Eurographics Symposium on Rendering (2007)
Jan Kautz and Sumanta Pattanaik (Editors)

High Dynamic Range Image Hallucination

Lvdi Wang[1,2] Li-Yi Wei[1] Kun Zhou[1] Baining Guo[1] Heung-Yeung Shum[1]

[1]Microsoft Research Asia [2]Tsinghua University

Abstract

We introduce high dynamic range image hallucination *for adding high dynamic range details to the over-exposed and under-exposed regions of a low dynamic range image. Our method is based on a simple assumption: there exist high quality patches in the image with similar textures as the regions that are over or under exposed. Hence, we can add high dynamic range details to a region by simply transferring texture details from another patch that may be under different illumination levels. In our approach, a user only needs to annotate the image with a few strokes to indicate textures that can be applied to the corresponding under-exposed or over-exposed regions, and these regions are automatically hallucinated by our algorithm. Experiments demonstrate that our simple, yet effective approach is able to significantly increase the amount of texture details in a wide range of common scenarios, with a modest amount of user interaction.*

Categories and Subject Descriptors (according to ACM CCS): I.3.1 [Computer Graphics]: Hardware Architecture: Graphics Processors; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism: Texture;

Keywords: high dynamic range, image hallucination, texture synthesis

1. Introduction

High dynamic range (HDR) imaging [RWPD05] has made significant progresses recently, with technologies ranging from content creation [DM97], tone mapping on low dynamic range (LDR) displays [DD02, RSSF02, FLW02, LFUS06], and novel HDR display systems [SHS*04, LCTS05]. However, capturing real-world HDR content is not yet a common practice, as it involves either expensive HDR cameras or using LDR cameras to capture the same scene under multiple exposures, a process prone to motion problems. Moreover, most existing contents such as historical photos are still in LDR. Consequently if we can add HDR details to enhance LDR contents, then common users can experience exciting new HDR hardware and software systems with their existing LDR contents, including images, videos, and environment maps.

One possible method to achieve this goal is to reverse the tone mapping process (e.g. a piece-wise linear mapping [MDS07] or a nonlinear mapping [BLDC06]). However, this is often quite insufficient, since the LDR features that need HDR details the most are precisely those parts that are completely over-exposed or under-exposed (See Figure 1). To our knowledge, no solution exists so far that can adequately address this issue.

Since reconstructing HDR information for over or under exposed LDR regions is an under-constrained problem, it bears similarity to image hallucination [FJP02, SZTS03], where a high-resolution image is *hallucinated* from a single low-resolution original. The process involves training a Bayesian algorithm over a database of high-resolution/low-resolution image pairs, and then inferring the high-resolution details by matching the low-resolution information in the database. Despite this similarity, however, in practice the same Bayesian technique for traditional image hallucination cannot be applied to HDR hallucination, since the over/under-exposed regions do not contain enough information for finding correspondences in a database. Moreover, creating a database suitable for hallucinating all possible HDR images is a daunting task.

In this paper, we propose a new method for HDR hallucination that does not require any additional information, except for a modest amount of user interaction. The user identifies corresponding textures for each under/over exposed region using an interactive stroke tool (Figure 1). Using user supplied strokes, we automatically transfer the appropriate

