Patch-Based High Dynamic Range Video

Nima Khademi Kalantari[1] Eli Shechtman[2] Connelly Barnes[2,3] Soheil Darabi[2] Dan B Goldman[2] Pradeep Sen[1]

[1]University of California, Santa Barbara [2]Adobe [3]University of Virginia

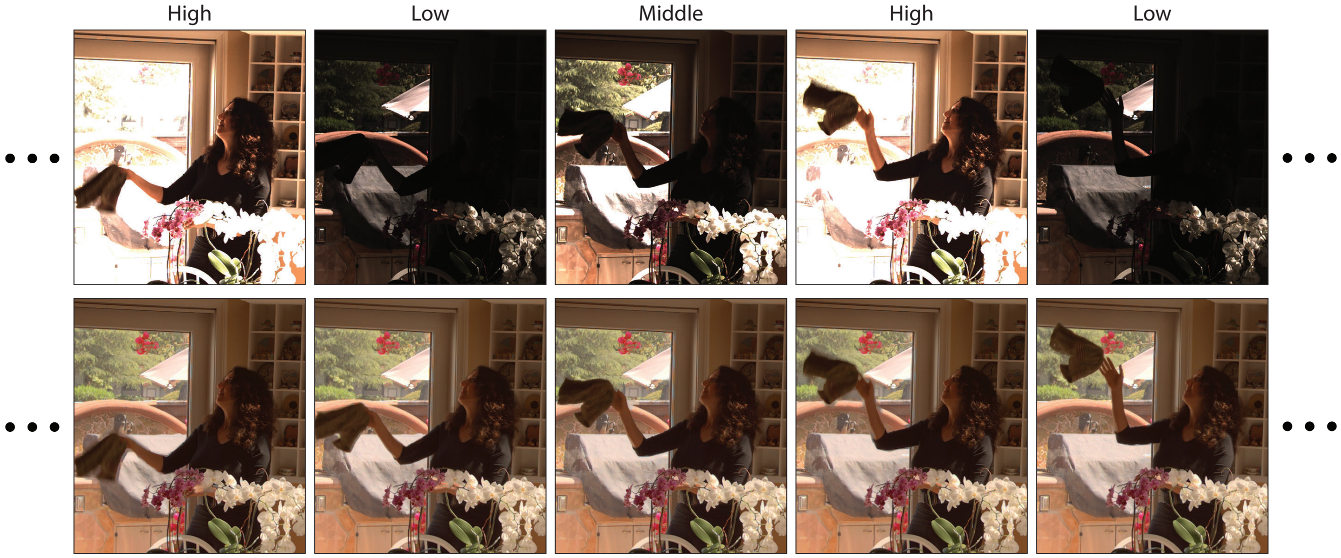

Figure 1: *(top row) Input video acquired using an off-the-shelf camera, which alternates between three exposures separated by two stops. (bottom row) Our algorithm reconstructs the missing LDR images and generates an HDR image at each frame. The HDR video result for this* `ThrowingTowel3Exp` *scene can be found in the supplementary materials. This layout is adapted from Kang et al. [2003].*

Abstract

Despite significant progress in high dynamic range (HDR) imaging over the years, it is still difficult to capture high-quality HDR video with a conventional, off-the-shelf camera. The most practical way to do this is to capture alternating exposures for every LDR frame and then use an alignment method based on optical flow to register the exposures together. However, this results in objectionable artifacts whenever there is complex motion and optical flow fails. To address this problem, we propose a new approach for HDR reconstruction from alternating exposure video sequences that combines the advantages of optical flow and recently introduced patch-based synthesis for HDR images. We use patch-based synthesis to enforce similarity between adjacent frames, increasing temporal continuity. To synthesize visually plausible solutions, we enforce constraints from motion estimation coupled with a search window map that guides the patch-based synthesis. This results in a novel reconstruction algorithm that can produce high-quality HDR videos with a standard camera. Furthermore, our method is able to synthesize plausible texture and motion in fast-moving regions, where either patch-based synthesis or optical flow alone would exhibit artifacts. We present results of our reconstructed HDR video sequences that are superior to those produced by current approaches.

CR Categories: I.4.1 [Computing Methodologies]: Image Processing and Computer Vision—Digitization and Image Capture

Keywords: High dynamic range video, patch-based synthesis

Links: DL PDF WEB

1 Introduction

High dynamic range (HDR) imaging is now popular and becoming more widespread. Most of the research to date, however, has focused on improving the capture of still HDR images, while HDR video capture has received considerably less attention. This is a serious deficit, since high-quality HDR video would significantly improve our ability to capture dynamic environments as our eyes perceive them. The reason for this lack of progress is that the bulk of HDR video research has focused on specialized HDR camera systems (e.g., [Nayar and Mitsunaga 2000; Unger and Gustavson 2007; Tocci et al. 2011; SpheronVR 2013; Kronander et al. 2013]). Unfortunately, the high cost and general unavailability of these cameras make them impractical for the average consumer.

On the other hand, still HDR photography has leveraged the fact that a typical consumer camera can acquire a set of low dynamic range (LDR) images at different exposures, which can then be merged into a single HDR image [Mann and Picard 1995; Debevec and Malik 1997]. However, most of the methods that address artifacts in dynamic scenes (e.g., [Zimmer et al. 2011; Sen et al. 2012]) only produce still images and cannot be used for HDR video.

The fundamental challenge is that producing high-quality HDR video from a set of alternating LDR exposures requires reconstructing well-aligned and temporally coherent LDR images. This needs to be done for each exposure in every frame so that the resulting HDR video is free of artifacts. Optical flow based solutions [Kang et al. 2003; Mangiat and Gibson 2010; Ginger HDR 2013] are suitable for scenes with small motion, but fail with complex motion. In these cases, they produce visible tearing and "ghosting" artifacts due to the failure of optical flow near motion boundaries.

Our method builds upon the recent work on HDR reconstruction for still images that poses the problem as a patch-based optimization [Sen et al. 2012]. Although this approach produces high-quality still HDR images, it is unsuitable for HDR video due to the lack of temporal coherency (see, e.g., `ThrowingTowel3Exp` in the supplementary materials[1]).

