EUROGRAPHICS 2014 / B. Lévy and J. Kautz
(Guest Editors)

Volume 33 (2014), Number 2

Spatio-Temporal Geometry Fusion for Multiple Hybrid Cameras using Moving Least Squares Surfaces

Claudia Kuster[1] Jean-Charles Bazin[1] Cengiz Öztireli[1] Teng Deng[2] Tobias Martin[1] Tiberiu Popa[3] Markus Gross[1]

[1]ETH Zurich [2]NTU Singapore [3]Concordia University

Figure 1: *a) Combined raw geometries obtained by a calibrated setup of two hybrid color+depth cameras rendered in green and red respectively. b) Result of geometry fusion obtained by our MLS-based approach. c) Textured geometry from a). Note the numerous visual artifacts due to the inaccurate and incomplete geometry, especially near depth discontinuities. d) Our optimized textured geometry from b).*

Abstract

Multiview reconstruction aims at computing the geometry of a scene observed by a set of cameras. Accurate 3D reconstruction of dynamic scenes is a key component in a large variety of applications, ranging from special effects to telepresence and medical imaging. In this paper we propose a method based on Moving Least Squares surfaces which robustly and efficiently reconstructs dynamic scenes captured by a set of hybrid color+depth cameras. Our reconstruction provides spatio-temporal consistency and seamlessly fuses color and geometric information. We illustrate our formulation on a variety of real sequences and demonstrate that it favorably compares to state-of-the-art methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Three-Dimensional Graphics and Realism—Display Algorithms

1. Introduction

High quality geometry acquisition is a fundamental problem in the fields of computer graphics and computer vision. State-of-the-art acquisition pipelines generally have three major components. First, data is acquired by a multi-sensor setup that consists of, for example, a set of calibrated color cameras [BPS*08, BHPS10] or a set of calibrated color cameras and projectors (structured light) [GWN*03]. Second, in an initial reconstruction stage, stereo matching is performed on image pairs or projected patterns. This generates an initial set of 3D points [BBH08]. Third, a fusion stage is applied to jointly combine these initial partial geometries into one consistent geometric model [Zac08, PSDB*10].

Traditionally, the initial reconstruction stage is time consuming, with the exception of very few systems [GWN*03]. However, the recent emergence of low-cost hybrid color+depth cameras, such as the Kinect, paved the way to real-time geometry acquisition, making the initial reconstruction stage virtually computationally free. On the downside, geometric data from such hybrid cameras is of very low quality, suffering from severe spatial as well as temporal artifacts. Moreover, these devices only capture 2.5D geometry (depth maps). In order to acquire the full 3D geometry of a scene or an object, multiple devices need to be used and their data fused into one coherent 3D model or, in the case of spatio-temporal surfaces, into one 4D model.

In this work we consider acquisition setups consisting of a set of calibrated hybrid cameras. We focus on the fusion stage that takes a sequence of depth maps and the associated color images as input and computes a spatio-temporal surface. Our goal is to outfit the acquisition pipeline with an efficient yet versatile fusion method that is tailored to the specific features and limitations of such setups.

There are several important challenges in fusing data acquired by multiple hybrid cameras. (i) The depth data is spatially and temporally noisy, particularly around depth discontinuities. This leads to incomplete geometry and disturbing flickering artifacts [KPZ*11]. (ii) Calibration errors can hinder the fusion process as the depth maps might not align properly [MF11]. Calibration of heterogeneous setups is typically more difficult and error prone than for homogeneous setups [HKH12]. (iii) Hybrid setups provide multiple data types (i.e., color and depth) that need to be combined for optimal reconstruction [KPZ*11]. (iv) A large variety of applications such as free viewpoint video and autostereoscopic 3D requires the generation of multiple views simultaneously. Therefore, it is important that the fusion of the geometry is view-independent. (v) The fusion process is generally memory intensive and computationally expensive, especially for spatio-temporal reconstruction [SAL*08].

We propose a fusion method for setups with multiple hybrid cameras that addresses these limitations. Our method is based on a point based surface representation and Moving Least Squares (MLS) reconstruction. Our main technical contribution is to adapt and extend current MLS reconstruction techniques for efficient, accurate and robust reconstruction that can naturally handle surface discontinuities and consistently incorporate color and temporal information.

2. Related work

Early methods of 3D surface acquisition using multi-camera setups rely on visual hulls to compute the rough geometry of an object [MBR*00]. Although very efficient, hull based methods miss details, especially in concave areas. Furthermore, they put a lot of constraints on what type of objects can be captured as well as on the camera positions. Therefore, these methods were complemented [dST*08] and eventually replaced altogether by techniques based on pairwise stereo reconstruction [BPS*08, VPB*09]. These pairwise 2.5D geometries need to be fused into one global 3D geometry. This can be achieved by fitting a template to the captured object [dST*08, LAGP09]. Such a template-based approach has the advantage of yielding a complete and consistent model spatially as well as temporally. However, in general template-based methods limit the scope of the acquisition. Templates for all objects in a scene are not always available and often impractical to build. Some methods require a priori a template of the captured object that has to be obtained by some other means. Other methods compute an ad-hoc template, but they are usually not online, i.e. they require the entire sequence as input [WJH*07, WAO*09].

Since the wide adoption of low-cost depth cameras, a lot of effort is focused on refining the depth maps by incorporating color and temporal information [RSD*12, LWA*12]. Although high-quality results can be achieved, processing the data from each device independently might lead to inconsistencies, such as geometric misalignments. The refined depth maps have to be jointly fused into a 3D (or 4D) surface using a surface reconstruction technique. In the following, we differentiate between surface reconstruction techniques based on their underlying surface representation.

Depth map representation: for a variety of applications, especially in graphics, vision and robotics, depth maps are a very popular representation [MAW*07, ZJWB09, CVHC08, GFP10]. One of the main benefits of depth maps is that the entire arsenal of image processing techniques is (i) readily available to refine the data very efficiently [PKT*11, RSD*12, MACNB12, YYLH12] and (ii) also suitable for spatio-temporal processing [LWA*12]. The downside of these methods is that the reconstruction is view-dependent. This implies that the processing has to be performed for each frame and virtual camera location and in the case of spatio-temporal reconstruction, the viewpoint must be fixed throughout the sequence.

