A Noise-Aware Filter for Real-Time Depth Upsampling

Derek Chan, Hylke Buisman, Christian Theobalt, Sebastian Thrun

A Noise-Aware Filter for Real-Time Depth Upsampling

Derek Chan Hylke Buisman Christian Theobalt Sebastian Thrun

Stanford University, USA

Abstract. A new generation of active 3D range sensors, such as time-of-flight cameras, enables recording of full-frame depth maps at video frame rate. Unfortunately, the captured data are typically starkly contaminated by noise and the sensors feature only a rather limited image resolution. We therefore present a pipeline to enhance the quality and increase the spatial resolution of range data in real-time by upsampling the range information with the data from a high resolution video camera. Our algorithm is an adaptive multi-lateral upsampling filter that takes into account the inherent noisy nature of real-time depth data. Thus, we can greatly improve reconstruction quality, boost the resolution of the data to that of the video sensor, and prevent unwanted artifacts like texture copy into geometry. Our technique has been crafted to achieve improvement in depth map quality while maintaining high computational efficiency for a real-time application. By implementing our approach on the GPU, the creation of a real-time 3D camera with video camera resolution is feasible.

1 Introduction

In recent years, a new type of camera has been developed that measures the time-of-flight (TOF) of infrared light to reconstruct 3D scene geometry at real-time frame rates, (e.g. [7]) largely independently of scene texture. Unfortunately, being a relatively young technology, time-of-flight imaging has not enjoyed the same advances with respect to image resolution, speed, and image quality, that have been made in traditional 2D intensity imaging. In consequence, current TOF sensors provide depth readings of comparably low image resolution (e.g. 176×144 pixels for the MESA Swissranger^{TM} SR3100) that are heavily contaminated with noise in the distance readings.

The ability to rapidly capture good quality, high-resolution depth images of dynamic scenes even in environments that are not suitable for image-based stereo (e.g. due to lack of texture), enables a variety of intriguing applications. Examples are environment perception in robotics, dynamic scene reconstruction for 3D video, marker-less motion capture and human computer interaction, to name only a few.

In this paper, we therefore propose a new method to capture depth data at high-resolution, at a minimal noise level, and at near real-time frame rates. To achieve this goal, we simultaneously record a dynamic scene with a high-resolution video camera and a frame-synchronized low-resolution depth sensor. Our method upsamples the low-resolution data of the depth sensor to the high resolution of the video camera. To serve this purpose, it enforces statistical relations between the high-resolution video images and the low-resolution depth maps, such as the collocation of intensity and depth edges,

as well as the assumption of smooth geometry in uniformly colored regions. In order to fulfill the real-time processing requirements, we pose our algorithm as a real-time filtering method. Our approach capitalizes on and extends the concept of the image-based bilateral filter, an edge-preserving image filter [11]. The bilateral filter adaptively shapes its spatial smoothing kernel by weighting it with a local color similarity term, thereby preventing blurring across intensity gradients.

In previous work, Kopf et al. [5] proposed a method called joint bilateral upsampling (JBU) to lift a low resolution tone-mapping result to the high resolution of a mega-pixel intensity image. This idea of splitting the filter domain into one intensity part and one depth part is also a core concept of our depth data upsampling. Unfortunately, the use of a color image to guide the up-sampling of a depth map admittedly runs the risk of inadvertently copying texture from the color image into actually smooth areas of the captured depth, Sect. 3. This effect is particularly noticeable if depth sensor readings contain a high amount of random noise in the first place.

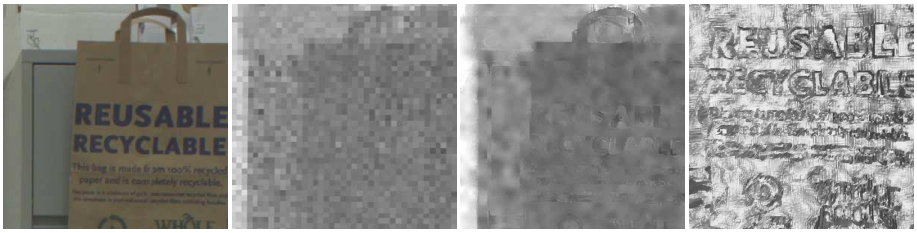
We therefore extend the original bilateral upsampling approach and propose a new noise-aware filter, termed NAFDU (**N**oise-**A**ware **F**ilter for **D**epth **U**psampling), that adaptively dampens the influence of color similarity in depth regions which are likely to be geometrically smooth but heavily noise affected, Sect. 4. This way, the unwanted effect of texture copying is almost entirely eliminated, which makes our method particularly suitable to be applied with currently available TOF imaging devices. Our algorithm can be effectively and efficiently implemented on state-of-the art graphics processing units (GPUs) to yield high-quality high-resolution depth readings in real-time. We have tested our approach using a prototype system comprising of one SR3100 time-of-flight depth camera and one high-resolution video camera, Sect. 5. Our results show that our noise-aware filtering concept enables faithful reconstruction of detailed 3D scene geometry at a high level of detail and with less unwanted artifacts than previous methods from the literature would deliver, Sect. 6.