In this work we propose a new, temporally coherent patch-based optimization algorithm that can produce high-quality HDR video from an input sequence of alternating exposures captured with an off-the-shelf camera. We show how optical flow can be utilized in conjunction with a patch-based method to achieve motion smoothness, providing robustness to failures of optical flow in areas of fast motion and occlusions. Where the optical flow fails, the patch-based method synthesizes plausible textures and the artifacts are typically confined to very small regions close to motion boundaries. Masking effects in the human visual system make these artifacts very difficult to detect in moving video.

Our key contribution is to combine optical flow with a patch-based synthesis approach similar to Sen et al. [2012] to achieve temporal coherency. We show that a simple combination of the two components does not work well and propose a method to compute spatially-varying search windows for handling complex motions. A secondary contribution is jitter suppression for temporal coherency, using multiple motion models to regularize the patch-based alignment in under-constrained regions. As a result of these contributions, we are able to demonstrate high-quality HDR videos for scenes with large camera and non-rigid scene motion.

2 Related work

The problem of HDR imaging has been extensively studied in the past, although most of the previous work has focused on the reconstruction of still HDR images. For brevity, we shall only consider methods that have been specifically developed for – or shown to handle – HDR video, and refer readers interested in general HDR imaging to texts on the subject [Reinhard et al. 2010].

As mentioned earlier, the systems that have produced perhaps the most high-quality results to date have been specialized cameras that capture HDR videos directly. These include cameras with special sensors to measure a larger dynamic range [Brajovic and Kanade 1996; Seger et al. 1999; Nayar and Mitsunaga 2000; Nayar and Branzoi 2003; Unger and Gustavson 2007; Portz et al. 2013], or with beam-splitters that split the light to different sensors so that each measures a different portion of the radiance domain simultaneously [Tocci et al. 2011; Kronander et al. 2013]. However, these approaches are limited by the fact that they require specialized, custom hardware, which make them expensive and less widespread.

One possible way to capture HDR video with conventional cameras is to use external beam-splitters [McGuire et al. 2007; Cole and Safai 2013]. However, this additional hardware makes the system bulky and difficult to use. Moreover, even simple tasks like changing the focus or zooming become difficult because of the necessary camera synchronization. Therefore, the more practical way is to use a single camera that alternates exposures for each frame. Although not all video cameras can currently do this, there are efforts to increase the programmability of digital cameras (e.g., [Adams et al. 2010]). Furthermore, it is not difficult to find off-the-shelf cameras that can alternate exposures (e.g., the Basler acA2000-50gc camera used in this work). This approach has been explored in the past [Kang et al. 2003; Mangiat and Gibson 2010; Magic Lantern 2013], and we use it for our capture as well.

Kang et al. [2003] demonstrate the first practical method for generating HDR video using an off-the-shelf camera with a system that acquires sequences that alternate between short and long exposures. They first use optical flow to unidirectionally warp the previous/next frames to a given frame. They then merge them together in the regions where the current frame is well-exposed with a weighted blend to reject ghosting. For the over/under-exposed regions of the current frame, they bidirectionally interpolate the previous/next frames using optical flow followed by a hierarchical homography algorithm to help with the alignment process. Although Kang et al.'s method can increase the dynamic range of videos, their algorithm has visible artifacts when the input video contains non-rigid or fast motion as can be seen in Figs. 6 and 7. This problem is due to the fact that the algorithm relies heavily on existing motion estimation methods that are still prone to errors in these cases.

The recent work of Mangiat and Gibson [2010] is perhaps the state-of-the-art for producing HDR video using off-the-shelf cameras. To overcome the problems of gradient-based optical flow used in Kang et al., they propose a block-based motion estimation approach to approximate motion between adjacent frames. Moreover, they propose a motion refinement stage and a filtering stage that uses a cross-bilateral filter to remove the block boundary artifacts. In follow-up work, Mangiat and Gibson [2011] demonstrate improved results by filtering the regions with large motion to hide the artifacts of mis-registration. However, their results still suffer from blocking artifacts, as shown in Fig. 6. Moreover, their method is designed to handle sequences with only two exposures.

Finally, some publicly-available software has been developed to capture alternating exposures and produce HDR video. For example, the MagicLantern firmware available for certain Canon DSLR cameras [2013] has an HDR video mode that allows for capturing video with alternating ISOs. The resulting video can then be used with Ginger HDR [2013], which features a stand-alone "Merger" tool that utilizes optical flow to register frames and produce an HDR output. However, like the optical flow based method of Kang et al., it has many artifacts that are visible in scenes with large motion.

3 Proposed algorithm

In order to acquire an HDR video stream with a conventional video camera, we must first capture an input video that alternates between different exposures for each frame, as shown in Fig. 2. Formally, given a set of N LDR images taken by alternating between M different exposures ($L_{\text{ref},1}, L_{\text{ref},2}, \ldots, L_{\text{ref},N}$), our goal is to reconstruct the N HDR frames ($H_n,\ n \in \{1, \ldots N\}$) for the entire video sequence[2]. To do this, our algorithm must reconstruct the missing LDR images at each frame ($L_{m,n},\ m \in \{1, \ldots, M\}, m \neq \text{ref}$), shown with dashed red squares in Fig. 2. Note we use the term "reference images" to refer to the LDR images captured by the camera.

[1] Some artifacts are difficult to observe in still images, and so in the paper we refer the reader to our supplementary video materials by scene name.

[2] Note that the exposure of the reference image is not fixed and depends on the frame number. Therefore, the correct notation would be $\text{ref}(n)$, but for the ease of notation we skip this formality.

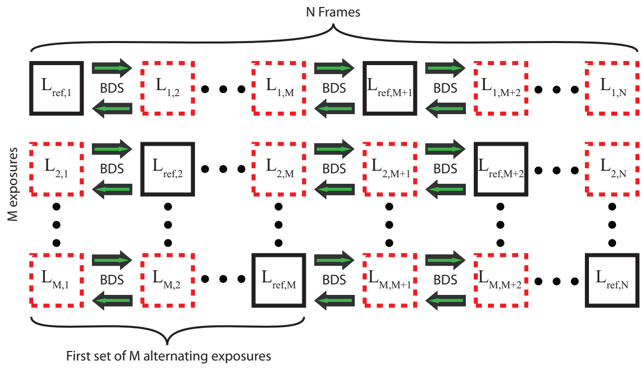

Figure 2: *An example video sequence with N frames. To capture HDR video, our off-the-shelf camera alternates between M different exposures, capturing only one specific exposure at each frame (shown with solid black squares). Our algorithm reconstructs the missing exposures at each frame (dashed red squares) by doing a patch search/vote on the two neighboring frames. To maximize the temporal coherency, the patch searches are performed around an estimated motion flow (given by the green arrows). Once these missing LDR frames have been reconstructed, the different exposures can be merged together for every frame to produce the final sequence of HDR images.*