Triangle mesh representation: mesh-based representations are widely used due to their simplicity and efficient execution on graphics hardware. For example, Maimone et al. [MF11] simultaneously acquire data from 10 Kinects and render the geometry from each individual device as a triangle mesh. As the rendered meshes are not consistent, the result exhibits spatial as well as temporal artifacts. To fuse the entire point cloud into one surface, most mesh methods use Voronoi diagram and Delaunay triangulation based algorithms to compute the connectivity [AB98, ACK01]. These methods are generally computationally expensive. Furthermore, they are challenged when large parts of the scene are occluded and they are not robust to noise. Other mesh based methods for temporal fusion like [PSDB*10, LLV*12] rely on parameterization and tracking to construct the temporal varying surface, but are dedicated to offline processing (in the order of several minutes per frame). In contrast, we obtain a processing rate in the order of one second per frame (see details in Section 5).

Volumetric representation: many popular volumetric fusion methods are based on the seminal work of Curless et al. [CL96], which maintains a regular grid enclosing the area of interest. The volumetric representation encodes the surface as a signed distance field where the implicit surface at distance zero represents the scene. This signed distance field is constructed by fusing depth maps from multiple views into the volume. Ray casting is generally employed to render the scene. This approach is especially useful when many views are merged into the volume [IKH*11], as the data size of

the volume remains constant throughout the fusion process which incrementally constructs a consistent surface. When many views are available, the resulting surface can be of high quality as the input data noise can be, to some extent, filtered out during the volumetric averaging process. However, this advantage does not play out when only a few views are available.

There are three additional shortcomings on this representation: First, updating the volume with dynamic data is a challenging task [MF12a]. Second, methods based on volumetric representations are memory intensive, as a relatively high volume resolution is required to achieve reasonable results. Therefore, in practice, the volume is moved around the scene to capture a larger area of interest [WKF*12, CBI13]. Third, it is difficult to incorporate color information in these volumetric approaches as it is not clear how textures can be fused into the volumetric representation as well. Texturing the reconstructed surface is generally based on texture projection [RNK97] or a variation of voxel coloring [SD99], or a hybrid of both [MF12a]. Texture projection projects the color image as seen from the color camera onto the reconstructed surface. Voxel coloring stores colors as weighted averages in the volume. Since the volume resolution is generally much lower than the input textures, methods based on texture projection have to address ghosting artifacts, whereas approaches based on voxel color exhibit blurring artifacts.

Variational methods [Zac08, KBH06] in the context of surface reconstruction explicitly store the volumetric grid on which the surface is reconstructed solving a global system of equations. They allow incorporating temporal consistency as a fourth dimension in the formulation (e.g., [SAL*08]). However, storing such a 4D volume does not only significantly increase the memory requirement, but also makes solving the respective (non-linear) problem computationally expensive.

Point based methods: 3D points are a natural primitive for geometry acquisition as most devices return point clouds or depth maps that can be trivially converted into points. Wand et al. [WJH*07, WAO*09] adopt a point based surface representation for temporal fusion to build a template of dynamic objects. However, the method is computationally demanding. Another possibility for fusing and consolidating point cloud data is applying filters on point locations, possibly augmented with attributes such as colors, similar to bilateral or robust filtering. Such filters can be used to remove noise and outliers [HLZ*09, HWG*13] as a preprocessing step for surface reconstruction, but cannot be used alone to generate and resample smooth surfaces.

A recent set of methods that combines the efficiency and locality of explicit surfaces with the robustness of the implicit methods is MLS based point set surfaces [Lev03, ABCO*03, ABCO*03]. They can locally estimate the implicit function and project points onto the surface. They have also been extended for improved stability [GG07], fast evaluation on the GPU [GGG08], and sharp feature and detail preservation [FCOS05, ÖGG09]. However, the current MLS definitions are not suitable to handle large depth discontinuities, and temporal and color dimensions have not been considered.

3. MLS based Point Set Surfaces

MLS based point set surfaces (PSS) are a class of robust surface definition and reconstruction methods that rely on a set of, possibly noisy, 3D sample points as input [ABCO*03, AA04, KB04]. In our application, the 3D point set is acquired by multiple depth cameras. MLS PSS approximate an implicit function $f : \mathbb{R}^3 \to \mathbb{R}$ and provide stable reconstructions based on local fitting. This approximation is computed such that $f(\mathbf{x}_i) \approx 0$ for the set of sample points $\mathbf{x}_i$ (for local surface fitting) with additional constraints such as $\|\nabla f\| = 1$ (to avoid the trivial solution $f(\mathbf{x}) = 0$). This is equivalent to performing a local kernel regression based fitting [ÖGG09], for which the following energy needs to be minimized:

$$\sum \rho(f(\mathbf{x}_i) - y_i) k(\mathbf{x}, \mathbf{x}_i) \tag{1}$$

Typically, a least squares system is solved such that $\rho(\cdot) = (\cdot)^2$, and the kernel function $k(\cdot)$ is defined to give more weights to the points $\mathbf{x}_i$ that are closer to the point $\mathbf{x}$. Due to this weighting, a local approximation is obtained by expanding $f(\mathbf{x}_i)$ in a Taylor series around $\mathbf{x}$ as $f(\mathbf{x}_i) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T (\mathbf{x}_i - \mathbf{x}) + \cdots$ and computing the minimizing $f, \nabla f, \cdots$. When using a first order expansion for f, a simple and efficient approximation can be computed by the following formula [AA03, AA04]:

$$f(\mathbf{x}) = \mathbf{n}(\mathbf{x})^T \left(\mathbf{x} - \mathbf{a}(\mathbf{x})\right), \tag{2}$$

where $\mathbf{n}(\mathbf{x})$ can be computed via a local principle component analysis or as $\mathbf{n}(\mathbf{x}) = \sum \mathbf{n}_i w_i(\mathbf{x})$ (if the normals $\mathbf{n}_i$ are provided), $\mathbf{a}(\mathbf{x}) = \sum \mathbf{x}_i w_i(\mathbf{x})$, and $w_i(\mathbf{x}) = k_i(\mathbf{x}) / \sum k_i(\mathbf{x})$. The kernel function $k(\cdot)$ determines the influence of each sample on the local approximation. We use a kernel of the form $k_i(\mathbf{x}) = \phi(\frac{\|\mathbf{x} - \mathbf{x}_i\|}{h_i(\mathbf{x})})$ where $h_i(\cdot)$ controls the smoothness of the surface (in our case, a simple constant parameter) and $\phi(\cdot)$ is a monotonically decreasing radial function. For $\phi(\cdot)$, we use a fast approximation of the Gaussian kernel [GG07]:

