Noname manuscript No.
(will be inserted by the editor)

Dynamic Depth of Field on Live Video Streams: A Stereo Solution

First Author · **Second Author**

Received: date / Accepted: date

Abstract The ability to produce dynamic Depth of Field effects in live video streams was until recently a quality unique to movie cameras. In this paper, we present a computational camera solution coupled with real-time GPU processing to produce *runtime* dynamic Depth of Field effects. We first construct a hybrid-resolution stereo camera with a high-res/low-res camera pair. We recover a low-res disparity map of the scene using GPU-based Belief Propagation and subsequently upsample it via fast Cross/Joint Bilateral Upsampling. With the recovered high-resolution disparity map, we warp the high-resolution video stream to nearby viewpoints to synthesize a light field towards the scene. We exploit parallel processing and atomic operations on the GPU to resolve visibility when multiple pixels warp to the same image location. To reduce aliasing, we further develop an image-space filtering technique to compensate for spatial undersampling. Finally, we generate dynamic Depth of Field effects from the synthesized light field rendering. All processing stages are mapped onto NVIDIA's CUDA architecture. Our system can produce Depth of Field effects with arbitrary aperture sizes and focal depths for the resolution of 640×480 at 15 fps.

Keywords First keyword · Second keyword · More

1 Introduction

Depth of field (DoF) effects are a useful tool in photography and cinematography because of their aesthetic value. In photography, they have been used to emphasize objects by creating a shallow plane of focus around the subject while blurring the rest of the scene. In cinematography, movie cameras use *dynamic* DoF effects to shift the viewer's attention from the foreground to the background or vice versa. Producing a high quality DoF, however, requires dedicated and often expensive lens systems. Commodity lenses such as the low cost Canon EF 50mm f/1.4 use a small number of aperture blades and produce blur artifacts caused by the polygon shaped apertures.

Varying the DoF effect and displaying the results in real-time was until recently a quality unique to movie cameras. For example, classical digital SLRs, while capable of modifying the DoF effect from shot to shot, can only produce static images. Of late, high-end DSLRs can stream video, but this capability comes at increased cost to the consumer and still yields a non-ideal video platform. Additionally, current DSLRs adjust focus using a ring on the lenses and DoF adjustments tend to produce momentary tilting of the captured video. Movie cameras, while capable of runtime dynamic DoF effects, tend to be bulky and unwieldy in order to accommodate complex corrective lenses arrays and storage media. Further, fully servoed studio lenses such as the Canon DIGI SUPER 25 XS can cost tens of thousands of dollars.

Our work is inspired by the recent light field imaging systems [9,18,19,8]. The Stanford light field camera array [20,21,16,17] is a two dimensional grid composed of 128 1.3 megapixel firewire cameras which stream live video to a stripped disk array. The large volume of data generated by this array forces the DoF effect to be rendered in post processing rather than in real-time. Furthermore, the system infrastructure such as the camera grid, interconnects, and workstations are bulky, making it less suitable for on-site tasks. The MIT light field camera array [22] uses a smaller grid of 64 1.3 megapixel usb webcams instead of firewire

F. Author
first address
Tel.: +123-45-678910
Fax: +123-45-678910
E-mail: fauthor@example.com

S. Author
second address

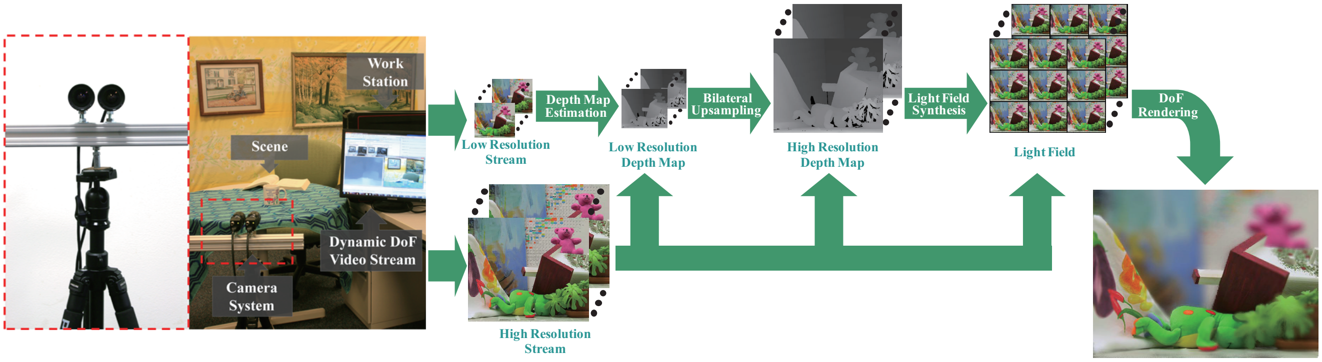

Fig. 1 The imaging hardware and the processing pipeline of our dynamic DoF video acquisition system. All processing modules are implemented on NVIDIA's CUDA to achieve real-time performance.

cameras and is capable of synthesizing real-time dynamic DoF effects. Both systems, however, still suffer from spatial aliasing because of the baseline between neighboring cameras. The camera spacing creates appreciable differences between the pixel locations of the same scene point in neighboring cameras producing an aliasing effect at the DoF boundary when their images are fused.

