

# Filtered Blending: A new, minimal Reconstruction Filter for Ghosting-Free Projective Texturing with Multiple Images

Martin Eisemann, Anita Sellent, Marcus Magnor

Computer Graphics Lab, TU Braunschweig  
Email: {eisemann,sellent,magnor}@cg.tu-bs.de

## Abstract

Whenever approximate 3D geometry is projectively texture-mapped from different directions simultaneously, annoyingly visible aliasing artifacts are the result. To prevent such ghosting in projective texturing and image-based rendering, we propose a new GPU-based rendering strategy and a new, view-dependent definition of ghosting. The algorithm is applicable to any kind of image-based rendering method, or general projective texture mapping, and adapts to arbitrary camera setups. It is able to cope with imprecise 3D geometry. Ghosting artifacts are efficiently eliminated at real-time rendering frame rates on standard graphics hardware. With the proposed rendering technique, better-quality rendering results are obtained from fewer images and coarser 3D geometry.

## 1 Introduction

Approximate geometry, camera calibration inaccuracies, and subcritical sampling are the reason for ghosting artifacts in light field rendering [15], lumigraph rendering [11], and view-dependent projective texture mapping [8]. In fact, ghosting/aliasing/double images<sup>1</sup> are a problem common to all image-based modeling and rendering applications whenever recorded image footage is to be merged with available geometry into one consistent representation. For highly accurate, laser-scanned geometry, a number of strategies have been devised for how to register photographs to 3D geometry in the presence of calibration inaccuracies [27, 3, 14], as well as how to generate a globally consistent texture map from multi-view footage [22, 2, 28]. While impressive digital models of real-world ob-

jects have been created this way, they come at the price of considerable user interaction, and not all approaches are suitable for reproducing view-dependent reflectance effects as the created texture map is usually not view-dependent.

In this paper, we present a new algorithm to achieve aliasing-free rendering results directly from a set of photographs in conjunction with some arbitrarily coarse geometry proxy. The contributions of this paper are

- a new view-dependent, anisotropic reconstruction filter that is able to take camera distribution, sampling density, the new virtual viewpoint and geometry inaccuracy into account and that is applicable to any kind of projective texturing;
- a real-time implementation of the proposed filter on standard graphics hardware;
- a new, conservative, yet more general definition for the causes of ghosting in projective texturing which takes the current viewpoint into account and provides a new upper bound on the highest representable frequency without ghosting, which results in more details in the output image.

Our main goal is to improve the visual quality of existing image-based modeling and rendering methods as well as to simplify the use of image-based approaches for representing real-world objects.

Our paper is organized as follows. After reviewing relevant previous work in Section 2 we examine the underlying problem of ghosting artifacts in multi-image projective texture mapping, Section 3. In Section 4 we describe *filtered blending* as a way to eliminate ghosting. Implementation details are given in Section 5, and experimental evaluation results are presented in Section 6 before we conclude with Section 7.

<sup>1</sup>Throughout the paper, we use the terms “ghosting”, “double images” and “aliasing” synonymously if not stated otherwise.



Figure 1: Images from our test data sets: for the synthetic *Bunny* and the real-world captured *Garfield*, approximated 3D geometry models are available. For the synthetic light fields *Buddha* and *Dragon*, a planar surface must suffice as geometry proxy, see Table 1 for more information on our test data sets.

	<b>Bunny</b>	<b>Garfield</b>	<b>Buddha</b>	<b>Dragon</b>
# geometry primitives	948	1280	1	1
Total images	49	24	256	256
Pixels per image	$512^2$	$768 \times 576$	$256^2$	$256^2$
Uncertainty offset	0.63%	1.18%	7.07%	8.13%
Band-limit filter support	12 pixels	10 pixels	12 pixels	10 pixels
Viewport	$360^\circ \times 360^\circ$	$360^\circ \times 180^\circ$	$90^\circ \times 90^\circ$	$90^\circ \times 90^\circ$
Output resolution (pixels)	$512^2$	$512^2$	$512^2$	$512^2$
Type	synthetic	real-world	synthetic	synthetic

Table 1: Information concerning our test data sets shown in Figure 1. The uncertainty offset along the viewing ray in positive and negative direction, in comparison to the diagonal of the geometries bounding box.