$$\phi_i(r) = \begin{cases} (1 - r^2)^4 & \text{if } r < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Once the implicit function $f(\mathbf{x})$ is obtained, the points can be projected onto the surface for rendering via splatting [ZPvBG01]. The projection can be simply performed with a gradient descent $\mathbf{x}_{k+1} = \mathbf{x}_k - \nabla f(\mathbf{x}_k) f(\mathbf{x}_k)$, where $\mathbf{x}_0$ is the point to be projected. For the definition in Equation 2, this can be made even more efficient by the approximate descent $\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{n}(\mathbf{x}_k) f(\mathbf{x}_k)$. Typically a few iterations are sufficient for a very accurate projection. Note that the projection is a fast operation involving only local neighbors at each iteration, making MLS surfaces suitable for efficient reconstruction and rendering of noisy point cloud data.

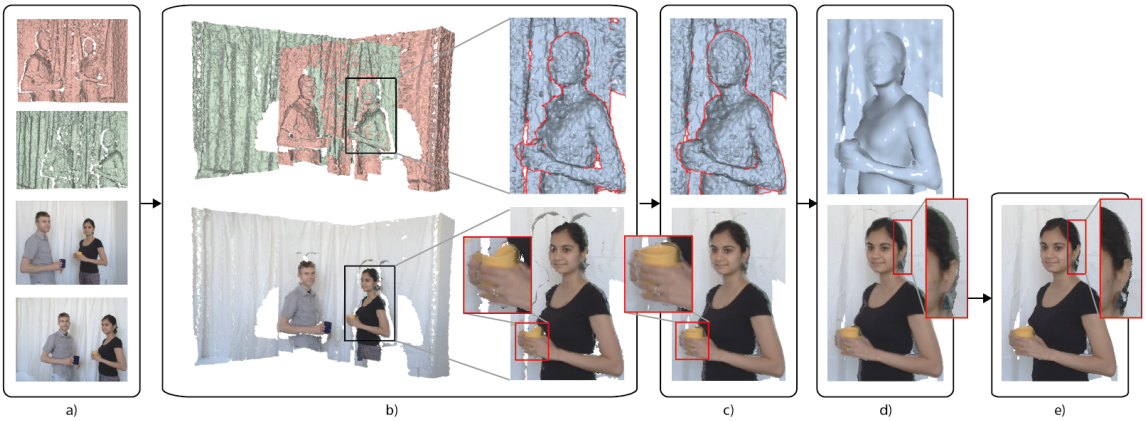

Figure 2: *Our pipeline: a) Raw input data: depth maps and color images of (here two) hybrid cameras. b) Combined geometry (top) and textured version (bottom). c) Boundary refinement and hole-filling. d) Final fused surface. e) Optimized textured rendering of the fused surface.*

3.1. Limitations of MLS PSS

Although celebrated for their robustness and accuracy, current MLS PSS have several limitations that need to be addressed particularly when using them for a reconstruction system based on multiple hybrid cameras. Below we outline these limitations and in the next section we detail how our method addresses them.

Color image consistency: in our setup, we simultaneously acquire depth maps and color images. Traditional MLS PSS are purely geometric and thus do not consider information from color images. In contrast, we propose taking advantage of the rich information available in these images to (i) enhance the geometry reconstruction quality and (ii) compute the 3D point color for texturing.

Temporal consistency: MLS PSS methods can be extended to the temporal domain by lifting the point representation from 3D to 4D. Adding the time dimension results in a temporally consistent geometry reconstruction. This conceptual extension faces two main technical challenges. The first one is to overcome the computational burden, while retaining the accuracy and theoretical smoothness properties of the spatio-temporal reconstruction (see performance section), and the second one is the ability to deal with the relatively sparse sampling of the time domain.

Performance: MLS PSS reconstruction relies on frequent neighborhood computations. Unlike meshes, neighbors of points cannot be efficiently retrieved unless appropriate data structures, such as kd-trees, are used. However, constructing and maintaining these structures is the main computational bottleneck of current MLS PSS methods. The complexity additionally increases in the case of spatio-temporal reconstruction due to the additional temporal dimension and number of points that define the surface to be reconstructed. We propose a solution that takes advantage of the structure inherent in the depth map and color data to perform the spatio-temporal neighborhood queries removing the need of computationally expensive data structures.

Discontinuities and boundaries: MLS PSS methods inherently assume that the underlying surface is a smooth manifold without boundaries. Although several variants of MLS surfaces address sharp features [FCOS05,ÖGG09] and boundaries [GG07], the current methods cannot handle the unreliable data points acquired by low-cost hybrid camera setups, especially near depth discontinuities as shown in Figure 6. The estimated location of these points can deviate significantly from their true locations, leading to an essentially random reconstruction in those regions. We propose a new solution that handles such cases robustly for hybrid camera setups.

4. Proposed Approach

Our novel geometry fusion method is designed for calibrated setups of multiple hybrid (color+depth) cameras. The depth cameras return depth maps that provide a set of 3D point clouds. They can be merged into a common coordinate frame using the extrinsic calibration. These merged point sets could be processed at each time instance using an MLS based reconstruction method. However, as discussed in Section 3.1, current MLS methods suffer from a number of fundamental limitations that make them unsuited for efficient dynamic scene reconstruction. The proposed method, as detailed in the following, handles fuzzy depth discontinuities robustly, ensures spatio-temporal consistency, and provides accurate texturing of the reconstructed model.

