EUROGRAPHICS 2012 / P. Cignoni, T. Ertl
(Guest Editors)

Volume 31 (2012), Number 2

Coherent Spatiotemporal Filtering, Upsampling and Rendering of RGBZ Videos

Christian Richardt[1,2] Carsten Stoll[1] Neil A. Dodgson[2] Hans-Peter Seidel[1] Christian Theobalt[1]

[1] MPI Informatik [2] University of Cambridge

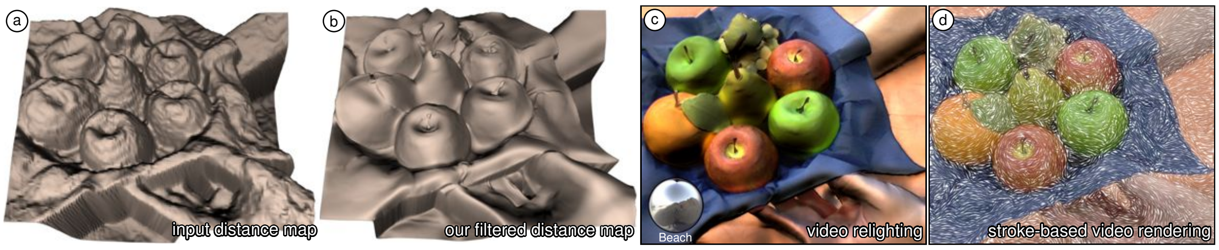

Figure 1: *Our RGBZ video processing techniques combine colour and depth videos into coherent videos with plausible per-pixel depth. This enables videos of real scenes to be rendered interactively in a range of rendering styles with unprecedented quality.*

Abstract

Sophisticated video processing effects require both image and geometry information. We explore the possibility to augment a video camera with a recent infrared time-of-flight depth camera, to capture high-resolution RGB and low-resolution, noisy depth at video frame rates. To turn such a setup into a practical RGBZ video camera, we develop efficient data filtering techniques that are tailored to the noise characteristics of IR depth cameras. We first remove typical artefacts in the RGBZ data and then apply an efficient spatiotemporal denoising and upsampling scheme. This allows us to record temporally coherent RGBZ videos at interactive frame rates and to use them to render a variety of effects in unprecedented quality. We show effects such as video relighting, geometry-based abstraction and stylisation, background segmentation and rendering in stereoscopic 3D.

1. Introduction

People increasingly express themselves through the pictures they take by producing art that goes beyond point-and-shoot photography. Computational photography has accelerated this trend: fitting a camera with new sensors, optics or computational power opens the door for groundbreaking new imagery. Some of the most compelling effects were enabled through specific modifications of the light path or sensor such that scene depth can be captured in an image in addition to colour information. The additional 'depth channel' enables previously impossible photographic effects, such as image refocusing and scene relighting after acquisition [VRA*07, OZM*06], and new non-photorealistic stylisations of real scenes [RTF*04, SZKC06].

While computational photography is relatively mature, *computational videography*, which produces similar enhancements of video technology through new sensor and optics designs, is still in its infancy. Therefore, we developed a set of essential algorithms and designed a new video camera device that enables effective and robust capture of RGBZ video (i.e., video with scene depth) of general scenes.

Our camera design uses a recently developed depth sensor, which measures the time-of-flight (ToF) of infrared light to recover scene depth, in conjunction with a normal colour video camera. The Microsoft Kinect sensor provides an alternative design that is equally suitable, but is based on an active stereo approach in the IR domain. Such depth sensors have key advantages over other geometry capture approaches: depth recovery is independent of scene texture (unlike stereo) and there is no interference with the scene in the visible spectrum (unlike structured light or photometric stereo). Additionally, such sensors can potentially be produced at low cost in high numbers, and the modifications required to a standard video camera are comparably small.

Unfortunately, ToF sensors have several limitations which prevent their data from being used directly: they produce very noisy depth data at low resolution, they become even noisier if scene motion is fast, and they do not deliver spatiotemporal correspondences which are essential for many advanced video effects, as we will show. Our research tackles these limitations and enables us to construct a computational video camera that is capable of producing spatiotemporally coherent RGBZ videos at interactive frame rates.

We demonstrate that the ability to reconstruct dynamic coherent scene geometry at interactive frame rates is a prerequisite for a variety of advanced video processing and video stylisation effects. This provides users with similar creative options for video processing as they were given by computational photography before. We show that more advanced and more appealing non-photorealistic rendering effects are possible for videos if good quality depth is at hand. In particular, we show how video relighting, geometry-based video abstraction, and stroke-based rendering benefit from our depth estimation pipeline. We also show that our high-quality noise-free depth reconstruction makes it easy to segment foreground elements, and to create stereoscopic 3D videos from the data recorded with our camera device.

Thus, the paper contributes in four ways:

1. By proposing a novel set of efficient and effective depth filtering and upsampling techniques for RGBZ videos: a fast fill-in procedure handling half-occluded regions and a spatiotemporal multi-lateral filtering approach tailored to IR-based depth cameras to increase the resolution of depth data, strongly reduce noise, and compute spatiotemporal correspondences between RGBZ video frames.
2. By presenting a prototype design for a practical computational RGBZ video camera which augments a regular video camera with a synchronised time-of-flight camera.
3. By illustrating the benefits of RGBZ video capture and processing to enable a variety of computational videographic effects either online or after short pre-processing.
4. By making source code, data sets and the design of our RGBZ camera prototype available on our project page at http://www.mpi-inf.mpg.de/resources/rgbz-camera/.

2. Previous Work

Geometry Capture There are four main approaches for capturing dynamic geometry (such as depth or normal maps) that can be used for video processing: (1) photometric stereo techniques compute normal maps from images using shape-from-shading [ZTCS99, BJK07]; (2) active stereo approaches project structured light and compute depth using triangulation [LT09]; (3) stereo vision and structure-from-motion techniques compute depth from image correspondences [SS02]; and (4) time-of-flight cameras derive depth maps from the time it takes light to travel to a point and back [KBKL10]. Out of the approaches that do not interfere with the scene, time-of-flight cameras provide the best data quality at video rates as they do not depend on scene texture. We are the first to use such cameras for RGBZ video processing.