## 2 Related Work

**Image-based rendering (IBR)** methods are able to achieve highly realistic rendering results of real-world objects or scenes from a collection of calibrated photographs. While some IBR methods rely solely on a large number of input images to minimize aliasing artifacts [15, 18], most IBR approaches make additional use of scene depth [11, 13, 4, 29], or full 3D geometry [8, 5, 26, 23, 24]. Potential sources for aliasing artifacts during rendering are (1) image calibration inaccuracies, (2) subcritical sampling in conjunction with insufficient pre-filtering [6, 16], and possibly (3) imprecise depth maps or inexact geometry.

**Image-based modeling (IBM)** extends the notion of IBR in that high-quality 3D geometry scans of an object are augmented with a collection of photos to capture the visual appearance [22, 27, 2, 14, 28]. Finite scanner resolution and tolerances, registration inaccuracies, and camera calibration errors all degrade overall image-to-texture mapping accuracy.

**Different reconstruction filters** for IBR have been investigated in the literature. Based on an analysis of the sampling problem in frequency and ge-

ometry space by Chai *et al.* [6] and Lin *et al.* [16], respectively, one can apply a low-pass filtering to the input/output images for ghosting-free “band-limited reconstruction” [25]. Isaksen *et al.* [13] propose a “wide-aperture reconstruction filter” which increases the spatial support or aperture size of the reconstruction filter. A combination of the two approaches is proposed by Stewart *et al.* [25]. Alternatively, Liu *et al.* [17] estimate scene geometry dynamically using a color similarity-based plane sweeping algorithm.

These approaches effectively reduce ghosting, but several issues remain unsolved. Not all of these approaches can be applied to general projective texturing. Many details are lost because the filtering operations are usually performed as a pre-processing step based on the maximum disparity, not taking the current viewing position into account. Not all approaches are able to preserve view-dependent reflectance characteristics. And a lot of user-interaction may be needed. In some approaches new artifacts may be introduced due to random color similarities. With wider camera baselines, these mismatch artifacts increase disproportionately.

**Warping Techniques** which establish dense correspondences between pixels in different images can produce most accurate results [7, 19]. However, specular and occluding surfaces pose a great challenge for the necessarily precise automatic camera calibration and depth acquisition. On the other hand, using only a sparse feature set, is usually problematic as feature detectors may select significantly different features in different images of the same scene. In addition many feature detectors are specialized to certain environments [1], and may not work as well for other settings.

### 3 Problem Description

In this section, we take a closer look at the causes of ghosting in projective texture mapping and light field rendering. We show that ghosting is solely dependent on the maximum disparity of a projected scene point to its real position in texture space. Therefore, ghosting can be detected even if only a single input camera and the virtual camera is taken into account.

Every pixel of the input images can be seen as the weighted integral of the light arriving at the image plane of the camera. Every scene point  $L$  of sufficiently small size, compared to the camera resolution, therefore contributes exactly to one pixel in the recorded images (for more details see [16]). During rendering, these are then reprojected onto an approximated surface. Lin *et al.* [16] state that the intensity contributions of each scene point  $L$  must at least touch each other in the output image to avoid ghosting. This is true for light field approaches if virtual and capturing cameras have the same resolution and the user is restricted to stay outside the convex hull defined by camera and focal plane. However, other rendering approaches, like projective texturing and geometry-assisted IBR, demand a more general definition of ghosting. Thus we propose that every scene point  $L$  must provide a single, resolution independent intensity maximum in the output image, Figure 2.

As one example, let's consider the viewing ray  $C_v L$ , Figure 2. By projecting the input image of camera  $C_1$  onto the approximate surface  $S$ , the contribution of  $L$  will not appear at point  $F_0$ , but rather at point  $F_1$ , revealing a disparity of  $d$ . If  $d$ , projected into the input image, is larger than half a pixel, ghosting artifacts may appear, as the

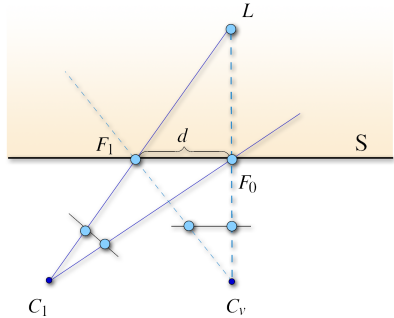


Figure 2: Ghosting in projective texture mapping, view-dependent texture mapping and light field rendering: The actual scene point  $L$ , as recorded from cameras  $C_1$  and  $C_v$ , is projected to two different points  $F_1$  and  $F_0$  on the approximate object surface  $S$ . If the projected distance  $d = F_0 - F_1$  is larger than half a pixel in the input images, ghosting occurs.

main contribution of  $L$  might not appear at  $F_0$  after blending different input image values together.