In an attempt to reduce aliasing artifacts in light-field based solutions, Ng [11] designed a light field camera that combines a single DSLR with a microlenslet array. Each lenslet captures the scene from a different viewpoint and the lenslet array effectively emulates a camera array. By using a large number of densely packed lenslets, one can significantly reduce the spatial aliasing artifacts. It also, however, comes at the cost of reduced resolution for each light field view. A light field camera is typically paired with an ultra-high-resolution static DSLR and is therefore not applicable to video streaming.

Instead of using either a camera array or a microlenslet array, we develop a novel hybrid stereo-lightfield solution. Our goal is to first recover a high-resolution disparity map of the scene and then synthesize a virtual light field for producing dynamic DoF effects. Despite recent advances in stereo matching, recovering high-resolution depth/disparity maps from images is still too expensive to perform in real-time. We, therefore, construct a hybrid-resolution stereo camera system by coupling a high-res/low-res camera pair. We recover a low-res disparity map and subsequently upsample it via fast cross bilateral filters. We then use the recovered high-resolution disparity map and its corresponding video frame to synthesize a light field. We implement a GPU-based disparity warping scheme and exploit atomic operations to resolve visibility. To reduce aliasing, we present an image-space filtering technique that compensates for spatial undersampling using MIPMAPPING. Finally, we generate dynamic DoF effects using light field rendering. The complete processing pipeline is shown in Figure 1.

We map all processing stages onto NVIDIA's CUDA architecture. Our system can produce DoF effects with arbitrary aperture sizes and focal depths for the resolution of 640×480 at 15 fps, as shown in the supplementary video. This indicates that if we capture the video streams at the same frame rate, we can display the refocused stream simultaneously. Our system thus provides a low-cost, computational imaging solution for runtime refocusing, an effect that is usually the domain of expensive movie cameras with servo-controlled lenses. Experiments on both indoor and outdoor scenes show that our framework can robustly handle complex, dynamic scenes and produce high quality results.

2 Hybrid-Resolution Stereo Camera

We first construct a hybrid stereo camera for recovering high-resolution disparity map in real-time. Our system uses the Pointgrey Flea2 camera pair to produce one *high-resolution color* video stream and one *low-resolution gray-scale* video stream. We synchronize frame capture to within 125μs by using the Pointgrey camera synchronization package. A unique feature of our stereo camera is the coupling of a high-resolution color camera with a low-resolution gray-scale camera. Such configurations have many advantages. First and foremost, it provides a multi-resolution stereo matching solution that can achieve real-time performance (Section 3). Second, the lower bandwidth requirement also allows our system to be implemented for less expense on a greater number of platforms. Stereo systems that stream two videos at 15 fps and 640×480 resolution can produce up to 27.6 MB of data per second. By comparison, our hybrid-resolution stereo camera only produces slightly more than half that rate of data. Although our current implementation uses Firewire cameras, the low bandwidth demands of our solution make it possible to use a less expensive and more common alternative like USB 2.0, even for streaming higher resolutions such as 1024×768. Finally, compared to off-the-shelf stereo cameras such as Pointgrey's Bumblebee, our system has several advantages in terms of image quality, cost, and flexibility. For example, the form factor of the Bumblebee forces its lenses to be small and it produces image with severe radial distortion. Our system is also less expensive ($1500 vs. $4000), and our setup allows us to dynamically adjust the camera baseline to best fit different types of scenes unlike the Bumblebee. We calibrate the stereo pair using a planar checker board pattern the algorithm outlined by Zhang [24]. It is not necessary, however, that the calibration be absolutely accurate as the disparity map is recovered from a severely downsampled image pair. Our experiments have shown that disparity map recovery using belief propagation on the low-resolution image pair is not affected by slight changes in the camera pair geometry. The intensity calibration on the camera pair is performed prior to capture via histogram equalization. The mappings for these processes are retained and applied to each incoming frame prior to stereo matching.

3 Real-time Stereo Matching

In order to efficiently generate a high-resolution disparity map from the input low-res/high-res image pairs, we implement a GPU-based stereo matching algorithm on CUDA.

