# Image Reconstruction with Burst of Low-Light Photography

Wanzi Zhou wanziz@stanford.edu
Chen Qian cqian23@stanford.edu
Tong Yang tongy@stanford.edu
Stanford University, Stanford, CA 94305

***Abstract*—Inspired by the Google HDR+ paper published in 2016[1], in this project, we implemented a thorough pipeline staring from a bursts of raw Bayer pattern photos with constant low exposure time taken from the Google HDR+ dataset[3] into a finely displayed high dynamic range(HDR) image. We give detailed descriptions of the algorithms we use in every stage of our pipeline and show the resulting images we get from each stage. We analyze the resulting images and present our analysis. We conclude that our pipeline is performing a good job in restoring natural color and enough details of the image. Due to the time limit, we also have some limitations and future work.**

## I. Introduction

There are mainly two motivations for this project. We want to do a project that is both highly related to the topics and theories we covered in class and is also up-to-date, interesting and influential.

One of the most common problems that we encounter when we shoot photos with smart phones is that there is not enough light, leading to a lot of noise and low dynamic range in the image. Usually there are two standard solutions to this problem: applying analog or digital gain and lengthening exposure time. The former would amplify the noise even more and the latter might lead to motion blur due to camera shake and subject motion[1]. We first find this paper by Google [1] published in 2016 which tries to solve the problem, and especially targeted at smart phone, by using a newly proposed computational photography pipeline called HDR plus (we will cover that more in the relational work), which excels in low-light and high dynamic range scenes and is computationally efficient and reliably artifact-free. We find this really interesting in that the original input photos of this pipeline are a burst of photos with the same low exposure time, while in class we only explored fusing photos with a burst of photos with different exposure time to provide a high dynamic range (from the well-known Debevec paper[2]). And according to this paper the HDR+ pipeline is very computationally efficient and has received very good user and press response. This topic is highly relevant to what we have learned in class and also has very high application values.

Although they have given many descriptions of their pipeline in the paper, in many places the details and algorithms are vaguely mentioned. So we want to start by absorbing some of their ideas and try to develop a competitively well-tuned pipeline on our own. We want to highlight one thing about the yet to illustrated pipeline before we go into the details: different from what we implemented in class, where we have a group of photos each with different exposures, the input photos of the pipeline are a burst of raw Bayer pattern images that have constant low exposure time. Why low exposures and why constant time? There are mainly three reasons. First, this pipeline is targeted at cell phones, thus taking long exposures is difficult for the user to control and thus there might be a lot of motion blur. Second, there will be misalignment issues between long exposed images and short exposed images, it's just easier to keep alignment in a burst of photos of the same scene that have constant and short exposure times. Thirdly and most importantly, this is because when in under exposure, all the details are well kept, i.e no place is overexposed to lose its details. You might say under exposure will introduce a lot of noise, however since these noises are random and thus distributed differently in different dark photos, when we fuse them together and apply all the denoising methods we discuss later, the noise problem is well handled.

Given this setting, we have two main challenges in this project. The first challenge is the misalignment issue. Because the input is a burst of images, there will be misalignment among themselves, if we do nothing there will be severe ghost blur effects. So when we merge these images together, we need a good way to mitigate the misalignment issue. Second, because all the input photos are underexposed, there will be a lot of noise. On the other hand, when we try to scale these dark images to get a high dynamic range image, the noises are also amplified. So we need to deal well with these noise in the merging stage and afterwards in order to get a good quality image.

There are two major parts of this project: the first is to find an appropriate method to merge different images in the burst, aiming to capture as many details as possible on the scene. By combining multiple observations of the scene over time, we can achieve high dynamic range and noise deduction. The other major part of the project is focused on post-processing of the HDR image, including correction, demosaicing, denoising and tone-mapping.

