sensors

Article

Surface Reconstruction from Structured Light Images Using Differentiable Rendering

Janus Nørtoft Jensen [1,*,†], Morten Hannemose [1,†], J. Andreas Bærentzen [1], Jakob Wilm [2], Jeppe Revall Frisvad [1] and Anders Bjorholm Dahl [1]

[1] DTU Compute, Technical University of Denmark, 2800 Kgs. Lyngby, Denmark; mohan@dtu.dk (M.H.); janba@dtu.dk (J.A.B.); jerf@dtu.dk (J.R.F.); abda@dtu.dk (A.B.D.)
[2] SDU Robotics, University of Southern Denmark, 5230 Odense, Denmark; jaw@mmmi.sdu.dk
[*] Correspondence: jnje@dtu.dk
[†] These authors contributed equally to this work.

Abstract: When 3D scanning objects, the objective is usually to obtain a continuous surface. However, most surface scanning methods, such as structured light scanning, yield a point cloud. Obtaining a continuous surface from a point cloud requires a subsequent surface reconstruction step, which is directly affected by any error from the computation of the point cloud. In this work, we propose a one-step approach in which we compute the surface directly from structured light images. Our method minimizes the least-squares error between photographs and renderings of a triangle mesh, where the vertex positions of the mesh are the parameters of the minimization problem. To ensure fast iterations during optimization, we use differentiable rendering, which computes images and gradients in a single pass. We present simulation experiments demonstrating that our method for computing a triangle mesh has several advantages over approaches that rely on an intermediate point cloud. Our method can produce accurate reconstructions when initializing the optimization from a sphere. We also show that our method is good at reconstructing sharp edges and that it is robust with respect to image noise. In addition, our method can improve the output from other reconstruction algorithms if we use these for initialization.

Keywords: 3D surface reconstruction; 3D scanning; structured light; differentiable rendering

check for updates

Citation: Jensen, J.N.; Hannemose, M.; Bærentzen, J.A.; Wilm, J.; Frisvad, J.R.; Dahl, A.B. Surface Reconstruction from Structured Light Images Using Differentiable Rendering. *Sensors* **2021**, *21*, 1068. https://doi.org/10.3390/s21041068

Academic Editors: Matteo Poggi and Thomas Moeslund

Received: 21 December 2020
Accepted: 31 January 2021
Published: 4 February 2021

1. Introduction

Structured light 3D scanning of an object can be used to produce a point cloud from which we can reconstruct a triangle mesh. The resulting mesh is a digital representation of the surface of the scanned object. This has many applications, including cultural heritage preservation and industrial quality control [1,2]. For most applications, the accuracy of the recovered surface of the reconstruction is of great importance. Typically, producing point clouds from phase-shifting structured light images is rather cumbersome. It involves determining the phases, unwrapping these, re-sampling the unwrapped phases due to image distortion and rectification, finding point correspondences and finally triangulating these. Afterwards, the point clouds from different sub-scans need to be merged before the final reconstruction of a triangle mesh. During this process of producing point clouds and subsequently a triangle mesh, the image noise propagates non-linearly to affect vertex positions in the reconstructed triangle mesh. We therefore propose that the point cloud and the process of creating it could be skipped, instead reconstructing surfaces directly from image intensities to investigate how that affects the accuracy of the reconstructions. Using vertex positions as model parameters, we minimize the least-squares error between rendered and recorded images to obtain a triangle mesh directly. An example is presented in Figure 1.

Our method has three major implications: (1) explicit point triangulation from image correspondences is no longer needed; (2) we understand how noise in the image data affects the final reconstruction; and (3) the reconstruction is done for all image data simultaneously.

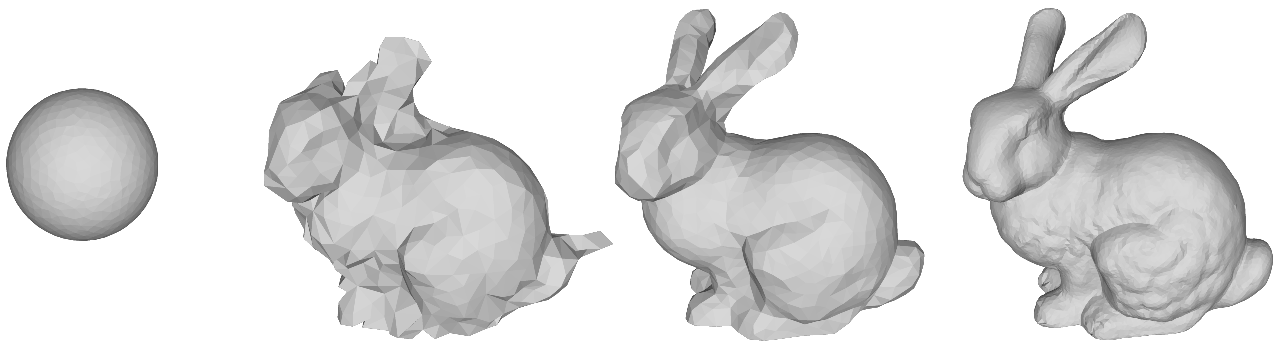


Figure 1. Starting from a sphere, our method reconstructs the Stanford Bunny [3] from images with low levels of noise. From left to right: Initial mesh (sphere), after 50 iterations, after 750 iterations and the converged result (1952 iterations). The final reconstruction has 13,780 vertices and a volume error (Δ_V) of 0.30% when compared to the ground truth (depicted in Figure 3j).

Similar approaches have been used for multi-view passive stereo but to the best of our knowledge not for structured light reconstructions. In structured light, dense correspondences can be established, usually yielding much higher accuracy in weakly textured areas.