3.1 CUDA Belief Propagation

Stereo matching is a long standing problem in computer vision [13]. Global methods based on belief propagation (BP) [15] and graph-cut [4,1] have been known to produce highly reliable and accurate results. These methods, however, are more expensive when compared to local optimization methods such as dynamic programming. Fortunately, BP lends itself well to parallelism on the GPU [2], where the core computations can be performed at every image pixel in parallel on the device.

We utilize the methods presented by Felzenwalb [3] to speed up our implementation without affecting the accuracy: we use a hierarchical implementation to decrease the number of iterations needed for message value convergence; we apply a checkerboard scheme to split the pixels when passing messages in order to reduce the number of necessary operations and halve the memory requirements; and we utilize a two-pass algorithm to reduce the running time to generate each message from $O(n^2)$ to $O(n)$ using the truncated linear model for data/smoothness costs.

Our CUDA BP implementation uses five separate kernels, whereas the CPU only calls the appropriate kernels and adjusts the current parameters/variables. A kernel is used to perform each of the following steps in parallel, with each thread mapping to computations at a distinct pixel:

1. Compute the data costs for each pixel at each possible disparity at the bottom level.
2. Iteratively compute the data cost for each pixel at each succeeding level by aggregating the appropriate data costs at the proceeding level.
3. For each level of the implementation:
   (a) Compute the message values at the current ‘checkerboard’ set of pixels and pass the values to the alternative set. Repeat for i iterations, alternating between the two sets.
   (b) If not at the bottom level, copy message values at each pixel to corresponding pixels of the succeeding level.
4. Compute the disparity estimate at each pixel using the data cost and current message values corresponding to each disparity.

Table 1 shows the performance of our algorithm on some of the Middlebury datasets at different resolutions. Despite the acceleration on the GPU, we find that it is necessary to use the lower resolution images (320×240 or lower) as inputs to our stereo algorithm in order to achieve real-time performance.

In our experiments described in the rest of the paper, we first smooth these low-resolution image pairs using a Gaussian filter where σ equals 1.0, then process them using our implementation with a disparity range from 0 to 35, maximum data cost and smoothness costs of 15.0 and 1.7, respectively, a data cost weight of 0.7 in relation to the smoothness cost, with 5 levels of belief propagation and 10 iterations per level. Each kernel is processed on the GPU using thread block dimensions of 32×4.

Data sets	Resolutions		
	128×96	320×240	640×480
Teddy	13ms	78 ms	446 ms
Tsukuba	8ms	55ms	357 ms
Cones	11ms	69 ms	424 ms

Table 1 Performance of our CUDA stereo matching at different resolutions. Note that the number of disparity levels is proportionally scaled to the resolution. The levels of belief propagation are all set to 5 and iterations per level are all set to 10.

3.2 Fast Cross Bilateral Upsampling

Given a low-resolution disparity map D' and a high-resolution image I, we intend to recover a high-resolution disparity map D using cross bilateral filters [23], where we apply a spatial Gaussian filter to D' and a color-space Gaussian filter to I. Assuming p and q are two pixels in I; W is the filter window size; I_p and I_q are the color of p and q in I; and q' is the corresponding pixel coordinate of q in D'. We also use

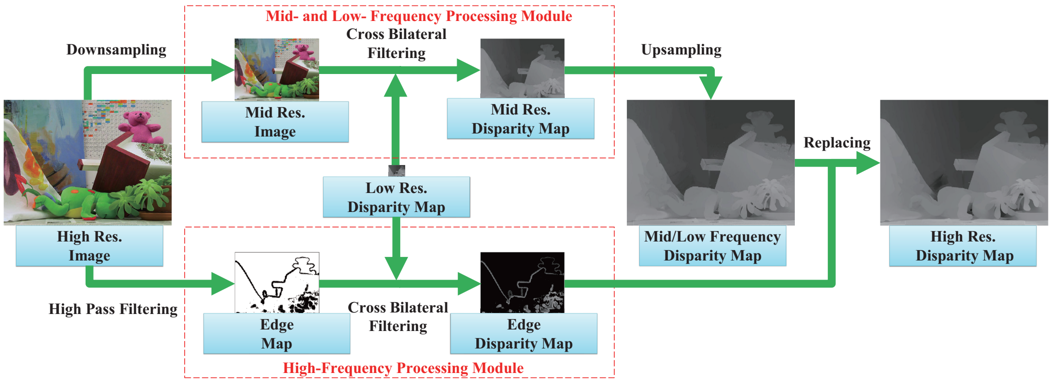

Fig. 2 Our fast cross bilateral upsampling scheme synthesizes a high-resolution disparity map from the low-resolution BP stereo matching result on CUDA.

