Burst photography for high dynamic range and low-light imaging on mobile cameras

Samuel W. Hasinoff Dillon Sharlet Ryan Geiss Andrew Adams
Jonathan T. Barron Florian Kainz Jiawen Chen Marc Levoy
Google Research

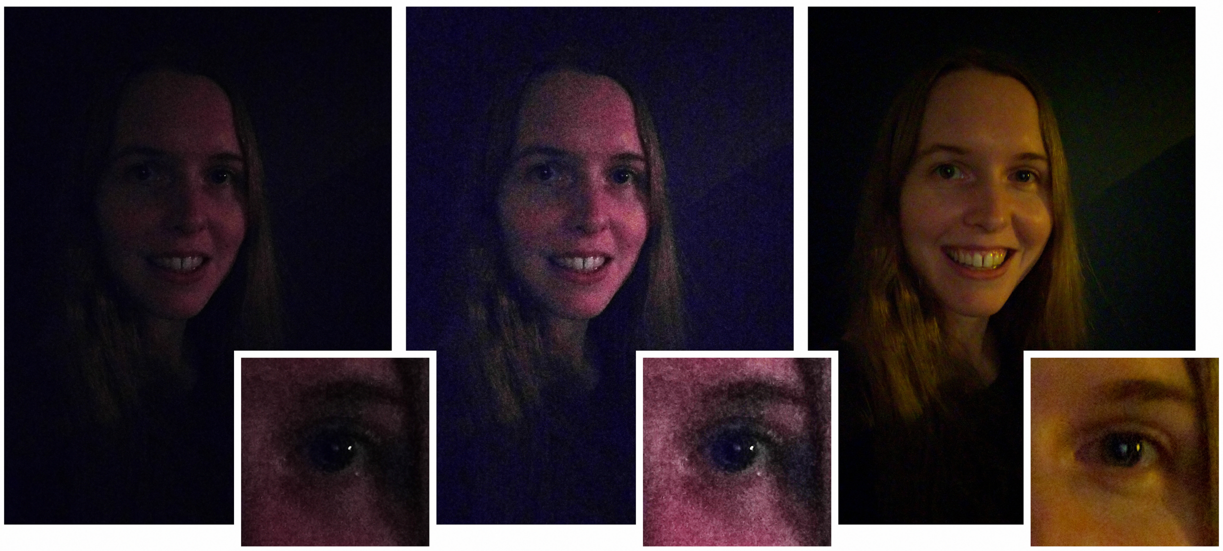


Figure 1: *A comparison of a conventional camera pipeline (left, middle) and our burst photography pipeline (right) running on the same cell-phone camera. In this low-light setting (about 0.7 lux), the conventional camera pipeline under-exposes (left). Brightening the image (middle) reveals heavy spatial denoising, which results in loss of detail and an unpleasantly blotchy appearance. Fusing a burst of images increases the signal-to-noise ratio, making aggressive spatial denoising unnecessary. We encourage the reader to zoom in. While our pipeline excels in low-light and high-dynamic-range scenes (for an example of the latter see figure 10), it is computationally efficient and reliably artifact-free, so it can be deployed on a mobile camera and used as a substitute for the conventional pipeline in almost all circumstances. For readability the figure has been made uniformly brighter than the original photographs.*

Abstract

Cell phone cameras have small apertures, which limits the number of photons they can gather, leading to noisy images in low light. They also have small sensor pixels, which limits the number of electrons each pixel can store, leading to limited dynamic range. We describe a computational photography pipeline that captures, aligns, and merges a burst of frames to reduce noise and increase dynamic range. Our solution differs from previous HDR systems in several ways. First, we do not use bracketed exposures. Instead, we capture frames of constant exposure, which makes alignment more robust, and we set this exposure low enough to avoid blowing out highlights. The resulting merged image has clean shadows and high bit depth, allowing us to apply standard HDR tone mapping methods. Second, we begin from Bayer raw frames rather than the demosaicked RGB (or YUV) frames produced by hardware Image Signal Processors (ISPs) common on mobile platforms. This gives us more bits per pixel and allows us to circumvent the ISP's unwanted tone mapping and spatial denoising. Third, we use a novel FFT-based alignment algorithm and a hybrid 2D/3D Wiener filter to denoise and merge the frames in a burst. Our implementation is built atop Android's Camera2 API, which provides per-frame camera control and access to raw imagery, and is written in the Halide domain-specific language (DSL). It runs in 4 seconds on device (for a 12 Mpix image), requires no user intervention, and ships on several mass-produced cell phones.

Keywords: computational photography, high dynamic range

1 Introduction

The main technical impediment to better photographs is lack of light. In indoor or night-time shots, the scene as a whole may provide insufficient light. The standard solution is either to apply analog or digital gain, which amplifies noise, or to lengthen exposure time, which causes motion blur due to camera shake or subject motion. Surprisingly, daytime shots with high dynamic range may also suffer from lack of light. In particular, if exposure time is reduced to avoid blowing out highlights, then insufficient light may be collected in shadowed areas. These areas can be brightened using local tone-mapping, but this again amplifies noise.

Ways to gather more light include using a larger-aperture lens, optical image stabilization, exposure bracketing, or flash. However, each method is a tradeoff. If the camera is a cell phone, then it is thickness-constrained, so making its aperture larger is difficult. Such devices are also power-constrained, making it challenging to create a synthetic aperture by increasing the number of cameras [Wilburn et al. 2005]. Optical image stabilization allows longer exposures while minimizing camera shake blur, but it cannot control blur caused by subject motion. With exposure bracketing followed by image fusion, different parts of the fused image represent the scene at different times, which makes it hard to achieve a single self-consistent composition. The most frequent artifact caused by incorrect fusion is ghosting (figure 2a), due to the difficulty of aligning images captured at different times. Sensors that alternate exposure times between adjacent scanlines ameliorate ghosting somewhat, but sacrifice detail and make accurate demosaicking difficult. To many photographers, an on-camera flash is the least palatable option. It adds light, but can change the scene in an unpleasant way. Flash/no-flash photography [Petschnigg et al. 2004] addresses this issue but is not sufficiently robust.

In this paper we describe a camera system that addresses these problems by capturing a burst of images and combining them with dynamic range compression. While algorithms for doing this are well known [Debevec and Malik 1997], building a system based on these algorithms and deploying it commercially on a mobile camera is challenging. In building our system we have found the following design principles to be important:

- **Be immediate.** The system must produce a photograph within a few seconds, and display it on the camera, even when the camera is not connected (wired or wirelessly). This means we cannot defer processing to a desktop computer or the cloud.
- **Be automatic.** The method must be parameter-free and fully automatic. Photographers should get better pictures without knowing the strategy used for capture or image processing.
- **Be natural.** The photographs we produce must be faithful to the appearance of the scene. In high-dynamic-range situations we must therefore limit the amount of local tonemapping we do to avoid cartoonish or surrealistic images. In very low-light scenes we must not brighten the image so much that it changes the apparent illumination or reveals excessive noise.
- **Be conservative.** It should be possible to use this as the default picture-taking mode. This means that the photographs produced must not contain artifacts, and must always be at least as good as conventional photographs. Moreover, in extreme situations it must degrade gradually to a conventional photograph.

Given this *conservative* constraint, we have found the most reliable approach to burst mode photography is to capture each image in the burst with the same exposure time. In other words we do not bracket. We arrived at this unexpected protocol because of the inherent difficulty in accurately aligning images captured using different exposure times. Small exposure variation may compromise alignment due to differing levels of noise and motion blur, and large variation may render local alignment impossible if a patch is exposed with no image content visible. Recent HDR fusion methods, e.g. [Hu et al. 2013], address this with sophisticated alignment and inpainting, but can produce physically inconsistent results.

Given this protocol, we choose an exposure that is low enough to avoid clipping (blowing out highlights) for the given scene. In other words we deliberately down-expose. We do this to capture more dynamic range. We also choose shorter than typical exposure times to mitigate camera shake blur, regardless of scene content [Telleen et al. 2007]. Although using lower exposures would seem to worsen noise, we offset this effect by capturing and merging multiple frames.

A second design decision arising from our *conservative* constraint is that we select one of the images in the burst as a "reference" frame, then align and merge into this frame those patches from other "alternate" frames where we are confident that we have imaged the same portion of the scene. Although we would gain the most signal-to-noise by being liberal and merging many patches, we choose instead to be conservative, merging image content only if the alternate patch appears similar to the reference patch. Furthermore, to reduce computational complexity, we merge only a single patch from each alternate frame. Our conservative merging strategy may cause some parts of the final image to appear noisier than others, but this artifact is seldom noticeable.