2 Related Work

Depending on the application in mind, different strategies can be used to enhance the image resolution of noisy range sensors, such as time-of-flight cameras [7]. In case the recorded scene is static, several measurements of the scene can be averaged or the integration time of the sensor can be increased to reduce the uncertainty in distance readings. For static scenes it is also feasible to combine depth readings from several slightly displaced viewpoints, and apply a superresolution scheme to the depth maps, which yields 3D data of higher fidelity [10, 9].

If real-time acquisition of dynamic scenes is the goal, however, all previously mentioned methods are not applicable. One way to meet the demanding performance requirements of real-time depth upsampling is to fuse the data of a high resolution video camera and a low-resolution depth sensor. A common upsampling recipe enforces heuristics between depth and intensity images, such as the collocation of depth and intensity gradients, and the smoothness of geometry in areas of similar color.

An approach that put this idea into practice was proposed by Diebel et al. [3]. They fuse depth maps of a laser range finder with intensity maps of a color camera by defining



(a) (b) (c) (d)

Fig. 1. A high resolution color image (a) and low resolution depth map (b) upsampled using conventional JBU (c). Note the copying of the intensity texture into geometry. The lettering of the words 'reusable' and 'recyclable' can be visually seen in the depth map upsampled when noise is not taken into account. The visible text in the depth is even more apparent when the range information is rendered as 3D geometry (d) (zoomed in for clarity).

the posterior probability of the high-resolution reconstruction as a Markov Random Field (MRF) and optimizing for the maximum-a-posteriori (MAP) solution.

Following similar heuristics, Kopf et al. proposed an alternative fusion method called joint bilateral upsampling [5]. Their algorithm uses a modification of the bilateral filter, an edge-preserving smoothing filter for intensity images [11]. The bilateral filter locally shapes the spatial smoothing kernel by multiplying with a color similarity term, a.k.a the range term, which yields an edge-preserving smoothing filter. Kopf et al. capitalize on this adaptive smoothing capability and bilaterally upsample a low-resolution tone mapping result such that it matches the resolution of a multi-megapixel intensity image. Recently, Crabb et al. applied bilateral upsampling to range data captured with a time-of-flight camera in order to do real-time matting [2].

At this point we would like to mention that researchers also investigated new bilateral filter decompositions and approximations to improve runtime performance [12, 8]. Although it can be assumed that variants of such techniques would also allow for further speedups of our new filter, we have currently not explored this direction further, since already with the current design a GPU implementation allows for real-time performance.

Yang et al. [13] also propose an upsampling technique based upon a bilateral filter. However, they do not use a joint bilateral technique to link the two images. Rather they use the high resolution image to create a cost volume to which they apply a standard bilateral filter. This requires them to run a 2D bilateral kernel over multiple slices of a volume. Even with GPU optimizations that we propose in this paper, their effective runtime would be our runtime multiplied by the number of cost slices, and thus orders of magnitude higher than for our approach.

All methods mentioned so far fail to address the problem that in the presence of severe noise in the depth readings, the fundamental heuristic assumptions about the relation between depth and intensity may lead to erroneous copying of texture into actually smooth geometry. In this paper, we therefore propose a new adaptive filter that prevents these artifacts, yet enables real-time high-quality upsampling when implemented on the GPU.

3 The Bilateral Filter and Standard Bilateral Upsampling

As background for the concepts developed later, we begin with a description of the bilateral filter, an edge-preserving smoothing filter for intensity images which was first described in [11]. The bilateral filter adaptively shapes a spatial smoothing kernel by multiplying it with a second term, typically referred to as the range term. The range term measures the photometric similarity of the filtered pixel to each pixel from the local neighborhood. The effect of this two-domain analysis is that in smoothly colored regions standard spatial smoothing is applied, while in the vicinity of color gradients unwanted smoothing across the edge is prevented. A bilateral filter can be expressed as follows:

$$J_p = \frac{1}{k_p} \sum_{q \in \Omega} I_q f(||p - q||) g(||I_p - I_q||) \ ,$$

where I_p is the intensity of pixel p in input image I, Ω is a spatial neighborhood around p, J_p is the value of pixel p in the filtered image J, and k_p is a normalizing factor. In this description we recognize first a data term $f(||p-q||)$ and then a range term $g(||I_p-I_q||)$. Generally, f and g are taken to be Gaussian functions with standard deviations σ_s and σ_r.