Depth Upsampling Diebel and Thrun [DT06] first fused data from a low-resolution depth scanner and a high-resolution colour camera using Markov Random Fields by observing that strong colour and depth edges coincide. Lindner et al. [LKH07] use projective texturing to combine ToF depth data with a colour video, which leaves holes in half-occlusion regions. More recent approaches are based on the bilateral filter [TM98], which can be used for depth super-resolution by evaluating the data and range terms on depth and intensity channels, respectively [KCLU07]. Yang et al. [YYDN07] propose an approach inspired by stereo matching, in which they create a cost space from the depth map, filter it joint-bilaterally using the colour images, then extract an improved depth map, and iterate this process. These techniques all have run times of several seconds per frame. Chan et al. [CBTT08] propose a real-time technique for joint-bilateral filtering that locally adjusts the filter to simple noise estimates. All these techniques only consider single frames from a video, and do not exploit the temporal coherence of video streams to further reduce noise levels, as we do. Beder et al. [BBK07] and Zhu et al. [ZWYD08] furthermore consider how the benefits of a ToF camera can be combined with those of stereo matching. Dolson et al. [DBPT10] work with sparse range data from a laser range finder. They use a spatiotemporal joint-bilateral filter only for missing data interpolation. Our captured distance maps are much noisier, and we perform spatiotemporal filtering and super-resolution on dense depth maps.

Depth-Based Stylisation Raskar et al. [RTF*04] introduced a camera which uses four flashes to detect and highlight depth edges in static and dynamic real-world scenes. However, no actual scene geometry is recovered apart from the location and orientation of depth discontinuities. Snavely et al. [SZKC06] describe how to process and stylise RGBZ videos acquired using an active space-time stereo approach. They smooth the raw depth maps bilaterally and estimate shape correspondence between frames for coherent stylisation. As temporal information is not taken into account during the filtering step, some temporal noise remains in the data. The first real-time photometric stereo system for capturing images with normals ("RGBN images") was demonstrated by Malzbender et al. [MWGA06]. They perform basic reflectance and normal transformations to emphasise surface detail of objects. Toler-Franklin et al. [TFR07] show the large variety of rendering styles that can be applied to such RGBN images. Wang et al. [WFF*10] also use photometric stereo to compute normals, which they use for transferring a new style onto an existing object by projecting the appropriate image. However, as no motion compensation is applied, any scene motion results in incorrect normals. All of these systems interfere with the scene in the visible spectrum, whereas our system uses a recent infrared depth camera to capture colour and depth images simultaneously without interference.

3. Hardware Setup

Our prototype camera consist of a MESA Imaging SR4000 time-of-flight camera and a Point Grey Flea2 colour video camera. We fitted them side by side to minimise their baseline and adjusted the video camera's lens to cover the ToF camera's field of view (Figure 2a). To synchronise frame capture, we connected the video camera's strobe output to the ToF camera's trigger input using a custom circuit. Due to limitations of the trigger circuitry, our setup is limited to capture video at 15 Hz. This is not a limitation of our approach, and it could be overcome by additional engineering. The frame rate of our prototype camera is nevertheless sufficient for demonstrating a range of computational videography effects.

We built our prototype camera prior to the release of the Microsoft Kinect, the first mass-market product to combine an IR-based active stereo system with a colour video camera in a single case. Our prototype camera gives us full hardware and software control over all components, with synchronised video streams and higher-quality colour videos. However, both systems suffer from the same general noise problem, and use separate depth and video cameras. We describe our method in detail using our prototype camera, and in Section 5 we show that our approach is also applicable to the Kinect.

4. RGBZ Video Processing

Our goal in this section is to combine the colour and depth videos captured using our prototype camera into a coherent RGBZ video – a video with plausible per-pixel depth at colour video resolution, low noise, and correspondence over time. We do not aim for geometric accuracy as plausible geometry is sufficient for many of our applications. The four main points to address are:

Video Alignment The colour and depth videos are captured from viewpoints that are laterally displaced and thus capture different views of the same scene.

Half-Occlusions Different viewpoints also mean that each camera can see into areas which are occluded in the other camera's view. This is illustrated in Figure 2b.

Resolution Mismatch The depth data captured by the ToF camera has a spatial resolution of 176×144, whereas the video camera has a maximum resolution of 1024×768.

Noisy Depth Data The accuracy of the depth readings is at best ±1 cm, but this is masked by temporal fluctuations with a standard deviation of several centimetres (see Figure 8a).

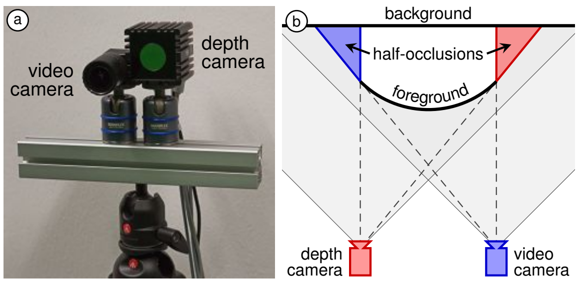

Figure 2: *(a) Our prototype camera mounted on a tripod; (b) schematic of our camera geometry with half-occlusions highlighted by the type of occluded information*

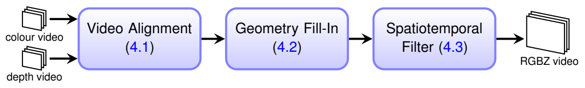

Figure 3: *Our filtering pipeline combines a colour video with a noisy, low-resolution depth video into a coherent RGBZ video with spatiotemporal correspondences.*

We propose a video processing pipeline (Figure 3) that addresses these points by aligning the colour and depth videos using a rigid transform (Section 4.1), filling in half-occluded areas (Section 4.2), and simultaneously performing super-resolution and denoising using a novel spatiotemporal filter (Section 4.3). Filtering and fill-in are data fusion operations that rely on the empirical observation that colour and depth discontinuities are usually collocated in natural scenes. As part of the filter, we compute dense spatiotemporal correspondence fields that can be used in further processing steps.

4.1. Video Alignment

The colour and depth videos from our prototype camera or the Microsoft Kinect are at first not registered or aligned. This alignment involves the choice of a common reference frame. Using the view of one of the two cameras limits half-occluded areas to be of a single type: either colour or geometry would be occluded in the reference view; any other reference frame would result in a mix of these occlusions, which would be harder to deal with than a single type. This leaves two options:

1. aligning the depth video to the colour camera's view using reprojection; or
2. aligning the colour video to the depth camera's view using projective texturing as in Lindner et al. [LKH07].