In many reconstruction methods, the raw image data are not considered during the reconstruction [4]. The reconstruction is instead done using unstructured point clouds that have been constructed from the image data. This means that noise due to imaging and point matching processes is not modeled in the reconstruction at all, and this noise is then non-linearly propagated through to the end result. With our approach, the only step in the method in which an error is minimized is in the image intensity domain. This enables us to minimize a more meaningful error, which is especially important in applications with a low signal-to-noise ratio of the reflected light; e.g., when scanning highly specular objects or when using short exposure times for fast acquisition.

2. Related Work

Suppose we know the configuration of the light and camera in a vision setup and the reflectance properties of the imaged object. Obtaining the shape of the object based on its shading in an acquired image is then referred to as the shape-from-shading problem [5]. The original shape-from-shading method by Horn [5,6] used so-called characteristic curves to describe the observed shape. The method was based on illumination from point-like sources, and the shading would then only allow the estimation of the gradient along a path. This was the reason for using a collection of curves to describe the object shape.

Curves are inconvenient in the sense that they require stitching to become a full surface description. One way to fit a mesh instead of curves is to use an optimization technique that does not require gradients. This has been done for a rectangular mesh using simulated annealing and simplex search [7]. Gradients are however preferable to ease the optimization problem. For triangle meshes, an approach has been developed based on image gradients [8]. Unfortunately, the shape can be difficult to recover from image gradients due to the color variance caused by normal variations in the surface. To keep gradient-based optimization while developing a method that is more robust to surface reflectance deviating from an assumption of a specific shading model, we use structured light with a differentiable pattern.

The combination of shape from shading with a structured light approach, such as phase-shifting, improves the performance of the shape estimation [9,10] and enables the simultaneous acquisition of shape and object color (diffuse reflectance) [11]. We do not include the estimation of spatially varying reflectance in this study, but we note that this is an option. Only a height field was reconstructed in this previous work. We take this concept one step further and reconstruct a closed 3D object as in the work of Zhang and Seitz [8] but also exploit structured light.

The mesh-based reconstruction method of Zhang and Seitz [8] was improved by Isidoro and Sclaroff [12,13] through the creation of a better initial mesh and by Yu et al. [14,15] using a model that more accurately models the physical reflectance. Another option is to use multiview stereo to acquire a good initial guess and then refine the mesh using a shape-from-shading approach [16]. Our method can be used similarly with the innovation of using structured light to improve the robustness of the mesh refinement.

The use of structured light has become a reliable technique for point-based 3D reconstruction [17] but has, to the best of our knowledge, not previously been tested in mesh-based 3D reconstruction. Our motivation is to have the benefits of a mesh-based technique. An important benefit is that the connectivity between points (vertices) is retained throughout the geometric refinement process.

The recent differentiable renderer from Loubet et al. [18] uses ray tracing and reverse mode automatic differentiation. The use of such a framework for mesh refinement is an option. However, each rendering is quite computationally demanding, so the optimization would have a significant run time. Liu et al. [19] introduced the soft rasterizer, which is a faster differentiable renderer based on a smoothed version of rasterization. This has shown promising results in other mesh reconstruction tasks, but to make it able to render the structured light of a projector is nontrivial.

A different but common approach to reconstructing a shape from structured light images is to first reconstruct a point cloud and then use one of the many methods for the reconstruction of meshes from point clouds [4,20]. Poisson reconstruction [21], which has seen several recent improvements [22], is among the most popular of these. Poisson reconstruction is a volumetric reconstruction method [4]. This means that the reconstruction first finds a smooth characteristic function and then extracts the polygon mesh using a method for iso-contouring [23]. Unfortunately, this means that there is no simple relationship between the final mesh vertices and the points in the intermediate point cloud, making the reconstruction of sharp features challenging, even if the screened variant [24] improved precision. To recapture sharp features, one might apply one of several anisotropic mesh-smoothing methods [25–28]. While these are effective, they operate solely on the mesh and do not make use of the original images. In particular, this can lead to sharp edges forming without a basis in data. Thus, the point cloud-based pipeline generally does not refer back to the original image data after the point cloud has been reconstructed, meaning that the quantification of the error in the output object is not intrinsic to this pipeline. In comparison, our differentiable rendering-based method optimizes the output mesh based on an energy that directly uses the structured light images.

Another approach is based on the additional information provided by a camera that includes a depth channel (RGB-D). Reconstruction techniques used with depth cameras are, however, either voxel-based or point-based according to a recent survey [29]. Some depth cameras retrieve depth images using structured infrared light. Thus, our technique could be useful for mesh-based reconstruction with a sensor of this kind. Finally, one can rely on the ability of a deep neural network to reconstruct a 3D shape from a single image (or multiple images) [30,31]. In this area, the conclusion is interestingly that a mesh-based method can generate 3D shapes with higher quality than voxel-based and point-based methods. However, while better with a single image, the deep learning-based methods currently lack the universality and adaptability of more traditional methods. Thus, in this work, we suggest a mesh-based method that is not based on deep learning.

3. Method

Our method fits a surface to a set of structured light images by minimizing the squared differences between rendered images and real images. We parameterize the object surface by a triangular mesh, and the parameters we optimize are thus the vertex positions $\mathbf{v}$. The real images are captured with structured light phase-shifting from multiple camera-projector positions. We denote these images $\mathbf{I}_{c,p}$, where $c \in \mathcal{C}$ is the index of the camera–projector pair and $p \in \mathcal{P}$ is the index of the projected pattern. We render images $\widetilde{\mathbf{I}}_{c,p}(\mathbf{v})$ of our parameterized surface from the same camera–projector positions and find the optimal vertex positions by solving the following minimization problem:

$$\underset{\mathbf{v}}{\operatorname{argmin}}\, \mathcal{L}(\mathbf{v}) = \underset{\mathbf{v}}{\operatorname{argmin}} \sum_{c \in \mathcal{C}} \sum_{p \in \mathcal{P}} \left\| \mathbf{I}_{c,p} - \widetilde{\mathbf{I}}_{c,p}(\mathbf{v}) \right\|_F^2, \tag{1}$$

where $\| \cdot \|_F$ is the Frobenius norm; i.e., we minimize the sum of squared differences over all pixels for all patterns and all camera–projector pairs.