To reconstruct the HDR images from the LDR inputs, Sen et al. [2012] had proposed a patch-based optimization system for still HDR photography that satisfied two properties: 1) the final HDR image H_n should be very close to the reference image n after mapping it to the radiance domain $h(L_{\text{ref},n})$ wherever $L_{\text{ref},n}$ is well-exposed, and 2) H_n should include information from the captured images at the M different exposures neighboring frame n. Although this often works well for still images, their method is unsuitable for our application since it lacks temporal coherency (see `ThrowingTowel3Exp` in the supplementary materials), a necessity for high-quality HDR video. Furthermore, their method can also generate unsatisfactory results when a large region of the reference image is under- or over-exposed. This is particularly relevant for our video application since the reference frame must vary in exposure for each time instant, resulting in large missing regions in many reference frames. Therefore, a direct application of the Sen et al. method to video yields unacceptable results, as shown in Fig. 3.

To address the problem of temporal coherence, we first observe that despite the motion from frame to frame in a video, the content of consecutive frames is very similar. For example, the LDR images of consecutive frames that have the same exposure (each of the rows in Fig. 2) will be very similar. The second observation is that many dynamic scenes can be approximated using multiple large regions that move coherently across consecutive frames. Guided by these observations and drawing some of the elements from the patch-based optimization framework of Sen et al. [2012], we propose the following energy function for HDR video reconstruction:

$$\begin{aligned} E(\text{all } L_{m,n}\text{'s}) = \sum_{n=1}^{N} \sum_{p \in \text{pixels}} \Big[& \alpha_{\text{ref},n_{(p)}} \cdot \big(h(L_{\text{ref},n})_{(p)} - H_{n(p)}\big)^2 \\ & + (1-\alpha_{\text{ref},n_{(p)}}) \cdot \sum_{m=1, m\neq \text{ref}}^{M} \Lambda(L_{m,n})\big(h(L_{m,n})_{(p)} - H_{n(p)}\big)^2 \\ & + (1-\alpha_{\text{ref},n_{(p)}}) \cdot \sum_{m=1}^{M} \text{TBDS}(L_{m,n}\ ,\ L_{m,n-1},\ L_{m,n+1}) \Big]. \end{aligned} \tag{1}$$

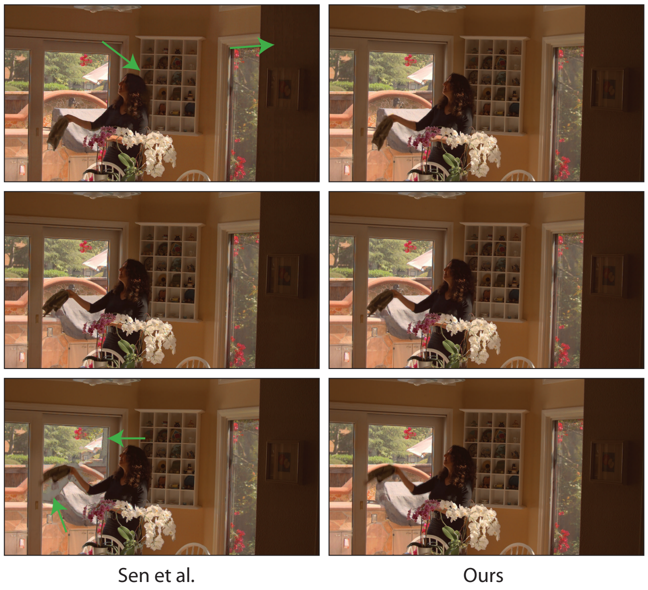
Sen et al.　　Ours

Figure 3: *Three HDR frames of the `ThrowingTowel3Exp` scene generated by both the method of Sen et al. [2012] and our method. The method of Sen et al. works best when the reference image is the middle exposure (middle). In the frames where the low or high exposed images are the reference (top and bottom, respectively), their method has artifacts, as indicated by the green arrows. Our method generates plausible results in all cases.*

In the first term, $h(L_{\text{ref},n})$ is a function that maps the LDR image $L_{\text{ref},n}$ to the linear radiance domain, and $\alpha_{\text{ref},n}$ is a function (Fig. 5) that approximates how well each pixel in $L_{\text{ref},n}$ is exposed. This term ensures that the HDR reconstruction H_n is similar to $h(L_{\text{ref},n})$ in an $\mathcal{L}_2$ sense in the well-exposed regions. The second term ensures that all the LDR images in one frame are similar to the HDR image in that frame in an $\mathcal{L}_2$ sense for the regions that are not well-exposed in the reference image. This term maintains the relationship between the HDR image and the LDR's that compose it, so it is weighted by the triangle function $\Lambda()$ used for merging [Debevec and Malik 1997]. Finally, the third term helps enforce temporal coherence by leveraging ideas from Regenerative Morphing [Shechtman et al. 2010]. In this case, we propose to use temporal bidirectional similarity (TBDS) to measure the bidirectional similarity of the LDR image $L_{m,n}$ to its counterparts in the previous ($L_{m,n-1}$) and next ($L_{m,n+1}$) frames:

$$\begin{aligned} \text{TBDS}(L_{m,n}\ ,\ L_{m,n-1}\ ,\ L_{m,n+1}) = {} & \text{BDS}(L_{m,n}\ ,\ L_{m,n-1}) \\ & + \text{BDS}(L_{m,n}\ ,\ L_{m,n+1}). \end{aligned} \tag{2}$$

Here we use the patch-based bidirectional similarity (BDS) metric proposed by Simakov et al. [2008], except that we constrain the search based on the estimated local motion to further improve temporal coherence:

$$\begin{aligned} \text{BDS}(T,\ S) = {} & \frac{1}{|S|} \sum_{p \in \text{pixels}} \min_{i \subset f_S^T(p) \pm w_S^T(p)} \text{D}(s(p), t(i)) \\ & + \frac{1}{|T|} \sum_{p \in \text{pixels}} \min_{i \subset f_T^S(p) \pm w_T^S(p)} \text{D}(t(p), s(i)), \end{aligned} \tag{3}$$

where $s(p)$ and $t(p)$ denote the patches centered at pixel p in the source and the target images, and $\text{D}()$ refers to the sum of the