The first is the only option for us, as we aim to upsample and denoise the depth map using the more reliable colour image data. Projective texturing would reduce the quality of the available colour image and hence the filtered depth map, due to texture interpolation and unavoidable texturing artefacts.

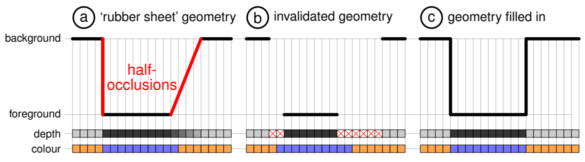

Figure 4: *Illustration of our fill-in procedure using a 1D slice through scene geometry seen from above: (a) 'rubber sheet' geometry with half-occlusions; (b) after geometry invalidation; (c) with multi-resolution geometry fill-in.*

Calibration To reproject the distance map to the video camera's view, we first calibrate the cameras intrinsically and extrinsically. ToF cameras exhibit a systematic depth bias and more sophisticated calibration approaches exist to address that [KBKL10, section 4]. However, in our experience, the basic calibration is sufficient, as the disparity between the colour and the depth cameras is small which makes the influence of systematic depth errors negligible. Using the calibrated camera parameters, we back-project the distance map to the world coordinate frame as a triangle mesh, and then project it into the video camera's view.

The output of this procedure is an aligned distance map at colour video resolution, but the true visible detail in the distance map has not been increased. Please see the accompanying video for an example.

4.2. Geometry Fill-In

The depth and video cameras capture slightly different views due to their displacement. This results in regions occluded in one view that are visible in the other (Figure 2b). Projecting the distance map onto the camera image thus introduces holes that need to be filled. ToF cameras also introduce so-called *flying pixels* at depth discontinuities which fluctuate at intermediate distances and need to be removed. They are caused by inaccuracies in the sensor's depth estimation when the area of a pixel covers surfaces at different distances [KBKL10].

We initially reproject the distance map using a triangle mesh that connects the centres of neighbouring pixels like a 'rubber sheet' (▨). In regions that are occluded in the depth camera's view, this geometry slopes to the background instead of showing a clear depth discontinuity (Figure 4a). Hence, we first invalidate these half-occluded and unreliable regions (Figure 4b), and then fill them in again from the surrounding geometry (Figure 4c).

Half-occlusions could be prevented using co-linear optics for depth and colour video capture, for instance using a beam splitter. However, all currently available systems, including the Kinect, use two displaced cameras. An algorithmic solution to overcome the resulting half-occlusions is thus highly relevant and ensures applicability to a wider range of possible RGBZ camera setups.

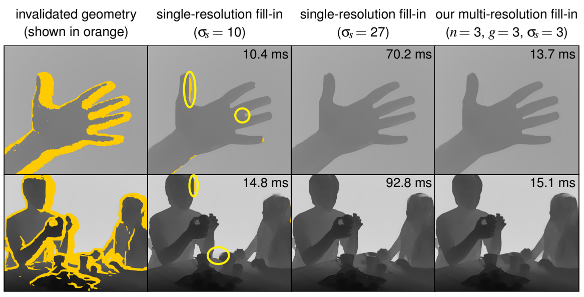
invalidated geometry (shown in orange)
single-resolution fill-in ($\sigma_s = 10$)
single-resolution fill-in ($\sigma_s = 27$)
our multi-resolution fill-in ($n = 3$, $g = 3$, $\sigma_s = 3$)
10.4 ms
70.2 ms
13.7 ms
14.8 ms
92.8 ms
15.1 ms

Figure 5: *Our multi-resolution geometry fill-in technique is 5–6× faster than the single-resolution fill-in with the large kernel size ($\sigma_s = 27$) necessary to fill holes, with comparable quality. A smaller kernel ($\sigma_s = 10$) has a similar run time, but shows errors (in yellow). Best viewed at high resolution.*

Invalidation Pixels near depth edges in the depth camera's view are unreliable and need to be invalidated. We remove these pixels by thresholding the depth gradient magnitude, which we compute using the 3×3 Sobel approximation. We use thresholds in $[0.1, 0.2]$ (smaller thresholds invalidate more geometry while larger thresholds leave too many unreliable pixels), assuming the depth maps are given in metres. In some cases, like the 'hand' sequence, visibly too little geometry is invalidated, and we manually choose to instead discard pixels that have surface normals which diverge too much from the view direction, as these tend to be noisy in depth cameras.

Fill-In The next step is to fill the holes in the aligned distance map. Here we rely on our assumption that depth and colour discontinuities coincide – a hypothesis exploited by joint-bilateral filters [TM98]. To fill large holes, as in our data, a large filter radius ($\sigma_s > 25$) is needed, which precludes fast online processing. Instead, we introduce a new multi-resolution fill-in technique inspired by a push/pull approach [GGSC96] applied to joint-bilateral upsampling [KCLU07], which produces results of comparable quality but is suitable for fast online processing, as demonstrated in Figure 5. We are aware of fast approximations to the bilateral filter, we but do not expect significant speedups that would justify their considerable additional implementation cost.

In the following paragraph, we explain our approach and illustrate it using a concrete example in Figure 6. Please refer to the illustration for every step (indicated by ①②③) of the explanation to see its effect. Our approach uses n resolution levels: 0 to $n-1$ from fine to coarse. Each level k has two inputs and one output, all of the same spatial resolution: the colour image i_k and aligned distance map d_k are inputs, and the filled-in distance map f_k is the output. The coarsest level, $n-1$, fills invalid pixels in the distance map d_{n-1} based on the corresponding colour image i_{n-1} using a standard joint-bilateral filter, resulting in the filled-in distance map f_{n-1}.