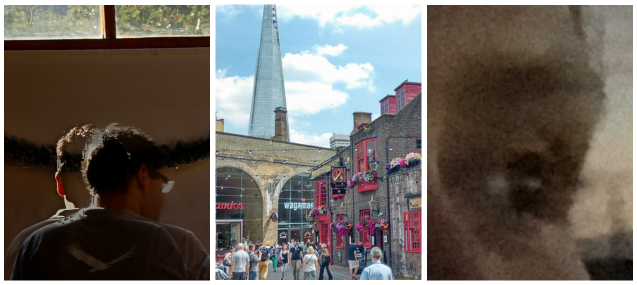

(a) *ghosting* **(b)** *excessive range compression* **(c)** *excessive denoising*

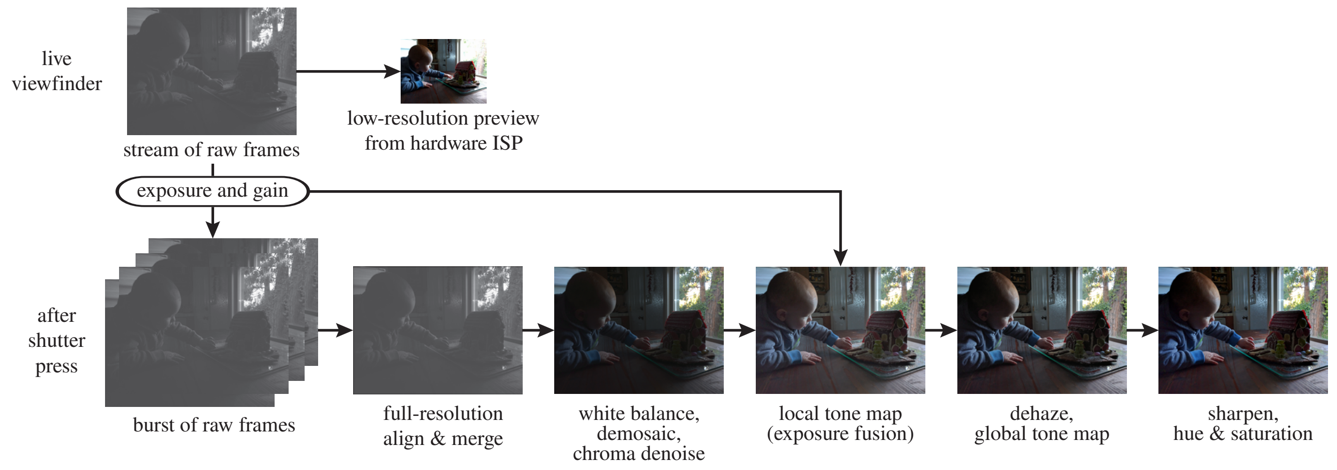
Figure 2: *Typical image processing artifacts for high dynamic range and low-light scenes. (a) Ghosting due to misalignment of different exposures. (b) Excessive range compression and saturation produces a flat "cartoony" rendition. (c) Excessive denoising under low illumination causes loss of fine detail and "splotchy" textures.*

By aligning and merging multiple frames, we produce an intermediate image with higher bit depth, higher dynamic range, and reduced noise compared to our input frames. This would let us produce a high-quality (albeit under-exposed) photograph merely by discarding the low-order bits. However, one of our goals is to produce *natural-looking* photographs even if the scene contains strong contrast. Therefore, we instead boost shadows, preserving local contrast while judiciously sacrificing global contrast. This process is called HDR tone mapping, and has been well studied [Reinhard et al. 2010]. Its effect is similar to that produced by traditional "dodging and burning" methods in print photography [Adams 1981]. We use a variant of exposure fusion [Mertens et al. 2007], because it is computationally efficient and produces natural-looking images; however, other algorithms are possible.

One challenge in writing a systems paper about a commercial computational photography system is that academic papers in this area describe only algorithms, not complete systems, and the algorithms in existing commercial systems are proprietary, not described in available textbooks, and not easily reverse-engineered. This situation is worse in the camera industry than in the computer graphics community, where textbooks do exist and public APIs have led to a tradition of openness and comparison [Levoy 2010]. This secrecy makes it hard for us to compare our results quantitatively with competing systems. To address this issue, we have structured this paper around an enumeration of design principles, a description of our implementation, and a sampling of our results—good and bad. We also present in supplemental material a detailed comparison with several state of the art JPEG-based fusion methods [Liu et al. 2014; Dabov et al. 2007a; Adobe Inc. 2016], evaluating our method for aligning and merging frames in isolation from the rest of our system. Finally, we have created an archive of several thousand raw input bursts (with associated capture metadata) and output photographs [Google Inc. 2016b], so others can improve upon or compare against our technique.

2 Overview of capture and processing

Figure 3 summarizes our processing system, which consists of a real-time pipeline (top row) that produces a continuous low-resolution viewfinder stream, and a non-real-time pipeline (bottom row) that produces a single high-resolution image.



live viewfinder
stream of raw frames
low-resolution preview from hardware ISP
exposure and gain
after shutter press
burst of raw frames
full-resolution align & merge
white balance, demosaic, chroma denoise
local tone map (exposure fusion)
dehaze, global tone map
sharpen, hue & saturation

Figure 3: *Overview of our two processing pipelines. The input to both pipelines is a stream of Bayer mosaic (raw) images at full sensor resolution (for example, 12 Mpix) at up to 30 frames per second. When the camera app is launched, only the viewfinder (top row) is active. This pipeline converts raw images into low-resolution images for display on the mobile device's screen, possibly at a lower frame rate. In our current implementation the viewfinder is 1.6 Mpix and is updated at 15–30 frames per second. When the shutter is pressed, this pipeline suspends briefly, a burst of frames is captured at constant exposure, stored temporarily in main memory, and the software pipeline (bottom row) is activated. This pipeline aligns and merges the frames in the burst (sections 4 and 5), producing a single intermediate image of high bit depth, then applies color and tone mapping (section 6) to produce a single full-resolution 8-bit output photograph for compression and storage in flash memory. In our implementation this photograph is 12 Mpix and is computed in about 4 seconds on the mobile device.*

In our current implementation the viewfinder stream is computed by a hardware Image Signal Processor (ISP) on the mobile device's System on a Chip (SoC). By contrast the high-resolution output image is computed in software running on the SoC's application processor. To achieve good performance this software is written in Halide [Ragan-Kelley et al. 2012]. We utilize an ISP to handle the viewfinder because it is power efficient. However, its images look different than those computed by our software. In other words our viewfinder is not WYSIWYG. Addressing this problem is future work.

A key enabling technology for our approach is the ability to request a specific exposure time and gain for each frame in a burst. For this we employ the Camera2 API [Google Inc. 2016a] available on select Android phones. Camera2 utilizes a request-based architecture based on the Frankencamera [Adams et al. 2010]. Another advantage of Camera2 is that it provides access to Bayer raw imagery, allowing us to bypass the ISP. As shown in figure 3 we use raw imagery in two places: (1) to determine exposure and gain from the same stream used by the ISP to produce the viewfinder, and (2) to capture the burst used to compute a high-resolution photograph. Using raw images conveys several advantages:

- **Increased dynamic range.** The pixels in raw images are typically 10 bits, whereas the YUV (or RGB) pixels produced by mobile ISPs are typically 8 bits. The actual advantage is less than 2 bits, because raw is linear and YUV already has a gamma curve, but it is not negligible.
- **Linearity.** After subtracting a black level offset, raw images are proportional to scene brightness, whereas images output by ISPs include nonlinear tone mapping. Linearity makes it easier to estimate gain and exposure, and lets us model sensor noise accurately, making patch comparisons for alignment more reliable.
- **Portability.** Merging the images produced by an ISP entails modeling and reversing its processing, which is proprietary and scene dependent [Kim et al. 2012]. By starting from raw images we can omit these steps, which also makes it easier to port our system to new cameras.

One drawback of raw imagery is that we need to implement the entire photographic pipeline, including correction of lens shading and chromatic aberration, and demosaicking. (These correction steps have been omitted from figure 3 for brevity.) Fortunately, since our alignment and merging algorithm operates on raw images, the expensive demosaicking step need only be performed once—on a single aligned and merged image, rather than on every frame in the burst.

3 Estimating exposure for the burst

An important function of a mobile ISP is to continuously adjust exposure time, gain, focus, and white balance as the user aims the camera. In principle we could read back exposure and gain using the Camera2 API and use them when requesting our constant-exposure burst.

For scenes having a moderate dynamic range this works well. Typical autoexposure algorithms will strive to avoid blowing out highlights, or allow only a small percentage of pixels to remain overexposed, hoping that these pixels represent light sources or specular reflections that we are content to see rendered as white. However, for scenes having a high dynamic range this strategy does not work, because either too many pixels will be overexposed or too many pixels will be left dark. Our solution to this problem consists of three steps:

1. **deliberately underexpose** so that fewer pixels saturate,
2. **capture multiple frames** to reduce noise in the shadows, and
3. **compress the dynamic range** using local tone mapping.

Underexposure is a well-known approach for high dynamic range capture, popularized for digital SLRs as "expose to the right" [Martinec 2008]. What makes underexposure viable in our solution is its combination with the noise reduction provided by capturing a burst.

In effect we treat HDR imaging as denoising [Hasinoff et al. 2010; Zhang et al. 2010].

Given this solution we must choose how much to underexpose, how many frames to capture, and how much to compress the dynamic range. If we underexpose too much our photograph will be noisy even if we capture multiple frames, and we cannot capture an unlimited number since capture and merging take time and power. Conversely, if we compress the dynamic range too much our photograph will look cartoony (see figure 2b). We therefore clamp our maximum compression factor to 8. Fortunately, few scenes in the real world require more compression than this (see figure 4).

