NormalFusion: Real-Time Acquisition of Surface Normals for High-Resolution RGB-D Scanning

Hyunho Ha[†] Joo Ho Lee[*] Andreas Meuleman[†] Min H. Kim[†]

[†] KAIST [*] University of Tuebingen

Abstract

Multiview shape-from-shading (SfS) has achieved high-detail geometry, but its computation is expensive for solving a multiview registration and an ill-posed inverse rendering problem. Therefore, it has been mainly used for offline methods. Volumetric fusion enables real-time scanning using a conventional RGB-D camera, but its geometry resolution has been limited by the grid resolution of the volumetric distance field and depth registration errors. In this paper, we propose a real-time scanning method that can acquire high-detail geometry by bridging volumetric fusion and multiview SfS in two steps. First, we propose the first real-time acquisition of photometric normals stored in texture space to achieve high-detail geometry. We also introduce geometry-aware texture mapping, which progressively refines geometric registration between the texture space and the volumetric distance field by means of normal texture, achieving real-time multiview SfS. We demonstrate our scanning of high-detail geometry using an RGB-D camera at ∼20 fps. Results verify that the geometry quality of our method is strongly competitive with that of offline multiview SfS methods.

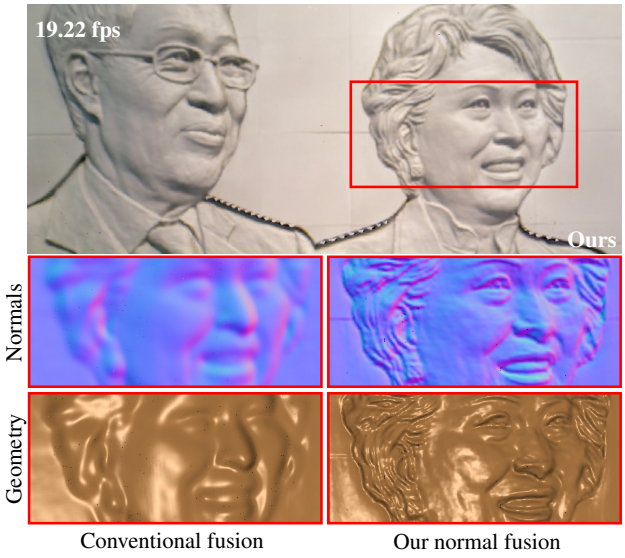


Figure 1: Result of our real-time normal fusion method, compared with the conventional fusion method that accumulates TSDFs of depth maps in the canonical space. We decompose camera signals to photometric normals and accumulate them in texture space associated with voxel grids of TSDFs, enabling high-resolution geometry in real-time. Refer to the supplemental video for real-time demo.

1. Introduction

Shape-from-shading (SfS) has been commonly used to enhance geometric details in 3D scanning. When surface reflectance and illumination are known, SfS factorizes reflected irradiance of camera signals to photometric normals in the camera's resolution [12]. When a high-resolution camera is used, the geometry quality can be improved significantly by combining base geometry and normals [21]. However, when reflectance and illumination are unavailable, SfS becomes a very ill-posed problem. *Multiview SfS* estimates distributions of illumination and albedo by leveraging multiview input [1, 10] and then obtains high-detail normals from shading. The geometry quality of these multiview SfS methods is significantly higher than that of real-time scanning methods using a conventional RGB-D camera. However, when a camera is unstructured and scenes are uncontrolled, multiview SfS becomes highly underdetermined; therefore, the ill-posed multiview SfS problem needs to be solved by expensive non-linear optimization with strong assumptions of scene and lighting conditions [29, 28] in addition to multiview registration.

Despite the strong benefits of multiview SfS to high-resolution geometry, it has been hardly achieved in real-time RGB-D scanning due to several challenges. First, multiview color and depth frames need to be registered by iterative closest point (ICP) [26] in general. However, perfect geometric registration by ICP is theoretically impossible with real systems due to noise in depth frames. It results in blurry reconstruction [18].

Second, to handle noise and inaccurate registration of depth information, the truncated signed distance function

(TSDF) [6] of depth maps has been accumulated in the canonical space. However, the TSDF-based algorithms introduce an inevitable tradeoff between spatial resolution of geometry and real-time performance [15, 22, 14, 11, 8]. A hashing technique was used to mitigate the tradeoff [23] by reducing memory footprint, but still, details of geometry often need to be compromised for performance.

To mitigate these challenges, we propose a real-time scanning method that can capture high-detail geometry by integrating two different techniques, volumetric fusion, and multiview SfS via geometry-aware texture mapping. First, we introduce the first *real-time acquisition method of photometric normals*, enabling us to capture the fine level of geometry stored in high-resolution texture space. Second, we propose *geometry-aware texture mapping*, which progressively refines geometric registration between the texture space and the canonical space of TSDFs so that we can solve multiview SfS with high accuracy.

We demonstrate that our method can acquire high-detail geometry, in addition to photometric normals and albedo textures in a high resolution at $\sim$20 fps using a conventional RGB-D camera. In particular, the geometry quality of our method is strongly competitive with that of offline multiview SfS methods. All codes and demo will be published online to ensure reproducibility.

2. Related Work

Online 3D Representation. Real-time 3D scanning methods [15, 22, 14, 11, 8, 23] accumulate TSDFs of depth maps in the canonical space. Since the depth frame is available in real-time, the camera pose for each frame is estimated by ICP. The spatial resolution of the reconstructed geometry is still determined by the voxel grid resolution in existing real-time methods. While the voxel hashing data structure can improve memory efficiency [23], the geometry resolution still needs to be compromised for real-time performance.

Traditionally, appearance attributes have been stored in the volumetric voxel grid [7, 14, 23, 15]. The camera pose is estimated by the depth information and is used to backproject the captured color information to the voxel grid. However, color and depth frames are captured asynchronously. Hence, accumulated color information tends to be blurred in real-time methods. Also, like the geometry information, the color information has been commonly stored in the voxel grid, which causes the aforementioned tradeoff between resolution and performance even applied for texture information. Recently, a tile texture atlas on the volumetric distance field has been proposed to enhance color texture, ensuring the quality of the texture more detailed than the geometry [18]. Also, they find out the photometric correspondence and register the current input frame to the existing texture space. However, they decouple geometric registration from texture mapping, focusing on texture alignment only, resulting in misalignment between texture and geometry. Our method achieves high-accuracy registration between the baseline geometry and texture space by means of photometric normals stored in the texture space.