We generate our structured light images by projecting sinusoidal patterns such that the intensity of each column of the projector is given by

$$\frac{1}{2} + \frac{1}{2} \sin\left(2\pi n_p x + \phi_p\right), \tag{2}$$

where x is the x-coordinate of the projector normalized to $[0,1]$, n_p is the frequency (number of periods) of the pattern and ϕ_p is a phase-shift. With structured light images, such as phase-shifted images made from Equation (2), the goal is usually to find the x-coordinate of the projector, which subsequently can be used for triangulation.

Our method is not specific to phase-shifting patterns; however, we use differentiable patterns to make the minimization problem tractable.

3.1. Rendering Images

To solve the minimization problem in Equation (1), we need to render the images $\widetilde{\mathbf{I}}_{c,p}(\mathbf{v})$ in each iteration. We want to adequately reproduce the structured light images that would have been obtained if our current parameterized surface was a real object. We achieve this by simulating the structured light process as seen from the viewpoint of the camera. We render the images using the formula

$$\widetilde{\mathbf{I}}_{c,p}(\mathbf{v}) = \mathbf{A}_c \sin\left(2\pi n_p \widetilde{\mathbf{X}}_c(\mathbf{v}) + \phi_p\right) + \mathbf{B}_c. \tag{3}$$

Here, $\sin(\cdot)$ is the element-wise application of the sine function, n_p is the number of periods as in Equation (2), ϕ_p is the phase-shift for the pth pattern and $\widetilde{\mathbf{X}}_c(\mathbf{v})$ contains the x-coordinates of the projector in the $[0,1]$ range for each pixel. We use the x-coordinate of the projector as the projector is offset from the camera along its x-axis. The matrices $\mathbf{A}_c$ and $\mathbf{B}_c$ are amplitudes and biases that are estimated from the ground truth images by fitting sinusoids at each pixel location. We find the elements of $\widetilde{\mathbf{X}}_c(\mathbf{v})$ by tracing a ray from the camera through the center of each pixel and projecting the point where it intersects the surface back to the projector. As we model the projector as a pinhole camera, the point is projected to the projector as follows:

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \mathbf{P} \begin{bmatrix} \mathbf{r} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ 1 \end{bmatrix}, \tag{4}$$

where $\mathbf{P}$ is the projection matrix of the projector, $\mathbf{p}_i$ is the ith column of $\mathbf{P}$ and $\mathbf{r}$ is the 3D point where the ray intersects the triangle face. The i,jth element of $\widetilde{X}_c$ is then given by

$$\widetilde{X}_c^{i,j} = \frac{q_1}{q_3}. \tag{5}$$

If the ray does not intersect the surface, we treat the pixel as background and set $\widetilde{X}_c^{i,j}(\mathbf{v}) := X_c^{i,j}$ such that the corresponding term in the loss $\mathcal{L}(\mathbf{v})$ will be zero.

Note that $2\mathbf{A}_c$ is the proportion of projector light that is reflected into the camera and $\mathbf{B}_c - \mathbf{A}_c$ is the amount of global light. We estimate these to allow the renderings to resemble the true images better. Fortunately, we need only estimate them once for each viewpoint, as we can then use these estimates repeatedly in each iteration of the optimization.

3.2. Optimizing the Surface

We use gradient descent to solve the optimization problem in Equation (1). In order to do this, we need the gradient of $\mathcal{L}(\mathbf{v})$, which in turn depends on the gradient of the elements in $\widetilde{\mathbf{X}}_c(\mathbf{v})$. Recall that each of these elements is computed by tracing a single ray from the camera to the surface of the object. The gradient of $\widetilde{X}_c^{i,j}$ will therefore only have contributions from the vertices spanning the triangle face that intersects the ray. We can compute the derivative for one of these three vertices ($\mathbf{p}_a$) as follows:

$$\frac{\partial \widetilde{X}_c^{i,j}}{\partial \mathbf{p}_a} = \left(\frac{\mathbf{p}_1 - \widetilde{X}_c^{i,j} \mathbf{p}_3}{q_3} \cdot \mathbf{d} \right) \frac{\lambda_a \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}, \tag{6}$$

where $\mathbf{d}$ is the ray direction, $\mathbf{n}$ is the normal of the face, and λ_a is the barycentric coordinate corresponding to $\mathbf{p}_a$. The equations for the remaining two vertices, $\mathbf{p}_b$ and $\mathbf{p}_c$, use λ_b and λ_c but are otherwise identical. For a derivation of Equation (6), see Appendix A.

As mentioned, we use gradient descent to update the vertex positions; that is,

$$\mathbf{v}_{i+1} = \mathbf{v}_i - \alpha_i \nabla \mathcal{L}(\mathbf{v}_\mathbf{i}), \tag{7}$$

where $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ are the vertex positions in the ith and $(i+1)$th iterations, respectively. To choose the step-length α_i, we use a simple backtracking line-search [32] to choose

$$\alpha_i = \frac{1}{2^n} \alpha, \tag{8}$$

where α is a fixed constant and n is the smallest non-negative integer such that $\mathcal{L}(\mathbf{v}_{i+1}) < \mathcal{L}(\mathbf{v}_i)$ when doing the update.

3.3. Initializing the Optimization

We have so far described how the iterative part of the optimization problem works, but this is only one half of the problem. A good initial guess is extremely important to ensure convergence. In the next two sections, we introduce two possible ways to obtain an initial guess for the minimization problem in Equation (1).