Auto-exposure using a database of examples A key difficulty in choosing exposure automatically in HDR situations is that this choice is often scene dependent. For example, it is usually acceptable to let the sun blow out, but if the scene is a sunset at the beach the sun should remain colored, and the rings of color around the sun should not be overexposed, even if the beach must be left dark. To address this problem we have created a database of scenes captured using traditional HDR bracketing, which we have hand-tuned to look as natural as possible when rendered using our tone mapping method (see section 6). The success of this approach depends on covering every kind of scene consumers are likely to encounter. Our database contains about $4,500$ labeled scenes, collected over the course of several years.

Given this labelled database and a viewfinder frame in raw format, we compute features of the frame and search our database for scenes that closely match it. The features we use are histogram quantiles, measured on a white-balanced and aggressively downsampled version of the viewfinder frame. We have found quantiles to be better features than histogram buckets when analyzing HDR scenes. Specifically, we compute four sets of 64 non-uniformly spaced quantiles, at two different scales, and for both the maximum and the average of the RGB channels. This helps us represent exposure at different frequencies while accounting for color clipping. In computing these quantiles, we apply a fixed weighting to favor the center of the image, and we strongly boost the weight of regions where faces are detected. We also restrict the set of candidates to examples whose luminance is within a factor of 8 of the current scene. This helps retain perception of scene brightness, avoiding, for example, unnatural day-for-night renditions.

Once we have found a set of candidate matching scenes, we compute a weighted blend of our hand-tuned parameters for those scenes. This blend yields two parameters: an overall exposure for capture, and an amount of dynamic range compression to apply to during tone mapping. Since the mapping from overall exposure (the product of exposure time, analog gain, and digital gain) to final image brightness is smooth and monotonic, we can recover the overall exposure using a few steps of binary search. Translating this overall exposure into capture settings entails factoring it into exposure time and gain (ISO). For this step we use a schedule that balances motion blur against noise. Specifically, for the brightest scenes we hold gain at its minimum level, allowing exposure times of up to $8\,\mathrm{ms}$. As scenes become darker this factorization changes. First we increase gain, up to $4\times$. Next, we begin to increase both exposure time and gain in parallel (in log space). This factorization proceeds up to the maximums of $100\,\mathrm{ms}$ exposure time, with $96\times$ gain. Camera sensors have a limited amount of analog gain they can apply; any gain above this limit we apply digitally in our pipeline.

In addition to determining exposure time, gain, and dynamic range compression, we must also decide how many frames to capture in a burst. The number we capture, N, is a tradeoff between signal-to-noise ratio, capture latency, and computation and memory. In low light, or in very high dynamic range scenes where we'll be boosting the shadows later, we want more frames to drive down noise, but they take more time and memory to capture, buffer, and process. In bright scenes, 1–2 images is usually sufficient, although more images are generally beneficial in combating camera shake blur. In practice, we limit our bursts to 2–8 images, informing the decision using our model for raw image noise (see section 5 for more detail).

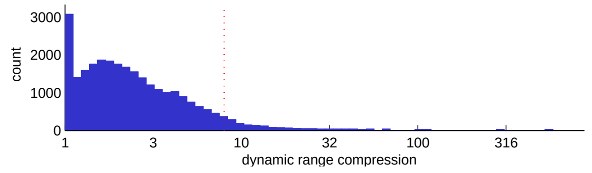


Figure 4: *The amount of dynamic range compression required to squeeze each scene in our quality verification database of 26,071 real world scenes into an 8-bit output image, as evaluated by the autoexposure algorithm described in section 3. The distribution shows that while most scenes benefit from some compression, indicated by a ratio more than 1, the recommended amount of compression is generally mild, and only 4% of scenes require a compression above 8 (dotted line). Such extreme scenes are hard to tone map in a way that looks natural.*

Although this algorithm takes only $10\,\mathrm{ms}$ to run in our implementation, analyzing every viewfinder frame is unnecessary, since rapid changes in scene brightness are uncommon. We therefore compute exposure on every 4th frame, mainly to save power.

One problem with our algorithm is that for a very HDR scene, a single viewfinder image will contain many overexposed pixels. This can make it tricky to estimate the right amount of underexposure to apply. To circumvent this problem we have experimented with continuous bracketing during viewfinding. However, differently-exposed images cannot be displayed to the user during viewfinding, and capturing them in the background disrupts the smoothness of the viewfinder. Fortunately, we have found that by designing our matching metric to be tolerant to clipped pixels, a single viewfinder image seldom produces bad matches. Our algorithm predicts exposures within 10% of the bracketing result for 87% of shots; the shots with larger variations tend to be both more strongly HDR and more forgiving to exposure precision.

Selecting a reference frame After capturing a burst of images, we have one more decision to make before beginning alignment—we must select a single *reference* frame to which all other frames are aligned. When the user releases the shutter, images are often blurred due to physical camera shake, which typically lasts for 50–$100\,\mathrm{ms}$. While optical image stabilization can mitigate this blur, we address blur induced by both hand and scene motion by choosing the reference frame to be the *sharpest* frame in the burst. This approach is commonly known as lucky imaging [Joshi and Cohen 2010]. To minimize perceived shutter lag, we restrict the reference frame to be from the first 3 frames in the burst.

4 Aligning Frames

In the context of our high-resolution pipeline, alignment consists of finding a dense correspondence from each alternate (non-reference) frame of our burst to the chosen reference frame. This correspondence problem is well-studied, with solutions ranging from optical flow [Horn and Schunk 1981; Lucas and Kanade 1981], which performs iterative optimization under assumptions of smoothness and brightness constancy, to more recent techniques that use patches or feature descriptors to construct and "densify" a sparse correspondence [Liu et al. 2011; Brox and Malik 2011], or that use image oversegmentations and directly reason about geometry and occlusion [Yamaguchi et al. 2014]. In the computer vision literature, optical flow techniques are evaluated primarily by quality on established benchmarks [Baker et al. 2011; Menze and Geiger 2015]. As a result, most techniques produce high-quality correspondences, but at a significant computational cost—at time of submission, the top 5 techniques on the KITTI optical flow benchmark [Menze and Geiger 2015] require between 1.7 and 107 minutes per megapixel in desktop environments.

Unfortunately, our strong constraints on speed, memory, and power preclude nearly all of these techniques. However, because our merging procedure (section 5) is robust to both small and gross alignment errors, we can construct a simple algorithm that meets our requirements. Much like systems for video compression [Wiegand et al. 2003], our approach is designed to strike a balance between computational cost and correspondence quality. Our alignment algorithm runs at 24 *milliseconds* per megapixel on a mobile device.

Handling raw images Because our input consists of Bayer raw images, alignment poses a special challenge. The four color planes of a raw image are undersampled, making alignment an ill-posed problem. Although we could demosaic the input to estimate RGB values for every pixel, running even a low-quality demosaic on all burst frames would be prohibitively expensive. We circumvent this problem by estimating displacements only up to a *multiple of* 2 *pixels*. Displacements subject to this constraint have the convenient property that displaced Bayer samples have coincident colors. In effect, our approach to Bayer alignment defers the undersampling problem to our merge stage, where image mismatch due to aliasing is treated like any other form of misalignment. We implement this by averaging 2×2 blocks of Bayer RGGB samples, so that we align downsampled 3 Mpix grayscale images instead of 12 Mpix raw images.

Hierarchical alignment To align an alternate frame to our reference frame, we perform a coarse-to-fine alignment on four-level Gaussian pyramids of the downsampled-to-gray raw input. As figure 5 illustrates, we produce a tile-based alignment for each pyramid level, using the alignments from the coarser scale as an initial guess. Each reference tile's alignment is the offset that minimizes the following distance measure relating it to candidate tiles in the alternate image:

$$D_p(u,v) = \sum_{y=0}^{n-1}\sum_{x=0}^{n-1} \left|T(x,y) - I(x+u+u_0, y+v+v_0)\right|^p \quad (1)$$

where T is a tile of the reference image, I is a larger search area of the alternate image, p is the power of the norm used for alignment (1 or 2, discussed later), n is the size of the tile (8 or 16, discussed later), and (u_0, v_0) is the initial alignment inherited by the tile from the coarser level of the pyramid.

The model in equation 1 implies several assumptions about motion in our bursts. We assume piecewise translation, which is true in the limit as the patch approaches a single pixel, but can be a limiting assumption for larger patches. By minimizing absolute error between image patches instead of, say, maximizing normalized cross-correlation, we are not invariant to changes in brightness and contrast. However, this is not a disadvantage, because camera exposure is fixed and illumination is unlikely to change quickly over the duration of our bursts.

Upsampling the coarse alignment to the next level of the pyramid is challenging when the coarse alignment straddles object or motion boundaries. In particular, standard upsampling methods like nearest-neighbor and bilinear interpolation can fail when the best displacement for an upsampled tile is not represented in the search area around the initial guess. In our system, we address this problem by evaluating multiple hypotheses for each upsampled alignment. We take as candidates the alignments for the nearest coarse-scale tiles, choosing the alignment with minimum L1 residual between the reference and alternate frames. This approach is similar in spirit to SimpleFlow [Tao et al. 2012], which also uses image content to inform the upsampling.