σ_c and σ_d as constants to threshold the color difference and filter size. We compute the disparity of pixel D_p as:

$$D_p = \frac{\sum_{q \in W} G_d(p,q) G_c(p,q) D'_q}{K_p}, \tag{1}$$

where $K_p = \sum_{q \in W} G_d(p,q) G_c(p,q)$, $G_d(p,q) = exp(\frac{-||p-q||}{\sigma_d})$, and $G_c(p,q) = exp(\frac{-||I_p - I_q||}{\sigma_c})$.

The complexity of cross bilateral upsampling (CBU) is $O(NW)$ where N is the output image size and W is the filter window size. Therefore the dominating factor to the processing time is the number of pixels that need to be upsampled, i.e., the resolution of the high-res image in the brute-force implementation.

To accelerate our algorithm, we implement a fast CBU scheme that effectively reduces the pixels to be upsampled. Paris *et al.* [12] have shown that the mid and low frequency components of an image remain approximately the same when downsampled. We therefore treat the high-frequency and the mid- and low- frequency components separately. Our method first applies a Gaussian high-pass filter to identify the pixels of high frequency in I and then uses a standard cross bilateral filter to estimate the disparity values at only these pixels. We store the resulting disparity map as D_{high}. We call this step the high-frequency processing module. In parallel, we downsample the color image to mid-resolution I_{mid}, apply CBU between D' and I_{mid} to obtain the mid-res disparity map D_{mid}; and subsequently upsample D_{mid} to D_{high} using standard bilinear upsampling. We call this step the mid- and low- frequency processing module. Finally, we perform high frequency compensation by *replacing* the disparity value at the identified high frequency pixels $\tilde{I}$ with D_{high}. Figure 2 shows the complete processing pipeline of our algorithm. Compared with standard CBU, our scheme only needs to upsample a small portion of the pixels and hence is much faster.

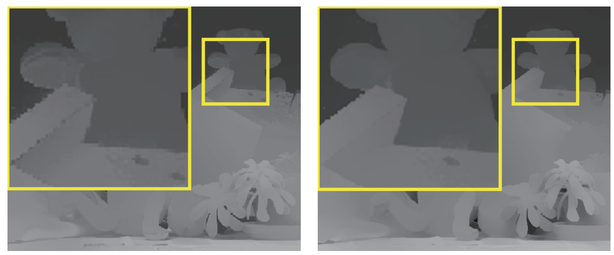

Fig. 3 Comparison of the result with(right) and without(left) high frequency compensation.

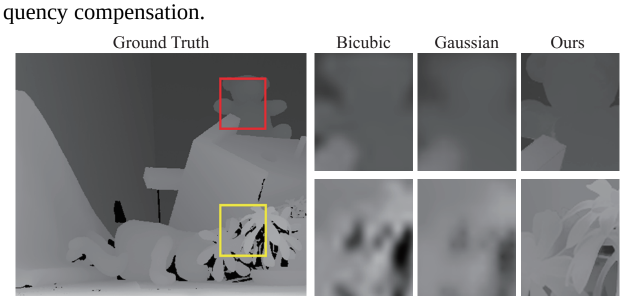

Fig. 4 Comparison of our method and other upsampling schemes on synthesize data. Both patches in the disparity map are upsampled from a resolution of 30×25 to 450×375.

3.3 CUDA Implementation.

We developed a GPU implementation of our algorithm on the CUDA architecture to tightly integrate with our CUDA BP stereo mapping algorithm. In our experiments, we found that it is most efficient to assign one thread to upsample each pixel in the disparity map. To further evaluate the throughput of our implementation, we upsampled 128×128 disparity maps with 1280×1280 color images. Our implementation achieves a processing speed of 22 ms per frame or 14 ms per megapixel with a 5×5 filter window, a significant speedup to the CPU-based scheme [5] (which was 2 seconds per megapixel).

To measure the accuracy of our scheme, we performed experiments using various stereo data sets. In Figure 3, we show using the Teddy data set that reintroducing high frequency compensation produces sharper edges and smoother

Fig. 5 Comparison between our method and bicubic upsampling on real scenes. The disparity map is upsampled from 320×240 to 640×480. Our method preserves sharp edges and maintains smoothness, which is critical to reliable DoF synthesis.

surfaces. Figure 4 illustrates our results in three regions on the Teddy data set. They are upsampled from 30×25 to 450×375. Compared with standard bicubic or Gaussian upsampling, our method preserves fine details near the edges. It is important to note that preserving edges while removing noise in the disparity map is crucial to our DoF synthesis as DoF effects are most apparent near the occlusion boundaries. To further measure the accuracy, we compared our estimation with the ground truth by computing the mean squared errors over all pixels. Table 2 compares the error incurred by our method under different upsampling scales on a variety of Middlebury stereo data sets, and the results show that our method is reliable and accurate even with very high upsampling scales.