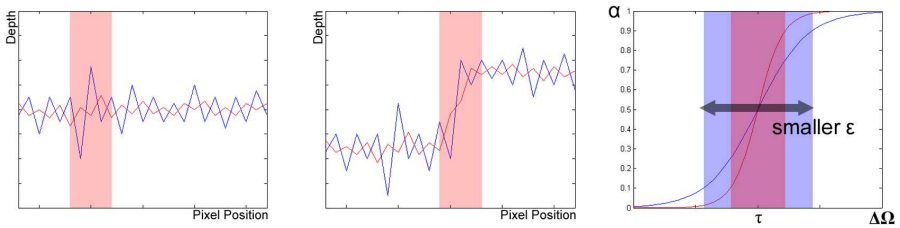
In [5], Kopf et al. propose joint bilateral upsampling, a variation of this filter which can be used to upsample a low resolution image I to the resolution of a high-resolution reference intensity image $\tilde{I}$. Instead of evaluating both components of the filter on I, the high resolution reference image is used in the range term, which enables edge-preserving upsampling of I to the resolution of $\tilde{I}$. To achieve this goal, the upsampled image $\tilde{S}_p$ is computed as follows:

$$\tilde{S}_p = \frac{1}{k_p} \sum_{q_\downarrow \in \Omega} I_{q_\downarrow} f(||p_\downarrow - q_\downarrow||) g(||\tilde{I}_p - \tilde{I}_q||) \ .$$

Here $p_\downarrow$ and $q_\downarrow$ are pixels in the low-resolution image and p and q the corresponding pixels in the high-resolution image.

In their original work Kopf et al. used this method to upsample low-resolution tone-mapping results to mega-pixel resolution. The technique can also apply the JBU concept to upsample depth maps captured with a range sensor, as further investigated by [2]. In this case the reference image $\tilde{I}$ is a high resolution color image, and I is the low-resolution depth map.

Unfortunately, our experiments showed that direct application of JBU for depth upsampling frequently exhibits artifacts in reconstructed geometry that can be attributed to erroneous assumptions about the correlation of color and depth data, Figs. 1 and 4. It is important to note that when JBU is applied to depth data up-sampling, the assumption is made that when neighboring pixels in the reference image have similar color, they also have a similar depth. Clearly, this heuristic assumption is not always valid: two objects with similar color can be at different depths in the scene. In many cases, though, this assumption is actually valid. However, in situations where it does not hold, we can observe two different types of artifacts:



(a) Case II (b) Case I (c)

Fig. 2. (a),(b) Cross-sections of depth maps (conceptual illustration). (a) In regions of low depth gradient, noise contamination of the raw data (blue) in the pink highlighted region can cause the difference between the max and min depth values to be high; however, the difference in the blurred depth input (red) remains small. (b) When an actual depth discontinuity occurs such as in the pink highlighted region, the difference between the max and min depth values is noticeable in the both the noisy raw data (blue) and the blurred depth (red). (c) We blend case I and case II of our filter using a sigmoid function. The width of the transition region can be increased by descreasing epsilon as shown from the red line to the blue line.

One type of artifact that may arise is *edge blurring* in the depth map. This generally occurs when the two objects on either side of a depth discontinuity have similar color. A second type of artifact is *texture copying*. Here, textures from the reference image are transferred into geometric patterns that appear in the upsampled depth maps, and thus the reconstructed geometry, Figs. 1 and 4. Texture copying either occurs near depth edges where some of the texture of one object has a similar color to (the texture of) the second object, or on actually smooth surfaces where the depth map is noisy, but the reference image has a distinct texture. Note that these assumptions and their related artifacts are also common with other aforementioned techniques, such as [13] and [3], which also use a color image to up-sample depth maps, see for instance Figs. 3 and 5 for examples of texture copy in MRF upsampling.

Especially, texture copying frequently occurs when bilaterally upsampling noisy time-of-flight depth maps. We therefore propose in the next section a new noise-aware filter that prevents these artifacts and still allows for real-time upsampling.