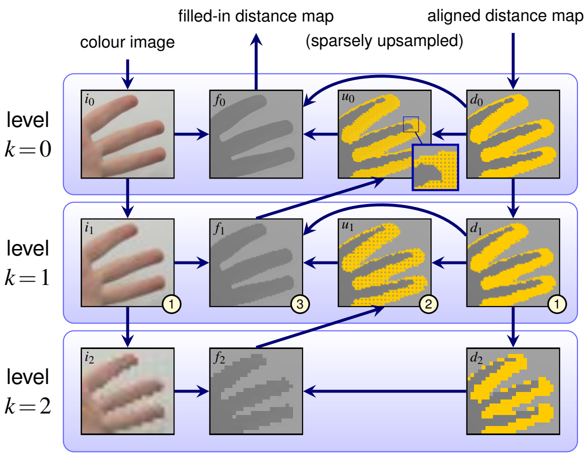

Figure 6: *Our multi-resolution fill-in technique effectively fills in the invalid orange pixels (using parameters $n=g=\sigma_s=3$). Please see Section 4.2 for a step-by-step explanation.*

All levels except the coarsest one, i.e. $k=0,\ldots,n-2$, work as follows:

① The image i_k and distance map d_k are downsampled by selecting every g^{th} pixel along the x and y axes, resulting in i_{k+1} and d_{k+1}. These downsampled images are passed to the next lower level, $k+1$, which returns a filled-in distance map f_{k+1} after all recursive processing has finished.

② The coarser levels have recursively filled all invalid pixels of d_{k+1} in f_{k+1}. These newly-filled pixels are now upsampled to a sparse grid of pixels, which is used to fill invalid regions in the distance map d_k. This results in the 'sparsely upsampled' distance map u_k with filled-in values at every g^{th} invalid pixel.

③ The same joint-bilateral filter as at the coarsest level is applied to the sparsely upsampled distance map u_k and the image i_k to fill in all invalid pixels in d_k. This recomputes the sparsely upsampled pixels (shown enlarged in Figure 6) to avoid artefacts.

We use the parameters $n=3$, $g=2$, $\sigma_s=10$ and $\sigma_r=0.05$ throughout this paper, except for the hand sequence which has the largest half-occlusion regions ($n=4$, $g=3$). Additionally, to reduce the influence of noise in the colour image, it is first filtered bilaterally using parameters $\sigma_s=3$ and $\sigma_r=0.1$.

4.3. Geometry Filtering

The result of the geometry fill-in step is passed into a spatiotemporal filter which simultaneously denoises and super-resolves distance maps: the denoising step strongly reduces the spatial and temporal noise recorded by depth cameras; and the super-resolution step uses the high-resolution colour video to increase the spatial resolution of distance maps by exploiting the coincidence of colour and depth edges.

A standard joint-bilateral filter applied at a single time step reduces spatial, but not temporal noise (Figure 8). We thus use a spatiotemporal filter that also incorporates information from previous frames, making it a 'multi-lateral' filter. As videos may contain significant motion, the filter needs to be motion-compensated. Our filter is related to the spatio-temporal upsampling technique by Herzog et al. [HEMS10] for efficient high-resolution rendering. They use one motion-compensated sample from the previous filtered frame to filter the current frame. This works well on clean geometry, but performs poorly on our noisy distance maps. We thus motion-compensate all kernel pixels, not just the centre (Figure 7).

In the following sections, we first explain a purely spatial version of our filter, f_S, and then combine it with the temporal version f_T into the spatiotemporal filter f_{ST}. We assume that values are represented as homogeneous quantities and filter the homogeneous coordinate like the others. This eliminates the usual division by the sum of weights in the filter notation.

Spatial Filter

Filtering the distance map in one time step can be achieved using a dual-joint-bilateral filter, which preserves edges in the colour image $\mathbf{i}(\mathbf{x},t)$ as well as the distance map $d(\mathbf{x},t)$. The spatially-filtered distance for pixel $\mathbf{x}$ at time step t is given by

$$f_S(\mathbf{x},t) = \sum_{\mathbf{y}\in N_{\mathbf{x}}} w_c(\mathbf{x},\mathbf{y}) \cdot w_d(\mathbf{x},\mathbf{y}) \cdot w_s(\mathbf{x},\mathbf{y}) \cdot d(\mathbf{y},t), \qquad (1)$$

where $N_{\mathbf{x}}$ is the set of pixels in the kernel of radius $2\sigma_s$ centred on $\mathbf{x}$, and the colour, distance and spatial weights are

$$w_c(\mathbf{x},\mathbf{y}) = \exp\left(-g_c \cdot \|\mathbf{i}(\mathbf{x},t) - \mathbf{i}(\mathbf{y},t)\|^2 / 2\sigma_c^2\right), \qquad (2)$$

$$w_d(\mathbf{x},\mathbf{y}) = \exp\left(-|d(\mathbf{x},t) - d(\mathbf{y},t)|^2 / 2\sigma_d^2\right) \text{ and} \qquad (3)$$

$$w_s(\mathbf{x},\mathbf{y}) = \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2 / 2\sigma_s^2\right), \qquad (4)$$

with $g_c=1$. The pixels $\mathbf{x}$ and $\mathbf{y}$ are always in the current frame, at time t, whereas $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ (which are introduced next) are in the previous frame $(t-1)$. Typical filter parameters are $\sigma_c \in [0.05, 0.1]$, $\sigma_d \in [0.075\text{ m}, 0.1\text{ m}]$, and $\sigma_s \in [4,8]$.

Spatio-Temporal Filter

Spatial filtering alone cannot suppress noise completely – residual low-frequency noise is still visible (see Figure 8 and our video). However, since this noise is independent for every time step, it can be further reduced by averaging frames from several time steps. The spatiotemporally-filtered distance at $\mathbf{x}$ and time step t is given by

$$f_{ST}(\mathbf{x},t) = \varphi \cdot f_S(\mathbf{x},t) + (1-\varphi) \cdot f_T(\mathbf{x},t), \qquad (5)$$

where the falloff parameter φ specifies the trade-off between spatial filtering (f_S) and temporal filtering (f_T), with $f_T(\mathbf{x},t)$ propagating filtered distances from the previous time step $t-1$ to the current time step t using motion compensation. The larger φ, the more weight is given to the current frame. We use $\varphi \in [0.01, 0.1]$ for the results reported here.