4 Real Time DoF Synthesis

Once we obtain the high-resolution disparity map, we set out to synthesize dynamic DoF effects. Previous single image based DoF synthesis algorithms attempt to estimate the circle of confusion at every pixel and then apply the spatially varying blurs on the image. These methods produce strong bleeding artifacts at the occlusion boundaries, as shown in Figure 7. In computer graphics, the distributed ray tracing and the accumulation buffer techniques have long served as the rendering method for dynamic DoF. Both approaches are computationally expensive as they either require tracing out a large number of rays or repeated rasterization of the scene.

Data sets	Upsampling Scales			
	20×20	10×10	5×5	2×2
Teddy	31.14	10.96	5.24	3.92
Plastic	20.79	11.56	6.50	7.29
Monopoly	30.91	9.73	4.71	3.45
Books	26.01	15.60	9.12	6.25
Baby2	11.09	5.11	3.35	2.02
Aloe	73.96	23.14	11.36	4.49
Cones	49.42	33.64	10.82	4.24
Art	67.40	24.70	8.70	4.67

Table 2 Mean squared errors of our upsampled disparity maps with different upsampling factors on the Middlebury data sets.

Furthermore, to apply ray-tracing or accumulation buffer in our application requires constructing a triangulation of the scene from the depth map, which would incur additional computational cost. Our solution is to dynamically generate a light field from the high-resolution video stream and its depth stream and then render DoF effects from the light field, as shown in Figure 6.

4.1 The Lens Light Field

Light fields provide an image-based representation that uses a set of rays to describe a scene in place of explicit geometry. A light field is commonly stored as a 2D array of images. Each pixel in the image can be indexed as an integer 4-tuple (s, t, u, v), where (s, t) is the image index in the array and (u, v) is the pixel index in the image.

In this paper, we construct a lens light field using the camera-pixel $st-uv$ parametrization. We assume that the lens plane is the st plane and we put an array of virtual light field cameras on the st plane. We further denote the sensor plane as the xy plane and the optical axis of the lens as the z direction.

To synthesize the light field, we use the high-resolution camera in our stereo pair as the reference camera R_{00} (i.e., $(s,t) = (0,0)$). We can then easily find all rays that pass through a 3D point A in terms of its disparity γ from the reference view. Assuming A's image is at pixel (u_0, v_0) in the reference camera, we can compute its image (pixel coordinate) in any light field camera R_{st} as:

$$(u,v) = (u_0, v_0) + (s,t) \cdot \gamma \qquad (2)$$

We use $L_{out}(s,t,u,v)$ to represent the out-of-lens light field and $L_{in}(x,y,s,t)$ to represent the in-camera light field. The image formed by a thin lens is proportional to the irradiance at a pixel a [14], which can be computed as a weighted integral of the incoming radiance through the lens:

$$a(x,y) \approx \sum_{(s,t)} L_{in}(x,y,s,t) cos^4 \phi \qquad (3)$$

To map the in-lens light field to the out-of-lens light field, it is easy to verify that pixel $a(x,y)$ on the sensor maps to pixel $(u_0,v_0) = (w-x, h-y)$ in R_{00}. Therefore, if we want to focus at the scene depth whose corresponding disparity is γ_f, we can find the pixel index in camera R_{st} using Equation 2. The irradiance at a can be approximated as:

$$a(x,y) = \sum_{(s,t)} L_{out}(s,t,u_0 + s \cdot \gamma_f, v_0 + t \cdot \gamma_f) \cdot cos^4 \phi$$

To estimate the attenuation $\cos^4 \phi$ term, we can directly compute $\cos^4 \phi$ for each ray (s,t,u,v). Notice that the ray has direction $(s,t,1)$. Therefore, we can compute $\cos^4 \phi = \frac{1}{(s^2+t^2+1)^2}$.

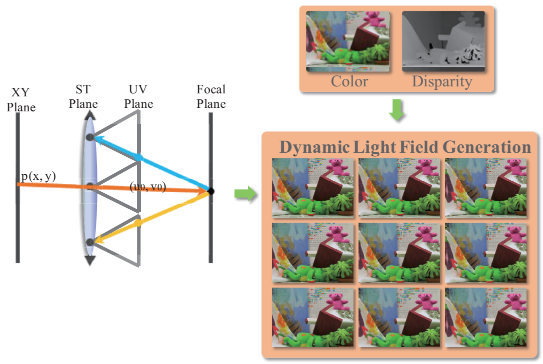

Fig. 6 We synthesize an in-lens light field (left) from the recovered high-resolution color image and disparity map (right).