Our fusion pipeline is illustrated in Figure 2. It has three main components: First, we position the points in the fuzzy depth discontinuities by classifying them in the most plausible locally defined foreground or background region. This is necessary for a stable reconstruction of those fuzzy areas. Second, we reconstruct the spatio-temporal geometry

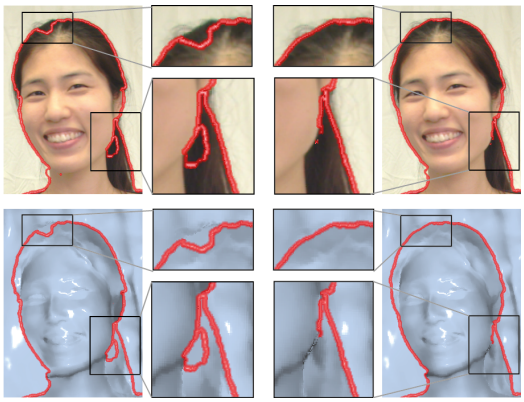

Figure 3: *Temporal kernel: 3D reconstruction without (left) and with (right) temporal information. Top: results in the color images, bottom: results in 3D space, both with close-up views.*

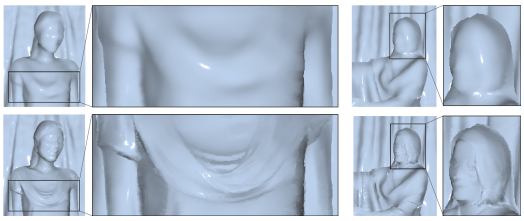

Figure 4: *Color aware MLS kernel: Top: without color information. Bottom: with color information.*

faithfully using an MLS reconstruction method that simultaneously incorporates depth and color information. Although our main goal is to reconstruct the geometry, for some applications such as novel view synthesis it is important to also provide texture information. This is typically achieved by blending the textures from different cameras, which may create unwanted visual artifacts [BBM*01]. In a third step we therefore optimize the texture based on the refined geometry.

4.1. Spatio-temporal Reconstruction

Incorporating the temporal dimension requires to reconstruct a 3-manifold surface in the 4D spatio-temporal space via the implicit function $f : \mathbb{R}^4 \rightarrow \mathbb{R}$. The same formulation as explained in Section 3 can be used to solve for f, except that the points and vectors are in 4D. These samples can be used to compute the normal $\mathbf{n}(\mathbf{x})$ via a local PCA, and $\mathbf{a}(\mathbf{x})$ via averaging, as defined in Equation 2.

The main benefit of the temporal kernel is that it provides a more accurate and consistent reconstruction that helps alleviate undesirable flickering artifacts during rendering. Figure 3 illustrates how the reconstructions obtained using our temporal kernel are more accurate and the accompanying video highlights the temporal consistency of the results.

Figure 5: *Conceptual approach of our boundary refinement on a 1D example. The dots represent the data points and the MLS resulting curve is shown in orange. a) Depth discontinuity. b) The color information of the data points (obtained from the color image) reveals that the discontinuity is inaccurate. c) Our method robustly classifies the points around the depth discontinuity as local foreground/background.*

Robustness to color differences. As observed by Öztireli et al. [ÖGG09], robustness to point-wise attributes can be obtained by adjusting the error function ρ in Equation 1. For certain classes of functions ρ, the resulting system can be solved by iteratively reweighted least squares with Gaussian weighting [ÖGG09]. For our purposes, the attributes are colors and we would like to get a surface definition that is robust to color differences. With our first order spatio-temporal definition, as explained in Sections 3 and 4.1, and assuming a single iteration, this leads to using a modified kernel k:

$$k_i(\mathbf{x}) = \phi\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h_i(\mathbf{x})}\right) \cdot \phi\left(\frac{\|c(\mathbf{x})-c(\mathbf{x}_i)\|}{h_i^c(\mathbf{x})}\right) \qquad (4)$$

where $\mathbf{x}$ and $\mathbf{x}_i$ are in the spatio-temporal domain, h_i and ϕ have been defined in Section 3, and $c(\mathbf{x})$ and $c(\mathbf{x}_i)$ are the colors of the points $\mathbf{x}$ and $\mathbf{x}_i$ respectively as seen by the camera from which they are constructed. The parameter h_i^c controls the smoothness of the color, similarly to h_i in the spatial domain. The color distance is computed in the Lab color space. Figure 4 shows the improvements in fine details when using the color aware kernel.

Fast neighborhood queries. As discussed previously, the main performance bottleneck of point based surfaces is the neighborhood search, particularly the construction and maintenance of spatio-temporal search structures. A key observation is that point clouds obtained from depth maps exhibit a particular structure: points are aligned to the 2D grid of their corresponding camera. For a given point $\mathbf{x}$, we compute its set of neighbors by projecting $\mathbf{x}$ in all the cameras and taking the union of all the points within an area in the 2D domain. The set of neighbors computed on the 2D grid is larger than the set of the actual 3D neighbors, but this is not a problem as the far away points are automatically eliminated by the MLS kernel that will assign a weight close to zero if they are too far in the 3D space.

We extended the neighborhood query to 4D by adding the time dimension as the fourth coordinate and scaling it linearly to match the spatial scaling. This scaling factor was computed experimentally and is the same for all the results shown in this paper.