## II. Related Works

In 1997, Debevec and Malik proposed a method for recovering high dynamic range radiance maps with burst photography[2]. The idea is to take a series of photographs with different exposures, and fuse into one high dynamic range photograph with the response function recovered using the assumption of reciprocity. The ideas works well with the photographs taken with conventional imaging equipment like digital camera, because they usually have relatively large sensor size and easily controlled ISO and exposure time. However, it is much more difficult in the case for cell phone cameras. Cameras on the phone usually have small apertures and sensor pixel size, which means that both the number of photons and the number of electrons each pixels can store are limited. In addition, different exposure time introduces challenges in aligning images because of the different levels of noises and motion blur, leading people to explore new approaches.

In [1], researchers at Google proposed a complete system called HDR plus in 2016. It captures a burst of underexposed frames, aligns and merges these frames to produce a single intermediate image of high bit depth, and tone maps this image to produce a high-resolution photograph. This paper tries to present this commercialized pipeline into an academia-stylized paper, and several main parts in this system include: it is composed of a hardware pipeline and a software pipeline. The input of both pipelines is Bayer mosaic image. When the camera app is launched, the hardware pipeline launches and displays a demosaiced low resolution on the phone screen. As soon as the shutter is pressed, a burst of frames is captured at constant exposure and stored in the main memory, then the software pipeline starts, where it aligns and merges the frames in a burst, produces a single intermediate image of high bit depth, and tone maps this image to produce a single full-resolution 8-bit output photograph for compression and storage in flash memory[1]. Several important features of system include: using constant exposure time in the burst of photos (as mentioned in Motivation part different exposure time leads to misalignment issue especially in phone cameras), using Bayer raw image as input instead of demosaiced RGB frame (providing more bits per pixel) and utilizing a novel FFT-based alignment algorithm and a hybrid 2D/3D Wiener filter to denoise and merge the frame. According to the paper, the entire system only takes 4 second on an Android phone, which is pretty efficient. They also put up an HDR+ dataset[3], which we think can be utilized in our project as an oracle baseline.

There are some other well-known denosing methods implemented previously which could potentially help us or give us ideas in our project. Kokaram[4] proposed the frequency-domain video denoising techniques by applying Wiener filtering in the 3D DFT domain. Another variation of the denoising method we can potentially explore is proposed by Delbracio and Sapiro in 2015[5]. We can view the captured burst images as nearby frames in the video and apply the same denoising method proposed for video denoising above. Based

on the computed weighted average in Fourier domain across the burst, the image is reconstructed by combining the least attenuated frequencies in each image.

## III. Algorithm

The pipeline of our project is shown in figure **??**, and we can divide it into two major parts. The first part is processing the burst of raw CFA images on R, G1, G2, B separately, and merging into one raw image, which contains Split Tiles, Merge Tiles, and Stack Tiles. Separating the color channel is intended to increase the robustness of the model.

In this section, we will mainly focus on major algorithms in our pipeline that are rather complex or contain innovative approach, mainly six of them: split tiles, merge tiles, stack tiles, white balancing, chromatic denoise and tone mapping. Other stages that involve more general and common approaches such as black level subtraction, demosaicing etc. are not included.

The second part is processing the merged raw image. We first did some basic adjustment through Black-level Subtraction, White Balancing, and Demosaicing. After these processes, we obtain an HDR image with bit depth of 12. Later we applied Tone Mapping, Global Tone Adjustment, and Sharpening to fit the final image into luminance range of display.

### A. Split Tiles

We obtained a burst of 10 underexposed 10-bit raw images of the same scene with the same exposure time. The unit of our image merging method is image tiles. Each tile contains $16 \times 16$ (or $32 \times 32$) pixels from color panels of Bayer raw input. Adjacent tiles have overlap size of $overlap$, which is a tuning parameter. $overlap$ allows image merging to consider local information while avoiding patchy results. In our final mode, the parameter is set to $8$.

### B. Merge Tiles

The major challenge of the project is about finding proper ways to merge 10 images into one. Since all images are of the same exposure time, we cannot merge them based on their weights(confidence) that a pixel is well-exposed. In addition, misalignment also needs to be taken into consideration when merging.

The merging is performed in the frequency domain. We denote the image index as $z \in \{1, ..., 10\}$, Fourier Transform of frequency $\omega$ as $T(\omega)$.