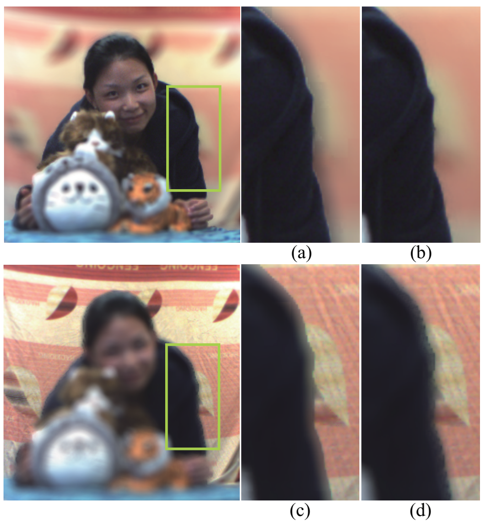

Fig. 7 Comparing results generated by image space blurring (a,c) and our light field synthesis method (b, d). Our approach effectively reduces both the intensity leakage (a) and boundary discontinuity (c) artifacts.

4.2 CUDA Implementation

To synthesize the light field from the reference camera R_{00} and its disparity map, we warp it onto the rest light field cameras using Equation 2. A naive approach would be to directly warp the RGB color of each pixel $a(u_0, v_0)$ in R_{00} onto other light field cameras. Specifically, using a's disparity value, we can directly compute its target pixel coordinate in camera R_{st} using Equation 2. Since the CUDA architecture supports parallel write, we can simultaneously warp all pixels in R_{00} onto other light field cameras. This simple approach, however, results in incorrect visibility: multiple pixels in R_{00} may warp to the same pixel a' in the light field camera R_{st}. While a' should pick the nearest one (with the largest disparity), the brute-force approach would simply use the last value written to the pixel. To resolve the visibility problem, we choose to warp the disparity value instead of the pixel color, and use CUDA's atomic max operation to resolve parallel write conflict.

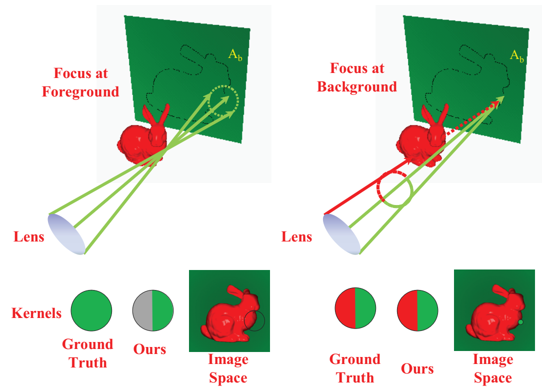

Fig. 8 Illustrations of two types of boundary artifacts. See Section 4.3 for details.

Although our warping algorithm is efficient, it also introduces two main artifacts. Due to speed requirements and limited texture memory sizes, we can only render a small light field with 36 to 48 cameras at a 640×480 image resolution. The low spatial resolution leads to strong aliasing artifacts due to undersampling. Second, since our reference view does not contain information from the occluded regions, the warped light field camera images will contain holes.

To reduce the image artifacts caused by undersampling and occlusions, we develop a simple technique similar to the cone tracing method to pre-filter the reference view [6]. Our method is based on the observation that out-of-focus regions exhibit most severe aliasing artifacts and occlusion artifacts since they blend rays corresponding to different 3D points. Our method compensates for undersampling by first blurring the out-of-focus rays and then blending them. A similar concept has been used in the Fourier slicing photography technique for generating a band-limited light field [10].

To simulate low-pass filtering in light field rendering, we first generate a Mipmap from the reference image using a 3×3 Gaussian kernel [7]. We then integrate the Gaussian Mipmap into the light field ray querying process.

4.3 Our Technique vs. Single-Image Blurring

Compared with single-image methods that apply spatially varying blurs, our light field based DoF synthesis technique significantly reduces two types of boundary artifacts. In instances where the camera focuses at the foreground, the ground truth result should blend points on the background. Conversely, single-image filtering techniques use a large kernel to blend the foreground and background pixels and hence, produce the *intensity leakage* artifact. Consider a point A_b lying on the background near the boundary, as shown in Figure 8. Our method attempts to blend rays originating from the background. Due to occlusions, it can only access a portion of them. Even if the background has consistent color or texture, our technique still produces reasonable approximations.