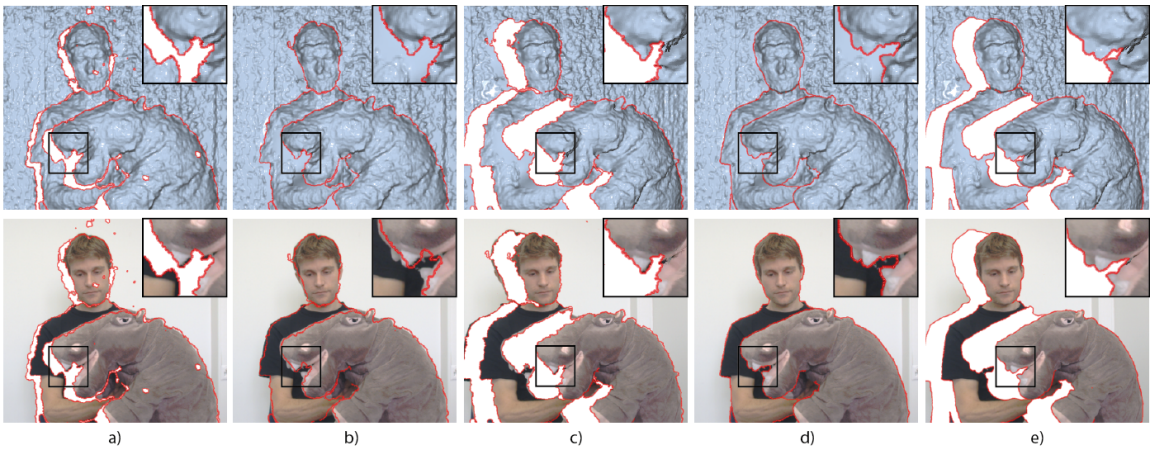

Figure 6: *The boundary refinement shown for clarity on the geometry of one depth camera. Top: Geometry. Bottom: Textured geometry. Red boundaries highlight depth discontinuities. a) Initial geometry with holes. b),c) Geometry filled, boundary is inconsistent with the color information. Rendered from the camera view and a virtual view, respectively. d),e) Corrected boundary, consistent with the color information. Rendered from the camera view and a virtual view, respectively. Note particularly the correct reconstruction of the teeth of the plush hypo highlighted in the zoomed-in area.*

4.2. Boundary Refinement

One of the most appealing features of MLS methods is their robustness to noise. However, although MLS surfaces are designed to tolerate some errors in the point set, they typically fail in the presence of very large errors. Specifically, very frequently with current depth camera technologies, points near depth discontinuities are inherently unreliable [KPZ*11]. They are often positioned incorrectly by the depth camera either onto the locally defined foreground or background of the depth discontinuity, as illustrated in Figures 5 and 6. These points can be too far away from the true geometry for the MLS reconstruction to work. Therefore it is important to detect and relocate them, such that the error is within the interval tolerated by the MLS reconstruction mechanism.

To address this issue we make a few key observations. First, as the error comes from the depth measurements, we need to search for the optimal positions for the problematic points only along the corresponding ray. Furthermore, similarly to a skewer, a ray intersects the geometry of a scene at a few discrete depths, which further limits the search space. Second, color consistency is a reliable indicator in selecting the appropriate depth along the ray and third, these points do not have to be relocated perfectly, but rather close enough for the MLS reconstruction to tolerate.

To put this into practice, for a given point $\mathbf{x}$ along a depth discontinuity we first perform a mean clustering of the neighboring pixels based on their depth. In nearly all cases there are only two clusters, therefore, for efficiency we assume two clusters. As shown in our examples, this assumption does not affect the quality of the results. The second step is to select which cluster is more consistent to the investigated point. To achieve this, we place the point sequentially at the average depth of each cluster and compute the fit function $g(\mathbf{x}_j) = \sum k_i(\mathbf{x})$ where $\mathbf{x}_j$ is the 3D position of the point $\mathbf{x}$ in the j-th cluster and $k_i()$ is defined as in Equation 4. This function g returns high values if the point $\mathbf{x}_j$ and its neighbors $\mathbf{x}_i$ are consistent in terms of both geometry and color and low values otherwise. Finally the cluster leading to the highest consistency value is selected. This step is illustrated schematically in Figure 5.

For completeness, a few additional technical details are now explained. In order for this method to work reliably, the depth map should contain no missing geometry such that all points have a full neighborhood. Therefore we fill the depth map in a similar way to [KPZ*11]. Points along depth discontinuities in one depth camera are not necessarily depth discontinuities when considering the union of all the depth maps. The boundary refinement is performed using the entire spatio-temporal geometry, thus producing robust and temporally consistent results. This procedure is repeated iteratively until the boundaries stabilize.

4.3. Texture Blending

The last step in the pipeline is to determine the color of each pixel. This is done by projecting the point into the visible cameras, and if two or more are available the information is fused together by performing a weighted average of the colors. The choice of these weights is not critical when the blended colors are similar. However, due to calibration and other data errors it happens frequently that the blended colors are quite different. As illustrated in Figure 2, this results in undesirable color bleeding artifacts. Several heuristics have

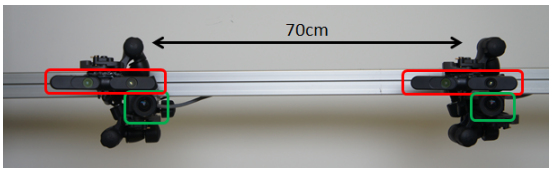

Figure 7: *Our hybrid setup composed of two Asus Xtion depth cameras (red) and two Point Grey's Grasshopper Express color cameras (green).*

been developed for this problem [BBM*01, KPZ*11]. They usually take into account the position of the cameras and/or the surface normals but not the local color consistency. We propose a new blending technique that computes the blending weights aware of the local color consistency.

Given a point $\mathbf{x}$, with colors $c_j(x)$ corresponding to M color cameras where $\mathbf{x}$ is unoccluded, we compute the unnormalized weights w_j as follows:

$$w_j(\mathbf{x}) = \sum \phi\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{h_i(\mathbf{x})}\right) \cdot \phi\left(\frac{\|c_j(\mathbf{x})-c_j(\mathbf{x}_i)\|}{h_i^c(\mathbf{x})}\right) \quad (5)$$

where $\mathbf{x}_i$ are neighborhood vertices and ϕ, h_i and h_i^c are defined as before. A high color consistency in the neighborhood of a given vertex will result in a high weight and vice-versa. The weights are normalized for the blending. As shown in Figure 9 and the accompanying video this blending method alleviates some of the color bleeding artifacts and leads to visually satisfying results.

5. Analysis and Results

We demonstrate our proposed approach on a calibrated hybrid setup consisting of two depth cameras (Asus Xtion devices) complemented with two high resolution color cameras (Grasshopper Express by Point Grey), as illustrated in Figure 7. The depth maps have a resolution of 640×480 and the color images 1280×960, and are both acquired at 30 frames per second (fps). Experiments have been conducted on a computer equipped with a 3.4GHz CPU, 16-GB RAM and a GeForce GTX 680 graphics card. Note that our approach is (i) general as its mathematical formulation does not put a restriction to the number of devices, and thus additional cameras can be installed if needed by the target application and is also (ii) scalable since its complexity increases only linearly with the number of devices. For rendering via splatting, we did not have to incorporate a resampling strategy [GGG08] as the projected splats already gave us a hole-free rendering.