Based on these two challenges, we first implemented a baseline method of merging, which takes the average of each frequency component as the output.

$$\tilde{T}(\omega) = \frac{1}{N} \sum_{z=1}^{N} T_z(\omega)$$

The problem of the method is that it is not robust to misalignment, and introduces severe ghosting effect, which will be discussed in more detail in the Analysis and Evaluation section.

To tackle the problem, we implemented a second merging method, which uses the idea of frequency-domain video denoising as mentioned in the Related Work, to increase the robustness and perform denoising at the same time. Here we take the first image as the reference, and add the weighted contribution of other images with a shrinkage factor $A_z$, where

$$A_z = \frac{|T_1(\omega) - T_z(\omega)|^2}{|T_1(\omega) - T_z(\omega)|^2 + c\sigma^2}$$

Here $\sigma^2$ is the noise variance, where we compute as the root-mean-square of the values in processed tile. $c$ is a tuning factor that controls the contribution of $\sigma^2$, which we set to $8$ in the final modeling.

Intuitively, $A_z$ controls the contribution of frequency component in other images where the values differ from the reference image. It reduces the contribution when the difference is introduced by misalignment, and increases the contribution when the difference is introduced by noise. The output tile can be computed as

$$\tilde{T}(\omega) = \frac{1}{N} \sum_{z=1}^{N} [(1 - A_z(\omega))T_z(\omega) + A_z T_1(\omega)]$$

To preserve the precision of the merging result, the bit depth is increased from 10 bits to 12 bits after this step.

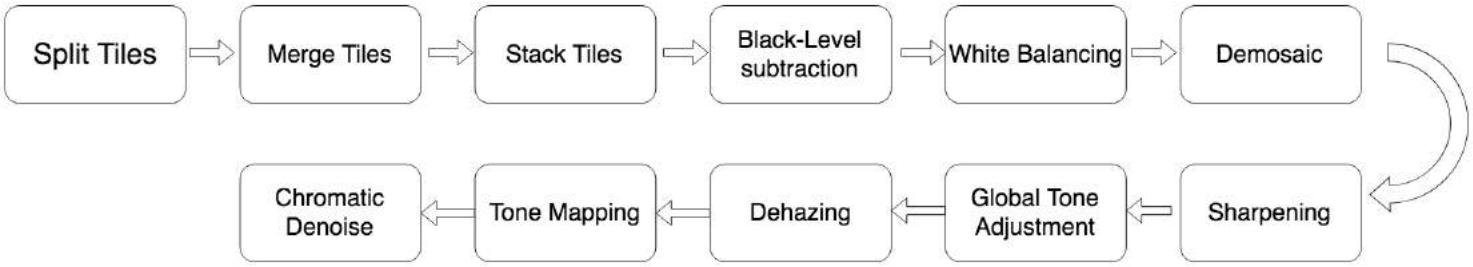


Fig. 1: Pipeline

### C. Stack Tiles

After obtaining the merged raw image, we reverse engine Split Tile with same overlap size to get a new Bayer raw image for further processing.

### D. White Balancing

Up to this part, we are still in the Bayer raw pattern. The purpose of white balancing is to adjust the balance of color in the image so by adjusting the white pixels to be actually white. The method we use for white balancing is to first find a reference point that is white and the $2\times2$ square at that pixel (which according to the Bayer pattern will definitely contain a red pixel, a blue pixel and two green pixels).

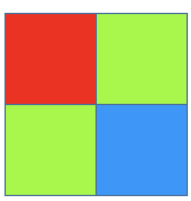

Fig. 2: example $2 \times 2$ square at the reference white point

Figure 2 shows an example square pattern at the reference white point. It might not be exactly R,G,G,B but according to the Bayer pattern the square must contain one red pixel, one blue pixel and two green pixels.