In instances where the camera focuses on the background, the ground truth result should blend both the foreground and background points. Single-image filtering techniques, however, would consider A_b in focus and hence directly use its color as the pixel's color. In this case, the transition from the foreground to the background appears abrupt, causing the *boundary discontinuity* artifacts. Consider a point A_b on the background near the occlusion boundary in the image as shown in Figure 8. Since rays originating from both the foreground and background are captured by our synthesized light field, our technique will produce the correct result.

Figure 7 compares the rendering results using our method and the single-image filtering approach on an indoor scene. Our technique exhibits fewer visual artifacts compared to the single-image filtering method, especially near the boundary of the girl. When examining the boundary of the sweater, the single-image method blurs the black sweater regions into the background and thus causes color bleeding, whereas our technique prevents such leakage. When focusing at the background, the single-image method exhibits discontinuous transitions from the girl to the background while our method preserves the smooth transition.

5 Results and Discussions

Our hybrid-resolution stereo system is connected to a work station through a single PCI-E Firewire card. The workstation is equipped with a 3.2GHz Intel Core i7 970 CPU, 4GB memory and an NVIDIA Geforce GTX 480 Graphic Card with 1.5GB memory. We implement all three processing modules (the disparity map estimation, fast CBU, dynamic DoF rendering) using NVIDIA's CUDA 3.1 with compute capability 2.0. Our system runs at the resolution of 640×480 with 15 fps.

We have conducted extensive experiments on both indoor and outdoor scenes. Some of our video results (live capture) can be found in the supplementary video. We expect to submit a demo to CVPR '11 to demonstrate our system. A crucial step in our real-time stereo matching module is choosing the proper parameters (e.g., the weight for the smooth/compatible terms of the energy function) to fit different types of scenes (indoor vs. outdoor). We have developed an interface to dynamically change the parameters, allowing us to see the results and fine tune the parameters, as shown in the supplementary video.

We first demonstrate our system on indoor scenes with controlled lighting. Figure 9 row 1 shows four frames captured by our system of a girl drinking coffee while reading. The coffee cup in the scene is textureless and very smooth. Our fast CBU scheme, however, still preserves the disparity edges, as shown in Figure 9 row 1. We then dynamically change the depth of the focal plane: column (a) and column (d) focus on the front of the table, column (b) focuses on the girl, and column (c) focuses on the background. Notice how the blur varies and the in-focus regions fade into the out-of-focus regions.

Figure 9 row 2 displays a scene of a girl moving a toy car on a table. The surface of the car is specular, and the background and car have similar colors, making it challenging to prevent the disparity of the background from merging with the disparity of the car. Moreover, the motion of the car is towards the camera, causing the body of the car to have several different disparities. This makes labeling each pixel using stereo correspondence methods even more difficult. Nevertheless, our algorithm preserves the edges of the car when it is in focus and correctly blurs portions of the scene outside of the focal plane. Our system performs well indoors because the background distance is often limited, therefore allowing one baseline to produce accurate disparity labels for the entire scene. In addition, artificially lit indoor scenes with diffuse walls and surfaces tend to have moderate dynamic range and have few poorly lit or saturated regions.

Indoor scenes undoubtedly aid the performance of our system. Our experiments on outdoor scenes, however, show promising results as well. Row 3 of Figure 9 shows an outdoor sequence with a distant background under dynamically varying lighting conditions. Notice that in column (a), the image is brighter than the rest of the frames in the sequence and the background contains noticeable shadows. In addition to incoherent illumination, large portions of the scene such as the sky and the ground are textureless, making it difficult to achieve robust stereo matching. Since our system allows us to dynamically change the camera baseline, we use its real-time feedback to tune the parameters and increase the camera baseline to obtain a satisfactory disparity map, as shown in the supplementary video. The use of large baseline may lead to holes near the occlusion boundaries on full-resolution images. These holes are, however, less significant in low-resolution stereo pairs and our upsampling scheme is able to borrow information from the color image to fill in the holes. The extracted frames show that we are able to correctly change the focus between the moving targets in both foreground and background.

6 Conclusion and Future Work

We have presented an affordable stereo solution for producing high quality live DoF effects. Our system shows promising results on indoor and outdoor scenes although it still has several limitations. First, 15 fps is a low frame rate and our resolution of 640×480 precludes our system from immediately being used in high quality HD video applications. Using multiple GPUs may address this problem as they allow

Fig. 9 Screen captures of live video streams produced by our system on both indoor (top two rows) and outdoor (bottom row) scenes.

greater exploitation of inherent parallelism in our computational pipeline. Second, although the high quality sensor and lens system on our camera pair significantly reduces image noise and optical distortions, this comes with a higher price. While less expensive than existing commercial movie cameras, our system is still twice the cost of most base level video cameras. Integrating existing real-time techniques to correct optical distortions and sensor noise into our pipeline would make it feasible to use lower cost webcams instead of the firewire Flea cameras.