4 NAFDU: A New Filter for Noise Aware Depth Upsampling

To address the specific requirements of depth superresolution and denoising for noisy real-time 3D sensors, like time-of-flight cameras, we propose a new multi-lateral noise-aware filter for depth upsampling (NAFDU). It is our goal to preserve the beneficial properties of bilateral upsampling in those areas where the data match our assumptions, and to prevent artifacts in those areas where standard bilateral upsampling is likely to cause erroneous texture copy. The NAFDU filter takes the following form:

$$\tilde{S}_p = \frac{1}{k'_p} \sum_{q_\downarrow \in \Omega} I_{q_\downarrow} f(||p_\downarrow - q_\downarrow||)[\alpha(\Delta_\Omega) g(||\tilde{I}_p - \tilde{I}_q||) + (1 - \alpha(\Delta_\Omega)) h(||I_{p_\downarrow} - I_{q_\downarrow}||)]$$

where f, g and h are all Gaussian functions, and $\alpha(\Delta_\Omega)$ is a blending function. A high weight α makes our filter behave like a standard bilateral upsampling filter, henceforth

referred to as *case I* of the filter. A low weight α gives higher influence to the modified range term $h(||I_{p_\downarrow} - I_{q_\downarrow}||)$, henceforth named *case II* of the filter. The latter range term makes the filter behave like a bilateral filter whose both domains are the same gray-level depth maps, i.e. it behaves like an edge preserving filter that smooths the 3D geometry independently of information from the color intensity images.

The crux is to decide when the filter ought to resort to case I and case II respectively. In a nutshell we want the filter to switch to case II only if areas are processed that are actually geometrically smooth but heavily contaminated with random noise in the distance measurement which would lead to texture copy. To serve this purpose, we define our blending function $\alpha(\Delta_\Omega)$ in the interval $[0, 1]$ as follows:

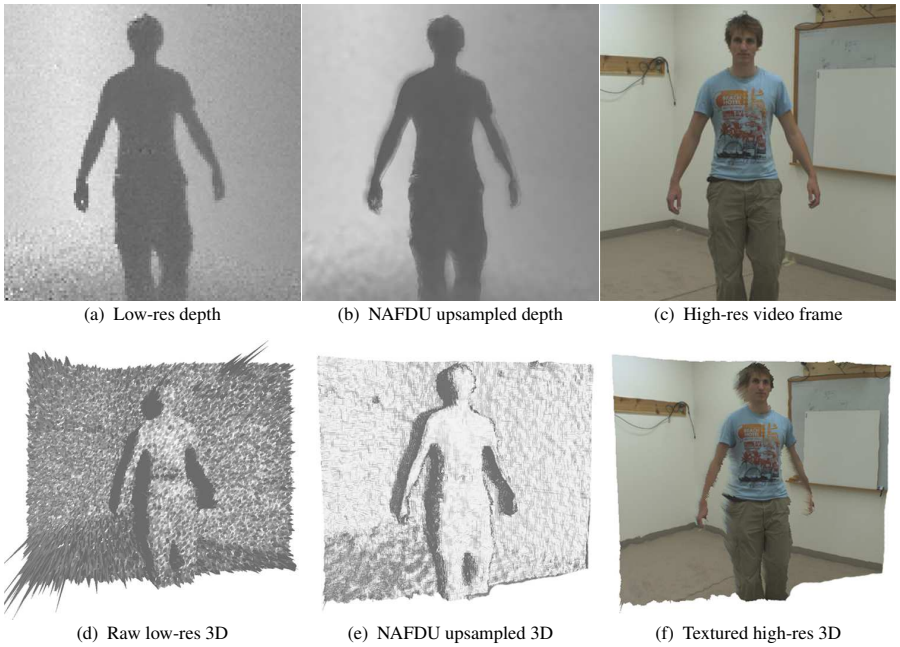
$$\alpha(\Delta_\Omega) = \frac{1}{1 + e^{-\epsilon \cdot (\Delta_\Omega - \tau)}} \ .$$

Here, Δ_Ω is the difference between the maximum and minimum gray-scale value (i.e. measured depth value) in pixel neighborhood Ω. This difference is evaluated on a low-pass filtered version of the raw depth map (3×3 Gaussian filter kernel). We reason that if this difference is below a certain range, the local surface patch is most likely to be smooth and only noise-affected - thus case II should be triggered, Fig. 2(a). If the min-max difference lies beyond that range, however, we are likely to look at a true geometric detail and case I is the appropriate choice, Fig. 2(b). Fig. 2(c) shows a plot of the weighting function α. The parameter ϵ controls how wide the transition area (in terms of the min-max difference) between case I and case II is. Parameter τ controls at what min-max difference the blending interval shall be centered. Appropriate values for ϵ and τ depend on the characteristics of the employed depth sensor and can be found through experiments. The design of α allows for a gradual blending between case I and case II which prevents artifacts in transition areas.