Online Geometry Enhancement. The quality of depth frames from conventional RGB-D cameras is lower than that of color frames in terms of spatial resolution and noise. Also, the noise level of depth frames is significantly higher than that of color frames. In order to mitigate the problem, a probabilistic uncertainty model of depth measurement was proposed for better alignment of the depth camera and improved fused geometry [4]. By leveraging high-quality RGB data, several shading-based geometric refinement techniques have also been proposed for real-time scanning by formulating an SfS problem in a single view to refine depth images [30, 24]. While these methods have improved the geometry quality clearly, the refined geometric information is still accumulated in the volumetric distance field, inheriting the tradeoff between resolution and performance. Also, the current real-time shading-based methods [30, 24] do not account for geometric misalignments of multiview frames when computing inverse rendering. To the best of our knowledge, we present the first real-time scanning method that can acquire photometric normals in real-time. It enables us to achieve high-quality geometry scanning by combining the baseline geometry and high-resolution normals.

Offline Geometry Enhancement. When surface reflectance and illumination are known, shape information can be decomposed from captured images. It is called shape-from-shading [12]. Leveraging multiview input, multiview SfS methods [25, 19, 29, 17, 16] have been proposed to enhance the detail of geometry. Multiview stereo is used to obtain the baseline geometry, which is refined later with shading cues via inverse rendering. However, as reflectance and illumination are unknown, they formulate inverse rendering problems with strong assumptions on scene and lighting conditions [29, 28] in addition to multiview registration. In order to solve nonlinear optimization, computational time increases significantly, even with a moderate resolution of geometry. These multiview SfS methods are thus inapplicable to real-time scanning. To mitigate the ill-posedness of the inverse rendering problem in uncontrolled scenes, multiview SfS has been applied to the multiscale voxel grid structure of TSDFs in offline scanning methods [32, 20]. However, they still require offline optimization of camera pose, per-voxel color, and geometry due to inaccurate depth information obtained from the RGB-D camera [32, 20]. In contrast, we reformulate *multiview SfS* by leveraging our geometry-aware texture mapping, which registers texture and normals from inverse rendering to the volumetric distance field with high accuracy.

3. Online Normal Fusion

Our real-time RGB-D scanning consists of three main steps: First, we estimate unknown illumination as spherical harmonics coefficients, allowing us to obtain photometric normals and diffuse albedos using SfS. Second, depths are integrated to the volumetric distance field. Third, photometric normals and albedos are warped via geometry-aware texture warping and blended in texture space. Figure 2 provides an overview.

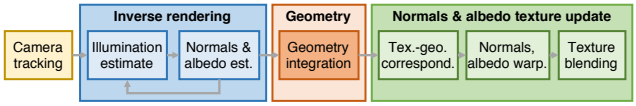


Figure 2: Overview of our reconstruction pipeline.

3.1. Online Inverse Rendering

3.1.1 Illumination Estimation

Under unknown illumination, we acquire color and depth streams as input using an RGB-D camera. Based on the diffuse surface assumption, we approximate unknown incident environment illumination using the first nine basis functions up to 2nd order spherical harmonics [11, 30]. Based on this simplification, the reflected irradiance B can be formulated as a function of diffuse albedo $\mathbf{a}$, surface normal $\mathbf{n}$ at pixel $\mathbf{u} = (i, j)$, and incident illumination:

$$B(\mathbf{u}) = \mathbf{a}(\mathbf{u}) \sum_{k=0}^{8} l_k H_k(\mathbf{n}(\mathbf{u})), \tag{1}$$

where H_k are the spherical harmonics basis functions, and l_k are the spherical harmonics coefficients of incident illumination.

We optimize the spherical harmonics coefficient set $\mathcal{L}^t$ at frame t by solving the following energy function: $E(\mathcal{L}^t) = E_{\text{ldata}} + \lambda_{\text{ltemp}} E_{\text{ltemp}}$, where E_{ldata} is the shading term, E_{ltemp} is the temporal illumination regularizer term, and λ_{ltemp} is its corresponding weight. Our shading term E_{ldata} minimizes the difference between the rendered image of diffuse reflected irradiance B^t and the input color image C^t at frame t as follows:

$$E_{\text{ldata}} = \sum\nolimits_{\mathbf{u} \in \mathcal{U}_{\mathbf{c}}^{\mathbf{t}}} ||Y(B^t(\mathbf{u})) - Y(C^t(\mathbf{u}))||^2, \tag{2}$$

where $\mathcal{U}_c^t$ is the set of valid pixels in the current color/depth frame, and Y is a luminance function that converts color values to a luminance intensity value. Solving the spherical harmonics coefficients of illumination is an overdetermined problem, which is solved by least-squares. In addition, we assume that illumination is consistent over time. The temporal regularizer of illumination E_{ltemp} is set to force the current light parameters similar to the previous estimated light parameters: $E_{\text{ltemp}} = \sum_{k=0}^{8} ||l_k^t - l_k^{t-1}||^2$.

3.1.2 Normal and Albedo Estimation

Factorizing normal and albedo from reflected irradiance is an ill-posed nonlinear problem, which needs to be solved in an iterative optimization. To reduce complexity, we optimize scalar depth values, rather than normals, as the latter would require an additional conversion step that converts normals to depth values before integrating them to TSDFs [9, 30, 24]. Note that normals can be estimated from depth values by computing finite differences over neighboring pixels.

We minimize the following energy function to optimize two unknown variables: depth $\hat{D}$ and albedo $\mathbf{a}$:

$$\begin{aligned} E(\hat{D}, \mathbf{a}) = E_{\text{data}} + \lambda_{\text{dreg}} E_{\text{dreg}} + \lambda_{\text{dsensor}} E_{\text{dsensor}} \\ + \lambda_{\text{areg}} E_{\text{areg}} + \lambda_{\text{atemp}} E_{\text{atemp}}, \end{aligned} \tag{3}$$

where E_{data} is the data term of shading, E_{dreg} is the spatial regularization term, E_{dsensor} is a depth constraint term, λ_{dreg} and λ_{dsensor} are corresponding weights for depth regularization/constraint terms, E_{areg} and E_{atemp} are spatial and temporal regularizer terms of albedo, λ_{areg} and λ_{atemp} are corresponding weights for albedo regularizers, respectively.