3.3.1. Using Other Reconstruction Methods

One way of getting a good initial guess is by using a reconstruction found via another reconstruction method. In this way, our method can be seen as a post-processing step that tries to adjust the reconstruction to fit to the original image data better. In some of our experiments, we have used Screened Poisson Reconstructions [24] at various depths as initialization. This method is used as it is able to produce decent reconstructions even from a very noisy point cloud and therefore yields a good initial guess. However, as Poisson reconstructions often produce undesired triangulations, we have found it to be beneficial to remesh the mesh before starting the optimization.

3.3.2. Using a Simple Shape

Another way of obtaining an initial guess is to use a simple shape such as a sphere. However, as the objective in Equation (1) has many high-frequency sinusoids, the optimization is prone to ending up in local minima when the initial guess is far from a global minimum. We therefore propose solving a related, but simpler, minimization problem and to use the solution as an initial guess for our problem in Equation (1). The simpler problem is given by

$$\underset{\mathbf{v}}{\operatorname{argmin}} \sum_{c \in \mathcal{C}} \left\| \mathbf{X}_c - \widetilde{\mathbf{X}}_c(\mathbf{v}) \right\|_F^2, \tag{9}$$

where $\mathbf{X}_c$ denotes the x-coordinates of the projector from the ground truth images. The method used to recover $\mathbf{X}_c$ depends on the patterns displayed, but for two sets of phase-shifted patterns, the heterodyne principle can be used [33].

To make the initial problem simpler, we start with a mesh that has few vertices and gradually increase the number of vertices by remeshing. This enables us to use a simple shape, e.g., a sphere, as our initial mesh.

3.3.3. Remeshing

As described in Sections 3.3.1 and 3.3.2, we use a remeshing algorithm. The algorithm we use is adapted from the Python geometry processing library Pymesh [34]. The first step is to remove any degenerate triangles and to split all edges that are longer than ℓ. Then, the algorithm repeatedly collapses any edges that are shorter than ℓ and splits any obtuse triangles where the angle is greater than 150°. This collapsing and splitting is repeated until the mesh no longer changes. Finally, any self-intersections are removed, the mesh is replaced by the outer hull of the mesh, obtuse triangles with an angle above 179° are split and any isolated vertices are removed.

4. Experiments

To demonstrate the usefulness of our method, we carried out a few experiments that we briefly introduce here. First, we showed that our method can reconstruct an object starting from a sphere; see Figure 1. Secondly, we reconstructed three different objects at multiple levels of noise; see Figure 2. Finally, we compared our method against a Poisson reconstruction at two levels of noise and showed that our method could reconstruct sharp edges.

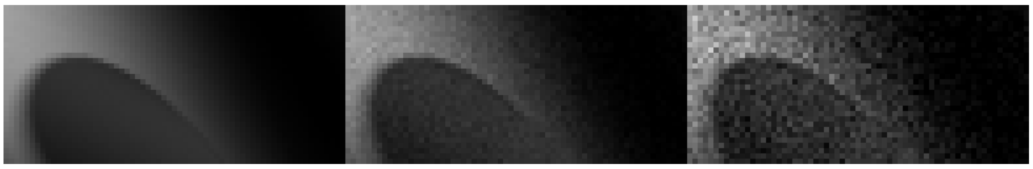


Figure 2. Crop of image shown with varying levels of noise. From left to right: $k = 1$, $k = 10^2$, $k = 10^3$.

4.1. Generating Ground Truth Images

We performed all experiments using synthetic ground truth images in order to have access to the ground truth shape of the mesh for comparison. Our ground truth images were made by projecting two sets of phase-shifted patterns with fifteen and sixteen periods, respectively, with sixteen shifts of the first pattern and eight shifts of the second, such that

$$n_p = \begin{cases} 15 & p \in [1, 2, \ldots, 16] \\ 16 & p \in [17, 18, \ldots, 24] \end{cases} \tag{10}$$

and

$$\phi_p = \begin{cases} 2\pi p \frac{1}{16} & p \in [1, 2, \ldots, 16] \\ 2\pi(p - 16)\frac{1}{8} & p \in [17, 18, \ldots, 24]. \end{cases} \tag{11}$$

4.1.1. Rendering

We rendered the ground truth images using ray tracing with 100 samples per pixel for anti-aliasing, and we used the Lambertian reflectance model to describe the optical properties of our objects of interest.

4.1.2. Noise

As these ground truth images were noise-free, we added noise to make the images more realistic. For this, we modeled the noise of a pixel with intensity x by a Gaussian distribution with mean x and the following variance:

$$\sigma^2 = \sigma_r^2 + x\sigma_p^2, \tag{12}$$

where the first term σ_r describes the signal-independent sensor read-out noise and the second term $x\sigma_p$ describes the signal-dependent shot noise. We chose σ_r and σ_p by using the noise levels from a baseline camera [35]. Our noise levels were then defined as multiples of this baseline noise level, controlled by k as follows:

$$\sigma^2(x,k) = k(4.5 \cdot 10^{-7} + x \cdot 2 \cdot 10^{-5}), \tag{13}$$

such that $k = 1$ gives the noise levels of a baseline camera for $x \in [0,1]$. After adding noise, we clamped pixel values to the $[0,1]$ range. Examples of images for different values of k are presented in Figure 2.

4.2. Quantitative Evaluation

We quantitatively evaluated the performance of our method using a metric Δ_V closely related to one minus the volumetric intersection over union (IoU):

$$\Delta_V = \frac{|(S \setminus \tilde{S}) \cup (\tilde{S} \setminus S)|}{|S|}, \tag{14}$$

where S and $\tilde{S}$ are the ground truth and reconstruction considered as solids, and $|\cdot|$ is the volume of a solid. Volumetric IoU has recently been used for 3D object detection [36] and for comparing volume-based surface representations [37–40]. As demonstrated by Kato et al. [40], the metric is equally useful for a mesh-based surface representation like the one we use.