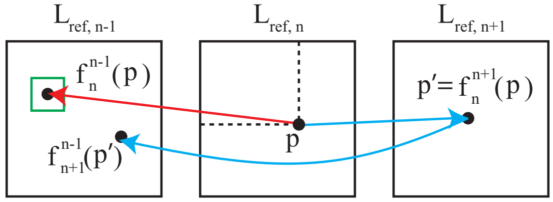

Figure 4: *To validate $f_n^{n-1}(p)$, the flow from L_n to L_{n-1} shown with red arrow, we first compute $f_n^{n+1}(p)$ and f_{n+1}^{n-1} shown with blue arrows. We then concatenate these two flows to get $f_{n+1}^{n-1}(p\prime)$ where $p\prime = f_n^{n+1}(p)$. If this flow is inside a small window (shown in green) around $f_n^{n-1}(p)$, we keep it, otherwise we discard it. In this case, the flow shown in red will be discarded since it does not pass the consistency check.*

squared differences (SSD) between two patches. We have modified the standard BDS equation by adding the $f_S^T(p)$ and $w_S^T(p)$ to constrain our search: $f_S^T(p)$ is the approximate motion flow at pixel p from the S to T and $w_S^T(p)$ scales the search window around it.

Intuitively, the first term (completeness) ensures that for every patch $s(p)$ in the source, there is a similar patch in the region defined by $f_S^T(p) \pm w_S^T(p)$ in the target image and vice versa for the second term (coherence). As shown by Simakov et al. [2008], minimizing this metric implies that the target image contains most of the content from the source image in a visually coherent way. As a result, minimizing the third term in Eq. 1 ensures that each LDR image $L_{m,n}$ contains similar content to its temporal neighbors. Moreover, constraining the patch searches around an initial motion estimation results in temporal coherency in the output video.

In our algorithm, we first estimate a rough initial motion, then use it to calculate a local search window size. We then minimize Eq. 1 using a two-stage iterative algorithm that iterates between the two stages until convergence. This method reconstructs the missing LDR images, which are finally combined to form the final HDR results. Therefore, our method consists of three main steps:

1. **Initial motion estimation (Sec. 3.1):** A rough motion is estimated in the two directions between consecutive frames ($f_S^T(p)$ and $f_T^S(p)$ in Eq. 3). We use a planar model (similarity transform) for the global motion and optical flow for the local motion estimation.

2. **Search window map computation (Sec. 3.2):** A window size is computed for every flow vector ($w_S^T(p)$ and $w_T^S(p)$ in Eq. 3). This search window map is used as the search window size around each initial estimate of the motion.

3. **HDR video reconstruction (Sec. 3.3):** A two-stage iterative method is used to minimize Eq. 1. In the first stage, a multiscale constrained patch search-and-vote is performed to minimize the last term of Eq. 1, and, in the second stage, an HDR merge step with reference injection [Sen et al. 2012] is used to minimize the first two terms. The algorithm iterates between these two stages until convergence. This reconstructs the missing LDR images and produces the final HDR frames.

We now discuss each of these steps in turn in the following sections.

3.1 Initial motion estimation

Computing the BDS between a pair of images requires performing a search in two directions, each requiring a motion flow estimation as per Eq. 3. Therefore, the two BDS terms in Eq. 2 involve the estimation of four motion flows at every frame n: $f_n^{n-1}(p)$, $f_{n-1}^{n}(p)$, $f_n^{n+1}(p)$, and $f_{n+1}^{n}(p)$, . Our motion estimation algorithm combines a similarity transform (rotation, translation, isometric scale) for the global motion followed by an optical flow computation. The camera motion can be approximately removed by a similarity transform since there is little camera movement between adjacent frames, while local scene motion is estimated by optical flow.

The first step is to find a similarity transform between the next and previous frames ($L_{\text{ref},n+1}$ and $L_{\text{ref},n-1}$) to the current frame $L_{\text{ref},n}$. This requires raising the exposure of the image with the lower exposure time to that of the other image to compensate for the exposure differences. To do this, we first apply the inverse camera response function to take the image with the lower exposure into the linear radiance domain. We then multiply it by the exposure ratio of the two images, and, finally apply the camera response function to map the radiance values into the LDR domain. After performing the exposure adjustment, we use RANSAC to find a dominant similarity model from the correspondences between the two images. Next, we warp the two neighboring images using the calculated similarity transforms to remove the global motion and facilitate the local motion estimation using optical flow. The rest of the process is performed on the warped images.

For simplicity, we only explain the process for estimating motion from frame n to $n-1$ (denoted by $f_n^{n-1}(p)$), but the other flows are calculated in a similar manner. Since most optical flow algorithms rely on the brightness constancy assumption, we first adjust the exposure of all three images $(n-1, n, n+1)$ to match the one with the highest exposure. This is necessary because our flow validation process, which will be explained later, works on all the three images under the assumption that they were captured under the same conditions. After adjusting the exposures, we use the optical flow method of Liu [2009] to compute $f_n^{n-1}(p)$.

As is well known, this flow might be inaccurate because of noise, saturated pixels, or complex motions. One common way for estimating erroneous flow is to compare $f_n^{n-1}(p)$ with $f_{n-1}^{n}(p)$ and keep the flows only if they are close to each other [Brox and Malik 2011]. However, we found this approach was not robust enough, often validating incorrect flow since errors are often symmetric. Therefore, we use a more robust flow consistency test based on *triplets* of frames, as shown in Fig. 4. To do this, we calculate the flows f_n^{n-1}, $f_n^{n+1}(p)$ and $f_{n+1}^{n-1}(p)$ and check if the concatenation $f_{n+1}^{n-1}(f_n^{n+1}(p))$ is inside a small window around $f_n^{n-1}(p)$. We keep the flow vectors where the concatenation is within a very small window $b_{\min}$, and otherwise we discard it as invalid. In addition, we discard the flows in the regions where $L_{\text{ref},n}$ is highly saturated (all three channels greater than δ_s) due to the lack of meaningful content. The final flow is obtained by concatenating this optical flow result with the similarity transform. In our implementation, we set $b_{\min}$ to 0.002 times the image size and δ_s to 0.99.

The estimated flow is used as a guide during the patch synthesis process to constrain the search to a small, local window around the flow vector. The size of the local window depends on the accuracy estimation of the optical flow, which is described next.