Please note that the computation of Δ_Ω on a low-pass filtered depth map is important as it enables us to reliably disambiguate between isolated random noise peaks and actual depth edges, as shown in Fig. 2(a),(b). We would also like to note that it is important that the range term h in case II takes the described form (a Gaussian) and cannot simply be set to a box filter in the range domain. With this design we achieve much better preservation of depth discontinuities if α lies in the transition zone. We would also like to mention that by choosing a small spatial support for our NAFDU filter (3×3 or 5×5), any form of texture copy around true depth edges can be minimized in practice while high-quality denoising is still feasible.

5 Implementation

The method described requires both 3D range maps and high resolution color images of the same scene that are pixel aligned. To obtain raw data, we built a test-system comprising of a Mesa Swissranger™ SR3100 time-of-flight camera and a Point Grey™ Flea2 video camera. The two cameras are placed side-by-side (as closely as possible) and are frame-synchronized. The Flea2 video camera provides 1024×768 pixels of image resolution, and the Swissranger produces depth maps at a granularity of 176×144 pixels. Jointly, the test system can record scenes at up to 25 fps. To ensure that capturing is



(a) Low-res depth (b) NAFDU upsampled depth (c) High-res video frame

(d) Raw low-res 3D (e) NAFDU upsampled 3D (f) Textured high-res 3D

Fig. 3. Our technique is applied to depth maps (top) which can be reconstructed as 3D geometry (bottom). Taking a low resolution depth image (a), we are able to upsample to a high resolution depth map (b). We can then apply the texture from the color image (c) to the reconstructed 3D geometry (f). The resolution, quality and detail level of the high-res geometry (e) is much higher than of the raw low-res depth (d).

occurring as fast as possible, the camera devices are accessed in a multi-threaded setup. Temporal synchronization is implemented in a software solution based upon the clock capture mechanism of the Flea2 camera.

To align the depth and video images we resort to the gray-scale IR intensity images that the depth camera provides in addition to depth maps. Given this, we can apply any of a number of standard multi-view camera calibration techniques or simple homographic warping [4]. In our experiments, we resort to the latter alternative.

The actual upsampling filter is implemented on a state-of-the-art graphics card. Since filter operations on individual pixels can be carried out independently, we can capitalize on the massive stream processing power of modern GPUs which by now feature up to 256 individual stream processors. In particular, we employ Nvidia's CUDA programming framework [6] to port our algorithm onto graphics hardware. Overall, we thereby achieve real-time up-sampling and denoising performance, see Sect. 6.

6 Results

As shown in Figs. 3, 4 and 5, our method successfully upsamples the low resolution depth maps to higher resolution. True geometry detail in the data, such as sharp depth

discontinuities, are preserved and enhanced, the random noise level is greatly reduced, and corroborating artifacts typical for previous fusion methods from the literature, in particular texture copying, are prevented. By exploiting the GPU as a fast stream processor, real-time noise aware upsampling is feasible. Overall, our NAFDU filter design successfully handles the data produced by state-of-the-art time-of-flight cameras which exhibit significantly higher random noise levels than most active scanning devices for static scenes.

To demonstrate the effectiveness of our approach we applied our technique to various scenes including our own recorded sequences as well as scenes from the Middlebury stereo benchmark data set (http://vision.middlebury.edu/stereo/data/). In the following, we analyze three main aspects of our method more closely. We first demonstrate the visual superiority of our results to results obtained with previous fusion-based upsampling methods, Sect. 6.1. Thereafter, we discuss the advantageous run time properties of our algorithm, and discuss practical performance gains in comparison to optimization-based upsampling methods, Sect. 6.2. Finally, we provide a quantitative comparison of the accuracy of our outputs to those from MRF-based upsampling [3] using data from the Middlebury data set, Sect. 6.3 We end the section by noting some overall findings from our results and by discussing some possible future directions of investigation for our work, Sect. 6.4.