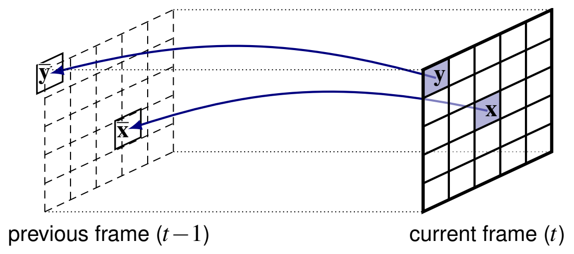

Figure 7: *Illustration of the filter kernel in the current frame (t), including arrows to indicate optical flow of the kernel centre* **x** *and a kernel pixel* **y** *to the previous frame (t − 1).*

A basic technique such as exponential averaging of the spatially-filtered distances over a window of time generates artefacts in areas of fast motion. This filters pixels which are not in correspondence, for example across depth discontinuities, and leads to 'smearing' artefacts. To address this issue, previous frames need to be motion-compensated to align moving objects across frames. Let $\bar{\mathbf{x}}$ at time step $t-1$ denote the motion-compensated location of $\mathbf{x}$ at time step t, as shown in Figure 7.

Now we compute the temporal contribution using

$$f_T(\mathbf{x},t) = \sum_{\mathbf{y}\in N_{\mathbf{x}}} w(\mathbf{x},\mathbf{y},\bar{\mathbf{x}},\bar{\mathbf{y}}) \cdot f_{ST}(\bar{\mathbf{y}},t-1) \quad (6)$$

$$w(\mathbf{x},\mathbf{y},\bar{\mathbf{x}},\bar{\mathbf{y}}) = w_c(\mathbf{x},\bar{\mathbf{y}}) \cdot w_d(\mathbf{x},\bar{\mathbf{y}}) \cdot w_s(\bar{\mathbf{x}},\bar{\mathbf{y}}) \cdot w_f(\mathbf{y},\bar{\mathbf{y}}) \quad (7)$$

$$w_f(\mathbf{y},\bar{\mathbf{y}}) = \exp\left(-\left\|\mathbf{y}-\bar{\mathbf{y}}\right\|^2 / 2\sigma_f^2\right). \quad (8)$$

There are a few things to highlight here. Equation 6 combines motion-compensated distances from the previous time step $t-1$; this reduces the noise. The weights in Equation 7 evaluate the similarity of the motion-compensated pixel $\bar{\mathbf{y}}$ to the centre $\mathbf{x}$ in the current time step t in terms of colour and distance, as before. However, the spatial weight w_s does not penalise distance from $\mathbf{x}$, but its motion-compensated location $\bar{\mathbf{x}}$. The flow weight w_f reduces the influence of past data in areas of fast motion as they tend to be unreliable. We found that $\sigma_f \in [4,5]$ produces good results.

Fast motion also leads to increased noise levels in ToF distance maps, as multiple images are sampled per frame [KBKL10]. To suppress this noise in areas of fast motion, the spatial filter is augmented using $g_c = [2 - \|\bar{\mathbf{y}}-\mathbf{y}\| / \sigma_f]_0^1$ where $[x]_a^b$ clamps x to the range $[a,b]$. The result is that in areas of fast scene motion the importance of spatial filtering is increased and distances can also be smoothed across colour edges. In practice this effectively prevents motion noise amplification. We use a full-kernel implementation of our filter, as our non-Gaussian colour weight w_c cannot be easily mapped to acceleration approaches [CPD07, AGDL09]. We compute correspondences across frames using optical flow [BBPW04], using an adapted GPU implementation by Eisemann et al. [EDM*08].

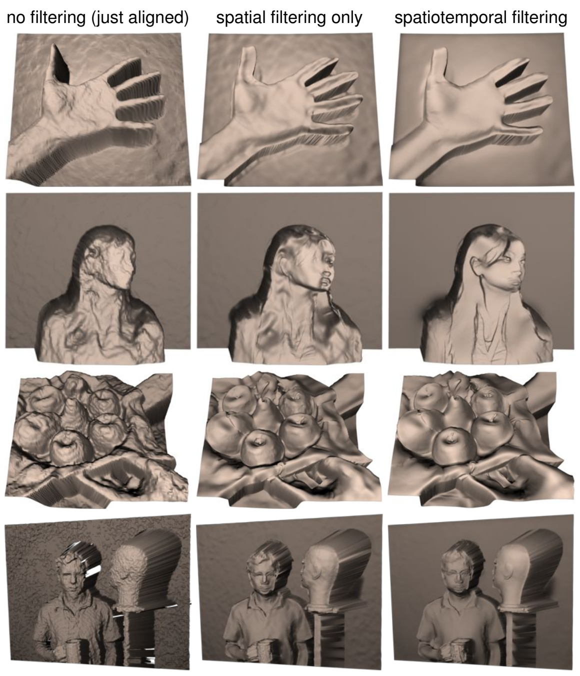

Figure 8: *Mesh renderings of distance maps without filtering, with spatial and spatiotemporal filtering. The last row shows data captured using a Microsoft Kinect sensor, with holes and depth quantisation steps. It is essential to refer to our video to see the full improvement of our approach.*

5. Results

We used our prototype RGBZ video camera to record a variety of scenes at a range of 0.5–2 m, including close-ups of objects, close-ups of people, and multiple people interacting. We recorded a total of 26 sequences with over 30,000 frames and about 35 minutes in length. Our techniques produce good results with remaining artefacts only in few frames. Noise is removed by smoothing surfaces while preserving depth and colour discontinuities. However, the strong smoothing also leads to some loss of detail, which is discussed below. Next, we use the recorded sequences to demonstrate the efficacy of the various steps in our RGBZ video filtering pipeline.

Our proposed geometry fill-in step removes and fills in unreliable and half-occluded geometry, as shown in Figure 5. While similar results could be achieved by a naïve joint-bilateral filter with a larger kernel size, our method is 5−6× faster, while outperforming a smaller kernel with similar run time in terms of quality.

Our spatiotemporal filtering step computes plausible high-quality distance maps from spatially and temporally noisy input data. In contrast, a simple spatial filtering approach still exhibits noise and flickering, which are effectively removed by our method (see Figure 8 and our video).

Microsoft Kinect We also tested our filtering approach on the Kinect sensor, an active stereo depth camera with combined video camera. Although the Kinect has different noise characteristics than a ToF camera, similar problems exist: low-resolution depth and the disparity between depth maps and video. The last row of Figure 8 shows that our filtering approach is similarly necessary and leads to clearly improved, coherent depth maps, with the Kinect's typical depth quantisation steps smoothed out (see also our video).