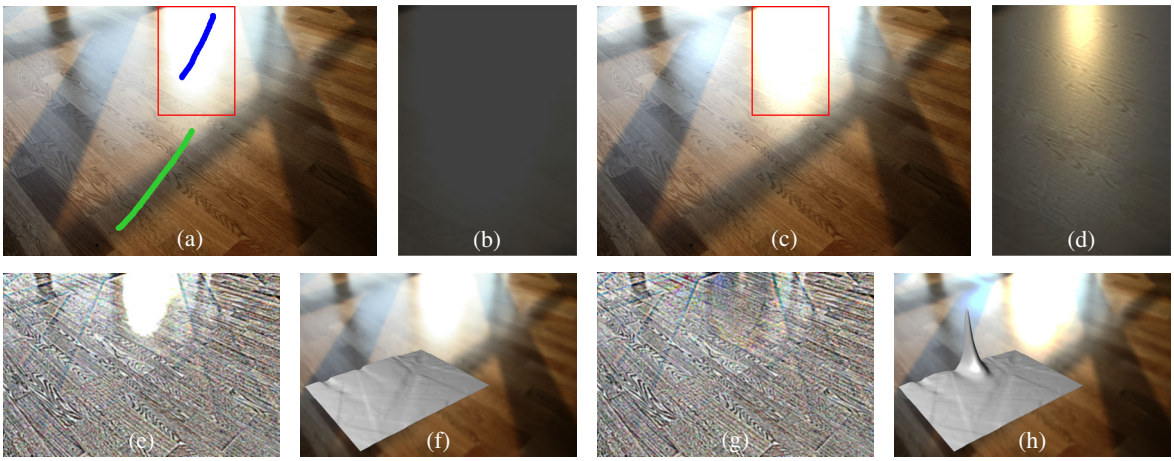

Figure 1: *HDR image hallucination. (a) Original image with an over-exposed region. The user selects this region via a blue stroke and a source region via a green stroke, and from these our algorithm automatically hallucinates the missing information with final result shown in (c). (b, d) are tone-mapped images of the red rectangular regions in (a, c). Our algorithm works by decomposing the original image into a high-frequency texture (e) and low-frequency illumination (f) components, hallucinating these two components separately (g, h), and combining these two components to yield the final result (c). Within the low-frequency illumination images (f, h) we also draw the corresponding illumination profiles for visualization.*

texture into the over/under exposed region and re-adjust the newly synthesized texture to an estimated brightness value in HDR radiance space.

Our underlying algorithm is based on the simple observation that many images contain repeating or nearly repeating texture patches under different illuminations. Hence, we can add HDR details to an over/under exposed region by transferring texture details from another patch of the same texture type that is under good illumination. By exploiting this observation, we take a constrained texture synthesis approach [EL99, DCOY03] for HDR hallucination.

However, unlike traditional texture synthesis where the entire texture is under roughly uniform lighting, in our scenario texture patches may exhibit drastically different illuminations. Consequently, direct application of traditional texture synthesis is inadequate. We address this issue by first decomposing the original image into a low-frequency illumination component and a high-frequency texture component using bilateral filtering [TM98, OCDD01, DD02]. We then hallucinate the high-frequency texture component via constrained synthesis and the low-frequency illumination component via elliptical Gaussian fitting. Finally, we combine these two components to yield the hallucination result.

Another issue with traditional texture synthesis is that it is not yet applicable to large-scale or semantic structures that require a level of image understanding or user interpretation, such as the wood planks in Figure 1. To handle this situation, we extend our stroke-based interface so that it can warp a source region into a destination region. Under the same user interaction, we have also provided a tool for adjusting local illumination levels. This is useful for artistic adjustments of results computed by our automatic Gaussian illumination fitting.

Using our user-friendly GUI with three stroke-based tools (texture, warp, and illumination), an ordinary user is capable of achieving a variety of convincing hallucination results. Beyond single images, our system can also be extended for hallucination from multiple images such as texture detail transfer from a different photograph when no detail is available from the original image, HDR environment map hallucination from an LDR environment map, and HDR video hallucination from an LDR video.

2. Algorithm

Our algorithm operates entirely in the radiance space. First, in the initialization phase of our algorithm, the input LDR, $\mathbf{I}_{ldr}$, is automatically converted into radiance space from a calibrated camera curve $f(x)$, i.e. let $\mathbf{I} = f(\mathbf{I}_{ldr})$, as shown in [Nay04]. Unfortunately, recovering the camera curve normally requires multiple images from the same camera. If this information is not available, the camera curve can be estimated from the distribution of luminance on image edges, as shown in [LGYS04]. However, in our current implementation we simply use a gamma curve with a 2.2 exponent value, and have found this works well in practice. Next, we identify the set of over-exposed or under-exposed pixels in the original LDR using simple thresholding on the relative luminance of the pixels.