Our pipeline executes entirely on the GPU, and currently runs at 0.5fps at full resolution and at 2fps at half resolution. Performance could be significantly improved with careful code optimization and next generation hardware. A strength of our method is its view-independence, i.e., the geometry does not have to be recomputed when the virtual camera is moved. Once the geometry is obtained, this allows real-time viewpoint modification, typically between 10 and 30fps in our setup, depending on the number of rendered points. Our reconstruction method relies on a few user parameters: we use a spatial neighborhood of 10 pixels, a temporal window of 4 frames and a color neighbourhood of 0.05 in the Lab perceptual color space for all the sequences of this paper.

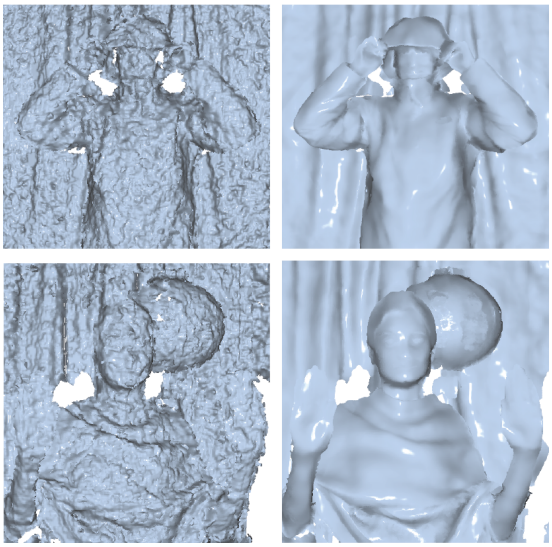

Figure 8: *Surface reconstruction using our method. Left: original geometry. Right: our surface reconstruction. Top example: Complex discontinuities are accurately preserved. Bottom example: many salient details such as the folds on the cloth are well preserved.*

Representative results obtained by our approach are available in Figures 1, 8 and 9. Some particular features have been discussed in the previous sections, such as color aware MLS kernel refinement in Figure 4 and boundary refinement in Figure 6. For a better assessment, particularly of the temporal consistency, we refer the reader to the accompanying video of this work. In addition to our own captures we have applied our algorithm to the well-known Breakdance dataset acquired by a set of color cameras [ZKU*04]. Note that here the input depth maps have been obtained by stereo matching. As shown in Figure 10 our method is able to fuse the multiple depth maps while removing quantization artifacts and preserving fine details.

5.1. Comparisons

We compare our approach against several competing state-of-the-art methods that use similar hybrid devices. In a first set of experiments, we apply two recently developed methods that integrate both color and temporal information acquired by a single hybrid camera: an advanced spatio-temporal filtering [RSD*12] and a temporal edge-aware filtering [LWA*12].

Figure 9: *Additional representative results of geometry fusion and texturing from a multi-hybrid camera setup. Raw geometry (a) and its textured version (c). Refined geometry (b) and its optimized textured version (d) obtained by our approach.*

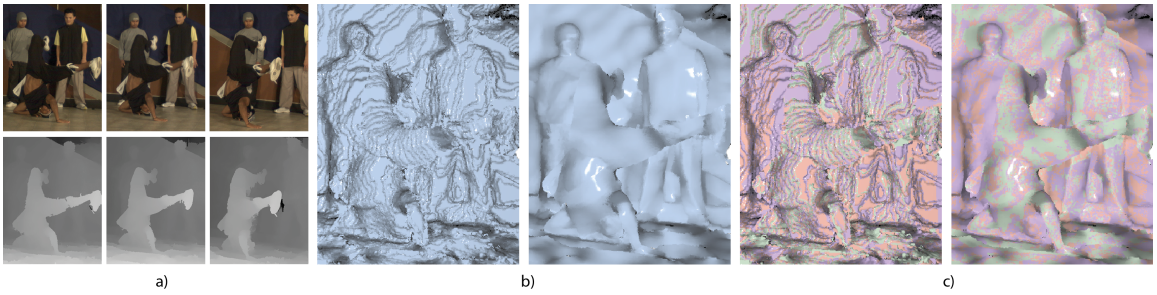

Figure 10: *Result using three cameras from the Breakdance dataset [ZKU*04]. (a) original depth maps and color images. (b) raw geometry and result. (c) color coded raw geometry and result (one color for each of the three cameras).*

Each hybrid camera is processed individually and then the resulting filtered geometries are combined into a common coordinate system using the extrinsic calibration of the setup. Results are shown in Figure 11 and in the accompanying video. Although their results are convincing on each individual device (especially the method of Richardt et al. [RSD*12]), these image-based methods cannot be straightforwardly applied to joint processing of data acquired by multiple hybrid cameras. An important practical consequence of independent processing is that the resulting geometries might not align correctly. For example, see the eye or mouth misalignment in Figure 11. While partly coming from small calibration inaccuracies of the setup (the same calibration has been used for all the methods), this geometry misalignment is mainly due to the fact that the temporal component strongly modifies the geometry. This has been experimentally verified by turning off the temporal component in the method of Richardt et al. [RSD*12] to the price of introducing temporal flickering artifacts. In contrast to independent processing, our method treats the data from all depth cameras as one geometry, thus providing more consistent results.

In a second set of experiments, we compare our work to a recent impressive system dedicated to free-viewpoint textured rendering using several hybrid cameras [MF11]. Their approach applies a specifically designed 3D data merger and renders the scene as a triangle mesh. While data from multiple cameras is simultaneously considered, experimental results show that the resulting geometry exhibits severe flickering artifacts, in contrast to our results. See Figure 12 and video.