3.2 Search window map computation

The search window map defines the size of the search window around each flow obtained in the previous step. This search window should be large enough so that the correct patch can be found during the patch search process, but not so large that it causes temporal jittering in the final result. The ideal size would be equal to the distance of the correct motion to the estimated flow, but, since we do not know the correct motion *a priori*, we need a method to estimate a window size where a good match can be found. Note that traditional optical flow confidence measures (e.g., [Jahne et al. 1999]) are not suitable for our purpose as they usually give a score map reflecting the *probability* to estimate correct motion.

We propose to use a patch search process to determine the size of the search window around each flow vector. We start with a small search window around the flow and perform a patch search to find a similar patch. If a good match is not found within a given threshold, the process is continued for several iterations, increasing the search window each time. Once a good patch is found, we use that search window size as the value in the search window map.

More explicitly, in order to find a search window $w_n^{n-1}(p)$ around a flow vector $f_n^{n-1}(p)$ from $L_{\text{ref},n}$ to $L_{\text{ref},n-1}$, we first match the exposure of the two images by raising the exposure of the lower one to match the higher one. For simplicity in this explanation, we simply use $L_{\text{ref},n}$ and $L_{\text{ref},n-1}$ to refer to the exposure adjusted versions of these images. Next, for a patch in $L_{\text{ref},n}$ centered on p, we look for the closest patch in an $\mathcal{L}_2$ sense in a very small window $b_{\min}$ around $f_n^{n-1}(p)$. If the distance in color space between these two patches is less than a threshold δ_n (0.04 in our implementation), we assign $w_n^{n-1}(p) = b_{\min}$.

In order to penalize patches that diverge greatly in one color channel, we compute the patch SSD for each color channel separately and take the maximum distance as the final value. If the distance is above the threshold, we exponentially increase the window size by a factor of two and continue the patch search and distance comparison. If a proper window size has not been found after four iterations, we assign a large window size to this flow $b_{\max}$, which we set equal to 0.4 times the image size.

The regions where $L_{\text{ref},n}$ is highly saturated (all three channels greater than δ_s) do not have enough content, so we use a different strategy to define the window search size. We first warp $L_{\text{ref},n-1}$ using $f_n^{n-1}(p)$. If the pixel value of the warped image in these highly saturated regions is smaller than δ_s, we assign a large search window $b_{\max}$, otherwise we assign a very small window $b_{\min}$. Since we use a patch-based method to compute the search window map, patches on the boundary between an accurate and inaccurate flow region will cover both regions. Therefore, the patch distances for these regions might be inaccurate, which makes the computed search window unreliable. To alleviate this problem and give more freedom to the patches in these regions, we dilate the search map by twice the patch width (7 in our implementation) to compute the final search map. This whole process is done for all other flow vectors that are used in our TBDS calculation.

3.3 HDR video reconstruction

Once we have computed the initial motion and the search window map, we minimize the energy in Eq. 1 using a two-stage algorithm. In the first stage, a constrained patch search-and-vote process is performed for each BDS term in Eq. 2, resulting in two voted images for each LDR image, shown with dashed red squares in Fig. 2. We then replace the LDR image with the average of these two voted images. We continue this search-and-vote process several times to minimize the third term in Eq. 1 [Shechtman et al. 2010]. The second stage, similar to Sen et al. [2012], consists of merging all the voted images and the reference image into an HDR image at each frame. This process simultaneously minimizes the second term of Eq. 1 and ensures that the first term is satisfied by injecting the well-exposed pixels of the reference image into the HDR frame. The algorithm iterates between these two stages until it converges.

Our algorithm begins by initializing all of the LDR images to the exposure-adjusted version of the reference image from the same frame. Then, for each LDR image $L_{m,n}$, we perform two bidirectional constrained patch searches against $L_{m,n+1}$ and $L_{m,n-1}$. These constrained searches are performed in a window (Sec. 3.2) around the initial motion flow estimate (Sec. 3.1). Next, in the voting process, the searched patches for completeness and coherence (the first and second terms in Eq. 3, respectively) are weighted averaged to generate a voted image for each BDS term in Eq. 2. The LDR image $L_{m,n}$ is then replaced with the average of these two voted images. We continue this search-and-vote process several times until convergence.

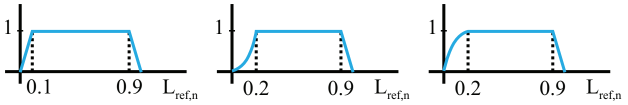

Figure 5: *The $\alpha_{ref,n}$ curves. (left) Sen et al. [2012], (middle) for search windows smaller than $b_{\max}$, (right) for search windows of size b_{max}. Note the curves only differ in the under-exposed regions and they are the same as Sen et al. in the over-exposed regions.*

In the next step, the averaged images from all M LDR sources in each frame ($L_{m,n}, m \in \{1, \ldots, M\}$) are combined using the HDR merge process, as proposed by Sen et al. [2012], to form an intermediate HDR frame H_n. The HDR merge process injects the well-exposed pixels of the reference image $L_{\text{ref},n}$ into the HDR frame. For the over/under exposed regions, we blend the reference image with the other LDR images in that frame using $\alpha_{\text{ref},n}$ (shown in Fig. 5 (middle)). Finally, we replace each missing LDR image $L_{m,n}$ with $l^m(H_n)$ which maps the radiance values of H_n to the exposure range of m. This process continues iteratively and in a multiscale fashion to minimize Eq. 1. Note that in coarse scales we reduce the size of the window according to the resolution of the image at that scale. In the coarsest scale, our images have 150 pixels in the smaller dimension and we have a total of 6 scales with a ratio of $\sqrt[5]{x/150}$, where x is the minimum dimension of input frames. We use 20 iterations at the coarsest scale and linearly decrease it to 5 at the finest scale. Because we constrain the search to a small window around the initial flow, our optimization converges faster and with fewer iterations and scales relative to Sen et al.