Now that we have an image occupying an LDR subset of an HDR radiance space, we must fill in missing regions

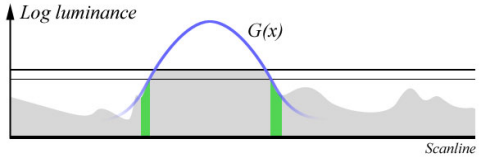

Figure 2: *Illustration of our Gaussian fitting process. We use a band of well illuminated pixels around the over-exposed region (shown in green) to estimate the Gaussian profile. The width of the (green) band is equivalent to the spatial variance parameter of the bilateral filter used in our texture/illumination decomposition.*

identified in previous steps. We achieve this by first decomposing the LDR in radiance space into a coarse illumination component and a detailed texture component. The process is as follows. First, we use bilateral filtering on $\mathbf{I}$ to produce the low frequency layer $\mathbf{L}_I$. (In theory it is better to perform bilateral filtering in the log space of luminance, but since our image starts out as in low dynamic range, empirically we have found it sufficient to filter in the original luminance space.) We then obtain the detail texture layer by simple division, i.e. $\mathbf{H}_I = \mathbf{I}/\mathbf{L}_I$ as in [OCDD01]. Next, we hallucinate each component independently. Note that this illumination/texture separation is essential for texture synthesis; otherwise regions with similar textures may have widely different illumination, causing difficulties in neighborhood search during constrained synthesis. While we may not achieve the correct separation in areas of high frequency illuminations and small scale shadows, it has the desired effect, since we do want to factor high frequency lighting variations into the texture part for synthesis as these effects are often repetitive in nature.

Our decomposition process just described is similar to the structure-and-texture algorithm in [BVSO03], but instead of their energy minimization, we opt to use bilateral filtering for its simplicity and computation speed. In addition, our goal is to separate texture details from large scale illumination instead of structures as in [BVSO03].

To hallucinate the illumination component, we estimate the radiance values for over exposed regions via interpolation from a linear combination of elliptical Gaussian kernels. More specifically, we first compute a weight $w(x)$ for each pixel x in the illumination layer:

$$w(x) = \begin{cases} \frac{C_{ue}-Y(x)}{C_{ue}} & Y(x) \in [0, C_{ue}) \\ 0 & Y(x) \in [C_{ue}, C_{oe}) \\ \frac{Y(x)-C_{oe}}{1-C_{oe}} & Y(x) \in [C_{oe}, 1] \end{cases} \quad (1)$$

where $Y(x) = r + 2g + b$ is the relative luminance and C_{ue}/C_{oe} are user adjustable threshold values. Specifically, pixels with $Y(x) \geq C_{oe}$ are considered to be over exposed, and pixels with $Y(x) \leq C_{ue}$ are considered to be under exposed. We then hallucinate illumination in log domain as

Figure 3: *Illustration of our segmentation + snapping process. The colored patches show segmentation results. The foreground stroke is from user painting (shown in green), whereas the background stroke is inferred directly from the segmentation boundary. The final snapped contour is shown in dashed contour.*

follows. We denote $L(x)$ as the logarithm of $Y(x)$. Then each individual over exposed region is separated and assigned a Gaussian lobe G to its centroid. The variances along x- and y-axis are decided by the size and shape of the region, usually the x- and y-extent of the region. The (log luminance) profile of the Gaussian is determined by an optimization procedure similar to [GKMD06] but using only pixels that are not over-exposed. See Figure 2 for an illustration. Finally, the new value $G(x)$ is blended with the original logarithmic luminance value $L(x)$ by the following formula:

$$O(x) = [1 - w(x)] \cdot L(x) + w(x) \cdot G(x) \quad (2)$$

This allows the well-illuminated regions largely unchanged (e.g. the wall in the gate scene shown in Figure 9). In addition, we have also implemented a stroke-based interface for this Gaussian fitting process to let the user interactively adjust illumination levels for artistic purposes. For a user selected brush size (σ_x, σ_y), we simply add a series of Gaussian lobes with variance (σ_x, σ_y) centered at the brush stroke.