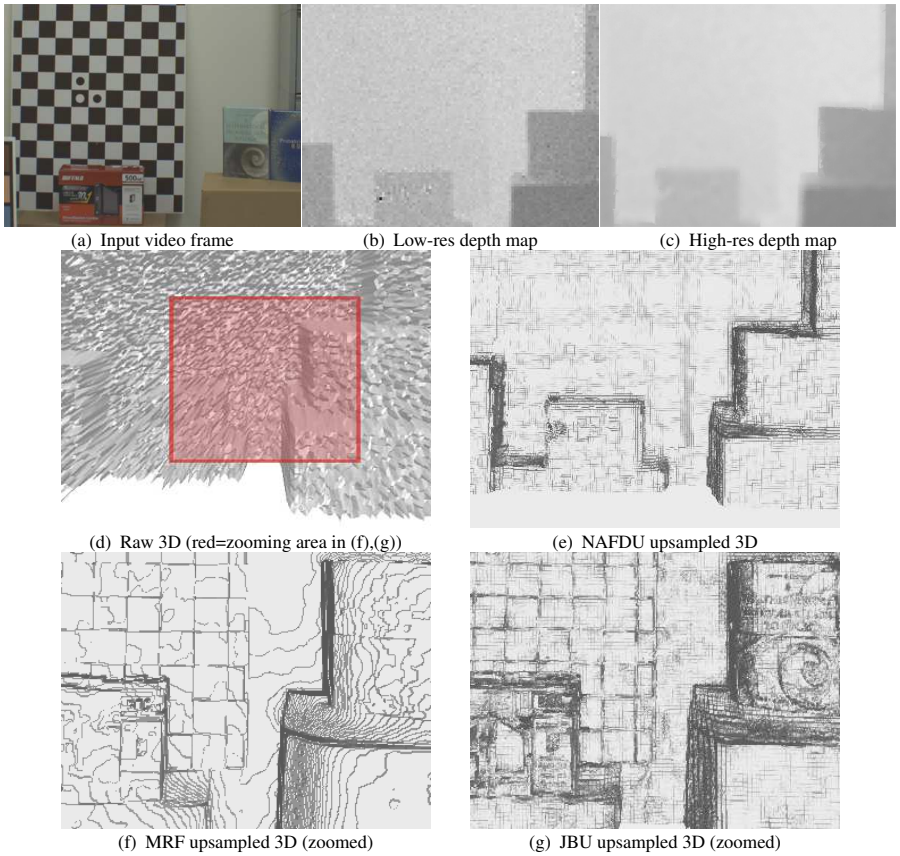
6.1 Upsampling Quality and Effectiveness of Noise-aware Filtering

At the low integration times required for scene capture at 25 fps, the depth data provided by the SR3100 are severely noise-affected, as can be seen in Figs. 3 and 4. We have tested our method on several different scenes recorded with our test rig, ranging from scenes with moving people to static scenes with objects of different textural appearance. In either case, the scenes have been recorded at a frame rate of 25 fps and upsampling was performed on each frame individually.

Fig. 3 shows that with our upsampling approach, the resolution of the depth data can be effectively raised to the level of the video camera. Overall, the noise level is greatly reduced and sharp details in the original scene, such as edges, are preserved and enhanced, Figs. 3(d) and 4(f),(g) show that all previous fusion-based upsampling methods, in this case exemplified by the MRF and JBU methods, fail to appropriately address the high noise level in time-of-flight imagers. Consequently, intensity texture patterns are embossed as 3D structure into the upsampled geometry, as it can be seen on the book cover in Fig. 4(g) when using JBU, the checkerboard in Fig. 4(f),(g) when using either MRF or JBU, or the shopping bag in Fig. 1(d) when using JBU. In contrast, our noise-aware scheme effectively prevents this texture copy, and thus our achieved reconstruction quality exceeds the one obtained with previous approaches.

6.2 Runtime Analysis

On the one hand, our technique has been designed to prevent specific noise-related upsampling artifacts, but is has also been designed with computational efficiency in mind. Our method's basis as a filter allows for effective parallelization when implemented on a GPU. For the following tests, we ran on an Athlon 5600+ with an Nvidia GeForce



(a) Input video frame (b) Low-res depth map (c) High-res depth map

(d) Raw 3D (red=zooming area in (f),(g)) (e) NAFDU upsampled 3D

(f) MRF upsampled 3D (zoomed) (g) JBU upsampled 3D (zoomed)

Fig. 4. From a video image (a) and low resolution depth map (b) we create a depth map at video resolution (c). When rendering 3D geometry, the significantly higher quality of our result (e) as opposed to the raw depth (d) is very obvious. Our algorithm also does not erroneously copy intensity texture from the checkerboard or the book cover into actually planar geometry as it happens with MRF (f) and JBU (g) upsampling (zoomed to red area in (d) for better visibility).