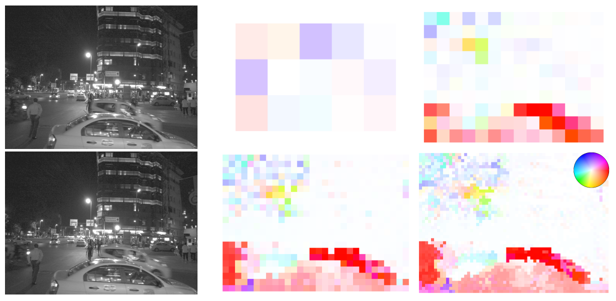

(a) *Image pair* **(b)** *Intermediate alignment fields*

Figure 5: *(a) A pair of 3 Mpix grayscale images. (b) The intermediate and final outputs of our multi-scale alignment, where hue and saturation indicate direction and magnitude of displacement (see the inset color circle). At the finest pyramid level (bottom right), tiles are* 32×32 *pixels and the maximum displacement is* 64 *pixels. The large regions of saturated colors show that a hierarchical algorithm is essential; our method supports displacements up to* 169 *pixels. Although our displacements contain errors, they are cheap to compute and sufficiently accurate to use as input to our merging stage.*

In our approach we make a number of heuristic decisions regarding decimation, patch size, search radius, and the choice of norm in equation 1. One crucial decision is to align differently depending on pyramid scale. In particular, at coarse scales we compute a sub-pixel alignment, minimize L2 residuals, and use a large search radius. Sub-pixel alignment is valuable at coarse scales because it increases the accuracy of initialization and allows aggressive pyramid decimation. At the finest scale of our pyramid, we instead compute pixel-level alignment, minimize L1 residuals, and limit ourselves to a small search radius. Only pixel-level alignment is needed here, as our current merging procedure cannot make use of sub-pixel alignment. More detail explaining these decisions, plus a description of how the computation of D_1 can be made fast with a brute-force implementation, can be found in the supplement. However, because D_2 uses a larger search radius, it must be made efficient using algorithmic techniques.

4.1 Fast subpixel L2 alignment

As just explained, naïvely computing equation 1 over a large search radius is prohibitively expensive. However, similar to the way normalized cross-correlation can be accelerated [Lewis 1995], the L2 version of equation 1 can be computed more efficiently with a box filter and a convolution:

$$D_2 = \|T\|_2^2 + \operatorname{box}(I \circ I, n) - 2\left(\mathcal{F}^{-1}\left\{\mathcal{F}\{I\}^* \circ \mathcal{F}\{T\}\right\}\right) \quad (2)$$

where the first term is the sum of the squared elements of T, the second term is the squared elements of I filtered with a non-normalized box filter of size $n \times n$ (the same size as T), and the third term is proportional to the cross-correlation of I and T, computed efficiently using a fast Fourier transform. For a complete derivation, see the supplement.

Having computed D_2 it is cheap to identify the integer displacement $(\hat{u}, \hat{v})$ that minimizes the displacement error. To produce a subpixel estimate of motion, we fit a bivariate polynomial to the 3×3 window surrounding $(\hat{u}, \hat{v})$ and find the minimum of that polynomial. This improves on the standard approach of fitting two separable functions [Stone et al. 2001] by avoiding the assumption that motion is independently constrained to the respective axes. Formally, we approximate:

$$D_2(u, v) \approx \frac{1}{2}[u\ v]\,\mathbf{A}\begin{bmatrix}u\\v\end{bmatrix} + \mathbf{b}^{\mathrm{T}}\begin{bmatrix}u\\v\end{bmatrix} + c \tag{3}$$

where $\mathbf{A}$ is a 2×2 positive semi-definite matrix, $\mathbf{b}$ is a 2×1 vector, and c is a scalar. We construct a weighted least-squares problem fitting a polynomial to the 3×3 patch of D_2 centered around $(\hat{u}, \hat{v})$. Solving this system is equivalent to taking the inner product of D_2 with a set of six 3×3 filters, derived in the supplement, each corresponding to a free parameter in $(\mathbf{A}, \mathbf{b}, c)$. The process is similar to the polynomial expansion approach of [Farnebäck 2002]. Once we have recovered the parameters of the quadratic, its minimum follows by completing the square:

$$\boldsymbol{\mu} = -\mathbf{A}^{-1}\mathbf{b} \tag{4}$$

The vector $\boldsymbol{\mu}$ represents the sub-pixel translation that must be added to our integer displacement $(\hat{u}, \hat{v})$.

5 Merging Multiple Frames

The key premise of burst photography is that we can realize noise reduction by combining multiple observations of the scene over time. However, to be useful in a photographic application, our merging method must be robust to alignment failures. As figure 6 shows, while alignment is important to help compensate for camera and object motion, we cannot rely on alignment alone, which can fail for a variety of reasons, such as occlusions, non-rigid motion, or changes in lighting.

With our performance goals in mind, we develop a merging method that is robust to misalignment, based on a pairwise frequency-domain temporal filter operating on the tiles of the input. In our setting, each tile in the reference is merged with one tile taken from each of the alternate frames, corresponding to the result of our alignment. Our approach takes inspiration from frequency-domain video denoising techniques that operate on 3D stacks of matching images patches [Kokaram 1993; Bennett and McMillan 2005; Dabov et al. 2007a]. In particular, Kokaram [1993] proposed a variant of classic Wiener filtering in the 3D DFT domain, attenuating small coefficients more likely to be noise.

V-BM3D [Dabov et al. 2007a] takes a similar approach, reinterpreting the Wiener filter and similar operators as "shrinkage" operators favoring the sparsity that is a statistical property of natural images in the transform domain. Techniques in this family are robust to misalignment because, for a given spatial frequency, any mismatch to the reference that cannot be ascribed to the expected noise level will be suppressed.

We adopt this strategy but depart from these methods in several ways. First, because we process raw images we have a simple model describing noise in the image. This improves robustness by letting us more reliably discriminate between alignment failures and noise. Second, instead of applying the DFT or another orthogonal transformation in the temporal dimension, we use a simpler pairwise filter, merging each alternate frame onto the reference frame independently. While this approach sacrifices some noise reduction for well-aligned images, it is cheaper to compute and degrades more gracefully with alignment failures (see figure 7). Third, as a consequence of this filter operating only over the temporal dimension, we run spatial denoising in a separate post-processing step, applied in the 2D DFT. Fourth, we apply our filter to the color planes of Bayer raw images independently, then reinterpret the filtered result as a new Bayer image. This method is simple but surprisingly robust, in that we observe little degradation even though we are ignoring Bayer undersampling. In the following, we expand on each these points and discuss artifacts that can result in extreme conditions.

Noise model and tiled approximation Because we operate on Bayer raw data, noise is independent for each pixel and takes a simple, signal-dependent form. In particular, for a signal level of x, the noise variance σ^2 can be expressed as $Ax + B$, following from the Poisson-distributed physical process of photon counting [Hasinoff et al. 2010]. The parameters A and B depend only on the analog and digital gain settings of the shot, which we control directly. To validate this model of sensor noise, we empirically measured how noise varies with different signal levels and gain settings.

In the transform domain where we apply our filtering, directly using a signal-dependent model of noise is impractical, as the DFT requires representing a full covariance matrix. While this could be addressed by applying a variance stabilizing transform [Mäkitalo and Foi 2013] to the input, for computational efficiency we instead approximate the noise as signal independent within a given tile. For each tile, we compute the variance by evaluating our noise model using the root-mean-square (RMS) of the samples in the windowed tile, normalized for the gain of the window function. Using RMS has the effect of biasing the signal estimate toward brighter image content. For low-contrast tiles, this is similar to using the mean; high-contrast tiles will be filtered more aggressively, as if they had a higher average signal level.

Robust pairwise temporal merge Our merge method operates on image tiles in the spatial frequency domain. For a given reference tile, we assemble a set of corresponding tiles across the burst, one per frame, and compute their respective 2D DFTs as $T_z(\boldsymbol{\omega})$, where $\boldsymbol{\omega} = (\omega_x, \omega_y)$ denotes spatial frequency, z is the frame index, and, without loss of generality, we take frame 0 to be the reference.

Where our method departs from other frequency-based denoising methods is our pairwise treatment of frames in the temporal dimension. To build intuition, a simple way to merge over the temporal dimension would be to compute the average for each frequency coefficient. This naïve averaging filter can be thought of as expressing an estimate for the denoised reference frame:

$$\tilde{T}_0(\boldsymbol{\omega}) = \frac{1}{N}\sum_{z=0}^{N-1} T_z(\boldsymbol{\omega}) \tag{5}$$

While this performs well when alignment is successful, it is not robust to alignment failure (see figure 6c). Because the 2D DFT is linear, this filter is actually equivalent to a temporal average in the spatial domain.