To hallucinate the texture component, we fill in the over/under exposed regions via constrained texture synthesis. To achieve this, our system first automatically performs a segmentation of the original image into regions of similar color via graph cut [BVZ01]. At run time, the user first draws a stroke to indicate the desired source for texture synthesis, and the source region is automatically segmented via the graph cut based method in lazy snapping [LSTS04]. However, unlike [LSTS04] which requires two strokes for indicating the foreground and background regions, our system requires only one foreground stroke. The background stroke is automatically inferred from the surrounding boundary of the segmentation regions. See Figure 3 for an illustration. We note that our approach is similar to points of interest in [DCOY03], except our selection process is more automated. From this user selected region, we perform constrained texture synthesis to hallucinate the target regions. In our implementation we adopt the K-coherence based constrained optimization [HZW*06] for interactive synthesis.

For under-exposed regions, we leave the illumination un-

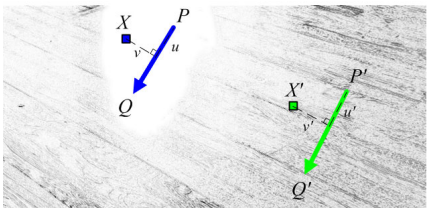

Figure 4: *Illustration of warping tool. For each target point X over the over/under exposed regions, its source location X' is determined via the relative positions of the source ($\overrightarrow{P'Q'}$) and destination ($\overrightarrow{PQ}$) strokes via [BN92].*

touched. We linearly scale the corresponding hallucinated texture values so that the maximum luminance $\leq C_{ue}$. In addition, we chose the value of C_{ue} so that it is greater than the noise level in the under-exposed regions.

In some cases, a level of image understanding is required to synthesize structured textons correctly into the image. For example, the wood planks (Figure 1) are highly structured and also contain some perspective information. It is unclear how they should be synthesized in a traditional texture optimization framework. In these cases, our algorithm adopts a warping tool. Here, the user selects an area similar to the region we are trying to recover using a stroke-based interface similar to our texture brush, and the target region is repaired via stroke-based image warping [BN92], as illustrated in Figure 4. In some sense, our warping tool is an extension of our optimization based texture synthesis since current texture synthesis algorithms cannot yet handle highly structured or large scale semantic information.

Finally, we blend the hallucinated textures, the hallucinated illumination map, as well as the original image to produce a final hallucinated HDR image. We perform blending via Poisson editing [PGB03] to smooth out the transition between our hallucinated areas and the original image.

3. Results and Discussion

Based on the algorithm described above, we have implemented a GUI system for interactive HDR hallucination. Our GUI provides three stroke-based tools: (1) a texture brush based on constrained texture synthesis, (2) a warping brush for structured or semantic information, and (3) an illumination brush for artistic adjustment of local lighting. All results shown in this paper are obtained using these three tools. The thresholds for over-exposure and under-exposure, as well as the standard deviations of the spatial and range Gaussians for bilateral filtering are empirically adjusted for each image. In most cases, our automatic illumination estimation suffices and the illumination brush is used sparingly. Once a user gets familiar with the tools, the interaction time for each example is from 1 to 5 minutes. Detailed statistics of user interactions for all our results are shown in Table 1.

For stochastic scenes such as those shown in Figure 9,

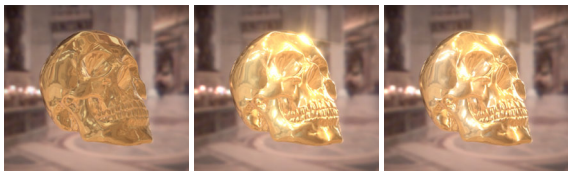

Figure 5: *Environment map hallucination. Original LDR map (left). Ground truth HDR map (middle). Our hallucination result (right).*