8600 GTS upsampling from 176×144 depth data to 800×600 resolution. The method scales well with the number of stream processors and even higher frame rates would be feasible with more recent GPUs. Although a similar performance range could be reached with normal JBU, our method produces better quality, as shown before.

As shown in the previous section, the output quality of MRF-based upsampling [3] is comparable to the JBU result and therefore inferior to our algorithm on highly noisy data. In addition to this quality difference, we in practice also see a performance advantage for our filter. In their paper, Diebel et al. use an iterative solver to find the MAP upsampled depth values based on a MRF. To obtain an as fair as possible comparison, we implemented their error metric and its gradient computation on the GPU to perform a runtime analysis. We also used an implementation of the error metric and gradient

computation on a cpu solved with an iterative L-BFGS-B solver [1] to derive depth results for the following quantitative comparison section.

To find an optimal set of MRF parameters, we regularly sampled the space of each parameter, optimized for the upsampled depth maps, and kept the parameter combination which yielded the minimal root mean square error with respect to a ground truth result. With the so-found optimal set of MRF parameters, it consistently took over 150 iterations to converge to a solution when ran upon upsampling several scenes from the Middlebury dataset. When performing the error metric and gradient computation on the GPU along with appropriate loading of data to and from the GPU, it took on average 755 ms to compute 150 iterations. This assumes connection of the GPU computation to our CPU solver, but the time is absent of the time actually required by the solver to compute new gradient directions. In the event of a full GPU solution, data would not have to transfer from the GPU, at which point the iterative runtime of the gradient computation is 279 ms. This value is also absent of the time required for a GPU based solver to find a new gradient direction. In contrast, the GPU implementation of our NAFDU method averages 49 ms, which includes transfering data to and from the GPU. The required number of iterations shows that even with all possible GPU optimizations the runtimes of our algorithm, that produces better results anyway, will be hard to match.

Finally, we would like to note that our approach performs much more efficiently than the multi-plane bilateral filtering and upsampling method of Yang et al. [13]. Although their method reportedly produces higher quality results than MRF or JBU methods, it will also fall short to prevent texture copying. Furthermore, since it requires many iterations of a bilateral filter at each time step, it is infeasible for real-time applications.

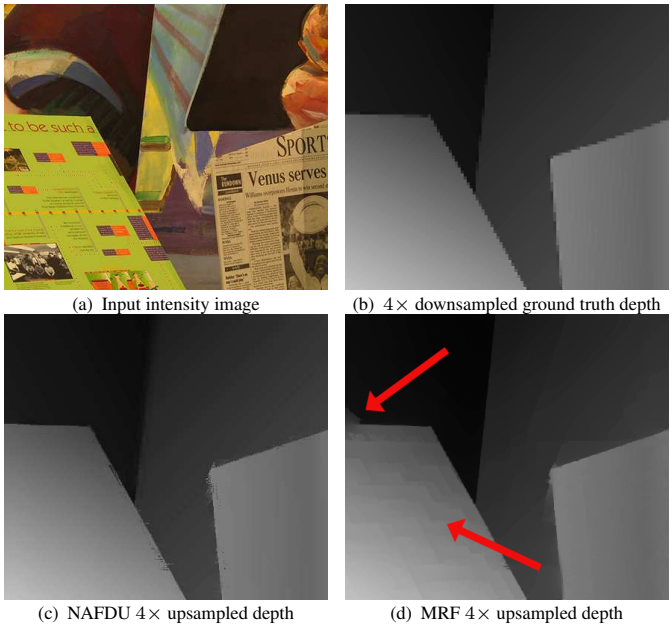
6.3 Quantitative Accuracy Comparison

To illustrate the working range of our NAFDU filter, we evaluate its performance also in the case of upsampling of depth data with little noise. To this end we quantitatively compare the output of our algorithm to the results obtained with MRF-upsampling. As test data we employ the *Cones*, *Teddy* and *Venus* scenes from the Middlebury stereo data set which also provides ground truth depth maps in addition to the intensity images, Fig. 5. The ground truth depth maps were downsampled by two, four, and eight times to use as input into both methods.

The comparison of root-mean-squared errors in Table 1 shows that under global error measure our method performs at least as good as other related methods from the literature when upsampling low noise depth maps. However, an inspection of individual parts of the upsampled images shows that the MRF approach still exhibits areas with texture copy (red arrows in Fig. 5(d)), whereas our upsampled result, Fig. 5(c), does not show this. It is thus fair to conclude that our method enables real-time performance, and at the same time delivers as good or even better results as previous methods from the literature.