Our data term E_{data} forces the reflected irradiance to be the same as color observation using Equation (1):

$$E_{\text{data}} = \sum\nolimits_{\mathbf{u} \in \mathcal{U}_{\mathbf{c}}^{\mathbf{t}}} ||B^t(\mathbf{u}) - C^t(\mathbf{u})||^2. \tag{4}$$

As commonly observed in previous SfS methods [24, 30], it is challenging to differentiate the impact of shading and reflectance from observation of a single view shading. There is a potential risk that diffuse albedo could be imprinted to the optimized surface normals.

Therefore, we design two regularization terms to prevent texture copy artifacts on normals. E_{dreg} enforces depth values to be spatially regularized using Laplacian smoothness:

$$E_{\text{dreg}} = \sum\nolimits_{\mathbf{u} \in \mathcal{U}_{\mathbf{c}}^{\mathbf{t}}} ||\nabla^2 \hat{D}^t(\mathbf{u})||^2, \tag{5}$$

where ∇^2 is the Laplacian operator that enforces the current depth value similar to the average value of neighboring pixels' depth values.

E_{dsensor} ensures that the optimized depth $\hat{D}^t$ does not deviate too much from the input depth image D^t: $E_{\text{dsensor}} = \sum_{\mathbf{u} \in \mathcal{U}_{\mathbf{c}}^{\mathbf{t}}} ||\hat{D}^t(\mathbf{u}) - D^t(\mathbf{u})||^2$.

In contrast to the Laplacian smoothness term of depth, E_{areg} imposes albedos to be similar to its neighboring pixels:

$$E_{\text{areg}} = \sum_{\mathbf{u} \in \mathcal{U}_c^t} \sum_{\mathbf{z} \in \mathcal{N}_{\mathbf{u}}} \phi(\Gamma^t(\mathbf{u}) - \Gamma^t(\mathbf{z})) ||\mathbf{a}^t(\mathbf{u}) - \mathbf{a}^t(\mathbf{z})||^2, \tag{6}$$

where $\mathcal{N}_{\mathbf{u}}$ is the set of neighbors of pixel $\mathbf{u}$, $\Gamma^t(\mathbf{u}) = C^t(\mathbf{u}) / Y(C^t(\mathbf{u}))$ is the chromacity at pixel $\mathbf{u}$, and $\phi(\mathbf{q}) =$

$1/(1+b||\mathbf{q}||)^3$ is the robust kernel function [32] with a predefined parameter b to avoid texture blurriness with chromaticity outliers. We set $b = 5$ for all results.

E_{atemp} is a temporal regularization term that makes the current albedo similar to the albedo we optimized in the previous frame. It prevents albedo values from overfitting by the data term over time, enforcing temporal smoothness: $E_{\text{atemp}} = \sum_{\mathbf{u}\in\mathcal{U}_c^t\cap\mathcal{U}_s^{t-1}} ||\mathbf{a}^t(\mathbf{u}) - \mathbf{a}^{t-1}(\mathbf{u})||^2$, where $\mathcal{U}_s^{t-1}$ is the set of valid pixels in the previous frame canonical mesh rendered using the current camera pose.

3.1.3 Hierarchical Nonlinear Optimization

Estimating two unknowns, normals and albedos, from reflected irradiance is severely ill-posed like other SfS methods [30, 32]. In order to solve the nonlinear optimization problem of inverse rendering using a GPU-based Gauss-Newton optimization [27] that sequentially computes two sparse matrix-vector (SpMV) multiplication kernels [33], we formulate a total energy function that seeks optimal depths and albedos $\mathbf{x} = \{\hat{D}, \mathbf{a}\}$. Refer to mathematical details in the supplemental document.

Given input color/depth frame, three different unknowns: albedos, normals, and illumination, need to be estimated simultaneously; therefore, this inverse rendering problem is severely ill-posed. And thus, we solve this SfS problem via a two-step optimization of estimating (1) illumination and (2) normals and albedos, iteratively repeating from the coarsest to the finest level. To estimate illumination at the coarsest level of the first frame, the reflected irradiance B^t in Equation (1) is calculated with the initial surface normals obtained from the current depth map and the initial diffuse albedo, set to uniform albedos, which is the averaged albedo of the color frame (Section 3.1.1). To estimate albedos and normals in the level, the optimized illumination is used. From the subsequent finer level, the previous results of albedos and normals are used for illumination estimation. The refined illumination is used for finer optimization of albedos and normals. From the second frame, the previously optimized albedo results are used for initialization in the optimization of normals and albedos at the coarsest level. Others are processed in the same way as the first frame.

3.2. Online Normal Mapping

Our inverse rendering step decomposes input depth and color images at the current frame into four different attributes: refined depth, diffuse albedos, surface normals, and illumination. The refined depth values are integrated to the canonical space of TSDFs. The surface normals and diffuse albedos are stored as a normal map and an albedo tile texture map, respectively. These high-resolution texture maps are associated with the voxel grid of the signed distance field. Here, our fusion method inherits the data structure of the recent real-time texture scanning method [18]. As shown in Figure 3, albedos, normals, and geometry accumulated up to the previous frame $(t-1)$ are updated using the output from the inverse rendering step in three folds: (1) geometry update and texture transfer, (2) geometry-aware texture warping, and (3) normals/albedos blending.

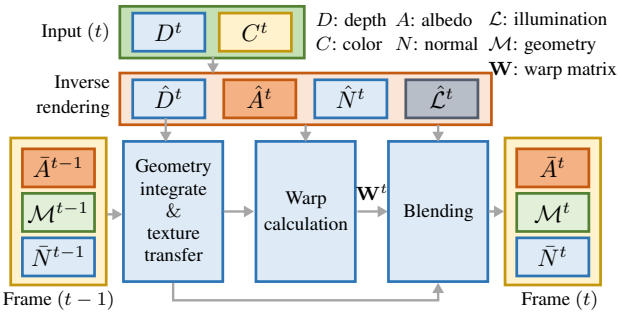


Figure 3: Overview of our normal/albedo texture optimization. Once refined depths, albedos, normals, and illumination are obtained in the current frame, these quantities are used to update albedos and normals in texture space, in addition to geometry in the canonical space of TSDFs.