This definition of ghosting is applicable to IBR in general because it gives room for view-dependent filtering of the input images, based on the current viewpoint, as we will show in Section 4. Note that our definition reduces to the one proposed by Lin *et al.* if their preconditions are fulfilled.

### 4 Filtered Blending

To motivate our approach, consider the diagram in Figure 3. The scene point  $L$  lies on the line of sight of the viewing ray  $C_v L_{p_0}$ , somewhere within the interval of maximum depth uncertainty  $d_{max}$  from the approximate geometry, which for this analysis we assume to be known. The line segment  $L_{p_1} L_{p_2}$  projected into the texture space of camera  $C_1$  reveals another line segment  $T_{L_{p_1}} T_{L_{p_2}}$ , which we call the line of disparity. Any value on this line could be the correct texture value. This is in fact similar to an epipolar geometry constraint [12].

We solve this uncertainty problem in a resampling process. Recall that ghosting is prevented if the projected disparity  $d$  in the texture images is less than  $1/2$  texel (see Section 3). In frequency domain, let  $\omega_t$  be the highest representable frequency in the texture function  $t$ , which is given by

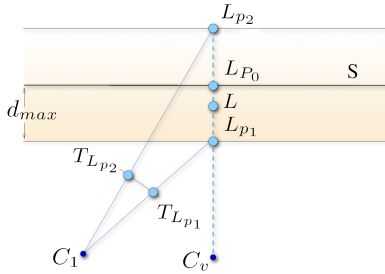


Figure 3: Scene point estimation. The scene point  $L$  observed from viewpoint  $C_v$  can only be estimated to lie somewhere between  $L_{p_1}$  and  $L_{p_2}$ , which are defined by the maximum depth uncertainty  $d_{max}$ . Its correct color value observed by camera  $C_1$  lies somewhere between the projected texture coordinates  $T_{L_{p_1}}$  and  $T_{L_{p_2}}$ .

its resolution and defined by the *Nyquist Theorem*. Then it follows that if we want to remove ghosting, frequencies higher than  $\frac{1}{2d}\omega_t$  have to be removed. But as we are dealing with discrete values, simply removing these frequencies is insufficient. One also has to consider appropriate sampling positions. Choosing  $\frac{1}{2}T_{L_{p_1}}T_{L_{p_2}}$  as the sampling position and  $|T_{L_{p_1}}T_{L_{p_2}}| - \epsilon$  as the sampling distance, with  $\epsilon \rightarrow +0$ , we anisotropically resample the texture function along the line of disparity at the highest possible frequency which assures that no ghosting will appear. This way we effectively avoided ghosting, since the correct texture values always contribute to the corresponding output pixels. As we take the current viewpoint into account, the closer the virtual camera is to one of the input cameras, the fewer frequencies are cut off from that input image and the output image will contain much more details than in a band-limiting approach. If the input camera and virtual camera coincide, all detail is preserved. This way, we implicitly take the input camera distribution into account, as the size of our filter is based on the geometric uncertainty and position of the input cameras.

The depth uncertainty itself can be established in different ways. In two-plane parameterized light field rendering, it is usually the difference along the  $z$ -axis from the focal plane, which is orthogonal to this axis by definition. For synthetic scenes the value of uncertainty is usually known in advance, it is to be estimated for real world scenes. Then

$L_{p_1}$  and  $L_{p_2}$  can be calculated by intersecting every viewing ray with the plane at  $Z_{min}$  and  $Z_{max}$ , which are the minimum and maximum  $z$ -values in the scene, respectively. In a more general setup, one could decide to either create an offset along the normal of the approximate surface with which the viewing ray is intersected, or the offset is created along the viewing ray. In the first case the calculated disparity becomes very large at objects silhouettes, as the normal is almost perpendicular to the viewing ray's direction. This leads to strong and distracting blurring artifacts. We therefore chose to calculate the offset along the viewing ray. This results in a small blur for images close to the current viewpoint and larger blur for those farther away. This is similar to an angular metric, and no sudden jumps at triangle borders appear.