Our future efforts include adapting our system to functional applications such as privacy protected surveillance. We plan to demonstrate the usefulness of our system in urban spaces to limit the focal plane to public areas, e.g., the sidewalks, while blurring more distant private areas like the interior of homes. Current urban surveillance networks are augmented with real-time recognition algorithms to detect illegal activity. When illegal activity is detected, our system could provide more information to law enforcement by removing the DoF effect using the stored disparity map stream for subsequent scene reconstruction. We can also leverage future gains in ubiquitous computing to produce a truly mobile platform which utilizes, for example, two camera phones for producing DSLR quality imagery.

References

1. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. PAMI, IEEE Transactions on **23**(11), 1222 –1239 (2001)
2. Brunton, A., Shu, C., Roth, G.: Belief propagation on the gpu for stereo vision. In: Computer and Robot Vision, The 3rd Canadian Conference on (2006)
3. Felzenszwalb, P., Huttenlocher, D.: Efficient belief propagation for early vision. In: CVPR (2004)
4. Kolmogorov, V., Zabin, R.: What energy functions can be minimized via graph cuts? PAMI, IEEE Transactions on **26**(2), 147 –159 (2004)
5. Kopf, J., Cohen, M.F., Lischinski, D., Uyttendaele, M.: Joint bilateral upsampling. In: SIGGRAPH (2007)
6. Lee, S., Eisemann, E., Seidel, H.P.: Depth-of-field rendering with multiview synthesis. In: SIGGRAPH Asia (2009)
7. Lee, S., Kim, G.J., Choi, S.: Real-time depth-of-field rendering using anisotropically filtered mipmap interpolation. IEEE Transactions on Visualization and Computer Graphics **15**(3), 453–464 (2009)
8. Levin, A., Hasinoff, S.W., Green, P., Durand, F., Freeman, W.T.: 4d frequency analysis of computational cameras for depth of field extension. ACM Trans. Graph. **28** (2009)
9. Mcmillan, L., Yang, J.C., Yang, J.C.: A light field camera for image based rendering (2000)
10. Ng, R.: Fourier slice photography. In: SIGGRAPH (2005)
11. Ng, R., Levoy, M., Brdif, M., Duval, G., Horowitz, M., Hanrahan, P.: Stanford tech report ctsr 2005-02 light field photography with a hand-held plenoptic camera
12. Paris, S., Durand, F.: A fast approximation of the bilateral filter using a signal processing approach. Int. J. Comput. Vision **81**(1), 24–52 (2009)
13. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vision **47**, 7–42 (2002)
14. Stroebel, L., Compton, J., Current, I., Zakia, R.: Photographic Materials and Processes (1986)
15. Sun, J., Zheng, N.N., Shum, H.Y.: Stereo matching using belief propagation. PAMI, IEEE Transactions on **25**(7), 787 – 800 (2003)
16. Vaish, V., Levoy, M., Szeliski, R., Zitnick, C., Kang, S.B.: Reconstructing occluded surfaces using synthetic apertures: Stereo, focus and robust measures. In: CVPR (2006)
17. Vaish, V., Wilburn, B., Joshi, N., Levoy, M.: Using plane + parallax for calibrating dense camera arrays. In: CVPR (2004)
18. Wang, H., Sun, M., Yang, R.: Space-time light field rendering. IEEE Transactions on Visualization and Computer Graphics **13**, 697–710 (2007)

19. Wang, H., Yang, R.: Towards space: time light field rendering. In: I3D (2005)
20. Wilburn, B., Joshi, N., Vaish, V., Levoy, M., Horowitz, M.: High-speed videography using a dense camera array. In: CVPR (2004)
21. Wilburn, B., Joshi, N., Vaish, V., Talvala, E.V., Antunez, E., Barth, A., Adams, A., Horowitz, M., Levoy, M.: High performance imaging using large camera arrays. ACM Trans. Graph. **24**, 765–776 (2005)
22. Yang, J.C., Everett, M., Buehler, C., McMillan, L.: A real-time distributed light field camera. In: Proceedings of the 13th Eurographics workshop on Rendering, EGRW '02, pp. 77–86 (2002)
23. Yang, Q., Yang, R., Davis, J., Nister, D.: Spatial-depth super resolution for range images. In: CVPR (2007)
24. Zhang, Z.: A flexible new technique for camera calibration. PAMI, IEEE Transactions on **22**(11), 1330 – 1334 (2000)