifacts at the cost of losing fine details. SA reduces zipper effects and color artifacts, but adds more artifacts where they weren't any apparent ones before. Finally, our approach, CNLM-BF, reduces zipper and color artifacts, but hits issues when it comes to small vibrant areas of images.

Given these trade-offs, both SSD and CNLM-BF are the more desirable options of those we've explored. They both only fail with smaller details, and ones that are less apparent than the speckling found in SA. We argue, however, that

Figure 8. Parameter tuning of SSD. Top: original parameters from the SSD paper, h = $\{16, 4, 1\}$. Bottom, our parameters h = $\{$ 0.1, 0.075, 0.05 $\}$.

CNLM-BF is more suited to the limitations of the human visual system, as small changes in chrominance are less apparent than losses of small details.

#### 5.0.1 Limitations and Future Work

As stated before, our method results in faded colors in smaller vibrant spots of images, as well as light blurring due to the final step. This creates problems for images that need to retain those colors or details on the scale of our 3x3 pixel neighborhoods. While the latter is not avoidable through our current method, the former might be remediable through a pre-NLM chrominance regularization step, in which we re-apply the original Bayer filter channels to the image before performing NLM, and is an area for future exploration.

### 5.1. Conclusion

In this paper, we've explored a few methods of non-local and iterative demosaicing, developed our own method,

Figure 9. Flowers in the balcony of the house in Kodak 24. From top to bottom, left to right: original, HQLI, SSD, SA, CNLM-BF, CNLM. Note the faded colors of SA, CNLM-BF, and CNLM. This is a 75x50 px area of the image.

and weighed the trade-offs of the results of all three. In the process, we've found a promising demosaicing method, CNLM-BF, which reduces zipper effects and color artifacts at minimal noticeable cost to the image, and of which produces high quantitative results.

## 6. Acknowledgements

Thank you to Gordon Wetzstein and David Lindell for providing the guidance and wisdom that led to this work.

## References

[1] A. Buades, B. Coll, J.-M. Morel, and C. Sbert. Self-similarity driven color demosaicking. *IEEE Transactions on Image Processing*, 18(6):1192–1202, 2009.

[2] J. F. Hamilton Jr and J. E. Adams Jr. Adaptive color plan interpolation in single sensor color electronic camera, May 13 1997. US Patent 5,629,734.

[3] D. Khashabi, S. Nowozin, J. Jancsary, and A. W. Fitzgibbon. Joint demosaicing and denoising via learned nonparametric random fields. *IEEE Transactions on Image Processing*, 23(12):4968–4981, 2014.

[4] X. Li. Demosaicing by successive approximation. *IEEE Transactions on Image Processing*, 14(3):370–379, 2005.

[5] H. S. Malvar, L.-w. He, and R. Cutler. High-quality linear interpolation for demosaicing of bayer-patterned color images. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 3, pages iii–485. IEEE, 2004.

[6] D. Menon and G. Calvagno. Color image demosaicking: An overview. *Signal Processing: Image Communication*, 26(8-9):518–533, 2011.

[7] N.-S. Syu, Y.-S. Chen, and Y.-Y. Chuang. Learning deep convolutional networks for demosaicing. *arXiv preprint arXiv:1802.03769*, 2018.

[8] R. Zhou, R. Achanta, and S. Süsstrunk. Deep residual network for joint demosaicing and super-resolution. In *Color and Imaging Conference*, volume 2018, pages 75–80. Society for Imaging Science and Technology, 2018.