Since the support of the applied low-pass filter can theoretically become arbitrarily large, we take two simple steps to alleviate the needed effort. First, we make strong use of GPU processing power. The whole filtering algorithm is implemented as a pair of vertex and fragment shaders. Second, we trade off detail for speed by applying a multi-resolution technique. We set a threshold  $\nu$  for the filtersize  $\mu$  in texture space. If this threshold is exceeded we use the  $n$ -th level of the input image-pyramid computed in a preprocess instead of the image itself, with  $n = \log_2(\frac{\mu}{\nu})$ . Note that this approach has almost no effect on the visual quality of the output, since a large filter size implicates a small weighting factor for an input camera and therefore only a small contribution to the output image, but speeds up the whole rendering process by a factor of roughly three.

Interestingly, depending on the movement, the varying change in blur in the output image can evoke the impression of repeatedly changing speed, even if a movement is in fact constant. We can solve this perceptual problem by applying a simple motion blur technique. If the viewpoint does not change, the image quickly converges to the optimally filtered solution.

Note that our algorithm is independent of the weighting scheme used for the image synthesis step, where the acquired texture values are combined to reveal the final pixel value, and can be used in conjunction with quadralinear interpolation [15], the unstructured lumigraph weighting scheme [4] or angular distance measures [9, 21].

## 5 Implementation

We implemented our algorithm on a NVidia GeForce 8800GTX graphics card using OpenGL/GLSL. For the filtered blending approach we add/subtract the estimated or a priori known depth uncertainty offset along the viewing ray from the vertex position. Reprojecting the new positions into the different input images yields the texture coordinates.

For the resampling process during filtered blending, we implemented the Mitchell-Netravali cubic B-spline filter as a fragment shader program [20]. We compared different filters, e.g., also truncated Gaussian and box filter, and found that the Mitchell-Netravali filter yields the visually most convincing rendering results.

## 6 Results, Discussion

Our test data sets include one real-world object, *Garfield*, one synthetically created 3D object, *Bunny*, and the two well-known Stanford light fields *Buddha* and *Dragon*. Figure 1 shows examples of the test data sets. Additional data, like the used depth uncertainty or size of the band-limiting filter, is listed in Table 1. To evaluate rendering quality, we compare our rendering strategies to direct (quadra-)linear interpolation as well as to pre-processed band-limited filtering. For band-limited filtering, the filter support is set to the smallest possible value to prevent ghosting. In projective texture mapping, we always select the three nearest cameras for interpolation as described in [21].

Our first test scene *Bunny* consists of 49 images rendered from randomly selected viewing directions. The upper two rows in Figure 4 depict the results obtained by the different rendering approaches. For better rendering quality assessment, some of the details are enlarged. In the first row, the virtual camera is close to one of the input cameras, in the second it is placed right between them. When comparing the two leftmost images, which correspond to standard linear blending (Figure 4a) and band-limited filtering (b) to our result on the right(c), note how the ghosting is smoothed away by the filtered blending, while discontinuities of the checkerboard texture are much better preserved. We achieve 342 fps when using our filtered blending approach.

To acquire the calibrated images for the *Garfield* scene a computer-controlled turntable and a digital camera with a lever arm was used to record 24 images in hemispherical configuration. The geometry is estimated using the voxel-based approach by Eisert *et al.* [10]. Rendering results are shown in Figure 4 third row. Again, the noticeable ghosting artifacts along the pupil in the image on the left are eliminated in our approach, while some of the details at the nose could be preserved. The *Garfield* model is rendered at approximately 334 fps.

We also tested our “ghost-busting” approach for light field rendering using the *Buddha* and *Dragon* data sets (Figure 1). Rendering results are shown in Figure 5. Notice how ghosting is prevented in our approach, Figure 5(c), while ghosting artifacts are obvious in standard quadrilinear interpolation (a). At the same time, much finer detail is preserved than if pre-processed band-limited filtering is used (b). In conjunction with light field rendering, we achieve around 105 fps with our approach.

## 7 Conclusions

We have presented an approach to achieve ghosting-free rendering results with viewpoint optimized, minimal, low-pass filtering for subcritically sampled light fields, as well as for general projective texture mapping with approximated geometry. In contrast to conventional methods based on prefiltering/band-limiting, our algorithm efficiently eliminates ghosting and preserves texture details considerably better, because our filtered blending takes the current viewpoint into account. Our definition of ghosting is able to conservatively establish the bounds for ghosting. Real-time rendering performance is achieved on a standard GPU. And the approach can be easily adapted to various different image-based rendering scenarios.