4.3. Experiment Details

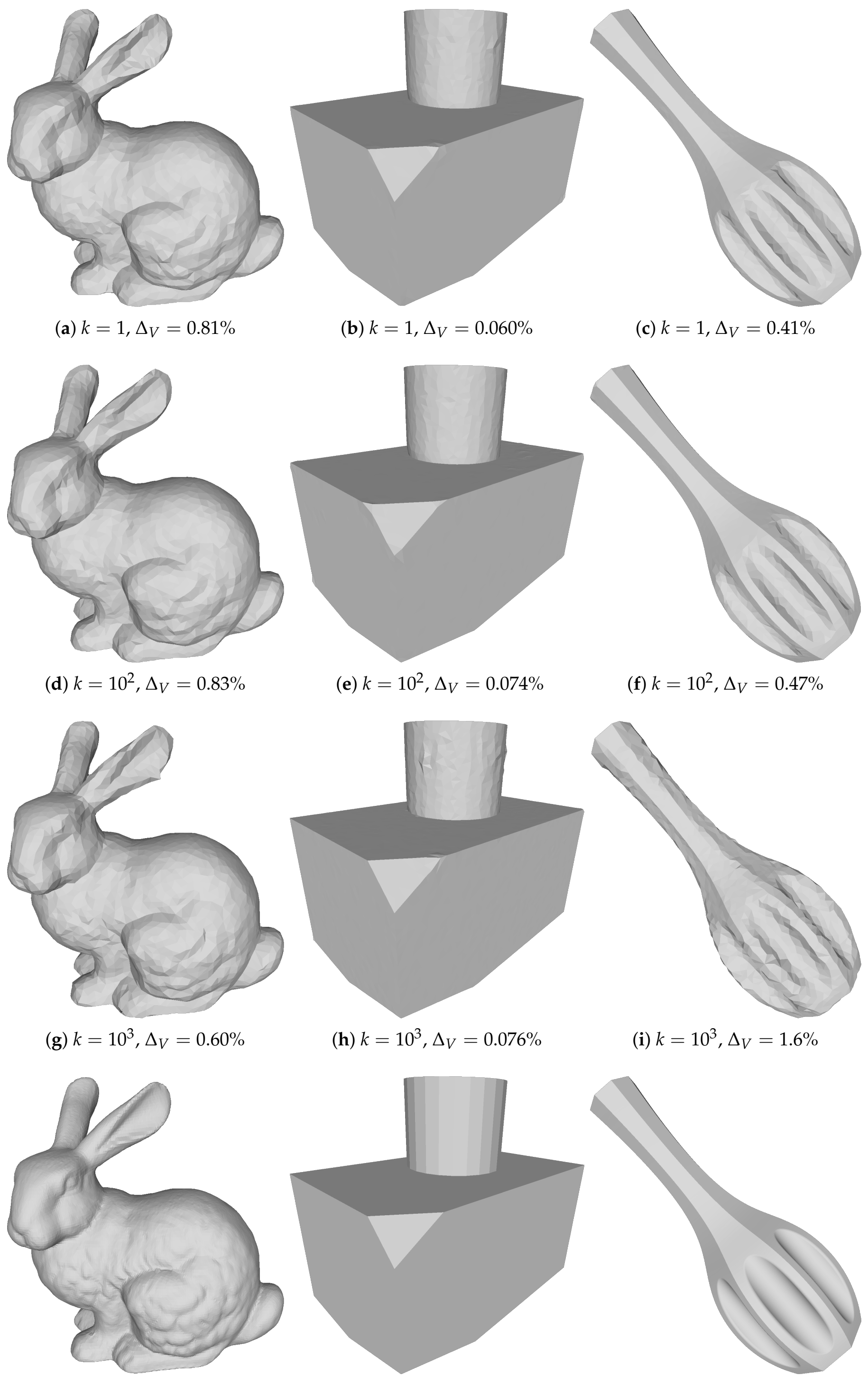
We evaluated our method on three different shapes: the Stanford Bunny [3], a combination of a cylinder and a box with various truncated corners and a dandelion vase [41]. These shapes are presented in the bottom row of Figure 3. When starting from a simple shape as described in Section 3.3.2, we used the same sphere across all our experiments. When optimizing the simpler problem in Equation (9), we performed remeshing for every 25th iteration. At each remeshing step, we decreased the target edge length, which yielded meshes with an increasingly fine resolution. For the ith remeshing step, we set the target to

$$\ell_i = 0.99^i \cdot 0.025 \cdot d_{BB}, \tag{15}$$

where d_{BB} is the largest diagonal of the bounding box of the current mesh. To be able to solve the problem in Equation (9), we estimated $\mathbf{X}_c$ using the heterodyne principle [33]. In all our experiments, we used 60 camera–projector positions organized in three circles, with 20 cameras in each circle, to mimic a structured light scanner with the object placed in three different poses on a turntable rotating 18° between each image. One of these circles is visualized in Figure 4. The camera resolution for all renderings was 1920×1080 pixels. Our method used approximately 1 s per iteration depending on the number of vertices and the resolution of the images.

In Figure 1, we show how our method was able to reconstruct the Stanford Bunny starting from a sphere. The reconstruction was done for $k = 1$. After finishing the optimization based on Equation (9), we remeshed the result by halving the target edge length ℓ from the last remeshing step to get a mesh with an even finer resolution. The two in-progress images shown are during the initial optimization directly on $\mathbf{X}_c$. The final reconstruction contained many of the fine details of the true bunny (Figure 3j).

To examine how our method is affected by noise, we reconstructed all three objects starting from a sphere with varying levels of image noise ($k \in [1, 10^2, 10^3]$), which we show in Figure 3. We see that our method was largely unaffected by noise and was still able to reconstruct the shape. Figure 5 shows the corresponding results when using a screened Poisson reconstruction.