Under-exposed regions must be treated carefully when estimating the HDR image to avoid artifacts from the alternating exposures. The parameter $\alpha_{\text{ref},n}$ in Eq. 1 determines what is over/under exposed and, therefore, controls the contribution of the reference image $L_{\text{ref},n}$ in the HDR image. Sen et al. used a fixed trapezoid function shown in Fig. 5 (left) as $\alpha_{\text{ref},n}$ (see Eq. 1) with a valid range of 0.1 to 0.9. This means that their method heavily relies on the reference image in the dark regions, which can be problematic when the reference image has low exposure. As can be seen in Fig. 3 (top) this washes out the details in the dark regions. Instead, to suppress the noise in the final HDR result, we set the minimum value of the valid range to 0.2 and use $(L_{\text{ref},n_{(p)}}/0.2)^2$ as $\alpha_{\text{ref},n}$ in the under-exposed regions ($L_{\text{ref},n_{(p)}} < 0.2$) as shown in Fig. 5 (middle).

Moreover, in the places that the search map has a large window $b_{\max}$, we use the $\alpha_{\text{ref},n}$ curve shown in Fig. 5 (right), which uses $(L_{\text{ref},n_{(p)}}/0.2)^{0.5}$ in the under-exposed regions. The reason is that the areas with large search windows are often occluded or undergoing very complex motion, so the reference needs to be injected more to avoid deviating from the reference. Since the motion is usually fast in these regions, artifacts are difficult to perceive.

Although we constrain the patch search to a small window around the rough initial motion flow, the HDR results might still exhibit a small amount of jittering. This jittering occurs in the under- and over-exposed regions of the reference image, where the valid infor-

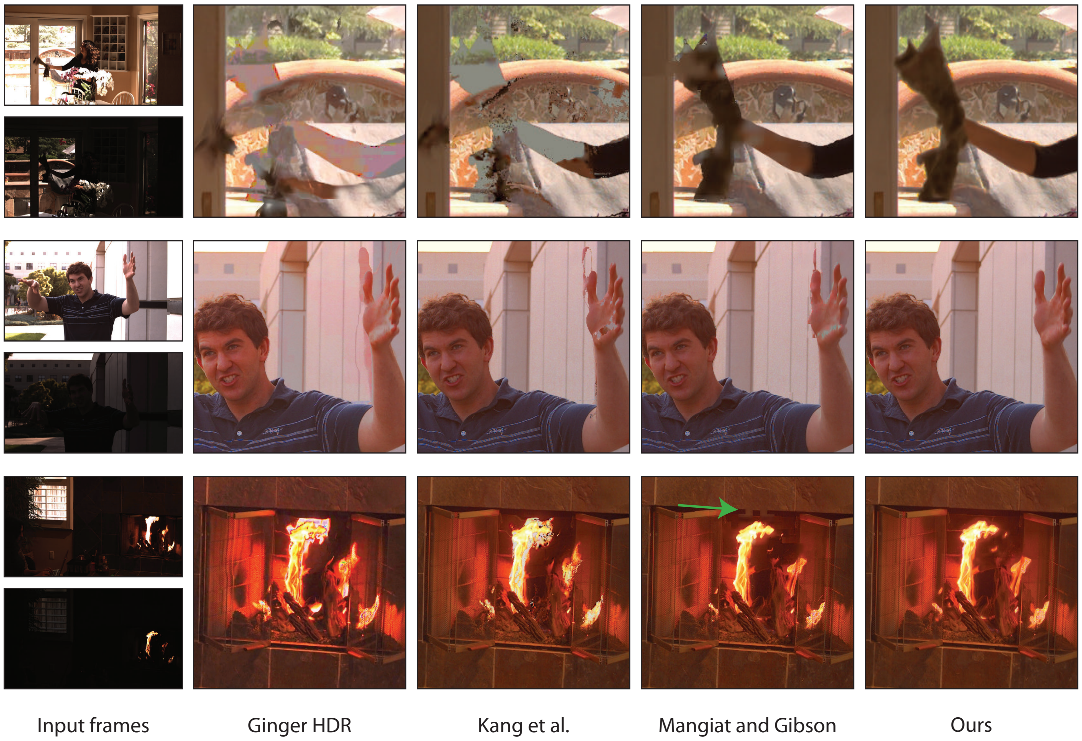

Figure 6: *A comparison of our algorithm and several other methods for a two-exposure input. From top to bottom,* `ThrowingTowel2Exp`*,* `WavingHands`*, and* `Fire`*.*

mation needs to be propagated from other exposures. To alleviate this problem, after performing the patch search, we find a few dominant similarity models in the nearest neighbor field (NNF) in the under/over exposed regions using RANSAC. We then overwrite the NNF values of the inliers using their corresponding model. Note that this process only removes jittering from regions where motion can be modeled with similarity transforms. Since no similarity model fits the regions with non-rigid motion, they will be detected as outliers and their NNF values will not be changed.

3.4 Acceleration and other details

To accelerate the search-and-vote process, we use the PatchMatch implementation of Barnes et al. [2009]. We found that, in most cases, one iteration of the search-and-vote process in the first stage of HDR video reconstruction algorithm results in convergence.

To compute the similarity transforms (Sec. 3.1), we observed that PatchMatch can find better correspondences in the smooth regions than the more commonly used SIFT features [Lowe 2004], providing a better similarity transform estimation. However, the PatchMatch correspondences are very dense, which slows down the similarity model estimation. Since we perform the similarity transform estimation twice, once in the motion estimation stage (Sec. 3.1) and then again in the HDR video reconstruction stage to remove the small jittering (Sec. 3.3), it is crucial to accelerate this process.

To do this, we use only a subset of the correspondences to estimate the model. In the HDR video reconstruction stage where we need to correct all the inliers, we first find a model using a uniform subset of the samples and then find all of the inliers using this model and correct them. For speed-up, we only perform this process at the finest scale. Empirically, we found one model is enough for correcting the inliers and fixing the problem of small jitter.

4 Results

All of the results shown in the paper were captured at 30 frames per second and with a resolution of 1280 × 720 using an off-the-shelf Basler acA2000-50gc camera. We captured input sequences with both two and three alternating exposures. We implemented our algorithm in MATLAB and compared against the method of Kang et al. [2003], Mangiat and Gibson [2010], and Ginger HDR [2013], a commercial software application that uses optical flow to register frames and merges them into HDR. We used our implementation of the method of Kang et al., but for Mangiat and Gibson's approach we asked the authors to run their algorithm on our datasets. Their core algorithm is proposed in [Mangiat and Gibson 2010] and includes improvements from [Mangiat 2012] that utilize hierarchical motion estimation. Tonemapping was done using the method of Reinhard et al. [2002] modified for temporal coherency, as proposed by Kang et al. Results of each method are in the appropriate folder in the supplementary material organized by scene name.