However there are certain limitations. If the input samples are too sparse, the image will still look blurry. Warping techniques might help to visually increase the resulting image even more. We are currently working on this.

Filtered blending greatly eases the constraints of image-based rendering: coarser 3D geometry, and fewer input images are sufficient to still achieve convincing rendering results. With this useful generalization, image-based rendering will hopefully find many new practical applications.

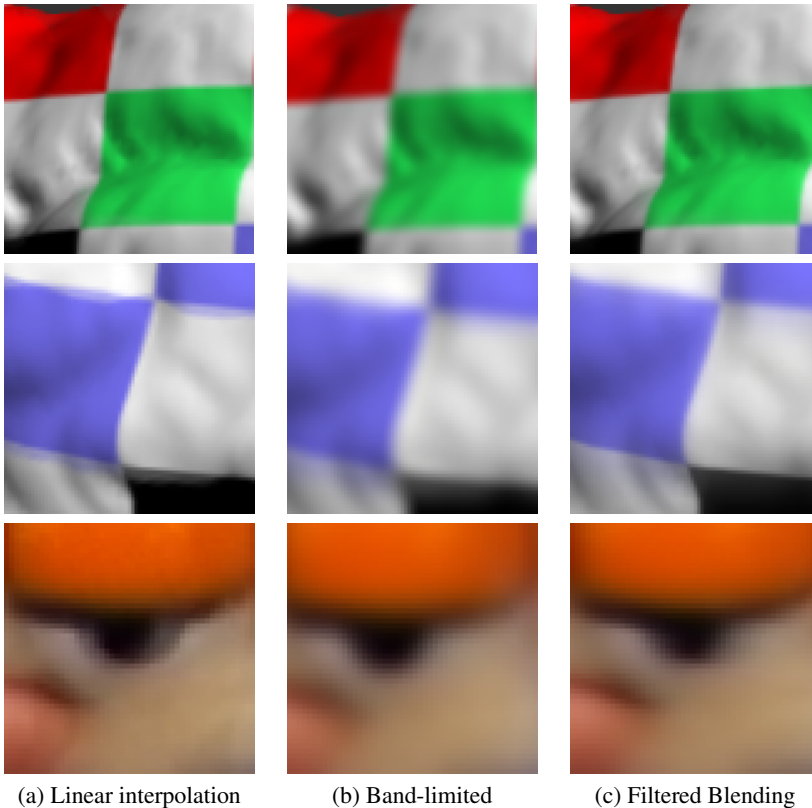
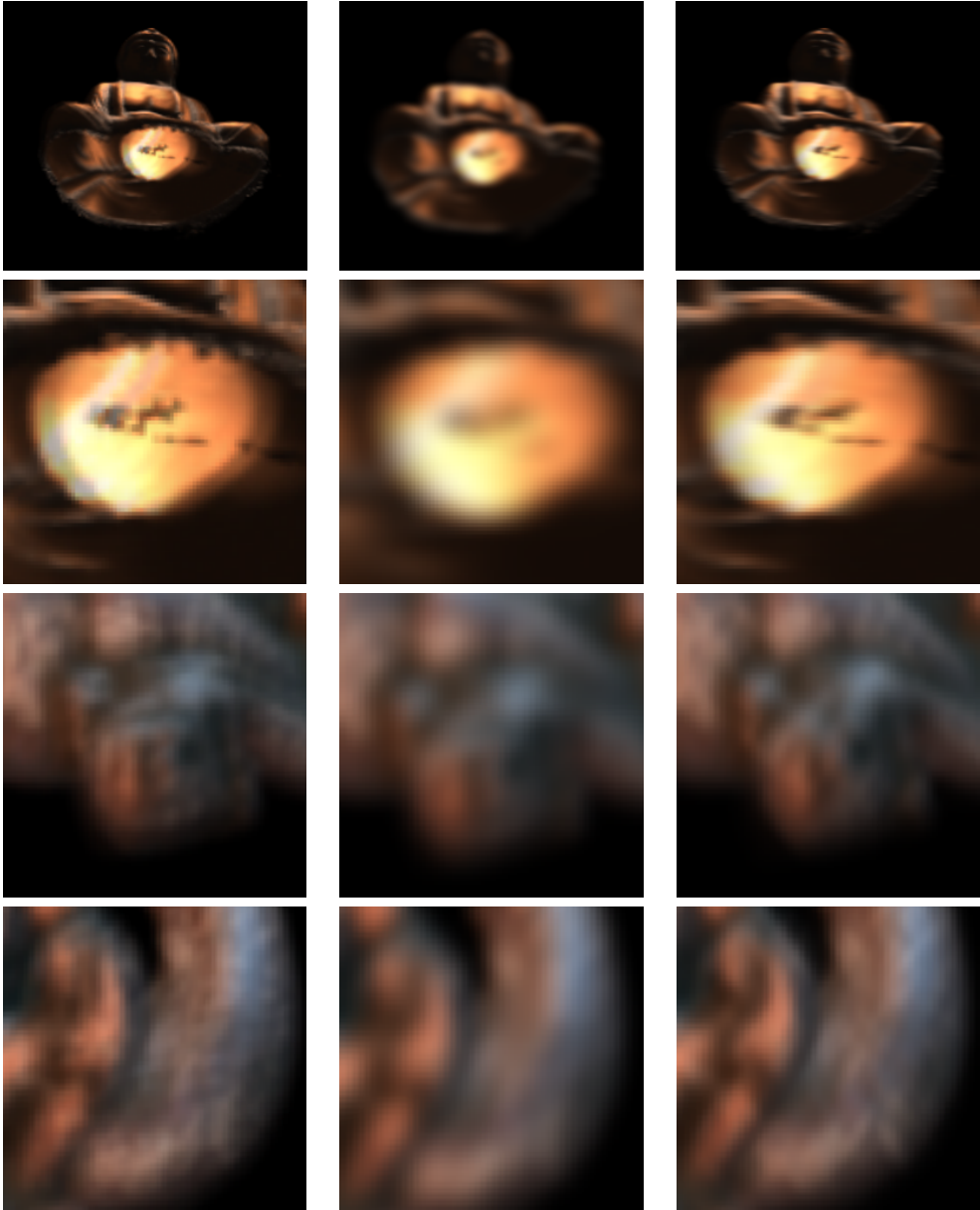