Figure 6: *Hallucination from different sources. Texture sample (left). The original over-exposed fire (middle). Our hallucination (right).*

we use primarily the texture brush for hallucination, including the waves in the stream scene, the bricks on the old city wall in the gate scene, the skin texture in the frog scene, and the clouds in the beach and bridge scenes. The church scene demonstrates a typical case in which traditional texture synthesis is not applicable - due to the large-scale structure of the windows we have to use our warping brush to hallucinate the rightmost window from the leftmost one. The illumination strokes have been applied to fine-tune the sky lighting on the beach, gate, and bridge scenes, as well as the detailed texture illumination of the stream scene. Note that due to our blending method (Equation 2), the illumination brush affects only over-exposed regions (this is why the wall in the Gate scene is unaffected). The frog scene demonstrates an under-exposed case.

In Figure 5, we have re-lighted a 3D environment with our HDR hallucination of St. Peter's Basilica. Even though our goal is not exact reconstruction, our result still compares favorably against the ground truth over the LDR original, which is much darker in several regions. In Figure 6, we have shown a case in which texture does not exist at all in the original LDR. However, we can simply transfer it from another image using our warp brush. Finally, our technique can also be applied to HDR video hallucination as shown in Figure 9, last case. There, we simply use our automatic Gaussian fitting to hallucinate the HDR illumination without any manual editing.

Figure 7 compares Adobe healing brush [Geo05] with our technique. The healing brush repairs a target region by copying from user selected source regions, so it does not work well for source regions that are small or fragmented. Our technique, in contrast, utilizes texture synthesis and is less sensitive to this problem. Furthermore, texture synthesis of-

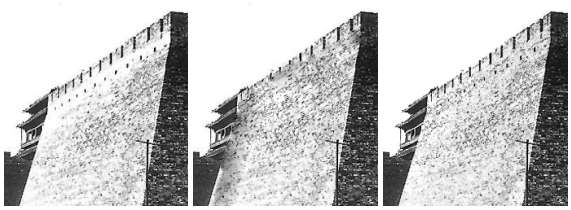

Figure 7: *Comparison with Adobe Photoshop healing brush. The cropped version of the original gate scene is on the left, the result by Photoshop healing brush is on the middle, and our hallucination result is on the right. Notice the major quality differences near the left side of the wall.*

Figure 8: *Limitations of our approach. The original images are on the left with our hallucination results on the right. Shown here are girl (top) and firework (bottom).*

ten avoids the unnatural repetitions caused by verbatim region copy in Photoshop healing brush. Also, since the healing brush is designed for LDR, but not HDR, images, it does not provide mechanisms to hallucinate the illumination component, nor do the copied regions comply with the target illumination level. Finally, the healing brush is more cumbersome to use and requires many more interactions, even for experienced users.

Figure 8 showcases limitations of our system. Basically, any scene that lacks proper texture information will be beyond our technique. In the girl scene, our technique can well recover the carpet. However, the girl's face and hair are impossible to recover due to their complex geometry and lack of texture information. In the firework scene, the sparks of the firework near the bottom-right corner are incorrectly hallucinated. The algorithm has no knowledge of the sparks shape in the over-exposed region, so the entire region is uniformly lit. However, it should be noted that our technique should never make an image worse, as the user can always decide which image regions to hallucinate.

Parameters To determine over/under exposed regions, we use $[C_{ue}, C_{oe}] = [0.05, 0.85]$ as "valid range" for all test images except the gate scene which we use $[0.0, 0.8]$. For bilateral filtering, we use a spatial parameter of 4 and a range parameter of 0.2 for all our scenes except wood, where parameters of 8 and 0.4 are used.

4. Conclusions and Future Work

In this paper, we have proposed a technique to hallucinate an HDR image from a LDR original with an interactive user interface. We have shown that excellent hallucinations can be obtained with surprisingly small amount of effort. A potential future work is to hallucinate secondary HDR effects such as scattering, glaring, and chromatic aberrations. Moreover, more advanced illumination models can be used along with geometry recovering methods, such as shape from shading and geometric completion, to generate more realistic hallucinations.