The latest systems that fuse data from multiple hybrid cameras [IKH*11, MF12b] are based on the volumetric representation following the approach proposed by Curless et al. [CL96]. As updating the volumetric representation with dynamic data is particulary challenging, the signed distance function is reconstructed at each time instance, taking into account all data provided by the multiple cameras. The color of a fragment is determined by projecting the position of the surface it corresponds to into the image of the color camera which best faces the surface point. Results obtained by the volumetric representation are shown in Figure 12 and in the video. Thanks to the volumetric fusion, spatial consistency

Figure 11: *Data from each depth sensor processed individually and then combined, resulting in an inconsistent surface. Richardt et al. [RSD*12] (left). Lang et al. [LWA*12] (middle). Our result: MLS-based reconstruction jointly fusing the geometry from all depth sensors (right).*

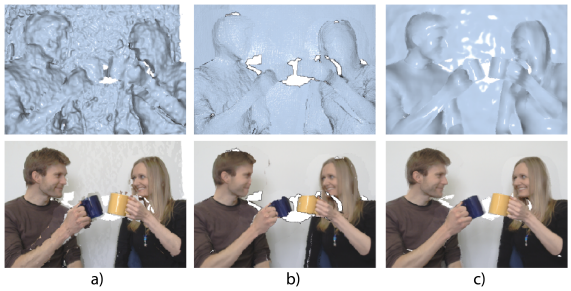

Figure 12: *Comparison of geometry reconstruction (top) and texturing (bottom) obtained by volumetric fusion [IKH*11] (a), triangle mesh-based representation [MF11] (b) and our MLS-based approach (c).*

and smoothness are ensured. However, since only a few views are available (here two cameras), the accumulation power of the volumetric representation cannot play and thus the quality of the reconstructed geometry is limited. Furthermore, flickering artifacts are noticeable because of the lack of a temporally consistent volumetric method for dynamic data.

5.2. Limitations

Despite that spatio-temporal reconstruction of dynamic data is both computationally and memory intensive [SAL*08], we managed to achieve a processing rate of 0.5fps at full resolution. Although this might not be sufficient for demanding applications such as real-time reconstruction with several additional cameras, in contrast to state-of-the-art offline spatio-temporal reconstruction methods [WJH*07, WAO*09, BPS*08, PSDB*10, LLV*12], our framework is designed bottom up for efficient reconstruction using only simple parallel local operations that each are computationally inexpensive. Thus, the current performance of our prototype is significantly faster than offline methods: in the order of half a second per frame v.s. in the order of minutes per frame and it has no conceptual performance limitations.

Because of the memory limitations of the graphics card used in our experiments, we can currently process a temporal window of only 4 frames which, while being sufficient for a wide range of sequences (as illustrated in the paper), it does not alleviate all artifacts.

6. Conclusion

In this paper we present a MLS methodology to fuse a set of 2.5D geometries with associated color information into one temporally coherent textured 3D geometry representation. Our method extends the traditional MLS point based surface representation by combining the efficiency inherent to grid aligned representations with the versatility and reconstruction power of point based representations. We apply our method to a setup consisting of hybrid cameras and demonstrate on various real-world sequences that our method produces superior output with minor flickering artifacts, even in the presence of significant spatial and temporal noise as well as small errors in the camera calibration.

References

[AA03] ADAMSON A., ALEXA M.: Approximating and intersecting surfaces from points. In *CGF (SGP)* (2003). 3

[AA04] ALEXA M., ADAMSON A.: *On normals and projection operators for surfaces defined by point sets*, 2004. 3

[AB98] AMENTA N., BERN M.: Surface reconstruction by voronoi filtering. In *SCG* (1998), ACM. 2

[ABCO*03] ALEXA M., BEHR J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE TVCG* (2003). 3

[ACK01] AMENTA N., CHOI S., KOLLURI R. K.: The power crust. In *ACM Symposium on Solid Modeling and Applications (SMA)* (2001). 2

[BBH08] BRADLEY D., BOUBEKEUR T., HEIDRICH W.: Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In *IEEE CVPR* (2008). 1

[BBM*01] BUEHLER C., BOSSE M., MCMILLAN L., GORTLER S., COHEN M.: Unstructured lumigraph rendering. In *SIGGRAPH* (2001). 5, 7

[BHPS10] BRADLEY D., HEIDRICH W., POPA T., SHEFFER A.: High resolution passive facial performance capture. *ACM TOG (SIGGRAPH)* (2010). 1

[BPS*08] BRADLEY D., POPA T., SHEFFER A., HEIDRICH W., BOUBEKEUR T.: Markerless garment capture. *ACM TOG (SIGGRAPH)* (2008). 1, 2, 9

[CBI13] CHEN J., BAUTEMBACH D., IZADI S.: Scalable real-time volumetric surface reconstruction. *ACM TOG (SIGGRAPH)* (2013). 3

[CL96] CURLESS B., LEVOY M.: A volumetric method for building complex models from range images. In *SIGGRAPH* (1996). 2, 8

[CVHC08] CAMPBELL N. D., VOGIATZIS G., HERNÁNDEZ C., CIPOLLA R.: Using multiple hypotheses to improve depth-maps for multi-view stereo. In *ECCV* (2008). 2

[dST*08] DE AGUIAR E., STOLL C., THEOBALT C., AHMED N., SEIDEL H.-P., THRUN S.: Performance capture from sparse multi-view video. *ACM TOG (SIGGRAPH)* (2008). 2

[FCOS05] FLEISHMAN S., COHEN-OR D., SILVA C. T.: Robust moving least-squares fitting with sharp features. *ACM TOG (SIGGRAPH)* (2005). 3, 4

[GFP10] GALLUP D., FRAHM J.-M., POLLEFEYS M.: A heightmap model for efficient 3D reconstruction from street-level video. In *3DPVT* (2010). 2

[GG07] GUENNEBAUD G., GROSS M.: Algebraic point set surfaces. *ACM TOG (SIGGRAPH)* (2007). 3, 4

[GGG08] GUENNEBAUD G., GERMANN M., , GROSS M.: Dynamic sampling and rendering of algebraic point set surfaces. *CGF (Eurographics)* (2008). 3, 7

[GWN*03] GROSS M., WÜRMLIN S., NAEF M., LAMBORAY E., SPAGNO C., KUNZ A., KOLLER-MEIER E., ŠVOBODA T., VAN GOOL L., LANG S., STREHLKE K., MOERE A. V., STAADT O.: Blue-c: a spatially immersive display and 3D video portal for telepresence. In *SIGGRAPH* (2003). 1

[HKH12] HERRERA C. D., KANNALA J., HEIKKILA J.: Joint depth and color camera calibration with distortion correction. *IEEE PAMI* (2012). 2

[HLZ*09] HUANG H., LI D., ZHANG H., ASCHER U., COHEN-OR D.: Consolidation of unorganized point clouds for surface reconstruction. In *ACM TOG (SIGGRAPH Asia)* (2009). 3