To add robustness, we instead construct an expression similar to equation 5, but incorporate a filter that lets us control the contribution of alternate frames:

$$\tilde{T}_0(\boldsymbol{\omega}) = \frac{1}{N}\sum_{z=0}^{N-1} T_z(\boldsymbol{\omega}) + A_z(\boldsymbol{\omega})[T_0(\boldsymbol{\omega}) - T_z(\boldsymbol{\omega})] \tag{6}$$

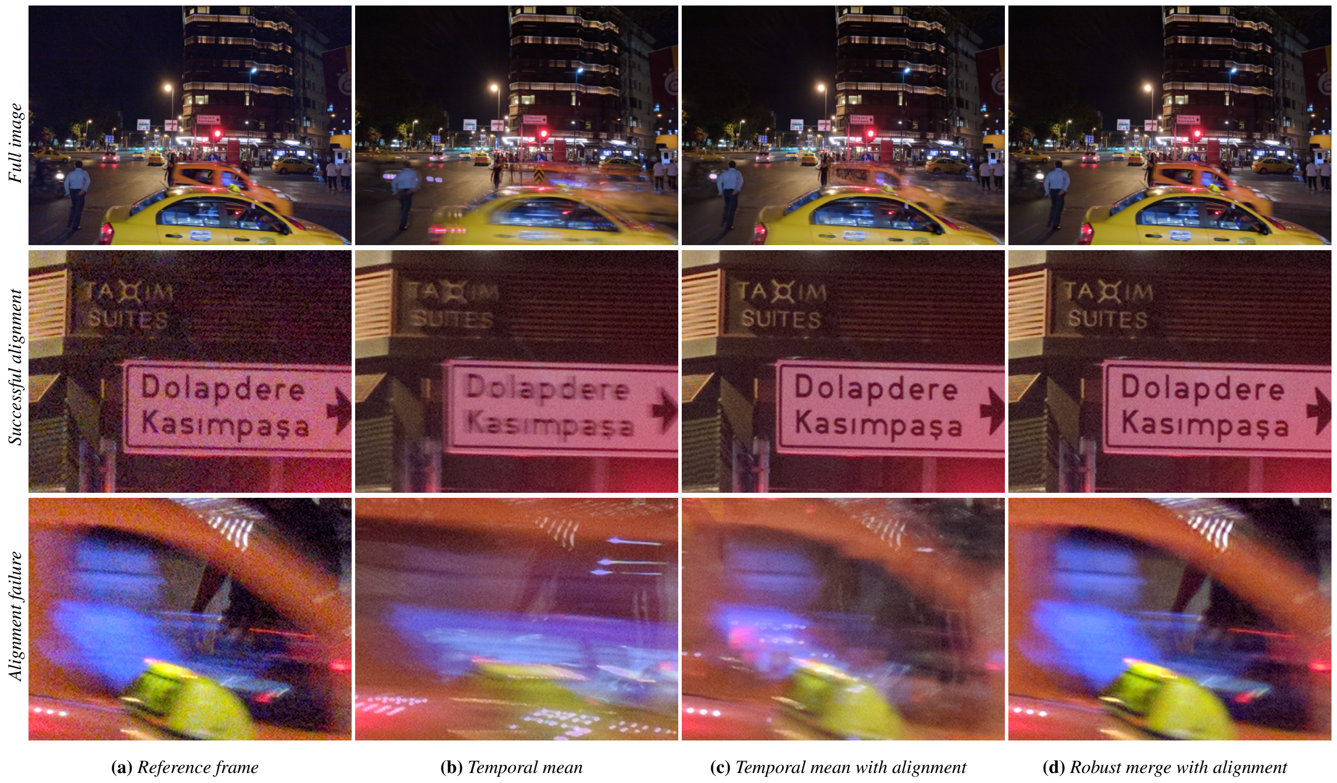


Figure 6: *Merging 8 frames of a moving scene. The top row is the full image, the middle row shows a crop where alignment succeeds, and the bottom row shows a crop where alignment partially fails. (a) One frame from the burst is chosen as the reference. (b) Averaging all 8 frames without alignment produces ghosts in regions exhibiting motion. (c) Averaging with alignment eliminates ghosts in some regions (middle), but fails in others (bottom). When alignment is successful (middle row), the temporal mean resembles the reference; however, when alignment fails (bottom) the mean is different from the reference frame, leaving ghosts. (d) The result of our robust merge closely resembles the reference frame even when alignment fails. Despite alignment failure, some features are partially denoised. Note the yellow roof light in the foreground and the person in the background.*

For a given frequency, A_z controls the degree to which we merge alternate frame z into the final result versus falling back to the reference frame. The body of this sum can be rewritten as $(1 - A_z) \cdot T_z + A_z \cdot T_0$ to emphasize that A_z controls a linear interpolation between T_z and T_0. Since the contribution of each alternate frame is adjusted on a per-frequency basis, alignment failure can be partial, in that rejected image content for one spatial frequency will not corrupt other frequencies.

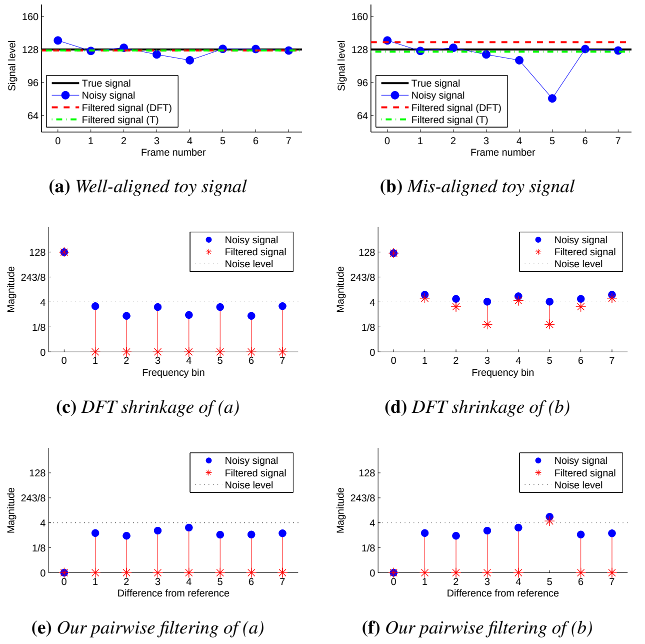
We are now left with the task of defining A_z to attenuate frequency coefficients that do not match the reference. In particular, we want T_z to contribute to the merged result when its difference from T_0 can be ascribed to noise, and for its contribution to be suppressed when it differs from T_0 due to poor alignment or other problems. In other words, A_z is a shrinkage operator. Our definition of A_z is a variant of the classic Wiener filter:

$$A_z(\boldsymbol{\omega}) = \frac{|D_z(\boldsymbol{\omega})|^2}{|D_z(\boldsymbol{\omega})|^2 + c\sigma^2} \qquad (7)$$

where $D_z(\boldsymbol{\omega}) = T_0(\boldsymbol{\omega}) - T_z(\boldsymbol{\omega})$, the noise variance σ^2 is provided by our noise model, and c is a constant that accounts for the scaling of noise variance in the construction of D_z and includes a further tuning factor (in our implementation, fixed to 8) that increases noise reduction at the expense of some robustness. The construction of D_z scales the noise variance by a factor of n^2 for the number of 2D DFT samples, a factor of $1/4^2$ for the window function, and a factor of 2 for its definition as a difference of two tiles. We tried several alternative shrinkage operators, such as hard and soft thresholding [Donoho 1995], and found this filter to provide the best balance between noise reduction strength and visual artifacts.

We found our pairwise temporal operator to produce higher quality images than a full 3D DFT, particularly in the presence of alignment failure. As figure 7 illustrates, a single poorly aligned frame renders the entire DFT transform domain non-sparse, leading the shrinkage operator to reject the contribution from all of the alternate frames, not only the poorly aligned one. By contrast, our temporal operator evaluates the contribution of each alternate frame independently, letting us degrade more gracefully with alignment failure. Our temporal filtering also has the advantage of being cheaper to compute and requiring less memory to evaluate. The contribution of each alternate frame can be computed and discarded before moving on to the next.

Spatial denoising Because our pairwise temporal filter above does not perform any spatial filtering, we apply spatial filtering as a separate post-processing step in the 2D DFT domain. Starting from the temporally filtered result, we perform spatial filtering by applying a pointwise shrinkage operator, of the same form as equation 7, to the spatial frequency coefficients. To be conservative, we limit the strength of denoising by assuming that all N frames were averaged perfectly. Accordingly, we update our estimate of the noise



(a) *Well-aligned toy signal* **(b)** *Mis-aligned toy signal*

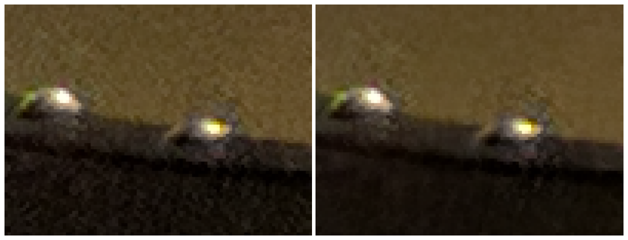
(c) *DFT shrinkage of (a)* **(d)** *DFT shrinkage of (b)*

(e) *Our pairwise filtering of (a)* **(f)** *Our pairwise filtering of (b)*

Figure 7: *Behavior of temporal filtering with alignment failure. For illustration, we created a toy sequence by sampling a single noisy pixel ($\sigma = 4$). When alignment is successful (left column), all differences from the reference (frame 0) are due to noise. In the well-aligned case, the DFT domain signal is strongly concentrated in the DC bin. Applying a pointwise shrinkage operator similar to equation 7 suppresses noise, for both the DFT (c) and our robust pairwise merge (e). The filtered output signal at the reference frame, as shown in (a), is very close to the true signal. When alignment is unsuccessful (right column), the two methods behave differently, even in the presence of a single outlier (frame 5). In the DFT domain (d), the outlier raises all coefficients above the noise level, i.e., makes the signal non-sparse, which reduces the effectiveness of shrinkage. In contrast, our pairwise temporal filter (f) allows shrinkage to be effective on all but the misaligned frame. The net result is that our robust pairwise merge has significantly more denoising than the DFT, producing an output signal closer to the true signal (b). For the DFT, a single outlier is enough to make the result degrade conservatively to the noisy reference signal.*

variance to be σ^2/N. In our experiments, we found that we can filter high spatial frequency content more aggressively than lower spatial frequency content without introducing noticeable artifacts. Therefore, we apply a "noise shaping" function $\tilde{\sigma} = f(\boldsymbol{\omega})\,\sigma$ which adjusts the effective noise level as a function of $\boldsymbol{\omega}$, increasing its magnitude for higher frequencies. We represent this function by defining a piecewise linear function, tuned to maximize subjective image quality.