Performance The RGBZ video filtering from Section 4 runs at a frame rate of 5.2 fps for a 584×506 RGBZ video in our prototype implementation – which uses a GeForce 295 GPU and a 2.8 GHz quad-core processor. The bottleneck is GPU time, which divides as follows: 28% each for optical flow and spatiotemporal filter, and 22% each for geometry fill-in and view alignment computations. The video effects in Section 6 run at interactive frame rates (10 Hz or more). Our complete filtering and rendering framework can be used interactively, which allows filtering and rendering parameters to be modified on the fly and their outcome to be observed. Alternatively, the video filtering can be performed in an offline preprocessing step, and the video effects can be rendered more smoothly. Fine-tuning and advances in hardware will soon make end-to-end real-time processing feasible.

Limitations The spatiotemporal filtering result depends on the quality of the correspondences between frames: in areas of fast motion, optical flow tends to be unreliable due to motion blur, large displacements and occlusions, which can lead to smearing artefacts in the filtered distance map. This could be ameliorated using a higher capture frame rate, which limits the maximum extent of motions, or by extending the optical flow to also respect depth discontinuities.

Our fill-in and filtering steps also rely on the assumption that similar colours imply similar depth or, equivalently, that strong colour and depth edges coincide. This assumption can be violated in two ways: (1) strong texture on smooth geometry may introduce 'texture copy' artefacts into the distance map; and (2) depth edges with small colour differences may not be preserved well, such as the shoulder in the second row of Figure 8. Nevertheless, we found this assumption to hold in many real scenes.

The depth cameras we use also have low spatial resolution, which limits the level of detail in the filtered distance map. Although our filtering increases the resolution beyond the physical limits of the depth sensor, not all fine details can be recovered. Even more detail may potentially be recovered by refining our result using shape-from-shading, but this would incur additional computational costs.

The bilateral filtering approach we take is not guaranteed to fill geometry optimally, as new values are computed as a linear combination of existing values. This may result in incorrect geometry in the geometry fill-in and spatiotemporal filtering steps. In practice, this only has limited influence on the quality of reconstruction, but it allows our system to run at interactive frame rates.

Comparison to Previous Work Our work is inspired by the work of Snavely et al. [SZKC06], but our approach differs from theirs in some important aspects. Our system is specifically designed for real-time performance, whereas their reconstruction and registration steps are computationally expensive and hence not suited for interactive processing of input data. Our proposed multi-resolution fill-in procedure produces higher quality depth maps than the simple interpolation used by Snavely et al. Our filtering pipeline is designed to handle the challenging noise characteristics of recent depth cameras, rather than the depth maps on which Snavely et al. worked, which are computed using complex spacetime active stereo processing. Due to this, our approach produces higher-quality depth maps at interactive frame rates.

6. Applications

We show the quality and coherence of our RGBZ videos using five applications that rely on jointly processing geometry and images to produce high-quality computational video effects: video relighting, video abstraction, stroke-based rendering, background segmentation, and stereoscopic 3D rendering. Our video shows these results in full quality and in motion.

Video Relighting Photographers and cinematographers routinely use lighting to great effect to achieve a desired aesthetic effect, such as setting the mood or directing attention. Changing the lighting after the recording requires either having captured a suitable lighting basis in a light stage [WGT*05] or manually retouching the images to add extra information such as normals [OZM*06]. This is clearly infeasible for videos, but our RGBZ videos provide the required geometry.

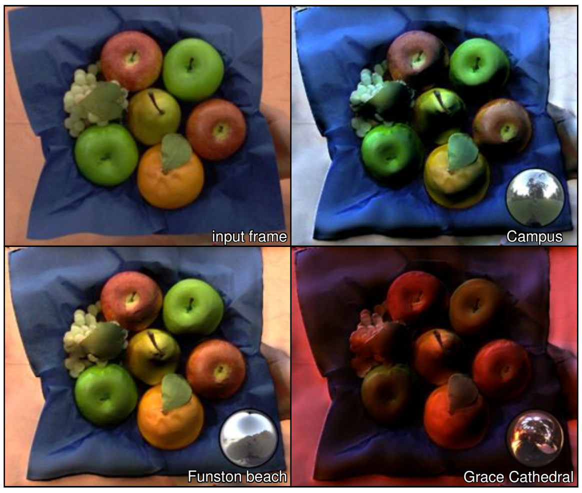

Figure 9: *Examples of our video relighting. The light probes are courtesy of Paul Debevec.*

We implemented a simple technique for relighting RGBZ videos in real time, using surface normals estimated from the depth data. We estimate the first 9 spherical harmonic coefficients of the environment lighting [RH01b, BJK07] in an offline process from a single video frame. In each frame, we obtain an albedo map by dividing each pixel's colour by the estimated illumination. We then use the same spherical harmonics technique for rendering new lighting conditions, as it efficiently approximates environment maps for diffuse objects [RH01a]. In principle, any lighting model could be used. We show several plausible relighting results in Figure 9.

Geometry-Based Video Abstraction We implemented a geometry-based video abstraction technique, which unifies video abstraction and line drawings into a rendering style that takes advantage of our coherent RGBZ videos. We first abstract the colour video using a bilateral filter, similar to Winnemöller et al. [WOG06], but also increase saturation to make the colours more vibrant. Next, we multiply the colour image with a cel-shaded version of the scene geometry, to create prominent, stylised shadow boundaries. Finally, we overlay lines using Lee et al.'s image-based line drawing technique [LMLH07], which detects ridges and valleys in a diffusely-shaded image. These lines help to visually communicate the shape of objects in the scene above and beyond what can be derived from a colour video alone. In Figure 10 we show results of our geometry-based video abstraction rendering style. Since our technique exploits scene geometry, it places feature lines at geometrically meaningful locations that correspond to perceptually important shape cues.

Stroke-Based Video Rendering We also implemented a method that performs stroke-based rendering by combining both colour and depth data from our RGBZ videos. Our technique is based on the stochastic sprite distribution approach proposed by Lu et al. [LSF10], which generates a set of textured sprites that uniformly cover the image area, advects them over time using optical flow, and handles sprite insertion and deletion to maintain uniform sprite density. However, instead of orienting strokes along image gradients, the principal curvatures of the distance map allow us to align them to meaningful geometric features. We can further attenuate strokes inversely to their diffuse shading, to concentrate them near object boundaries and more closely resemble a hatching style [PHWF01]. For exemplary results see Figure 11 and our video, where we also show a variety of results achieved by modifying rendering parameters.