Acknowledgement We would like to thank Josh Bao for his participation in the early stage of this project, Becky Sundling for video dubbing, Dwight Daniels for suggestions on writing styles, and anonymous reviewers for their comments. Image courtesy of Paul Debevec, M. Gatton, Shane Brinkman-Davis, National Geographic, and Wikipedia.

References

[BLDC06] BANTERLE F., LEDDA P., DEBATTISTA K., CHALMERS A.: Inverse tone mapping. In *GRAPHITE '06* (2006), pp. 349–356.

[BN92] BEIER T., NEELY S.: Feature-based image metamorphosis. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques* (1992), pp. 35–42.

[BVSO03] BERTALMIO M., VESE L., SAPIRO G., OSHER S.: Simultaneous structure and texture image inpainting. *IEEE Transactions on Image Processing 12*, 8 (aug 2003), 882–889.

[BVZ01] BOYKOV Y., VEKSLER O., ZABIH R.: Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 11 (2001), 1222–1239.

[DCOY03] DRORI I., COHEN-OR D., YESHURUN H.: Fragment-based image completion. *ACM Trans. Graph. 22*, 3 (2003), 303–312.

[DD02] DURAND F., DORSEY J.: Fast bilateral filtering for the display of high-dynamic-range images. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), pp. 257–266.

[DM97] DEBEVEC P. E., MALIK J.: Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques* (1997), pp. 369–378.

[EL99] EFROS A. A., LEUNG T. K.: Texture synthesis by non-parametric sampling. In *ICCV '99: Proceedings of the International Conference on Computer Vision-Volume 2* (1999), p. 1033.

[FJP02] FREEMAN W. T., JONES T. R., PASZTOR E. C.: Example-based super-resolution. *IEEE Comput. Graph. Appl. 22*, 2 (2002), 56–65.

[FLW02] FATTAL R., LISCHINSKI D., WERMAN M.: Gradient domain high dynamic range compression. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), pp. 249–256.

[Geo05] GEORGIEV T.: Vision, healing brush, and fiber bundles. In *IS&T/SPIE Conf. on Hum. Vis. and Electronic Imaging X* (2005), vol. 5666, pp. 293–305.

[GKMD06] GREEN P., KAUTZ J., MATUSIK W., DURAND F.: View-dependent precomputed light transport using nonlinear gaussian function approximations. In *SI3D '06: Proceedings of the 2006 symposium on Interactive 3D graphics and games* (2006), pp. 7–14.

[HZW*06] HAN J., ZHOU K., WEI L.-Y., GONG M., BAO H., ZHANG X., GUO B.: Fast example-based surface texture synthesis via discrete optimization. *Vis. Comput. 22*, 9 (2006), 918–925.

[LCTS05] LEDDA P., CHALMERS A., TROSCIANKO T., SEETZEN H.: Evaluation of tone mapping operators using a high dynamic range display. *ACM Trans. Graph. 24*, 3 (2005), 640–648.

[LFUS06] LISCHINSKI D., FARBMAN Z., UYTTENDAELE M., SZELISKI R.: Interactive local adjustment of tonal values. *ACM Trans. Graph. 25*, 3 (2006), 646–653.

[LGYS04] LIN S., GU J., YAMAZAKI S., SHUM H.-Y.: Radiometric calibration from a single image. *Proceedings of Computer Vision and Pattern Recognition 02* (2004), 938–945.

[LSTS04] LI Y., SUN J., TANG C.-K., SHUM H.-Y.: Lazy snapping. *ACM Trans. Graph. 23*, 3 (2004), 303–308.

[MDS07] MEYLAN L., DALY S., SUSSTRUNK S.: Tone mapping for high dynamic range displays. In *Proc. of Human Vision and Electronic Imaging XII* (2007), vol. 6492.

[Nay04] NAYAR S. K.: Modeling the space of camera response functions. *IEEE Trans. Pattern Anal. Mach. Intell. 26* (2004), 1272–1282.

[OCDD01] OH B. M., CHEN M., DORSEY J., DURAND F.: Image-based modeling and photo editing. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), pp. 433–442.