(**a**) $k = 1$, $\Delta_V = 0.81\%$ (**b**) $k = 1$, $\Delta_V = 0.060\%$ (**c**) $k = 1$, $\Delta_V = 0.41\%$

(**d**) $k = 10^2$, $\Delta_V = 0.83\%$ (**e**) $k = 10^2$, $\Delta_V = 0.074\%$ (**f**) $k = 10^2$, $\Delta_V = 0.47\%$

(**g**) $k = 10^3$, $\Delta_V = 0.60\%$ (**h**) $k = 10^3$, $\Delta_V = 0.076\%$ (**i**) $k = 10^3$, $\Delta_V = 1.6\%$

(**j**) Ground truth (**k**) Ground truth (**l**) Ground truth

Figure 3. Reconstructions made with our method on three different objects, for increasing levels of noise k.

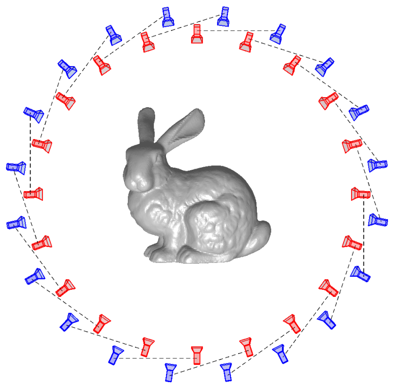


Figure 4. Visualization of one of the circles from our camera–projector setup. In all experiments, we used three circles with 20 cameras each. Red indicates a camera and blue indicates a projector. The dotted lines show which cameras and projectors belong together.

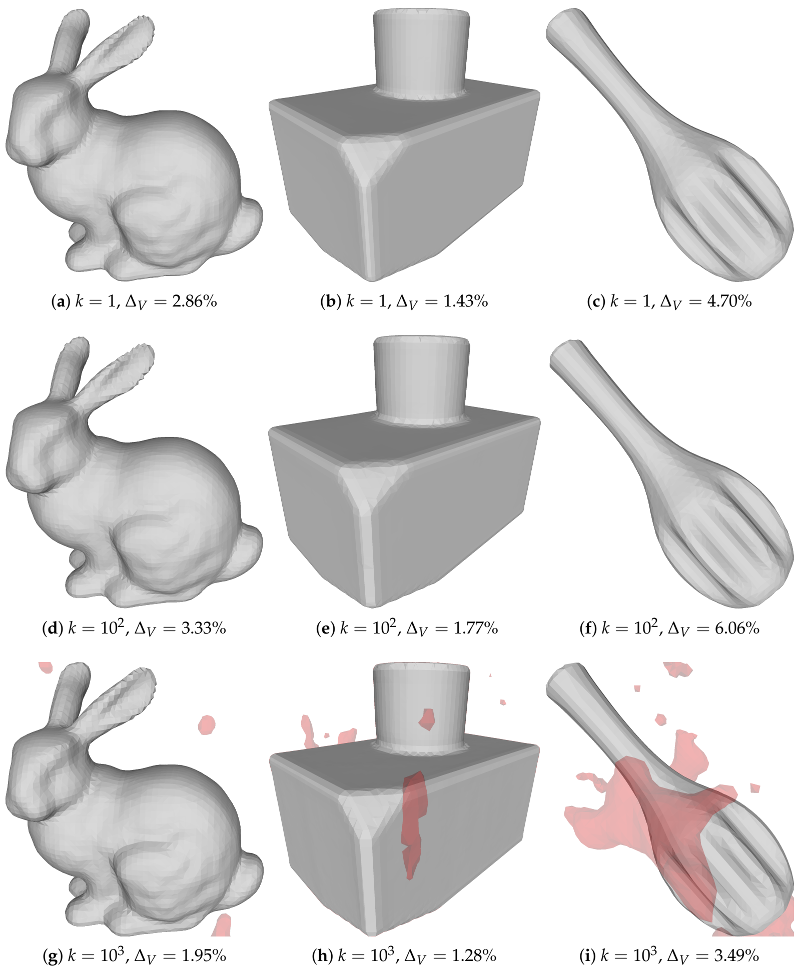
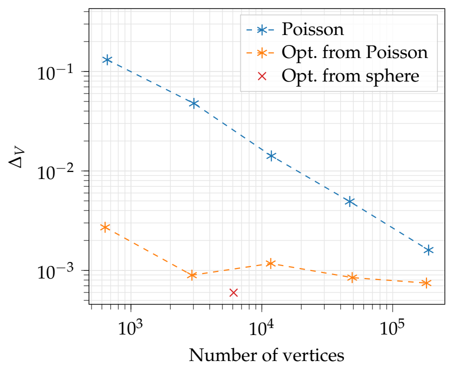

(a) $k = 1$, $\Delta_V = 2.86\%$
(b) $k = 1$, $\Delta_V = 1.43\%$
(c) $k = 1$, $\Delta_V = 4.70\%$
(d) $k = 10^2$, $\Delta_V = 3.33\%$
(e) $k = 10^2$, $\Delta_V = 1.77\%$
(f) $k = 10^2$, $\Delta_V = 6.06\%$
(g) $k = 10^3$, $\Delta_V = 1.95\%$
(h) $k = 10^3$, $\Delta_V = 1.28\%$
(i) $k = 10^3$, $\Delta_V = 3.49\%$

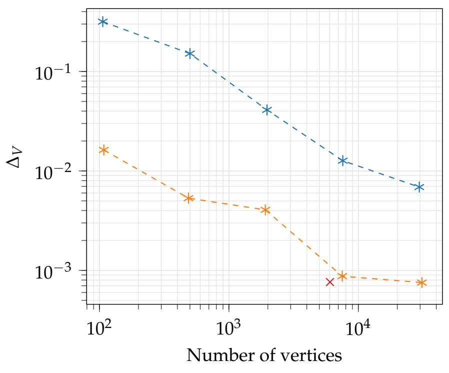
Figure 5. Poisson reconstructions of three different objects, for increasing levels of noise k. The reconstructions for $k = 10^3$ had a lower Δ_V than the reconstructions at lower noise levels, as these reconstructions were done at an increased octree depth to ensure that the reconstructions had more vertices than the corresponding reconstructions in Figure 3. Red areas do not belong to the largest connected component and have thus not been used in the computation of Δ_V.