Background Segmentation We further use the depth data of RGBZ videos to cleanly segment foreground objects from the background by matting out pixels behind a given plane. To soften the boundaries, we apply a 3×3 Gaussian blur to the binary matte; an additional matting step would likely further improve results. Figure 12 shows an example which compares segmentation results based on unfiltered depth and improved segmentation using spatiotemporally-filtered depth.

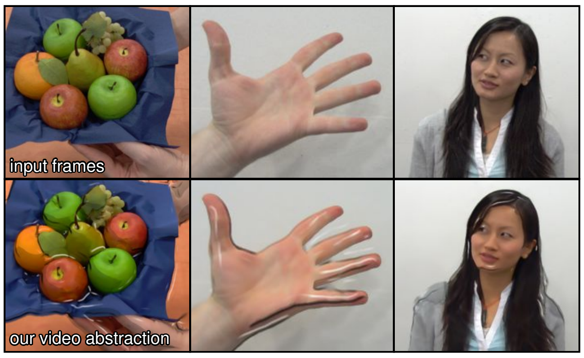

Figure 10: *Examples of our video abstraction style.*

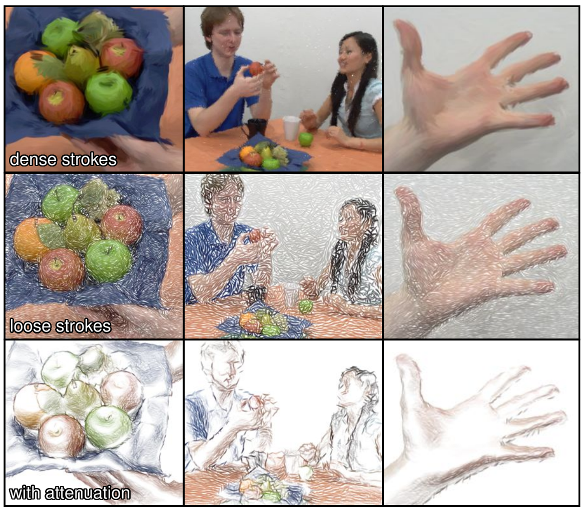

Figure 11: *Examples of our stroke-based rendering style.*

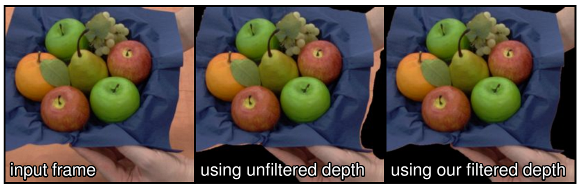

Figure 12: *Examples of our video background segmentation. Our filtered depth produces clean object outlines.*

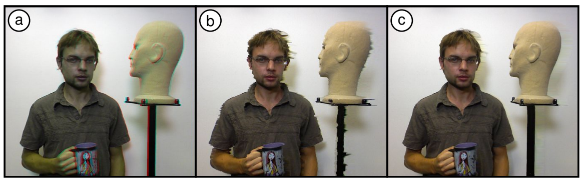

Figure 13: *Stereoscopic 3D rendering from RGBZ videos using Microsoft Kinect data: (a) red-cyan anaglyph image (best viewed at high resolution); wide-baseline right views using (b) the unfiltered and (c) our filtered depth map.*

Stereoscopic 3D Rendering By reprojecting RGBZ videos as textured triangle meshes, like in our video alignment step, new views can be synthesised, for example in stereoscopic 3D. We use parallel virtual stereo cameras that are horizontally displaced from the real camera position, according to a given interocular distance, and also shift the cameras' image plane so that the screen plane (of zero disparity) is at a given depth. The shifted views show disocclusions at depth discontinuities, which we fill using the background colour. Figure 13 shows that the accurate depth boundaries produced by our filtering approach are required to avoid artefacts. The new views are RGBZ videos with dense, high-quality depth, unlike depth maps derived from stereo matching. Thus, they can be used for applying video processing effects in stereoscopic 3D, as we show in our video.

Discussion The quality of the geometry directly influences the quality of the rendered videos. In our video, we show that purely spatial filtering of distance maps results in artefacts across all rendering styles, such as lighting artefacts and flickery lines. In contrast, distance maps filtered using our spatiotemporal approach are temporally coherent and thus avoid these artefacts. Errors in the filtered geometry, such as blurred depth discontinuities, can still lead to artefacts, such as misplaced lines in the abstraction style, halos in the relit video or small errors in video segmentation. Relighting is most sensitive to geometry inaccuracies and simplifying assumptions, such as Lambertian reflectance. On some scenes, the quality of geometry and lighting estimation was not sufficient to produce plausible results. Texture-less regions in the video further result in poor optical flow, which may lead to correspondence errors becoming visible in some effects, such as "swimming" strokes. This limitation is inherent to optical flow, not our approach. Nevertheless, our RGBZ processing pipeline enables fast high-quality depth and video capture, opening the path for computational video effects of higher quality and complexity than possible before.

7. Conclusion

We have presented essential device-independent algorithms that combine videos from a colour video camera and a depth camera into a coherent, plausible RGBZ video. To accomplish this, we propose a video processing pipeline consisting of three components: (1) a video alignment step to register the input videos; (2) a geometry invalidation and fill-in step that addresses spurious geometry; and (3) a multi-lateral spatio-temporal filter that simultaneously denoises the depth video and increases its spatial resolution to match the colour video. We also demonstrate that our approach improves the depth maps generated by the Kinect. We further showed a range of video processing effects that critically rely on the high quality of the geometry information available through RGBZ videos to achieve results that are unobtainable from a colour video alone.

Acknowledgements We thank Tadas Baltrušaitis, Gaurav Bharaj, Ian Davies, Lech Świrski, James Tompkin and Cheng-lei Wu for helpful discussions and contributions; we further thank the anonymous reviewers for their helpful feedback. Supported by the Intel Visual Computing Institute (IVCI).