3.2.1 Progressive Normals Transfer

Once we refine depth via coarse-to-fine inverse rendering, the depth values are integrated to the canonical space of TSDFs, updating geometry at the current frame. However, this geometry update breaks the relationship between the geometry in the canonical space and the associated texture maps as discussed in [18]. We, therefore, transfer the previous texture to the updated geometry by projecting the previous textures to the current one in normal orientations of the updated geometry by means of ray casting, following [18]. Different from the existing texture fusion method [18], our method needs to transfer not only albedos but also normal texture. Since normals are directional attributes, we must account for rotation when transferring normals.

Once we find out correspondence between the previous normals $\mathbf{n}^{t-1}$ and the current ones $\mathbf{n}^t$, we compute rotation matrices of normals $\mathbf{R}$ using Rodrigues' rotation formula as follows: $\mathbf{R} = \mathbf{I} + [\mathbf{v}]_\times + [\mathbf{v}]_\times^2 \frac{1-c}{s^2}$, where $\mathbf{v} = \mathbf{n}^{t-1} \times \mathbf{n}^t$, $[\mathbf{v}]_\times$ is a skew symmetric matrix of $\mathbf{v}$, $c = \mathbf{n}^{t-1} \cdot \mathbf{n}^t$ is the cosine of the angle between two normal vectors, and $s = ||\mathbf{v}||$ the sine of the angle between two normal vectors.

3.2.2 Geometry-aware Texture Mapping

Depth vs. Geometry. In the standard fusion framework, we first need to estimate camera pose per frame using depth information via ICP. Depth frames are aligned with respect to the intermediate geometry in the canonical space. There is an inevitable camera pose error due to the imperfectness

of the progressively estimated geometry and noise in the depth map. It results in an inaccurate projection of input depth values in subsequent frames. This error has been mitigated using the *truncated* signed distance functions in the voxel grid. However, geometry results tend to be smoothed.

Color vs. Geometry. It is worth noting that unavoidable error of geometric registration causes the mismatch between color frames and intermediate geometry (even with the perfectly synchronized camera). When resolving the mismatch problem between color frames and intermediate geometry (accumulated from erroneous depth frames), there is no *direct correspondence* between appearance and geometry.

The matching problem has been tackled by offline optimization, exhaustively creating synthetic textures [2, 13], or is not even explicitly handled, potentially including mismatch errors between texture and geometry [18]. Enhancing 3D geometry using SfS requires *ideal registration* of the captured shading normals to the base geometry. The existing work [18] can align only a group of textures *regardless of geometry* due to lack of the constraint for image-to-geometry alignment. It suffers from wrong projection of textures.

In contrast, we additionally optimize geometric correspondence between normals in the texture space and geometry in the canonical space of TSDFs by formulating an energy optimization problem as follows:

$$E(\mathcal{W}^t) = \lambda_{\text{nwarp}} E_{\text{nwarp}} + \lambda_{\text{cwarp}} E_{\text{cwarp}}, \tag{7}$$

where $\mathcal{W}$ is a set of spatially-varying warp functions, E_{nwarp} is a normal-based energy term, E_{cwarp} is a color-based energy term, and λ_{nwarp} and λ_{cwarp} are corresponding weights, respectively.

In particular, our novel normal-based energy term E_{nwarp} enforces the current texture normal map $\hat{N}$ to the surface normals of the canonical geometry $\tilde{N}$ through the local grid window Ω based on the grid-based warping method [18]. This energy term suppresses the misalignment between two different normals:

$$E_{\text{nwarp}} = \sum_{\mathbf{z}\in\Omega(\mathbf{u})} \omega(\mathbf{z}) \left|\left| \hat{N}(\pi(\mathbf{W}(\mathbf{u})\mathbf{T}\tilde{V}(\mathbf{z}))) - \tilde{N}(\mathbf{z}) \right|\right|^2, \tag{8}$$

where $\mathbf{z}$ is a pixel within the local grid window $\Omega(\mathbf{u})$, $\tilde{V}$ is the 3D vertex map of the canonical geometry in the camera space, $\mathbf{T}$ is the extrinsic camera transformation, π is the camera projection matrix, ω is a Gaussian weight $\omega(\mathbf{z}) = \exp(-||\mathbf{u}-\mathbf{z}||^2/\sigma)$ with a spatial parameter σ to control the regularity of the estimated camera motion, and $\mathbf{W}$ is the unknown spatially-varying warping field in the matrix form at the grid node $\mathbf{u}$. We estimate the camera motions $\mathbf{W}$ at each regular lattice grid $\mathbf{u}$ at each level and then interpolate them via the Gaussian weight ω for every pixel. We empirically found that three levels are sufficient and that the width between grid points is set to 64 pixels. We optimize this energy term $\mathbf{W}$ with weight decay. Note that normals from the canonical space $\tilde{N}$ have less details of normals from SfS. To match the level of detail in optimization, we apply a low-pass filter to high-detail normals from SfS $\hat{N}$.

For color texture, we include a color-based energy term E_{cwarp} to enforce photometric consistency of the current color frame C to the rendering reflected irradiance image $\bar{B}^t$ through the local window Ω following [18]:

$$E_{\text{cwarp}} = \sum_{\mathbf{z}\in\Omega(\mathbf{u})} \omega(\mathbf{z}) \left|\left| Y(C(\pi(\mathbf{W}(\mathbf{u})\mathbf{T}\tilde{V}(\mathbf{z})))) - Y(\dot{B}(\mathbf{z})) \right|\right|^2, \tag{9}$$

where $\dot{B}$ is the reflected irradiance image with the current camera motion by using the newly updated geometry with the transferred normal and albedo texture.