[PGB03] PEREZ P., GANGNET M., BLAKE A.: Poisson image editing. *ACM Trans. Graph. 22*, 3 (2003), 313–318.

[RSSF02] REINHARD E., STARK M., SHIRLEY P., FERWERDA J.: Photographic tone reproduction for digital images. *ACM Trans. Graph. 21*, 3 (2002), 267–276.

[RWPD05] REINHARD E., WARD G., PATTANAIK S., DEBEVEC P.: *High Dynamic Range Imaging: Acquisition, Display, and Image-Based Lighting*. Morgan Kaufmann Publishers, 2005.

[SHS*04] SEETZEN H., HEIDRICH W., STUERZLINGER W., WARD G., WHITEHEAD L., TRENTACOSTE M., GHOSH A., VOROZCOVS A.: High dynamic range display systems. *ACM Trans. Graph. 23*, 3 (2004), 760–768.

[SZTS03] SUN J., ZHENG N.-N., TAO H., SHUM H.-Y.: Image hallucination with primal sketch priors. *CVPR 02* (2003), 729–736.

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *ICCV* (1998), pp. 839–846.

scene	# texture strk	# warp strk	# illumi strk
wood	0	2	0
bridge	4	0	2
beach	4	0	1
frog	2	0	0
stream	4	0	2
gate	4	0	2
church	0	4	0
X'mas tree	0	0	0
light bulb	2	0	1
env-map	0	0	1
fire	0	2	0
video	0	0	0
girl	2	0	0
firework	0	0	0

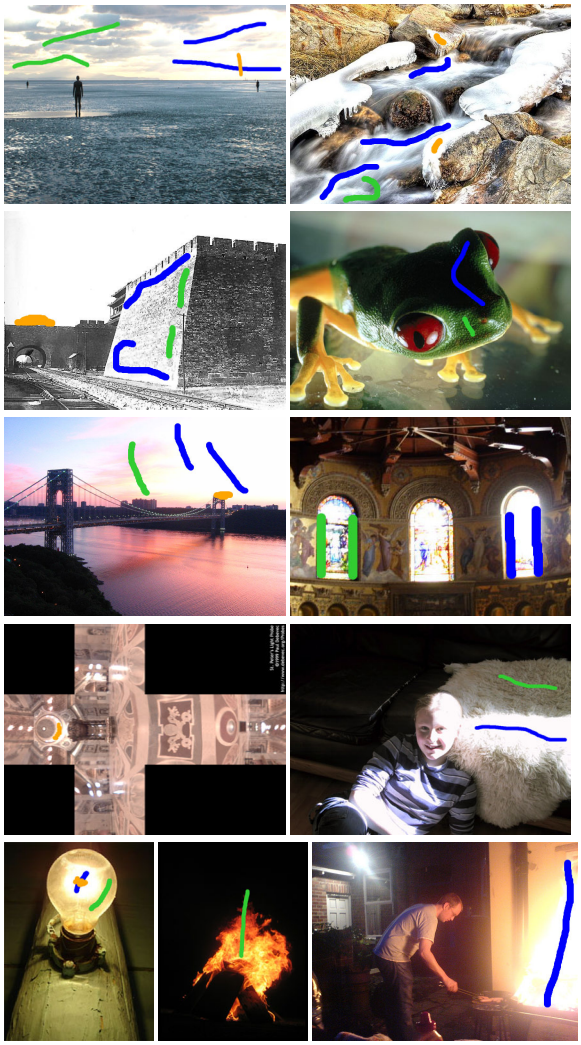

Table 1: *User interaction statistics for our results. Green: source texture/warp strokes. Blue: destination texture/warp strokes. Orange: illumination strokes. The # of texture and warp strokes include both source and destination. The Christmas tree, firework, and video results are produced automatically with no user interaction.*