Figure 4: Projective texture mapping of the *Stanford Bunny* (rows 1 and 2) and of the real-world data set *Garfield* (row 3). In row 1 the virtual camera is close to one of the input cameras, in row 2 and 3 it is set in-between two of them: (a) Linear interpolation reveals strong ghosting around high frequency details (row 2 and 3). (b) Band-limited reconstruction removes ghosting, but the result is excessively blurred. (c) Our filtered blending approach preserves discontinuities considerably better and completely removes aliasing. Notice the much sharper edges on the bunny in row 1 and 2 and the preserved brown stripe on the *Garfield*'s nose.

## References

- [1] Daniel G. Aliaga, Dimah Yanovsky, Thomas Funkhouser, and Ingrid Isenhardt. Interactive Image-Based Rendering Using Feature Globalization. In *13D '03: Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 163–170, New York, NY, USA, 2003. ACM Press.
- [2] Adam Baumberg. Blending images for texturing 3d models. In Paul L. Rosin and A. David Marshall, editors, *Proceedings of the British Machine Vision Conference*, 2002.
- [3] Fausto Bernardini, Ioana M. Martin, and Holly Rushmeier. High-quality texture reconstruction from multiple scans. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):318–332, 2001.
- [4] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured Lumigraph Rendering. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 425–432, 2001.
- [5] J. Carranza, C. Theobalt, M. Magnor, and H. P. Seidel. Free-viewpoint video of human actors. In *SIGGRAPH '03: Proceedings of the 30th annual conference on Computer graphics and interactive techniques*, pages 569–577, 2003.
- [6] Jin-Xiang Chai, Shing-Chow Chan, Heung-Yeung Shum, and Xin Tong. Plenoptic Sampling. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 307–318, 2000.