3.2.3 Normal/Albedo Blending

Once we know the spatially-varying warp function $\mathcal{W}^t$, we are ready to blend normals $\hat{N}^t$ with the transferred normals $\dot{N}^t$ at the current frame t to the canonical texture space $\bar{N}^t$. For each texel of a 3D point $\mathbf{p}$ in the canonical space of TSDFs, we evaluate the spatial resolution and registration certainty of each image pixel by computing blending weights following the current methods [18, 3] as follows:

$$\bar{N}^t(\mathbf{p}) = \frac{\Psi^{t-1}(\mathbf{p})\dot{N}^t(\mathbf{p}) + \psi(\mathbf{p})\hat{N}^t(\tilde{\mathbf{u}})}{\Psi^{t-1}(\mathbf{p}) + \psi(\mathbf{p})}, \tag{10}$$

where $\tilde{\mathbf{u}}$ is a pixel that corresponds to point $\mathbf{p}$ via the warping function $\mathbf{W}$ at the current frame t. In addition, weight Ψ is the accumulated weight for normal/albedo blending at the current frame: $\Psi^t(\mathbf{p}) = \min(\Psi^{t-1}(\mathbf{p}) + \psi(\mathbf{p}), \psi_{\text{max}})$, where ψ_{max} is a predefined parameter that controls the upper bound of the blending weight, $\psi(\mathbf{p})$ is the blending weight for a given camera pose. The current blending weight $\psi(\mathbf{p})$ can be computed by accounting for the area size, the camera angle, and occlusion: $\psi(\mathbf{p}) = \psi_{\text{area}}(\mathbf{p}) \cdot \psi_{\text{angle}}(\mathbf{p}) \cdot \psi_{\text{occ}}(\mathbf{p})$, following the weight formulae defined in [18].

4. Results

Implementation Details. We implement our normal fusion method based on the ground of two real-time methods [23, 18]. It runs on a conventional desktop environment equipped with an Intel Core I9-10920X CPU of 3.5 GHz with 64 GB RAM, and an NVIDIA RTX TITAN graphic card. We acquire color and depth frames using a PrimeSense Carmine RGB-D camera of 640×480 resolution. We use a 4 mm voxel size of the voxel grid of TSDFs with 4×4 tile textures for most of our results. For the Fountain [31], Lion, Bricks, and Tomb scenes from Intrinsic3D [20] in our results, we use the image resolution of 640×480. When

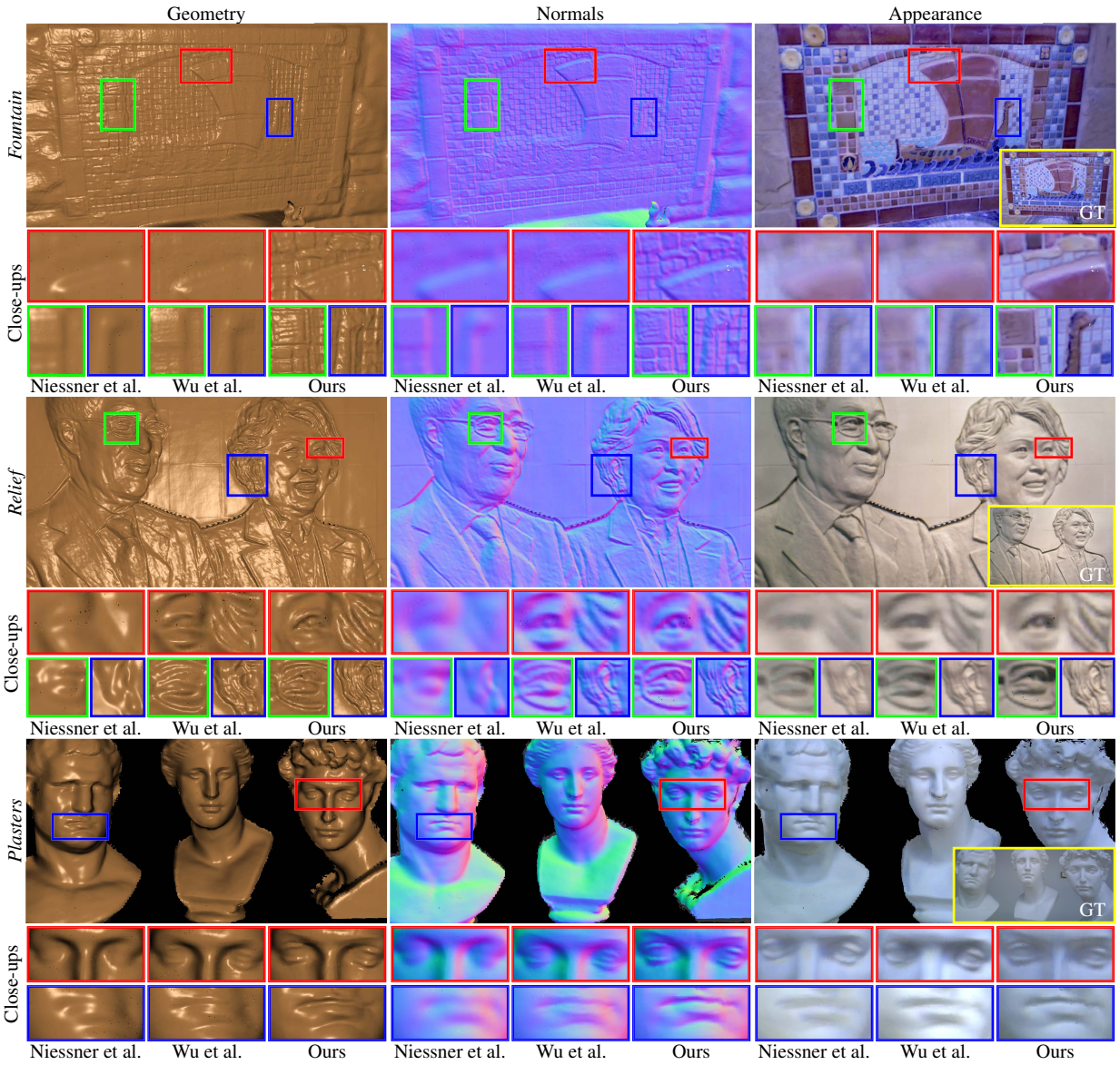


Figure 4: We compare our scanned results with two real-time methods [23, 30]. For fair comparison, we make the camera pixel resolution projected to the voxel grids with the same size. Our method can achieve the high level of geometric details significantly against the state-of-the-art methods thanks to geometry-aware texture mapping of high-resolution normals.

input frames were severely degraded by motion blur, we discarded the input frames using a blurriness metric [5]. To produce our results, our hyperparameters were consistently set to: λ_{ltemp}=200 (except for Lion, λ_{ltemp}=500), λ_{dreg}=100, λ_{dtemp}=100 (except for Fountain, λ_{dtemp}=50), λ_{areg}=0.1, λ_{atemp}=0.01, λ_{nwarp}=0.3, and λ_{cwarp}=1.0.