Merging Bayer raw Note that up to this point, we have presented our merging algorithm in terms of single-channel images. However, as mentioned above, both our input and output consist of Bayer-mosaicked raw images. Our design handles raw images in the simplest way possible: we merge each plane of the Bayer image independently, and we do not use alignment any more precise than pixel level in the Bayer color planes. Aligning to higher precision would require interpolation for both align and merge, which would increase computational cost significantly. While our approach is fast and effective, it is less sophisticated than multi-frame demosaicking algorithms (e.g., [Farsiu et al. 2006]) designed to recover high



(a) *Without spatial denoising* **(b)** *With spatial denoising*

Figure 8: *Spatial denoising failing to suppress noise around strong high contrast features. Note the spatial denoising is effective throughout the image, except near the strong specular highlights on the metallic rivets.*

frequency content lost to Bayer undersampling.

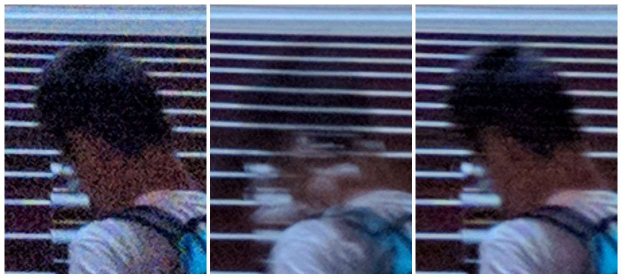
Because Bayer color planes are undersampled by a factor of four, one might pessimistically assume that 75% of frames will be rejected on average, leading to compromised denoising. While our robust filter will indeed reject aliased image content not fitting our noise model, this rejection only happens on a per-DFT bin basis and aliasing issues are likely to be confined to a subset of DFT bins. The same behavior can be observed in figure 6, where despite poor alignment (figure 6c, bottom), our robust temporal filter is able to significantly reduce noise without introducing any visible ghosting (figure 6d, bottom).

Overlapped tiles Our merge method operates on tiles overlapped by half in each spatial dimension. By smoothly blending between overlapped tiles, we avoid visually objectionable discontinuities at tile boundaries. Additionally, we must apply a window function to the tiles to avoid edge artifacts when operating in the DFT domain. We use a modified raised cosine window, $\frac{1}{2} - \frac{1}{2}\cos(2\pi(x + \frac{1}{2})/n)$ for $0 \leq x < n$, and 0 otherwise. This differs from the conventional definition: first, the denominator of the cosine argument is n, not $n-1$. Unlike the conventional window, when this function is repeated with $n/2$ samples of overlap, the total contribution from all tiles sum to one at every position. Second, the window is shifted by half to avoid zeros in the window resulting from the modified denominator. Zeros in the window correspond to pixels not contributing to the output, which implies we could have used a smaller tile size (with the associated computational savings) to achieve the same result.

Artifacts We have observed several classes of artifacts resulting from this system. First, this filter tends to fail to suppress noise around strong high contrast features, as shown in figure 8. This is a result of high contrast features having a non-sparse representation in the spatial DFT domain, reducing the effectiveness of spatial denoising.

Second, because our shrinkage function never fully rejects a poorly aligned tile, mild ghosting artifacts can sometimes occur, as shown in figure 9. In our experience, these ghosting artifacts are subtle, and very often are difficult to distinguish from motion blur.

Finally, our filter can occasionally produce ringing artifacts typically associated with frequency-domain filters. While ringing is largely mitigated by our windowing approach, in challenging situations classic Gibbs phenomenon can be visible, particularly after being amplified by sharpening and other steps in our finishing pipeline. Ringing is most frequently visible in the neighborhood of poorly-aligned clipped highlights, which exhibit high spatio-temporal contrast. In our experience, ringing has a negligible visual effect for most scenes.



(a) *Reference frame* **(b)** *Temporal mean with alignment* **(c)** *Robust merge*

Figure 9: *Example of subtle ghosting artifacts produced by our merge algorithm due to large motion. Note the horizontal structure from the benches visible through the moving person's head in the robust merged result (c), which is not visible in the reference frame (a). The temporal mean with alignment is included to demonstrate the scale of motion and degree of alignment failure (b).*

6 Finishing

Aligning and merging the captured Bayer raw frames produces a single raw image with higher bit depth and SNR. This image must now undergo correction, demosaicking, and tone mapping–operations that would normally be performed by an ISP, but in our case is implemented are software and include the key additional step of dynamic range compression. In order of application, these operations are:

1. **Black-level subtraction** deducts an offset from all pixels, so that pixels receiving no light become zero. We obtain this offset from optically shielded pixels on the sensor.
2. **Lens shading correction** brightens the corners of the image to compensate for lens vignetting and corrects for spatially varying color due to light striking the sensor at an oblique angle. These corrections are performed using a low-resolution RGGB image supplied by the ISP.
3. **White balancing** linearly scales the four (RGGB) channels so that grays in the scene map to grays in the image. These scale factors are supplied by the ISP.
4. **Demosaicking** converts the image from a Bayer raw image to a full-resolution linear RGB image with 12 bits per pixel. We use a combination of techniques from Gunturk et al. [2005], including edge directed interpolation with weighted averaging, constant-hue based interpolation, and second order gradients as correction terms.
5. **Chroma denoising** to reduce red and green splotches in dark areas of low-light images. For this we use an approximate bilateral filter, implemented using a sparse 3x3 tap non-linear kernel applied in two passes in YUV.
6. **Color correction** converts the image from sensor RGB to linear sRGB using a 3x3 matrix supplied by the ISP.
7. **Dynamic range compression** See description below.
8. **Dehazing** reduces the effect of veiling glare by applying a global tone curve that pushes low pixel values even lower while preserving midtones and highlights. Specifically, we allow up to 0.1% of pixels to be clamped to zero, but only pixels below 7% of the white level.
9. **Global tone adjustment**, to increase contrast and apply sRGB gamma correction, by concatenating an S-shaped contrast-enhancing tone curve with the standard sRGB color component transfer function.
10. **Chromatic aberration correction** to hide lateral and longitudinal chromatic aberration. We do not assume a lens model, but instead look for pixels along high-contrast edges, and replace their chroma from nearby pixels less likely to be affected by chromatic aberration.
11. **Sharpening** using unsharp masking, implemented using a sum of Gaussian kernels constructed from a 3-level convolution pyramid [Farbman et al. 2011].
12. **Hue-specific color adjustments** to make blue skies and vegetation look more appealing, implemented by shifting bluish cyans and purples towards light blue, and increasing the saturation of blues and greens generally.
13. **Dithering** to avoid quantization artifacts when reducing from 12 bits per pixel to 8 bits for display, implemented by adding blue noise from a precomputed table.

Dynamic range compression For high dynamic range scenes we use local tonemapping to reduce the contrast between highlights and shadows while preserving local contrast. The tonemapping method we have chosen is a variant of exposure fusion [Mertens et al. 2007]. Given input images that depict the same scene at different brightness levels, exposure fusion uses image pyramids to blend the best-exposed parts of the input images to produce a single output image that looks natural and has fewer badly exposed areas than the inputs.

Exposure fusion is typically applied to images captured using bracketing. In our pipeline we capture multiple frames with constant exposure, not bracketing. To adapt exposure fusion to our pipeline, we derive "synthetic exposures" from our intermediate HDR image by applying gain and gamma correction to it, then fuse these as if they had been captured using bracketing. We perform these extractions in grayscale, and we create only two synthetic exposures–one short and one long. The short exposure tells us how many pixels will blow out, and becomes the overall exposure used during capture, while the ratio between the short and long exposures tells us how much dynamic range compression we are applying. Both values come from our autoexposure algorithm.

Fusing grayscale instead of color images, and using only two synthetic exposures, reduces computation and memory requirements. It also allows us to simplify the per-pixel blend-weights compared to those in the work by Mertens et al. [2007]. In particular, we use a fixed weighting function of luma that favors moderately bright pixels. This function can be expressed as a one-dimensional lookup table. After fusing the synthetic exposures we undo the gamma-correction of the resulting grayscale image and re-colorize it by copying per-pixel chroma ratios from the original linear RGB image.