We begin by demonstrating the results of the naïve combination of optical flow with the method of Sen et al. [2012]. For this, we constrained all the patch search processes in their method to a fixed window size around the optical flow. We experimented with small and large window sizes. The results of this experiment for the `ThrowingTowel3Exp` scene can be found in the `NaivePatchHDR` folder of the supplementary materials. As can be seen, the results are not temporally coherent.

Next, we demonstrate the importance of each stage of our algorithm. For this, we ran four different experiments on the `ThrowingTowel3Exp` scene. These results can be found in the `AlgorithmBreakdown` folder of the supplementary materials, where they are appropriately named by their corresponding experiment. In the first experiment, initial motion flows (Sec. 3.1) are used to simply warp the next and previous frames to the current frame and generate HDR video. This results in large artifacts in regions of motion, and jittering problems arise due to the inaccuracy of the motion estimation. Second, we applied a search window of a small fixed size around all of the flows and generated results with our HDR video reconstruction system (Sec. 3.3). Due to this limited window search, our HDR reconstruction system cannot correct the inaccuracies of initial motion flow and produces visible ghosting artifacts around moving objects. Third, we repeated this experiment with a large search window instead. The accompanying video shows that even our model fitting (Sec. 3.3) fails to correct the jittering caused by the broader search window. Finally, our full method, excluding model fitting, results in small jittering in the output. For comparison, our full algorithm exhibits the final result with minimal artifacts.

Since the method of Mangiat and Gibson and Ginger HDR can only handle two-exposures, we first compare the results of our algorithm with all the other methods on sequences with two alternating exposures separated by three stops. Fig. 6 shows the result of this comparison on the `ThrowingTowel2Exp`, `WavingHands`, and `Fire` scenes (from top to bottom). Ginger HDR and the method of Kang et al. have similar artifacts around moving objects due to failure of optical flow. Specifically, the method of Kang et al. relies on the interpolated frames in the under-constrained regions, so it sometimes cannot reconstruct fast-moving objects. Moreover, the method of Mangiat and Gibson shows visible blocking artifacts around the moving objects. On the other hand, our method can plausibly reconstruct the areas containing fast-moving objects.

Next, we show our results on videos with three alternating exposures separated by two stops, which has not been demonstrated before. Among the three previous methods, only the method of Kang et al. can be extended to work with three exposures. We note that Kang et al. was only previously demonstrated for two exposure inputs and, thus, a three exposure input may not be ideal for their system. Fig. 7 shows the results of our comparison with Kang et al. on the `Dog`, `CheckingEmail`, and `Skateboarder` scenes (from top to bottom). As can be seen, the method of Kang et al. has visible artifacts around the moving objects, while ours reconstructs visually-pleasing HDR video.

As for timing, our implementation takes roughly three and a half minutes per frame for a two exposure sequence, in most cases. This timing consists of the following: initial motion estimation (Sec 3.1): 40 secs, search window map computation (Sec 3.2): 30 secs, search/vote (first stage in Sec 3.3): 125 secs, and HDR merge (second stage in Sec 3.3): 25 secs. We note that these timings are obtained by decreasing from 12 iterations at the coarsest scale to 3 iterations at the finest scale during the HDR video reconstruction stage (Sec. 3.3). In practice, we found that these iteration counts generate high-quality results for most cases. However, a few scenes (e.g. `WavingHands` in Fig. 6) required additional iterations (20 decreased to 5, as explained in Sec 3.3) to generate satisfactory results, increasing timings by 70%.

5 Limitations and future work

Our algorithm relies on motion estimation, so this can occasionally result in problems for the output video. For example, the `Skateboarder` scene, shown at the bottom of Fig. 7, exhibits some frames where limbs blur or partially disappear due to misestimated motion. However, these artifacts are difficult to perceive, since they are small, infrequent, and occur around motion boundaries. Thus, because our algorithm does not rely too strongly on optical flow and can also synthesize plausible texture in these regions, it avoids generating noticeable artifacts.

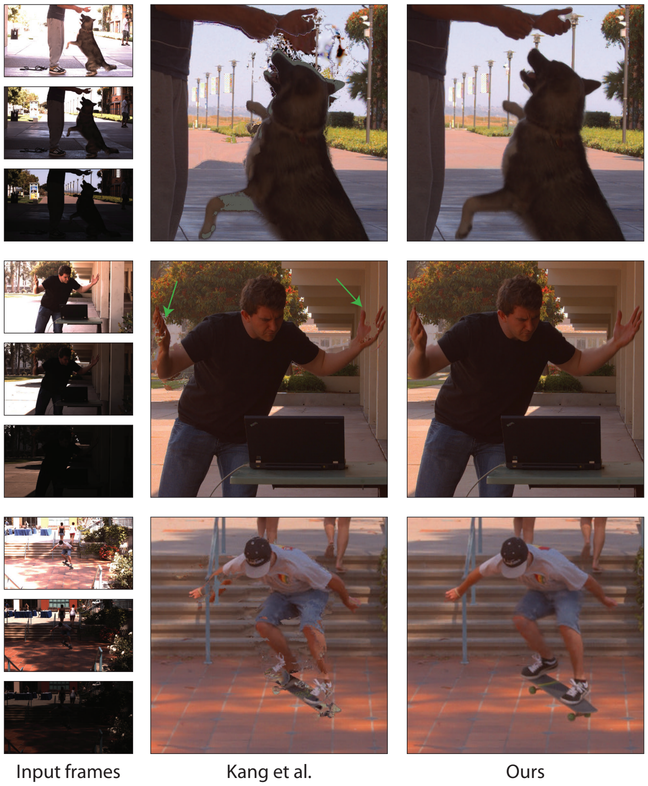

Figure 7: *A comparison of our method and Kang et al. for a three-exposure input. From top to bottom,* `Dog`*,* `CheckingEmail`*, and* `Skateboarder`*.*

Furthermore, our search window map can sometimes be inaccurate due to our reliance on similarity transform in the saturated regions. In these cases, the patch search will be limited to small regions around an inaccurate flow and, therefore, our method is unable to place the patches in the correct position, resulting in artifacts. An example is the `Skateboarder` scene, where the shoulder of the skateboarder exhibits slight jittering. Although our artifacts are still more plausible than those of Kang et al., a better way of handling the saturated regions can be investigated in the future.

