High Dynamic Range Image Hallucination

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.

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.

Keywords: high dynamic range, image hallucination, texture synthesis

1 Introduction

High dynamic range (HDR) imaging [Reinhard et al. 2005] has made significant progresses recently, with technologies ranging from content creation [Debevec and Malik 1997; Mantiuk et al. 2006], tone mapping on low dynamic range (LDR) displays [Durand and Dorsey 2002; Fattal et al. 2002; Lischinski et al. 2006], and novel HDR display systems [Seetzen et al. 2004; Ledda et al. 2005]. 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. [Banterle et al. 2006]). 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 [Freeman et al. 2002; Sun et al. 2003], 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 novel 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 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 [Efros and Leung 1999; Drori et al. 2003] 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 [Tomasi and Manduchi 1998; Durand and Dorsey 2002]. 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, $I_{ldr}$, is automatically converted into radiance space from a calibrated camera curve $f(x)$, i.e. let $I = f(I_{ldr})$, as shown in [Nayar 2004]. Unfortunately, recovering the camera curve from the distribution of luminance on image edges, as shown in [Lin et al. 2004]. 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. Since our algorithm is texture-based, we also perform a denoising pass at under-exposed regions to avoid confusing noise for texture details.

Now that we have a clean image occupying an LDR subset of an HDR radiance space, we must fill in missing regions identified in previous steps. We achieve this by first decomposing the LDR in radiance space into a low frequency illumination component and a high frequency texture component as follows. First, we use bilateral filtering on $I$ to produce the low frequency layer $L_l$. We then obtain the high frequency layer by simple division, i.e. $H_l = I / L_l$ as in [Oh et al. 2001]. 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.

<table>
<thead>
<tr>
<th>scene</th>
<th># texture strokes</th>
<th># warp strokes</th>
<th># illum strokes</th>
</tr>
</thead>
<tbody>
<tr>
<td>wood</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>bridge</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>beach</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>carpet</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>stream</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>gate</td>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>church</td>
<td>0</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Xmas tree</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>light bulb</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>env-map</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>fire</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>video</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: User interaction statistics for our results. The # of texture and warp strokes include both source and destination strokes. The Christmas tree and video results are produced automatically with no user interaction.

To hallucinate the illumination component, we estimate the range values for over exposed regions via interpolation from a linear combination of elliptical Gaussian kernels [Green et al. 2006]. This process is fully automatic. 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. Specifically, for a user selected brush size $(\sigma_x, \sigma_y)$, we simply add a Gaussian blob with variance $(\sigma_x, \sigma_y)$ centered at the brush stroke point.

To hallucinate the texture component, we fill in the over/under exposed regions via constrained texture synthesis. To achieve this, 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 [Li et al. 2004]. We have increased the penalty for pixel differences in [Li et al. 2004] to produce a conservative area within the texture boundaries, since we are only interested in selecting texture samples, not to segment out the texture boundary. We note that our approach is similar to points of interest in [Drori et al. 2003], 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 [Han et al. 2006] for interactive synthesis.

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 [Beier and Neely 1992]. 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 high frequency textures, the hallucinated low frequency illumination map, as well as the original image to produce a final hallucinated HDR image. We perform blending via Poisson editing [Perez et al. 2003] 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 4, 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 fur texture in the carpet 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 and carpet scenes.

In Figure 2, 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 3, 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 4, last case. There, we simply use our automatic Gaussian fitting to hallucinate the HDR illumination without any manual editing.

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. In the future, we will build on this by using Bayesian learning to simplify the process even further. 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. For example, once geometry is recovered, we can then use texture shop [Fang and Hart 2004] to synthesize over the recovered surface, and use more advanced illumination models as well.

References

Figure 4: 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, carpet, stream, gate, church, Christmas tree, light bulb, and video. Please refer to our accompanied video for details.