7 Results

Figure 10 shows example photos taken with our system side-by-side with single-exposure photos produced by a conventional imaging pipeline. Our system almost always produces results superior to a conventional single-exposure pipeline, and in scenes with high dynamic range or low light the improvement is often dramatic–fewer blown-out highlights or crushed shadows, less noise, less motion blur, better color, sharper details, and more texture.

For a more detailed evaluation of our system's align and merge method, demonstrating its robustness compared to state of the art JPEG-based fusion [Liu et al. 2014; Dabov et al. 2007a; Adobe Inc. 2016], please refer to the supplement.

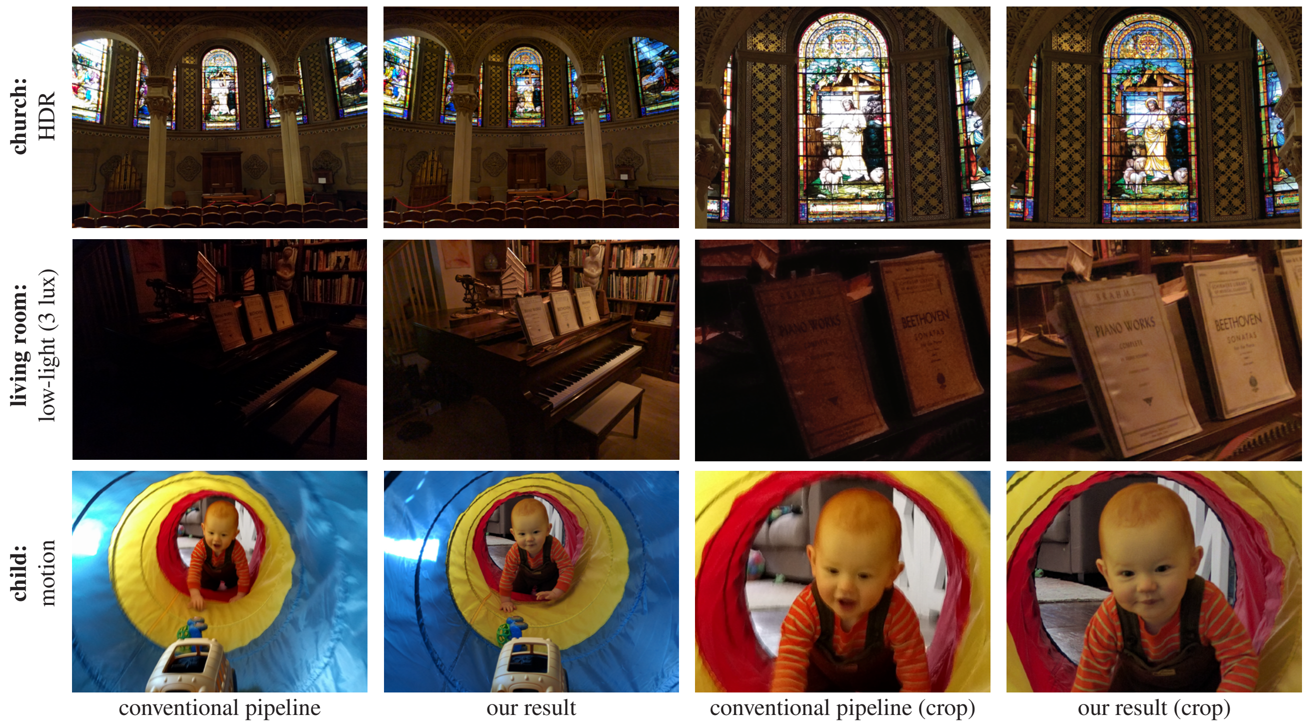


Figure 10: *A comparison of photos produced by our method with similar photos produced by a standard single-exposure processing pipeline on the same device. Readers are encouraged to zoom in to these figures. The top row shows a classic HDR scene: stained-glass windows in a church. In this example our method retains more detail in both the bright windows and the darker walls around them. The middle row shows a dark scene (3 lux is roughly equivalent to candlelight). Here our method produces a brighter and image than a standard pipeline. Also, by relying less on spatial denoising we are able to preserve low-contrast detail. The bottom row shows a fast-moving subject in moderately low light. In this situation we use shorter exposure times for each frame in our burst than the single exposure in a conventional pipeline, reducing motion blur. We also benefit in this scene from lucky imaging, which selects the sharpest frame it can find near the beginning of the burst.*

Failure cases Despite generally good image quality, our system does fail in extreme situations. We have designed it to degrade gracefully in these situations, but we wish it were even better. Some of these situations are shown in figure 11.

In addition, if a scene's dynamic range is so high that exposure fusion using two synthetic exposures would yield cartoony results, then we treat the scene as if its dynamic range was low and allow more pixels to blow out. Using three synthetic exposures might work better, but is expensive to compute and requires more subtle tuning of our autoexposure database. Also, if a scene contains such fast motions that features blur despite our short exposure time, then alignment might fail, leaving excessive noise in the output photograph.

Our most serious failure mode is that at very low light levels the ISP's autofocus and white balance estimation begin failing. Although merging and alignment may still work, the photograph might be out of focus or have a color cast. Slight casts are visible in figure 1.

Performance To make our pipelines fast enough to deploy on mobile devices, we have selected algorithms for their computational efficiency. This means avoiding non-local communication and data dependencies that preclude parallelization, consuming as little memory as possible, and employing fixed point arithmetic wherever possible. These same concerns preclude using algorithms with global iteration (e.g., FlexISP [Heide et al. 2014]), large or dynamic spatial support (e.g., BM3D [Dabov et al. 2007b]), or expensive tonemapping (e.g., local Laplacian filters [Aubry et al. 2014]).

Our system has shipped on devices having 12–13 Mpix sensors, on which we capture bursts of up to 8 frames. Thus, we may be required to store and process as much as 104 Mpix per output photograph. Although we have selected algorithms for efficiency, processing this much data still requires a highly-optimized implementation. Most of our code is written in Halide [Ragan-Kelley et al. 2012], which enables us to more easily fuse pipeline stages for locality and to make use of SIMD and thread parallelism. In addition, since we compute many small 2D real DFTs for align and merge, we have implemented our own FFT in Halide. For the small DFTs in our pipeline, this implementation is five times faster than FFTW [Frigo and Johnson 2005] on an ARM-based mobile phone.

Summarizing, on a Qualcomm Snapdragon 810 not subject to thermal throttling the time required to produce an output photograph ranges from 2.5 to 4 seconds, depending on the number of frames in the burst. For a low light shot taking 4 seconds, this breaks down as 1 second to capture the frames, 500 ms for alignment, 1200 ms for merging, and 1600 ms for finishing. For a daylight shot taking 2.5 seconds, we measure 100 ms for capture, 250 ms for alignment, 580 ms for merging, and 1600 ms for finishing.

8 Conclusions

In this paper we have described a system for capturing a burst of underexposed frames, aligning and merging these frames to produce a single intermediate image of high bit depth, and tone mapping this image to produce a high-resolution photograph. Our results have better image quality than single-exposure photos produced by a conventional imaging pipeline, especially in high dynamic

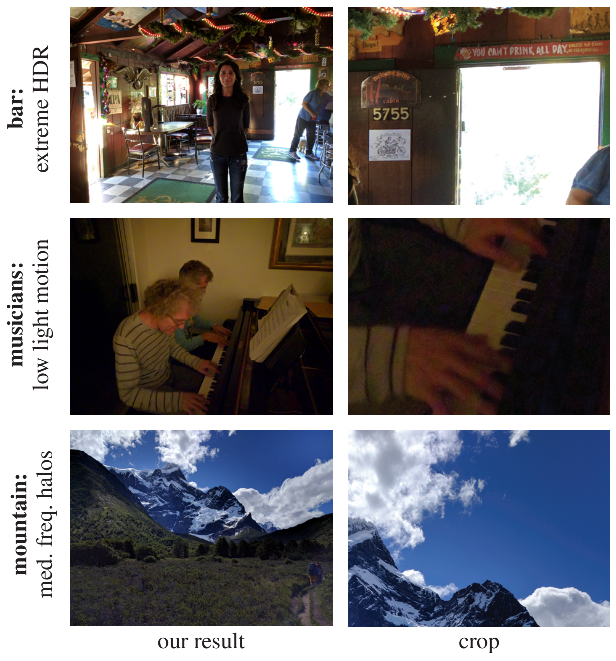


Figure 11: *Situations we do not handle well. Top: in this extremely high dynamic scene, we preferentially exposed for the face and building interior, thereby losing detail in the bright open doorways. Middle: In low light scenes with fast motions, we clamp to a short exposure time, producing excessive noise to avoid motion blur. Bottom: High contrast scenes can exhibit mild medium frequency halos (dark blue patches in sky) due to our use of exposure fusion.*

range or low-light scenes, and almost never exhibit objectionable artifacts. The system is deployed on several mass-produced cell phones, marketed as "HDR+" in the Nexus 6, 5X, and 6P. Consumers using our system are unaware that they are capturing bursts of frames with each shutter press, or that their final photograph is generated from multiple images using computational photography.