[HWG*13] HUANG H., WU S., GONG M., COHEN-OR D., ASCHER U., ZHANG H. R.: Edge-aware point set resampling. *TOG* (2013). 3

[IKH*11] IZADI S., KIM D., HILLIGES O., MOLYNEAUX D., NEWCOMBE R., KOHLI P., SHOTTON J., HODGES S., FREEMAN D., DAVISON A., ET AL.: KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. *ACM UIST* (2011). 2, 8, 9

[KB04] KOBBELT L., BOTSCH M.: A survey of point-based techniques in computer graphics. *CGF* (2004). 3

[KBH06] KAZHDAN M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *CGF (SGP)* (2006). 3

[KPZ*11] KUSTER C., POPA T., ZACH C., GOTSMAN C., GROSS M.: FreeCam: A hybrid camera system for interactive free-viewpoint video. In *VMV* (2011). 2, 6, 7

[LAGP09] LI H., ADAMS B., GUIBAS L. J., PAULY M.: Robust single-view geometry and motion reconstruction. *ACM TOG (SIGGRAPH Asia)* (2009). 2

[Lev03] LEVIN D.: Mesh-independent surface interpolation. *GMSV* (2003). 3

[LLV*12] LI H., LUO L., VLASIC D., PEERS P., POPOVIĆ J., PAULY M., RUSINKIEWICZ S.: Temporally coherent completion of dynamic shapes. *ACM TOG (SIGGRAPH)* (2012). 2, 9

[LWA*12] LANG M., WANG O., AYDIN T., SMOLIC A., GROSS M.: Practical temporal consistency for image-based graphics applications. *ACM TOG (SIGGRAPH)* (2012). 2, 7, 9

[MACNB12] MAC AODHA O., CAMPBELL N. D., NAIR A., BROSTOW G. J.: Patch based synthesis for single depth image super-resolution. In *ECCV* (2012). 2

[MAW*07] MERRELL P., AKBARZADEH A., WANG L., MORDOHAI P., FRAHM J.-M., YANG R., NISTER D., POLLEFEYS M.: Real-time visibility-based fusion of depth maps. In *ICCV* (2007). 2

[MBR*00] MATUSIK W., BUEHLER C., RASKAR R., GORTLER S. J., MCMILLAN L.: Image-based visual hulls. In *SIGGRAPH* (2000). 2

[MF11] MAIMONE A., FUCHS H.: Encumbrance-free telepresence system with real-time 3D capture and display using commodity depth cameras. In *IEEE/ACM ISMAR* (2011). 2, 8, 9

[MF12a] MAIMONE A., FUCHS H.: Real-time volumetric 3D capture of room-sized scenes for telepresence. *3DTV* (2012). 3

[MF12b] MAIMONE A., FUCHS H.: Reducing interference between multiple structured light depth sensors using motion. *IEEE VR* (2012). 8

[ÖGG09] ÖZTIRELI C., GUENNEBAUD G., GROSS M.: Feature preserving point set surfaces based on non-linear kernel regression. *CGF (Eurographics)* (2009). 3, 4, 5

[PKT*11] PARK J., KIM H., TAI Y.-W., BROWN M. S., KWEON I.: High quality depth map upsampling for 3D-TOF cameras. In *ICCV* (2011). 2

[PSDB*10] POPA T., SOUTH-DICKINSON I., BRADLEY D., SHEFFER A., HEIDRICH W.: Globally consistent space-time reconstruction. *CGF (SGP)* (2010). 1, 2, 9

[RNK97] RANDER P., NARAYANAN P. J., KANADE T.: Virtualized reality: constructing time-varying virtual worlds from real world events. In *IEEE VIS* (1997). 3

[RSD*12] RICHARDT C., STOLL C., DODGSON N. A., SEIDEL H.-P., THEOBALT C.: Coherent spatiotemporal filtering, upsampling and rendering of RGBZ videos. *CGF (Eurographics)* (2012). 2, 7, 8, 9

[SAL*08] SHARF A., ALCANTARA D. A., LEWINER T., GREIF C., SHEFFER A., AMENTA N., COHEN-OR D.: Space-time surface reconstruction using incompressible flow. In *ACM TOG (SIGGRAPH Asia)* (2008). 2, 3, 9

[SD99] SEITZ S. M., DYER C. R.: Photorealistic scene reconstruction by voxel coloring. *IJCV* (1999). 3

[VPB*09] VLASIC D., PEERS P., BARAN I., DEBEVEC P., POPOVIĆ J., RUSINKIEWICZ S., MATUSIK W.: Dynamic shape capture using multi-view photometric stereo. In *ACM TOG* (2009), ACM. 2

[WAO*09] WAND M., ADAMS B., OVSJANIKOV M., BERNER A., BOKELOH M., JENKE P., GUIBAS L., SEIDEL H.-P., SCHILLING A.: Efficient reconstruction of nonrigid shape and motion from real-time 3D scanner data. *ACM TOG* (2009). 2, 3, 9

[WJH*07] WAND M., JENKE P., HUANG Q., BOKELOH M., GUIBAS L., SCHILLING A.: Reconstruction of deforming geometry from time-varying point clouds. In *CGF (SGP)* (2007). 2, 3, 9

[WKF*12] WHELAN T., KAESS M., FALLON M., JOHANNSSON H., LEONARD J., MCDONALD J.: Kintinuous: Spatially extended KinectFusion. In *RSS Workshop on RGB-D* (2012). 3

[YYLH12] YANG J., YE X., LI K., HOU C.: Depth recovery using an adaptive color-guided auto-regressive model. In *ECCV* (2012). 2

[Zac08] ZACH C.: Fast and high quality fusion of depth maps. In *3DPVT* (2008). 1, 3

[ZJWB09] ZHANG G., JIA J., WONG T.-T., BAO H.: Consistent depth maps recovery from a video sequence. *IEEE TPAMI* (2009). 2

[ZKU*04] ZITNICK C. L., KANG S. B., UYTTENDAELE M., WINDER S., SZELISKI R.: High-quality video view interpolation using a layered representation. *SIGGRAPH* (2004). 7, 8

[ZPvBG01] ZWICKER M., PFISTER H., VAN BAAR J., GROSS M.: Surface splatting. In *SIGGRAPH* (2001). 3