- [7] Shenchang Eric Chen and Lance Williams. View Interpolation for Image Synthesis. Number Annual Conference Series, pages 279–288, 1993.
- [8] Paul Debevec, George Boshokov, and Yizhou Yu. Efficient View-Dependent Image-Based Rendering with Projective Texture-Mapping. *9th Eurographics Rendering Workshop*, pages 105–116, 1998.
- [9] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and Rendering Architecture from Photographs: A Hybrid Geometry- and Image-Based Approach. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, 1996.
- [10] Peter Eisert, Eckehard Steinbach, and Bern Girod. Multi-hypothesis volumetric reconstruction of 3-d objects from multiple calibrated camera views. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP'99)*, pages 3509–3512, 1999.
- [11] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The Lumigraph. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 43–54, 1996.
- [12] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. 2 edition, 2003.
- [13] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. Dynamically Reparameterized Light Fields. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 297–306, 2000.
- [14] Hendrik P. A. Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Trans. Graph.*, 22(2):234–257, 2003.
- [15] Marc Levoy and Pat Hanrahan. Light Field Rendering. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 31–42, 1996.
- [16] Zhouchen Lin and Heung-Yeung Shum. A Geometric Analysis of Light Field Rendering. *International Journal of Computer Vision*, 58(2):121–138, 2004.
- [17] Yang Liu, George Chen, Nelson Max, Christian Hofsetz, and Peter McGuinness. Undersampled Light Field Rendering by a Plane Sweep. *Computer Graphics Forum*, 25(2):225–236, June 2006.
- [18] W. Matusik and H. Pfister. 3D TV: A scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. In *SIGGRAPH '04: Proceedings of the 31st annual conference on Computer graphics and interactive techniques*, pages 814–824, 2004.
- [19] Leonard McMillan and Gary Bishop. Plenoptic Modeling: An Image-Based Rendering System. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 39–46, 1995.
- [20] Don P. Mitchell and Arun N. Netravali. Reconstruction Filters in Computer-Graphics. In *SIGGRAPH '88: Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 221–228, 1988.
- [21] Kari Pulli, Michael Cohen, Tom Duchamp, Hugues Hoppe, Linda Shapiro, and Werner Stuetzle. View-Based Rendering: Visualizing Real Objects from Scanned Range and Color Data. In Julie Dorsey and Phillipp Slusallek, editors, *Proceedings of the Eurographics Workshop on Rendering*, pages 23–34, 1997.
- [22] Claudio Rocchini, Paolo Cignoni, Claudio Montani, and Roberto Scopigno. Multiple textures stitching and blending on 3D objects. In *Proceedings of the Eurographics Workshop on Rendering*, pages 119–130, 1999.
- [23] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. In *SIGGRAPH '06: Proceedings of the 33rd annual conference on Computer graphics and interactive techniques*, pages 835–846, 2006.
- [24] J. Starck and A. Hilton. Surface capture for performance based animation. *IEEE Computer Graphics and Applications*, 27(3):21–31, 2007.
- [25] J. Stewart, J. Yu, S. J. Gortler, and L. McMillan. A New Reconstruction Filter for Undersampled Light Fields. In *Proceedings of the Eurographics Workshop on Rendering*, pages 150–156, 2003.
- [26] S. Vedula, S. Baker, and T. Kanade. Image based spatio-temporal modeling and view interpolation of dynamic events. *ACM Transactions on Graphics*, 24(2):240–261, April 2005.
- [27] Daniel N. Wood, Daniel I. Azuma, Ken Aldinger, Brian Curless, Tom Duchamp, David H. Salesin, and Werner Stuetzle. Surface light fields for 3d photography. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 287–296, 2000.
- [28] Kun Zhou, Xi Wang, Yiyang Tong, Mathieu Desbrun, Baining Guo, and Heung-Yeung Shum. Texturemontage: Seamless texturing of arbitrary surfaces from multiple images. In *SIGGRAPH '05: Proceedings of the 32nd annual conference on Computer graphics and interactive techniques*, pages 1148–1155, 2005.
- [29] C. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. In *SIGGRAPH '04: Proceedings of the 31st annual conference on Computer graphics and interactive techniques*, pages 600–608, 2004.



(a) Quadra-linear interpolation

(b) Band-limited

(c) Filtered Blending

Figure 5: Comparison for the two sub-critically sampled light fields *Buddha* (top rows) and *Dragon* (bottom rows): (a) Quadralinear interpolation cannot suppress ghosting artifacts. (b) Band-limiting the entire light field leads to excessively blurry results. (c) Our filtered blending approach preserves most of the details while ghosting is completely avoided.