Qualitative Comparison. We qualitatively evaluate the spatial resolution of scanned geometry by our algorithm, compared with two real-time fusion algorithms [23, 30] with synthetic and real scenes. For fairness of comparison of all three methods, we make the camera pixel resolution projected to the voxel grids with the same size. To this end, we employ a denser voxel resolution (1 mm voxel size) for both Niessner et al. [23] and Wu et al. [30] (our implementation) than that of ours (4 mm voxel size mapped with 4×4 texture tiles). Note that our voxel grid resolution is 4^3 times lower than those of these compared methods in this experiment. For comparisons, we obtain high-resolution mesh model by combining our normal texture with the baseline geometry from the voxel grid using Nehab's method [21]. Again, for fairness, we use subdivision surfaces two times to match our spatial resolution to other methods. Refer to the supplemental video for real-time demo and more results.

Figure 4 compares scanned results. The traditional fusion method [23] accumulates TSDFs to the hashed canonical space directly. Since there is no correction for both

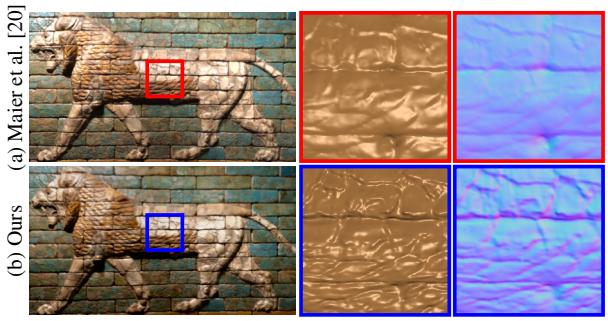


Figure 5: Geometry results of our real-time scanning results are highly competitive to those of an offline SfS-based method [20].

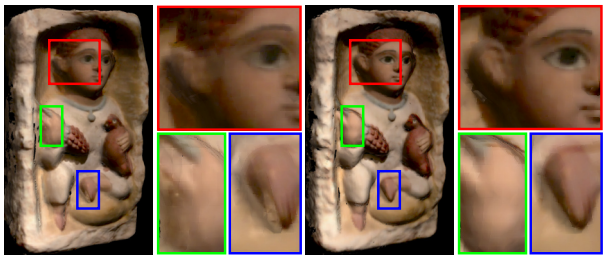


Figure 6: We render two scanned objects using novel-light/view rendering. Compared with the existing texture fusion [18], our method presents improved texture appearance and better shading and shadow thanks to our geometry-aware texture mapping algorithm.

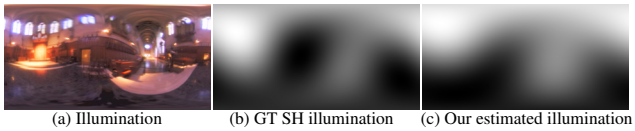


Figure 7: (a) Illumination (a.k.a. Grace) used for creating a synthetic dataset, shown in Figure 8. (b) GT illumination projected to 2nd-order spherical harmonics. (c) Our estimated illumination. Our method successfully estimates low-frequency illumination.

depth noise and ICP registration errors in the traditional method, the reconstructed results look blurred. The SfS-based fusion method [30] improves the geometry through the shading-based refinement of depth maps. However, the shading-based method's results still appear blurry due to disagreement of inter-frame ICPs and misalignment between color and depth frames. Even though the corresponding output voxel sizes are the same, our method can achieve a higher level of details in captured geometry, thanks to the high-resolution normal information in the texture space.

We compare the quality of geometry with an existing offline SfS-based method [20]. As shown in Figure 5, our method can capture high-frequency details of geometry in terms of geometry and normals, and our results are strongly competitive with those of the state-of-the-art offline method. Moreover, our method takes only 58.5 ms for this frame, achieving real-time performance without compromising geometric quality.

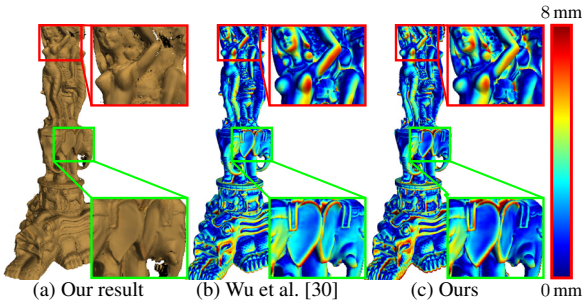


Figure 8: We compare the geometry accuracy of our scanned model with a SfS-based fusion method [30]. (a) presents 3D model scanned by ours. (b) and (c) compare Hausdorff distance errors between the results of two methods and the ground-truth. Our method improves the geometric accuracy against the state-of-the-art method.

We compare the accuracy of our texture mapping with the existing texture fusion method [18] (Figure 6). We rendered two scanned results using a novel light in a novel view. Thanks to our geometry-aware texture mapping that mitigates geometric misalignment between geometry and texture (Section 3.2.2), the accuracy of registered texture maps improves significantly than the state-of-the-art method. Our method can achieve the high-quality appearance of textured objects in terms of shading and shadow.

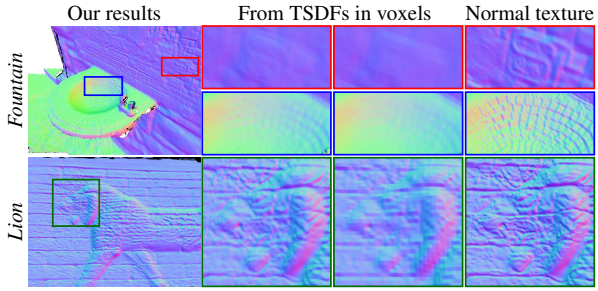
In addition, Figure 7 compares the estimated environment illumination profile against the ground-truth. The GT illumination profile is used for creating the synthetic dataset, shown in Figure 8.