We then take the average of the two green pixels as the a single green pixel value. Obtaining the three red, green and blue pixel values, we find the maximum value between them and compute a scale ratio for all three channels as defined below in $r_{\text{red}}$, $r_{\text{green}}$ and $r_{\text{blue}}$.

$$\text{greenValue} = \frac{\text{greenValue1} + \text{greenValue2}}{2}$$

$$\text{maxV} = \max\{\text{redValue} + \text{greenValue} + \text{blueValue}\}$$

$$r_{\text{red}} = \text{maxV}/\text{redValue}$$

$$r_{\text{green}} = \text{maxV}/\text{greenValue}$$

$$r_{\text{blue}} = \text{maxV}/\text{blueValue}$$

We then apply the scale ratio for all values in the three channels respectively. Finally we apply a global scaling in the whole image where we make sure the largest value across all channels is less than 1 (because we use double value to process the image).

### E. Chroma Denoising

The goal of chroma denosing is to reduce red and greens splotches in dark areas of low-light images. We convert the RGB image after demosaicing to YCbCr and perform bilateral filtering on the chromatic channels Cb and Cr. We then map YCbCr back to the RGB soace to view the effects.

### F. Tone Mapping

Since our processed image now has bit depth of 12, we would like to fit the image into luminance range of display, $i.e.$ 8 bits with range $(0-255)$. We implemented local tone mapping by applying a non-linear bilateral filter to the intensity while preserving the detailed information.

$$I_{\text{intensity}} = (20R + 40G + B)/61$$

$$\text{Chrominance} = (R/I, G/I, B/I)$$

We apply bilateral filter to $L = \log(I_{\text{intensity}})$ to get the base layer $B$, and compute the detail layer $D = L - B$ to preserve the texture. After applying offset and scale to be base layer $B$ and add back with detail layer $D$, we convert back to the linear scale and merge with Chrominance to obtain the final image for display.

## IV. Analysis and Evaluation

In addition to comparing the final output of our pipeline to the originally taken images to test the performance, we are interested in how each modules in this pipeline contributes to the output image's quality. Thus, we ran comparison experiments with respect to tile merging, black-level subtraction, white balancing, chromatic denoising and tone mapping by simply removing and adding these modules to see the difference. In our experiments, we did not perform comparison test on demosaicing as it is a necessary step, and not on dehazing, global tone adjustment and sharpening because the performance of these modules is well-known.

It worth notice that in our project, there is no ground truth for evaluation purpose, as we should not rely on some recovered images to claim that they are our final targets. If so, there will be no value inside our project. Hence, we cannot apply methods such as computing PSNR with respect to the original images as we did in homework, but instead judging the image quality by eyes.

### A. Split Tiles

In order to study the effect of tile size and overlapping size on the image quality, we performed experiments on different choice of tile size $L*$ and overlapping size $O$. See experiment results in Fig. 3.

Fig. 3: Comparison among different tile sizes and overlapping sizes

The parameter of Fig. 3 is shown in Table. **??**.

TABLE I: Parameters of Fig. 3

| | | | |
|---|---|---|---|
| L=2, O =0 | L=16, O =0 | L=64, O =0 | L=1024, O=0 |
| L=2, O =1 | L=16, O =8 | L=64, O =32 | L=1024, O =512 |

It is found that although a too large or small tile/overlapping size will be harmful to the image quality, as long as tile/overlapping size falls in some certain region, we can hardly detect difference on the output images.

### B. Merge Tiles

As discussed in the previous section, we proposed two ways to merge tiles: one is simply averaging over the frequency domain (baseline), and the other is averaging depending on shrinkage factor $A(z)$. The performance of two methods is shown in Fig. 4.

Fig. 4: Merging Frame

In Fig. 4, the left image is the output of simple averaging, while the right one is achieved by weighted averaging. It is clear that there exists severe ghosts effect in the left images, especially on the parts

having sharp edges. However, there is almost no ghost effect in the right image. Thus, we conclude that the weighted averaging method is much more robust to misalignment.

### C. Black-level Subtraction

The difference between the output of demosaiced image undergoing black-level subtraction and not is shown in Fig. 5.