It is difficult in a technical paper to prove our general claim of superior image quality, or to cover the range of corner cases our system handles robustly. What we can say is that our system has received positive reviews in the press, has scored higher than most competing commercial systems in evaluations by independent agencies [DxO Inc. 2015], and that in millions of pictures captured by consumers each week, we have not seen disastrous results.

In order that others may judge our image quality and improve on our algorithms, we have created an archive of several thousand bursts of raw images in DNG format [Google Inc. 2016b]. For each burst we include our merged raw output and final JPEG output. EXIF tags and additional files describe our camera parameters, noise model, and other metadata used to generate our results.

Limitations and future work The most significant drawback of our system is that after the user presses the shutter there is a sensible lag before the burst begins and the reference frame is captured. Since this frame sets the composition for the photograph, it can be difficult to capture the right moment in an action scene. Some of this lag is due to our autoexposure algorithm, some to Camera2's software structure, and some to our use of lucky imaging, which adds a variable delay, depending on which frame was chosen as the reference.

To avoid shutter lag, many mobile phones employ zero shutter lag (ZSL), in which the camera continuously captures full-resolution YUV frames, stores them in a circular buffer, and responds to shutter press by selecting one image from this buffer to finish and store. Since focus, exposure, and white balance change continuously during aiming, it's not obvious how ZSL can be adapted to capture constant-exposure bursts. This is a topic for future work.

Another limitation of our system is that computing the output photograph takes several seconds and occupies a significant amount of memory until it finishes. If the user presses the shutter several times in rapid succession, we can easily run out of memory, causing the camera app to stall. Programmable hardware might solve this problem, but incorporating such hardware into mass-produced mobile devices is not easy.

Finally, the discrepancy between our ISP-generated viewfinder and software-generated photograph produces a non-ideal user experience. In extreme situations, a user might abandon photographing a scene because it looks poor in the viewfinder, when in fact our software would produce a usable photograph of that scene. Programmable hardware might solve this problem as well, and is a topic for future work.

References

ADAMS, A., TALVALA, E.-V., PARK, S. H., JACOBS, D. E., AJDIN, B., GELFAND, N., DOLSON, J., VAQUERO, D., BAEK, J., TICO, M., LENSCH, H. P. A., MATUSIK, W., PULLI, K., HOROWITZ, M., AND LEVOY, M. 2010. The Frankencamera: an experimental platform for computational photography. *SIGGRAPH*.

ADAMS, A. 1981. *The Print, The Ansel Adams Photography Series 3*. New York Graphic Society.

ADOBE INC., 2016. Photoshop CC 2015.1.2, http://www.adobe.com/creativecloud.html.

AUBRY, M., PARIS, S., HASINOFF, S. W., KAUTZ, J., AND DURAND, F. 2014. Fast local laplacian filters: Theory and applications. *TOG*.

BAKER, S., SCHARSTEIN, D., LEWIS, J. P., ROTH, S., BLACK, M. J., AND SZELISKI, R. 2011. A database and evaluation methodology for optical flow. *IJCV*.

BENNETT, E. P., AND MCMILLAN, L. 2005. Video enhancement using per-pixel virtual exposures. *SIGGRAPH*.

BROX, T., AND MALIK, J. 2011. Large displacement optical flow: Descriptor matching in variational motion estimation. *TPAMI*.

DABOV, K., FOI, A., AND EGIAZARIAN, K. 2007. Video denoising by sparse 3D transform-domain collaborative filtering. *EUSIPCO*.

DABOV, K., FOI, A., KATKOVNIK, V., AND EGIAZARIAN, K. 2007. Image denoising by sparse 3-D transform-domain collaborative filtering. *TIP*.

DEBEVEC, P. E., AND MALIK, J. 1997. Recovering high dynamic range radiance maps from photographs. *SIGGRAPH*.

DONOHO, D. L. 1995. De-noising by soft-thresholding. *Information Theory, IEEE Transactions on 41*, 3, 613–627.

DXO INC., 2015. Google Nexus 6P review, http://www.dxomark.com/Mobiles.

FARBMAN, Z., FATTAL, R., AND LISCHINSKI, D. 2011. Convolution pyramids. *SIGGRAPH*.

FARNEBÄCK, G. 2002. *Polynomial Expansion for Orientation and Motion Estimation*. PhD thesis, Linköping University, Sweden.

Farsiu, S., Elad, M., and Milanfar, P. 2006. Multi-frame demosaicing and super-resolution of color images. *TIP*.

Frigo, M., and Johnson, S. G. 2005. The design and implementation of FFTW3. *Proc. IEEE*.

Google Inc., 2016. Android Camera2 API, http://developer.android.com/reference/android/hardware/camera2/package-summary.html.

Google Inc., 2016. HDR+ burst photography dataset, http://www.hdrplusdata.org.

Gunturk, B., Glotzbach, J., Altunbasak, Y., Schafer, R., and Mersereau, R. 2005. Demosaicking: color filter array interpolation. *IEEE Signal Processing Magazine*.

Hasinoff, S. W., Durand, F., and Freeman, W. T. 2010. Noise-optimal capture for high dynamic range photography. *CVPR*.

Heide, F., Steinberger, M., Tsai, Y.-T., Rouf, M., Pajk, D., Reddy, D., Gallo, O., Liu, J., Heidrich, W., Egiazarian, K., Kautz, J., and Pulli, K. 2014. FlexISP: A flexible camera image processing framework. *SIGGRAPH Asia*.

Horn, B. K. P., and Schunk, B. G. 1981. Determining optical flow. *Artificial Intelligence*.

Hu, J., Gallo, O., Pulli, K., and Sun, X. 2013. HDR deghosting: How to deal with saturation? *CVPR*.

Joshi, N., and Cohen, M. F. 2010. Seeing Mt. Rainier: Lucky imaging for multi-image denoising, sharpening, and haze removal. *ICCP*.

Kim, S. J., Lin, H. T., Lu, Z., Süsstrunk, S., Lin, S., and Brown, M. S. 2012. A new in-camera imaging model for color computer vision and its application. *TPAMI*.

Kokaram, A. C. 1993. *Motion picture restoration*. PhD thesis, Churchill College, University of Cambridge. Section 8.1.

Levoy, M. 2010. Experimental platforms for computational photography. *IEEE CG&A 30*.

Lewis, J. 1995. Fast normalized cross-correlation. *Vision interface*.

Liu, C., Yuen, J., and Torralba, A. 2011. Sift flow: Dense correspondence across scenes and its applications. *TPAMI*.

Liu, Z., Yuan, L., Tang, X., Uyttendaele, M., and Sun, J. 2014. Fast burst images denoising. *SIGGRAPH Asia*.

Lucas, B. D., and Kanade, T. 1981. An iterative image registration technique with an application to stereo vision. *IJCAI*.

Mäkitalo, M., and Foi, A. 2013. Optimal inversion of the generalized anscombe transformation for poisson-gaussian noise. *TIP*.

Martinec, E., 2008. Noise, dynamic range and bit depth in digital SLRs, http://theory.uchicago.edu/œejm/pix/20d/tests/noise.

Menze, M., and Geiger, A. 2015. Object scene flow for autonomous vehicles. *CVPR*.

Mertens, T., Kautz, J., and Reeth, F. V. 2007. Exposure fusion. *Pacific Graphics*.

Petschnigg, G., Szeliski, R., Agrawala, M., Cohen, M., Hoppe, H., and Toyama, K. 2004. Digital photography with flash and no-flash image pairs. *SIGGRAPH*.

Ragan-Kelley, J., Adams, A., Paris, S., Levoy, M., Amarasinghe, S., and Durand, F. 2012. Decoupling algorithms from schedules for easy optimization of image processing pipelines. *SIGGRAPH*.

Reinhard, E., Ward, G., Pattanaik, S. N., Debevec, P. E., and Heidrich, W. 2010. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*, 2nd ed. Academic Press.

Stone, H. S., Orchard, M. T., Chang, E.-C., Martucci, S., et al. 2001. A fast direct Fourier-based algorithm for subpixel registration of images. *IEEE Transactions on Geoscience and Remote Sensing*.

Tao, M. W., Bai, J., Kohli, P., and Paris, S. 2012. Simpleflow: A non-iterative, sublinear optical flow algorithm. *Computer Graphics Forum (Eurographics 2012)*.

Telleen, J., Sullivan, A., Yee, J., Wang, O., Gunawardane, P., Collins, I., and Davis, J. 2007. Synthetic shutter speed imaging. *Computer Graphics Forum*.

Wiegand, T., Sullivan, G. J., Bjøntegaard, G., and Luthra, A. 2003. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*.

Wilburn, B., Joshi, N., Vaish, V., Talvala, E.-V., Antunez, E., Barth, A., Adams, A., Horowitz, M., and Levoy, M. 2005. High performance imaging using large camera arrays. *SIGGRAPH*.

Yamaguchi, K., McAllester, D., and Urtasun, R. 2014. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. *ECCV*.

Zhang, L., Deshpande, A., and Chen, X. 2010. Denoising vs. deblurring: HDR imaging techniques using moving cameras. *CVPR*.