Quantitative Comparison. In order to evaluate the accuracy of the reconstructed geometry, we built a synthetic RGB-D image dataset by rendering an object through a synthetic RGB-D camera model. We add spatial blur to the ground-truth depth rendering and also add Gaussian noise to the depth and color pixel signals. The standard deviations of depth and color images are set to 0.002 and 1, respectively. Also, to simulate asynchronous capture of color and depth frames in real RGB-D cameras, we apply random offsets to translation ($0 - 3$ mm) and rotation ($0 - 0.1°$) of the synthetic RGB-D camera. We make the projected pixel size be the same size by setting the voxel resolution of Wu et al. [30] is 2^2 times higher than that of ours. Figure 8 compares the Hausdorff distance errors of our scanned geometry against the ground-truth with that of the real-time SfS-based fusion method. Over convex- and concave-shaped surfaces, our method improves the geometric accuracy clearly. The average distance error of ours is 3.11 mm, which is smaller than that of the compared method (3.32 mm).

Performance Evaluation. Table 1 presents the computation times for our algorithms with different configura-

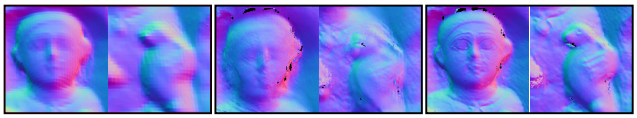
Scene	Texture tile, voxel unit	Geometry integration	Input enhancement	Texture optimization	Total time per frame
Fountain	4×4 tile, 4 mm	8.3 ms	12.1 ms	54.7 ms	75.1 ms
	8×8 tile, 10 mm	3.5 ms	10.1 ms	43.3 ms	56.9 ms
Tomb	4×4 tile, 4 mm	4.1 ms	10.6 ms	27.5 ms	42.2 ms
	8×8 tile, 10 mm	2.9 ms	10.5 ms	27.1 ms	40.5 ms
Lion	4×4 tile, 4 mm	5.0 ms	11.1 ms	42.4 ms	58.5 ms
	8×8 tile, 10 mm	3.2 ms	11.1 ms	37.0 ms	51.3 ms
Bricks	4×4 tile, 4 mm	4.5 ms	10.5 ms	33.7 ms	49.8 ms
	8×8 tile, 10 mm	3.2 ms	9.8 ms	31.6 ms	44.7 ms
Relief	4×4 tile, 4 mm	3.7 ms	10.3 ms	38.1 ms	52.1 ms
	8×8 tile, 10 mm	2.6 ms	9.7 ms	35.7 ms	48.1 ms
Plasters	4×4 tile, 4 mm	3.5 ms	10.0 ms	26.1 ms	39.6 ms
	8×8 tile, 10 mm	2.4 ms	9.5 ms	22.7 ms	34.6 ms

Table 1: Performance of our method with different configurations of voxel grids and tile textures. Our algorithm runs in real-time with average 20 fps. We mainly use the first row of each scene, which is 4×4 texture tile and 4 mm voxel size for our results.



(a) 4 mm (b) 10 mm (c)10 mm, 8×8

Figure 9: Impact of voxel resolution. (a) and (b) present reconstructed normals using two different voxel size: (a) 4 mm, (b) 10 mm. (c) shows normals reconstructed from 10 mm voxels with 8×8 tile texture. Our method achieves high-detail geometry with significantly less memory.



(a) Base geometry (b) Without warp (c) With warp

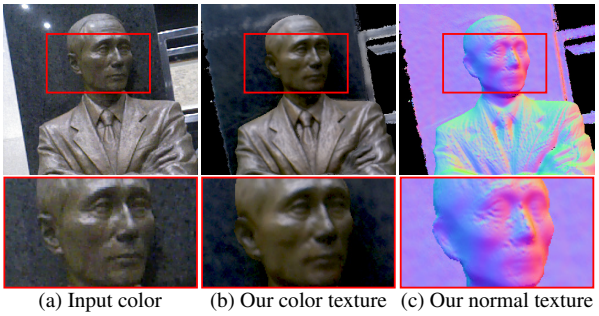
Figure 10: Verification of our geometry aware texture warp. (a) Reconstructed baseline geometry. (b) Normal texture blended without our geometry-aware warping. (c) Normal texture blended with our geometry-aware warping.

tions of voxel grids and title textures when producing result scenes. Our algorithm runs in real-time with average 20 fps. For comparison, the offline method [20] runs in about six hours for the Lion scene while ours is able to update the geometry in real time, capturing the scene in 30 seconds in total. Figure 9 compares the impact of the resolution of texture mapping against those of the baseline geometry from the canonical space of TSDFs.

Impact of Geometry-aware Texture Mapping. Figure 10 presents the impact of our geometry-aware texture warping algorithm. Images (a) shows geometry from the canonical space of TSDFs. They look blurry by the resolution of the voxel grids. Images (b) and (c) compare normal-textured geometry with and without our geometry-aware texture-warp term (Equation (9) in Section 3.2.2). As shown in (a) and (c), our normal texture is well aligned with the baseline geometry in the voxel grid. This high-accuracy registration between texture and baseline geometry allows us to export our high-detail normal texture map as a high-detail geometry model by combing it with the baseline geometry [21].

5. Discussion

Our algorithm is not free from limitations. First, our algorithm proceeds under the assumption of Lambertian surfaces in a static illumination environment like other SfS methods. Thus, it is difficult to obtain accurate results when the illumination changes or objects are highly specular. Figure 11 shows that the accuracy of our inverse rendering method is degraded by specular reflection. Estimating SVBRDF in real-time will be an interesting future work.



(a) Input color (b) Our color texture (c) Our normal texture

Figure 11: Limitations. (a) Input color from sensor. (b) Our color texture result. (c) Our normal texture result.

6. Conclusion

We have presented a real-time 3D scanning method that can capture high-detail geometry and albedo texture using a conventional RGB-D camera. Our main contributions are summarized as follows: First, our multiview SfS algorithm solves the inverse rendering problem in real-time, yielding photometric normals, diffuse albedos, and illumination, in addition to the baseline geometry in voxel grids. Second, our geometry-aware texture mapping registers the texture space to the canonical space of TSDFs with high accuracy, enabling geometric enhancement with high-detail normals and baseline geometry. The level of details of our method is strongly competitive with that of the existing offline SfS-based methods while our method can run in real-time on a conventional desktop computing environment.