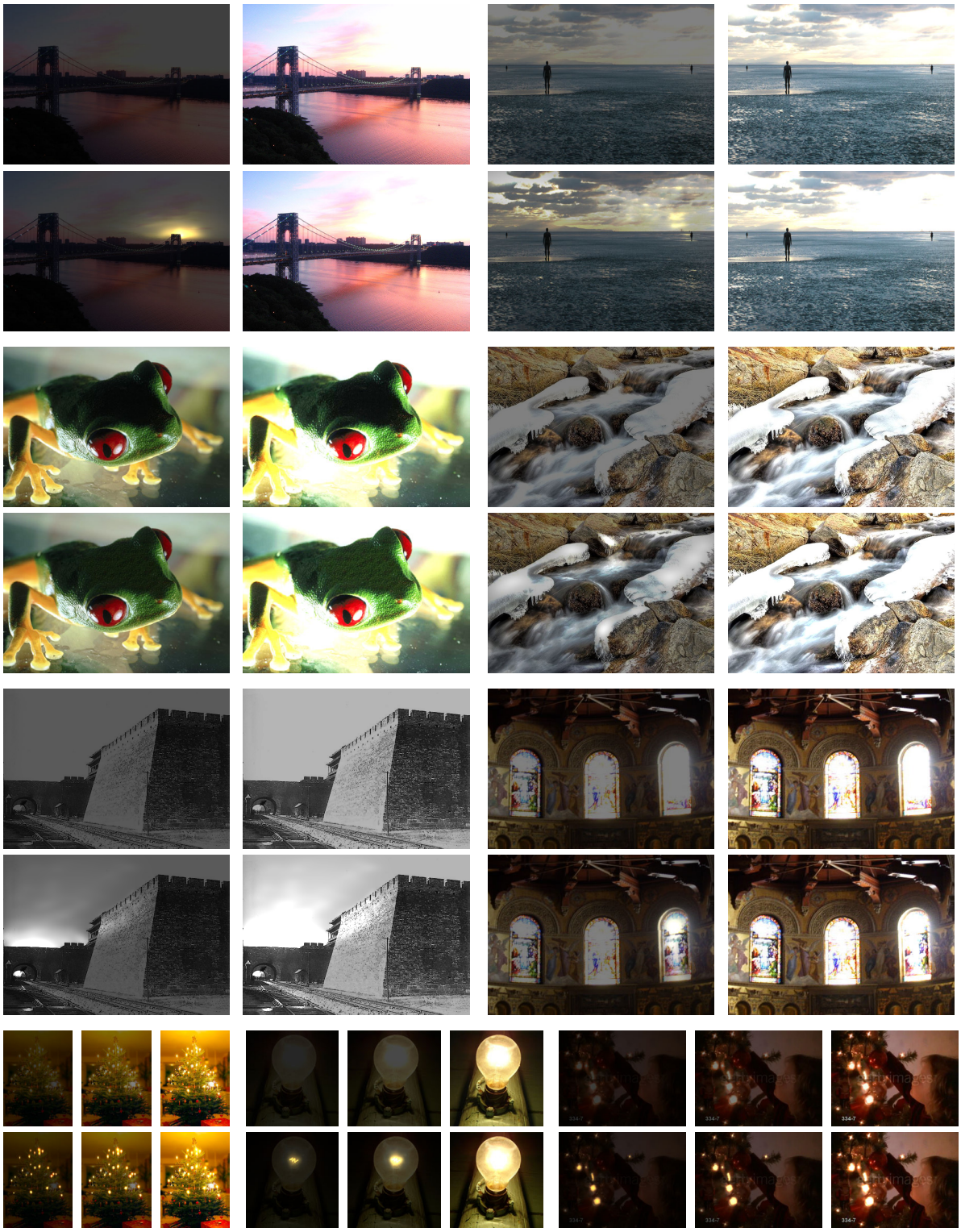

Figure 9: *HDR hallucination results. For each group of images, the original with different exposures is on the top, with our hallucination on the bottom. From top to bottom, left to right: bridge, beach, frog, stream, gate, church, Christmas tree, light bulb, and video. The frog case demonstrates texture hallucination over an under-exposed region (top of the head) while the other cases demonstrate over-exposed regions. Please refer to our accompanied video for details.*