References

[AGDL09] ADAMS A., GELFAND N., DOLSON J., LEVOY M.: Gaussian KD-trees for fast high-dimensional filtering. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 3 (2009), 21. 6

[BBK07] BEDER C., BARTCZAK B., KOCH R.: A combined approach for estimating patchlets from PMD depth images and stereo intensity images. In *Proc. DAGM Symposium* (2007), no. 4713 in LNCS, pp. 11–20. 2

[BBPW04] BROX T., BRUHN A., PAPENBERG N., WEICKERT J.: High accuracy optical flow estimation based on a theory for warping. In *Proc. ECCV* (2004), vol. 3024 of *LNCS*, pp. 25–36. 6

[BJK07] BASRI R., JACOBS D., KEMELMACHER I.: Photometric stereo with general, unknown lighting. *International Journal of Computer Vision 72*, 3 (2007), 239–257. 2, 8

[CBTT08] CHAN D., BUISMAN H., THEOBALT C., THRUN S.: A noise-aware filter for real-time depth upsampling. In *ECCV Workshop on multi-camera & multi-modal sensor fusion* (2008). 2

[CPD07] CHEN J., PARIS S., DURAND F.: Real-time edge-aware image processing with the bilateral grid. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3 (2007), 103. 6

[DBPT10] DOLSON J., BAEK J., PLAGEMANN C., THRUN S.: Upsampling range data in dynamic environments. In *Proc. CVPR* (2010), pp. 1141–1148. 2

[DT06] DIEBEL J., THRUN S.: An application of Markov Random Fields to range sensing. In *Proc. NIPS* (2006), pp. 291–298. 2

[EDM*08] EISEMANN M., DECKER B. D., MAGNOR M., BEKAERT P., DE AGUIAR E., AHMED N., THEOBALT C., SELLENT A.: Floating textures. *Computer Graphics Forum (Proc. Eurographics) 27*, 2 (April 2008), 409–418. 6

[GGSC96] GORTLER S. J., GRZESZCZUK R., SZELISKI R., COHEN M. F.: The lumigraph. In *Proc. SIGGRAPH* (1996), pp. 43–54. 4

[HEMS10] HERZOG R., EISEMANN E., MYSZKOWSKI K., SEIDEL H.-P.: Spatio-temporal upsampling on the GPU. In *Proc. I3D* (2010), pp. 91–98. 5

[KBKL10] KOLB A., BARTH E., KOCH R., LARSEN R.: Time-of-flight cameras in computer graphics. *Computer Graphics Forum 29*, 1 (2010), 141–159. 2, 4, 6

[KCLU07] KOPF J., COHEN M. F., LISCHINSKI D., UYTTENDAELE M.: Joint bilateral upsampling. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3 (2007), 96. 2, 4

[LKH07] LINDNER M., KOLB A., HARTMANN K.: Data-fusion of PMD-based distance-information and high-resolution RGB-images. In *Proc. ISSCS* (July 2007), pp. 121–124. 2, 3

[LMLH07] LEE Y., MARKOSIAN L., LEE S., HUGHES J. F.: Line drawings via abstracted shading. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3 (2007), 18. 8

[LSF10] LU J., SANDER P. V., FINKELSTEIN A.: Interactive painterly stylization of images, videos and 3D animations. In *Proc. I3D* (2010), pp. 127–134. 8

[LT09] LANMAN D., TAUBIN G.: Build your own 3D scanner: 3D photography for beginners. In *SIGGRAPH Courses* (2009). 2

[MWGA06] MALZBENDER T., WILBURN B., GELB D., AMBRISCO B.: Surface enhancement using real-time photometric stereo and reflectance transformation. In *Proc. EGSR* (2006), pp. 245–250. 2

[OZM*06] OKABE M., ZENG G., MATSUSHITA Y., IGARASHI T., QUAN L., SHUM H.-Y.: Single-view relighting with normal map painting. In *Proc. Pacific Graphics* (2006), pp. 27–34. 1, 7

[PHWF01] PRAUN E., HOPPE H., WEBB M., FINKELSTEIN A.: Real-time hatching. In *Proc. SIGGRAPH* (2001), pp. 581–586. 8

[RH01a] RAMAMOORTHI R., HANRAHAN P.: An efficient representation for irradiance environment maps. In *Proc. SIGGRAPH* (2001), pp. 497–500. 8

[RH01b] RAMAMOORTHI R., HANRAHAN P.: A signal-processing framework for inverse rendering. In *Proc. SIGGRAPH* (2001), pp. 117–128. 8

[RTF*04] RASKAR R., TAN K.-H., FERIS R., YU J., TURK M.: Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3 (2004), 679–688. 1, 2

[SS02] SCHARSTEIN D., SZELISKI R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision 47*, 1–3 (2002), 7–42. 2

[SZKC06] SNAVELY N., ZITNICK C. L., KANG S. B., COHEN M.: Stylizing 2.5-D video. In *Proc. NPAR* (2006), pp. 63–69. 1, 2, 7

[TFR07] TOLER-FRANKLIN C., FINKELSTEIN A., RUSINKIEWICZ S.: Illustration of complex real-world objects using images with normals. In *Proc. NPAR* (2007). 2

[TM98] TOMASI C., MANDUCHI R.: Bilateral filtering for gray and color images. In *Proc. ICCV* (1998), pp. 839–846. 2, 4

[VRA*07] VEERARAGHAVAN A., RASKAR R., AGRAWAL A., MOHAN A., TUMBLIN J.: Dappled photography: mask enhanced cameras for heterodyned light fields and coded aperture refocusing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26* (2007), 69. 1

[WFF*10] WANG O., FUCHS M., FUCHS C., DAVIS J., SEIDEL H.-P., LENSCH H. P. A.: A context-aware light source. In *Proc. ICCP* (2010). 2

[WGT*05] WENGER A., GARDNER A., TCHOU C., UNGER J., HAWKINS T., DEBEVEC P.: Performance relighting and reflectance transformation with time-multiplexed illumination. *ACM Transactions on Graphics (Proc. SIGGRAPH) 24*, 3 (2005), 756–764. 7

[WOG06] WINNEMÖLLER H., OLSEN S. C., GOOCH B.: Real-time video abstraction. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3 (2006), 1221–1226. 8

[YYDN07] YANG Q., YANG R., DAVIS J., NISTÉR D.: Spatial-depth super resolution for range images. In *Proc. CVPR* (2007). 2

[ZTCS99] ZHANG R., TSAI P.-S., CRYER J., SHAH M.: Shape-from-shading: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*, 8 (1999), 690–706. 2

[ZWYD08] ZHU J., WANG L., YANG R., DAVIS J.: Fusion of time-of-flight depth and stereo for high accuracy depth maps. In *Proc. CVPR* (2008). 2