Acknowledgment

Min H. Kim acknowledges Samsung Research Funding Center of Samsung Electronics (SRFC-IT2001-04) for developing partial 3D imaging algorithms, in addition to a partial support of Korea NRF grants (2019R1A2C3007229), MSIT/IITP of Korea (2017-0-00072), and MSRA.

References

[1] Thabo Beeler, Derek Bradley, Henning Zimmer, and Markus Gross. Improved reconstruction of deforming surfaces by cancelling ambient occlusion. In *European Conference on Computer Vision*, pages 30–43. Springer, 2012.

[2] Sai Bi, Nima Khademi Kalantari, and Ravi Ramamoorthi. Patch-based optimization for image-based texture mapping. *ACM Trans. Graph.*, 36(4):106–1, 2017.

[3] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 425–432, 2001.

[4] Yan-Pei Cao, Leif Kobbelt, and Shi-Min Hu. Real-time high-accuracy three-dimensional reconstruction with consumer rgb-d cameras. *ACM Transactions on Graphics (TOG)*, 37(5):1–16, 2018.

[5] Frederique Crete, Thierry Dolmiere, Patricia Ladret, and Marina Nicolas. The blur effect: perception and estimation with a new no-reference perceptual blur metric. In *Human vision and electronic imaging XII*, volume 6492, page 64920I. International Society for Optics and Photonics, 2007.

[6] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. pages 303–312, 1996.

[7] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4):1, 2017.

[8] Mingsong Dou, Philip Davidson, Sean Ryan Fanello, Sameh Khamis, Adarsh Kowdle, Christoph Rhemann, Vladimir Tankovich, and Shahram Izadi. Motion2fusion: Real-time volumetric performance capture. *ACM Transactions on Graphics (TOG)*, 36(6):1–16, 2017.

[9] David A Forsyth. Variable-source shading analysis. *International Journal of Computer Vision*, 91(3):280–302, 2011.

[10] Bastian Goldlücke, Mathieu Aubry, Kalin Kolev, and Daniel Cremers. A super-resolution framework for high-accuracy multiview reconstruction. *International journal of computer vision*, 106(2):172–191, 2014.

[11] Kaiwen Guo, Feng Xu, Tao Yu, Xiaoyang Liu, Qionghai Dai, and Yebin Liu. Real-time geometry, albedo, and motion reconstruction using a single RGB-D camera. *ACM Transactions on Graphics (ToG)*, 36(4):1, 2017.

[12] Berthold KP Horn. Obtaining shape from shading information. *The psychology of computer vision*, pages 115–155, 1975.

[13] Jingwei Huang, Justus Thies, Angela Dai, Abhijit Kundu, Chiyu Jiang, Leonidas J Guibas, Matthias Nießner, Thomas Funkhouser, et al. Adversarial texture optimization from rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1559–1568, 2020.

[14] Matthias Innmann, Michael Zollhöfer, Matthias Nießner, Christian Theobalt, and Marc Stamminger. Volumedeform: Real-time volumetric non-rigid reconstruction. In *European Conference on Computer Vision*, pages 362–379. Springer, 2016.

[15] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568, 2011.

[16] Kichang Kim, Akihiko Torii, and Masatoshi Okutomi. Multi-view inverse rendering under arbitrary illumination and albedo. In *European conference on computer vision*, pages 750–767. Springer, 2016.

[17] Fabian Langguth, Kalyan Sunkavalli, Sunil Hadap, and Michael Goesele. Shading-aware multi-view stereo. In *European Conference on Computer Vision*, pages 469–485. Springer, 2016.

[18] Joo Ho Lee, Hyunho Ha, Yue Dong, Xin Tong, and Min H Kim. Texturefusion: High-quality texture acquisition for real-time rgb-d scanning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1272–1280, 2020.

[19] Zhe Liang, Chao Xu, Jing Hu, Yushi Li, and Zhaopeng Meng. Better together: Shading cues and multi-view stereo for reconstruction depth optimization. *IEEE Access*, 8:112348–112356, 2020.

[20] Robert Maier, Kihwan Kim, Daniel Cremers, Jan Kautz, and Matthias Nießner. Intrinsic3d: High-quality 3d reconstruction by joint appearance and geometry optimization with spatially-varying lighting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3114–3122, 2017.

[21] Diego Nehab, Szymon Rusinkiewicz, James Davis, and Ravi Ramamoorthi. Efficiently combining positions and normals for precise 3d geometry. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, page 536–543, New York, NY, USA, 2005. Association for Computing Machinery.

[22] Richard A Newcombe, Dieter Fox, and Steven M Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015.

[23] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):1–11, 2013.

[24] Roy Or-El, Guy Rosman, Aaron Wetzler, Ron Kimmel, and Alfred M Bruckstein. "rgbd-fusion: Real-time high precision depth recovery". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5407–5416, 2015.

[25] Yvain Quéau, Jean Mélou, Jean-Denis Durou, and Daniel Cremers. Dense multi-view 3d-reconstruction without dense correspondences. *arXiv preprint arXiv:1704.00337*, 2017.

[26] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, May 2001.

[27] Daniel Weber, Jan Bender, Markus Schnoes, André Stork, and Dieter Fellner. Efficient gpu data structures and methods to solve sparse linear systems in dynamics applications. In *Computer Graphics Forum*, volume 32, pages 16–26. Wiley Online Library, 2013.

[28] Chenglei Wu, Carsten Stoll, Levi Valgaerts, and Christian Theobalt. On-set performance capture of multiple actors with a stereo camera. *ACM Transactions on Graphics (TOG)*, 32(6):1–11, 2013.

[29] Chenglei Wu, Kiran Varanasi, Yebin Liu, Hans-Peter Seidel, and Christian Theobalt. Shading-based dynamic shape refinement from multi-view video under general illumination. In *2011 International Conference on Computer Vision*, pages 1108–1115. IEEE, 2011.

[30] Chenglei Wu, Michael Zollhöfer, Matthias Nießner, Marc Stamminger, Shahram Izadi, and Christian Theobalt. Real-time shading-based refinement for consumer depth cameras. *ACM Transactions on Graphics (ToG)*, 33(6):1–10, 2014.

[31] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Transactions on Graphics (TOG)*, 33(4):1–10, 2014.

[32] Michael Zollhöfer, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics (TOG)*, 34(4):1–14, 2015.

[33] Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (ToG)*, 33(4):1–12, 2014.