6.4 Discussion and Future Work

In previous sections we showed that our noise-aware upsampling filter allows us to produce high-resolution noise-reduced depth data in real-time even with a highly noisy



(a) Input intensity image (b) 4× downsampled ground truth depth

(c) NAFDU 4× upsampled depth (d) MRF 4× upsampled depth

Fig. 5. Venus scene from the Middlebury data set. We applied our method (c) and MRF upsampling (d) to the four-times downsampled ground truth depth (b). While both methods reconstruct high-frequency content like edges faithfully, in contrast to our result, the MRF result exhibits slight texture copy in the areas marked with red arrows (best visible in electronic version).

TOF camera. The real-time requirement, however, makes some simplifying assumptions necessary for which we would like to develop improved alternatives in the future. First, our filter switches between its two operating modes using a fundamentally heuristic model that requires manual parameter setting. In future, we plan to investigate ways to learn the correct blending function from real data. Also, our 2D filter is defined in the image plane of the sensor, and we plan to research if it is feasible to define it in a local 3D domain which may enable further improvement of our results. Additional testing would be useful to determine the limits of effective spatial resolution increase; we have focused on demonstrating the techniques ability to improve range data to video resolutions.

Furthermore, some stepping artifacts in the results are due to 8-bit quantization which we will resolve in future by moving to a full floating point pipeline. Finally, we would like to note that our current warping-based alignment may lead to non-exact depth and video registration in some areas, which may explain remaining blur in our results around some actually sharp depth edges. An ideal hardware solution would have the video and depth sensors record through the same optics which would greatly facilitate alignment.

	Venus			Teddy			Cones		
Downsampled	*Raw*	*MRF*	*Our Method*	Raw	MRF	Our Method	Raw	MRF	Our Method
2x	2	2.1	**1.73**	4.32	4.1	4.09	4.91	5.07	5.41
4x	2.97	2.28	2.18	5.38	4.41	4.5	7.17	5.93	5.8
8x	4.86	3.13	2.95	7.61	5.45	5.64	10.55	8.03	**6.54**

Table 1. RMSE quantitative comparison on depth map pixel values against the MRF method using the Middlebury data set. Raw denotes the RMSE of the downsampled raw depth map. Note that our method produces results similar to MRF on these less challenging data, and in two cases (bolded) performs clearly better.

7 Conclusion

In this paper we presented a new noise-aware upsampling filter that enables us to fuse in real-time the data from a low-resolution noisy depth sensor and a high resolution video camera. Real-time fusion enables us to upsample the low resolution 3D data to the image resolution of the video sensor while preserving features, reducing random noise, and eliminating erroneous texture embossing artifacts. We have shown that the results of our method exceed the reconstruction quality obtainable with related methods from the literature. Using the parallel stream processing power of a modern graphics processor, the construction of a high-resolution real-time 3D video camera is feasible.

References

1. R. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comp.*, 16(5):1190–1208, 1995.
2. R. Crabb, C. Tracey, A. Puranik, and J. Davis. Real-time foreground segmentation via range and color imaging. In *Proc. of CVPR Workshop on Time-of-flight Computer Vision*, page NN, 2008.
3. J. Diebel and S. Thrun. An application of markov random fields to range sensing. *Advances in Neural Information Processing Systems*, 18:291–298, 2005.
4. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
5. J. Kopf, M. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. *ACM Transactions on Graphics (TOG)*, 26(3), 2007.
6. Nvidia. http://www.nvidia.com/object/cuda_get.html, 2007.
7. T. Oggier, K. Griesbach, et al. 3D-Imaging in Real-Time with Miniaturized Optical Range Camera. *6th Int. Conf. for Opt. Technologies OPTO*, pages 89–94, 2004.
8. S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. The extended version of the 2006 conference paper, 2006.
9. G. Rosenbush, T. Hong, and R. D. Eastman. Super-resolution enhancement of flash LADAR range data. volume 6736, page 673614. SPIE, 2007.
10. S. Schuon, C. Theobalt, J. Davis, and S. Thrun. High-quality scanning using time-of-flight depth superresolution. In *IEEE CVPR Workshop on Time-of-Flight Computer Vision*, page NN, 2008 (In Press).
11. C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846, 1998.
12. B. Weiss. Fast median and bilateral filtering. *ACM Transactions on Graphics (TOG)*, 25(3):519–526, 2006.
13. Q. Yang, R. Yang, J. Davis, and D. Nister. Spatial-Depth Super Resolution for Range Images. pages 1–8, 2007.