Finally, we compared our method against Screened Poisson reconstruction, as shown in Figure 6, for multiple depths of the reconstruction. The point cloud used for the Poisson reconstruction for $k = 1$ contained 22 million points, and the point cloud for $k = 10^3$ contained 17 million points. Our method consistently achieved a lower error than the

Poisson reconstruction. The meshes from some of these data points are shown in Figure 7 along with a Robust Implicit Moving Least Squares (RIMLS) reconstruction. Figure 7 also shows that our method was able to reconstruct sharp features even in the presence of large amounts of noise compared to other feature-preserving reconstruction methods such as RIMLS [42]. The result from RIMLS was very poor in this case due to the large amount of noise in the point cloud.

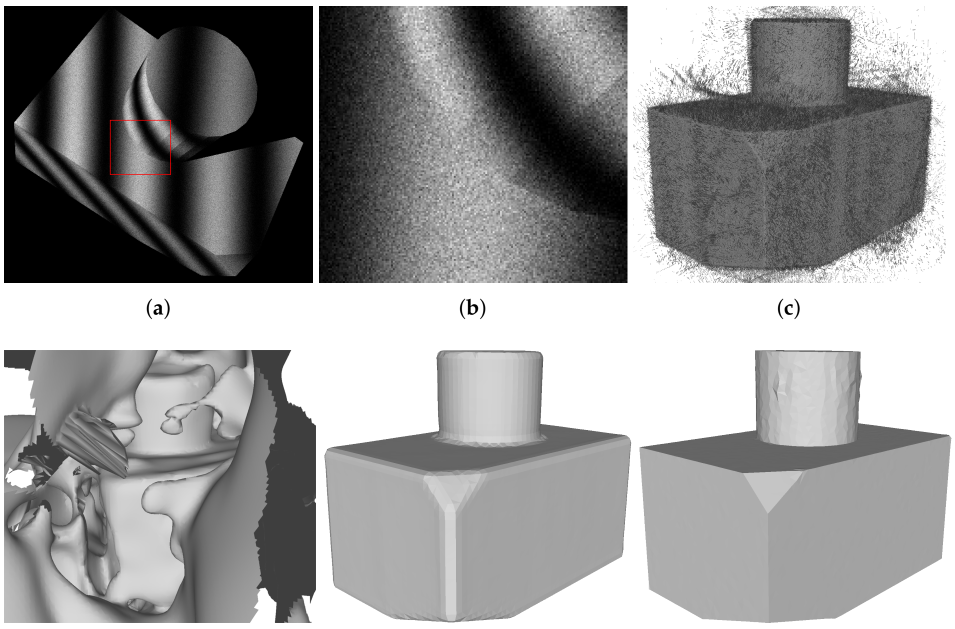


(**a**) Low image noise ($k = 1$)



(**b**) High image noise ($k = 10^3$)

Figure 6. Comparison showing Δ_V as a function of the number of vertices in the mesh for reconstructions of a box with a cylinder. Poisson is the Poisson reconstruction, where only the connected component with the largest volume has been kept, as the Poisson reconstruction sometimes produces multiple connected components for high levels of noise. Opt. from Poisson and from sphere presents our method using the initial guesses described in Section 3.3. The Poisson reconstruction was done for spatial octree depths of 4 to 8.


(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 7. Our method was able to reconstruct meshes with sharp edges from very noisy images ($k = 10^3$). Our method had a lower error (Δ_V) than a Screened Poisson reconstruction with a comparable number of vertices (spatial octree depth 7). The error of the Robust Implicit Moving Least Squares (RIMLS) reconstruction could not be evaluated. (**a**) A noisy input image. (**b**) Crop of noisy image. (**c**) Point cloud. (**d**) RIMLS reconstruction [42]. (**e**) Poisson reconstruction [24] with 7064 vertices. $\Delta_V = 1.28\%$. (**f**) our method with 6038 vertices. $\Delta_V = 0.08\%$.

5. Discussion

Our method has the same implicit assumption about the appearance of the object that is necessary for structured light. This assumption is that the amount of global light in each pixel, i.e., the light that is not directly reflected, is the same for all patterns. This is approximately true for high-frequency sinusoidal patterns [43].

Although we used the Lambertian reflectance model to render our synthetic data, we do not expect this to be a limitation of our method. It does not rely on the assumption of Lambertian reflectance due to the use of structured light, and therefore does not rely on estimating a texture map of the object, neither implicitly nor explicitly.

In our experiments, we had the advantage of knowing the camera and projector positions exactly, which would not be the case when working with real data. However, this can be remedied by including the positions of these as parameters in the optimization problem. Implementing this is relatively easy as our method only requires first-order derivatives, which can be computed analytically using an approach similar to that used in Appendix A. Additionally, we do not need to purely rely on our estimates of $\mathbf{A}_c$, $\mathbf{B}_c$, as these can also be part of the optimized parameters. However, we expect both $\mathbf{A}_c$, $\mathbf{B}_c$, and the camera–projector positions to be known quite accurately and would suggest allowing them to be part of the optimization only once the original optimization problem has converged.