Fig. 5: Performance Test of Black-level Subtraction

It is shown in Fig. 5 the figure without black-level subtraction looks redder than the one with black-level subtraction, which is closer to natural color.

### D. White Balancing

The difference between the output after demosaicing with white balancing and not is shown in Fig. 6.


Fig. 6: Performance Test of White Balancing

The left image is with white balancing while the right is produced without white balancing. As shown by Fig. 6, the output image without white-balancing possesses large amount of red, which looks unnatural. However, the image undergoing white-balancing takes a more natural color. Since none of the remaining steps can recover the unnatural colors, we conclude that white balancing is necessary in our pipeline.

### E. Tone Mapping

We need tone mapping to show an HDR image in lower bit depth. The image before and after tone mapping is shown in Fig. 7.

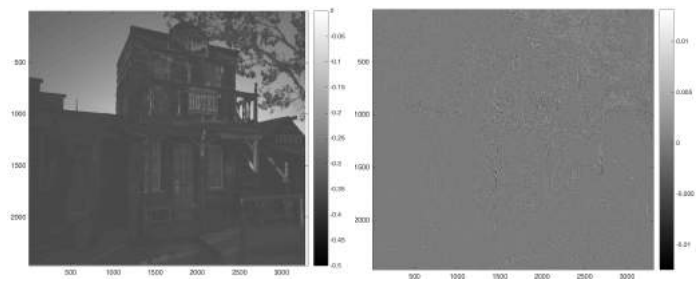

Fig. 7: Tone Mapping Scaled Base and Detail Layer


Fig. 8: Performance Test of Tone Mapping

It is observed that after tone mapping, dark areas are much brighter than before, and color details are restored.

### F. Final Result

The original image (dng file) preview and the final output is shown in Fig. 9.


Fig. 9: Comparison between Original Image and Our Output

The left image in Fig. 9 is the preview of original DNG file, which only have average pixel value of $0.143$. The right is our final output, which has average pixel value of $0.508$. We can see that our pipeline recovers an HDR image, with clear color details in both bright and dark regions in the original image.

## V. Conclusions and future work

We conclude that our pipeline is performing equally good as the HDR+ algorithm when not performing frame aligning. Using the merging algorithm with shrinkage factor effectively removes the ghost blur due to misalignment. The details are well recovered in HDR scenes (women in lower left corner). The tone mapped color is natural and well saturated (blue sky, brown house, green leaves).

Due to the limited time, we list the following limitations and future work for our project.

- We did not perform frame aligning, thus the edges are not sharp enough ($e.g.$ edges of the house). If there is more time, we think we should definitely do this part.
- We can try similar idea in high dynamic range videos. We can view nearby frames as the burst of image input in our project and apply similar techniques. However, there are several things that we need to consider. First of all, we need a high-speed camera, if the camera cannot capture high speed motion well, then there will be severe motion blur effect, making the alignment process even harder. Second, we definitely need to perform the alignment part, basically this is because the video is not static and different frames will have different perspectives of the scene captured. After these two steps, we can perform similar steps as mentioned above in our work.
- There are some artifacts in the blue sky (some parts are more green than blue), we think this needs further parameter adjustment in demosaicing and denoising, and probably includes steps such as hue specific color adjustment.
- We should also try to apply our algorithm in actual smart phones so that its computational efficiency and practical value can be better measured.

## References

[1] Sam Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T. Barron, Florian Kainz, Jiawen Chen, Marc Levoy, Burst photography for high dynamic range and low-light imaging on mobile cameras, 2016, ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2016)

[2] Paul E. Debevec, Jitendra Malik, Recovering High Dynamic Range Radiance Maps from Photographs, 1997

[3] GOOGLE INC., HDR+ burst photography dataset, 2016, http://www. hdrplusdata.org

[4] Kokaram, A. C., Motion picture restoration (doctoral thesis), 1993, https://doi.org/10.17863/CAM.711

[5] Mauricio Delbracio, Guillermo Sapiro, Hand-Held Video Deblurring Via Efficient Fourier Aggregation, 2015