In terms of speed, our algorithm's runtime can be significantly improved with a more optimized implementation. We observed that, in practice, most of the regions have a very small search window. In these areas, our optimization system is more constrained and converges faster, enabling us to decrease the number of iterations and improve runtime. We leave the acceleration of our algorithm for future work.

6 Conclusion

In conclusion, we have demonstrated a new method for producing HDR video with an off-the-shelf camera, which combines the advantages of patch-based synthesis and optical flow. We observed that patch-based synthesis lacks temporal coherency and that optical flow can fail in the presence of complex motion. To solve this issue, we combine the two methods through spatially varying search maps. Our HDR reconstruction is solved as a simultaneous optimization of a single energy over all known and unknown LDR images. We demonstrated that our method can generate visually pleasing results with good temporal coherency that are superior to the existing approaches.

Acknowledgments

The authors would like to gratefully thank Steve Bako for his help in acquiring the datasets and generating the results. Juli Shulem, Carlos Torres, and Keenan Dalton also helped with the datasets. Stephen Mangiat ran his algorithm on our datasets to produce comparisons for Fig. 6. We acknowledge support from the UCSB Center for Scientific Computing at the CNSI and MRL: an NSF MRSEC (DMR-1121053) and NSF CNS-0960316. Finally, we thank the anonymous reviewers for their helpful comments, which greatly improved the final version of the paper. This work was partially funded by National Science Foundation CAREER grants IIS-1342931 and IIS-1321168, and by Adobe.

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. *ACM Trans. Graph. 29*, 4 (July), 29:1–29:12.

Barnes, C., Shechtman, E., Finkelstein, A., and Goldman, D. B. 2009. Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. Graph. 28* (July), 24:1–24:11.

Brajovic, V., and Kanade, T. 1996. A sorting image sensor: an example of massively parallel intensity-to-time processing for low-latency computational sensors. In *Proceedings of ICRA, 1996*, vol. 2, 1638–1643.

Brox, T., and Malik, J. 2011. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Trans. Pattern Anal. Mach. Intell. 33*, 3 (Mar.), 500–513.

Cole, A., and Safai, M., 2013. Soviet Montage Productions. `http://www.sovietmontage.com/`.

Debevec, P. E., and Malik, J. 1997. Recovering high dynamic range radiance maps from photographs. In *Proceedings of ACM SIGGRAPH 1997*, 369–378.

Ginger HDR, 2013. A commercial HDR merging application. http://www.19lights.com/.

Jahne, B., Geissler, P., and Haussecker, H., Eds. 1999. *Handbook of Computer Vision and Applications with Cdrom*, 1st ed., vol. 2. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Kang, S. B., Uyttendaele, M., Winder, S., and Szeliski, R. 2003. High dynamic range video. *ACM Trans. Graph. 22*, 3 (July), 319–325.

Kronander, J., Gustavson, S., Bonnet, G., and Unger, J. 2013. Unified HDR reconstruction from raw CFA data. *IEEE International Conference on Computational Photography (ICCP)*.

Liu, C. 2009. *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. Doctoral thesis, Massachusetts Institute of Technology.

Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2 (Nov.), 91–110.

Magic Lantern, 2013. Canon DSLR camera firmware. http://www.magiclantern.fm/.

Mangiat, S., and Gibson, J. 2010. High dynamic range video with ghost removal. In *Proc. SPIE 7798*, no. 779812, 1–8.

Mangiat, S., and Gibson, J. 2011. Spatially adaptive filtering for registration artifact removal in HDR video. In *ICIP 2011*, 1317–1320.

Mangiat, S. 2012. *High Dynamic Range and 3D Video Communications for Handheld Devices*. Doctoral thesis, University of California, Santa Barbara.

Mann, S., and Picard, R. W. 1995. On being 'undigital' with digital cameras: Extending dynamic range by combining differently exposed pictures. In *Proc. of Society for Imaging Science and Technology*, 442–448.

McGuire, M., Matusik, W., Pfister, H., Chen, B., Hughes, J., and Nayar, S. 2007. Optical splitting trees for high-precision monocular imaging. *IEEE Computer Graphics and Applications 27*, 2 (march-april), 32–42.

Nayar, S., and Branzoi, V. 2003. Adaptive dynamic range imaging: optical control of pixel exposures over space and time. In *Proceedings of ICCV 2003*, 1168 – 1175.

Nayar, S., and Mitsunaga, T. 2000. High dynamic range imaging: spatially varying pixel exposures. In *CVPR 2000*, 472 – 479.

Portz, T., Zhang, L., and Jiang, H. 2013. Adaptive dynamic range imaging: optical control of pixel exposures over space and time. In *Proceedings of ICCP 2013*.

Reinhard, E., Stark, M., Shirley, P., and Ferwerda, J. 2002. Photographic tone reproduction for digital images. *ACM Trans. Graph. 21*, 3 (July), 267–276.

Reinhard, E., Heidrich, W., Debevec, P., Pattanaik, S., Ward, G., and Myszkowski, K. 2010. *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*, second ed. Morgan Kaufmann.

Seger, U., Apel, U., and Höfflinger, B. 1999. HDRC-Imagers for natural visual perception. In *Handbook of Computer Vision and Application*, B. Jähne, H. Haußecker, and P. Geißler, Eds., vol. 1. Academic Press, 223–235.

Sen, P., Kalantari, N. K., Yaesoubi, M., Darabi, S., Goldman, D. B., and Shechtman, E. 2012. Robust patch-based HDR reconstruction of dynamic scenes. *ACM Trans. Graph. 31*, 6 (Nov.), 203:1–203:11.

Shechtman, E., Rav-Acha, A., Irani, M., and Seitz, S. 2010. Regenerative morphing. In *CVPR 2010*, 615–622.

Simakov, D., Caspi, Y., Shechtman, E., and Irani, M. 2008. Summarizing visual data using bidirectional similarity. In *CVPR 2008*, 1–8.

SpheronVR, 2013. http://www.spheron.com/.

Tocci, M. D., Kiser, C., Tocci, N., and Sen, P. 2011. A versatile HDR video production system. *ACM Trans. Graph. 30*, 4 (July), 41:1–41:10.

Unger, J., and Gustavson, S. 2007. High-dynamic-range video for photometric measurement of illumination. SPIE, vol. 6501, 65010E.

Zimmer, H., Bruhn, A., and Weickert, J. 2011. Freehand HDR imaging of moving scenes with simultaneous resolution enhancement. *Computer Graphics Forum 30*, 2 (Apr.), 405–414.