In order to solve the optimization problem, we computed the derivative of our loss function, which involved the derivative of our rendering. This is potentially problematic as our rendering is not differentiable at depth discontinuities; i.e., where non-neighbouring faces of the mesh are bordering each other in the image space. This could, e.g., be the ear and body of the Stanford Bunny. Our method will in this case experience aliasing error in the renderings and derivatives. However, since the faces are observed from multiple views simultaneously, there is often another view in which the same edges are observed with continuous depth. The problem is thus mitigated, and since it occurs only for a small percentage of the pixels for each iteration, it was not a problem in our optimization experiments. Although the optimization problem in Equation (9) is much less prone to ending up in a local minimum, it is still not guaranteed to find the global optimum. However, as the graphs in Figure 6 demonstrate, our method is able to find a good local minimum.

The choice of using a mesh as the surface representation to optimize has some advantages. It is very efficient to compute the gradient of our loss function for a mesh, as each pixel only influences a constant number of elements in the gradient, which makes it suitable for parallel implementation on a GPU (Graphics Processing Unit).

Limitations

A disadvantage of our method is that it is not able to handle topology changes. In practice, this means that the initial mesh must have the same topology as the true object, but even then it is possible that the initial shape would need to be closer to the final shape for objects containing holes. Whether starting with a simple shape with the correct topology is sufficient is yet to be determined. However, this problem is avoided when using a Poisson reconstruction as the initial guess.

While our method is quite robust to image noise, there will be extreme situations where the structured light decoding scheme will fail in so many pixels that either of our initialization methods will fail. This is because both of our initial guesses to the method rely on having access to $\tilde{\mathbf{X}}_c$, which in the extreme case only contains noise. The limit to how much noise we can handle is yet to be determined, but as seen in Figures 2 and 3, we are able to handle substantial amounts of image noise.

We can control the complexity of our mesh by adjusting the target number of vertices. This is a feature and in some cases also a limitation. The maximum target number of vertices is bounded by the amount of image noise. When the target number of vertices is increased, each face in the mesh becomes smaller and therefore is more affected by noise. Being able to set the number of vertices in the mesh, however, also allows the user to determine how many vertices are required to describe the given geometry. The number of vertices also implicitly controls the amount of smoothing our method does, which enables control over the trade-off between bias and variance.

Areas of the object that have not been observed by any of the cameras are not affected by any gradient updates and therefore retain the shape of the initial guess. This means that full coverage of the object is necessary to obtain an accurate reconstruction. However, our method is naturally able to identify which areas are not seen by any camera in the final solution, which implies that our method can be extended to remove or smooth these areas depending on the requirements.

6. Conclusions

Our primary contribution in this work is a novel model for computing a surface mesh directly from structured light images without the need for an intermediate point cloud. With this model, we have shown that the direct computation of a triangle mesh gives high accuracy and is particularly good at reconstructing sharp features such as corners and edges, which are smoothed out using the two-step Screened Poisson reconstruction that we used for comparison. Further, we have obtained very high robustness to noise by optimizing the vertex positions directly from the images. Finally, our approach completely avoids partial scans that must subsequently be aligned, because the optimization is done for all images at once. This demonstrates the advantage of directly reconstructing a surface from structured light images using differentiable rendering. Our method opens up many interesting avenues of future work, including bringing it from the realm of synthetic data to real-world data and testing the limits with respect to image noise. We conjecture that our method may be well-suited for scanning highly specular objects and other objects with a low signal-to-noise-ratio where aggregating information from many views is beneficial.

Author Contributions: Conceptualization, J.W., J.R.F. and A.B.D.; methodology, J.N.J., M.H. and J.A.B.; software, J.N.J. and M.H.; investigation, J.N.J. and M.H.; visualization, J.N.J.; supervision, A.B.D., J.A.B. and J.R.F.; writing—original draft preparation, J.N.J., M.H., J.R.F. and A.B.D.; writing—review and editing, all authors. All authors have read and agreed to the published version of the manuscript.

Funding: The work presented in this paper is part of the research project "FlexDraperProduct: Developing an integrated production-ready solution for automated layup of prepregfiber plies", funded by Innovation Fund Denmark (Grant No. 8057-00011B). The support is gratefully acknowledged.

Data Availability Statement: The data used in this study are available at https://eco3d.compute.dtu.dk/pages/scanning.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A. Derivation of Equation (6)

In this appendix, we derive the gradients of the elements of $\widetilde{\mathbf{X}}_c(\mathbf{v})$ with respect to the vertices $\mathbf{v}$. Let $\widetilde{X}_c^{i,j}$ be the i,jth element of $\widetilde{\mathbf{X}}_c(\mathbf{v})$. This is found by tracing a ray through the center of the i,jth pixel in the camera until it intersects the surface at a point $\mathbf{r}$; that is,

$$\mathbf{r} = \mathbf{o} + t\mathbf{d}, \tag{A1}$$

for some t, where $\mathbf{o}$ is the position of the camera and $\mathbf{d}$ is the direction of the ray. Modeling the projector as a pinhole camera, this point is projected back into the projector by

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & P_{1,4} \\ P_{2,1} & P_{2,2} & P_{2,3} & P_{2,4} \\ P_{3,1} & P_{3,2} & P_{3,3} & P_{3,4} \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ 1 \end{bmatrix}. \tag{A2}$$

If we define $\mathbf{p}_1 = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} \end{bmatrix}$ and $\mathbf{p}_3 = \begin{bmatrix} P_{3,1} & P_{3,2} & P_{3,3} \end{bmatrix}$ then,

$$\widetilde{X}_c^{i,j} = \frac{q_1}{q_3} = \frac{\mathbf{p}_1 \cdot \mathbf{r} + P_{1,4}}{\mathbf{p}_3 \cdot \mathbf{r} + P_{3,4}}. \tag{A3}$$

The point $\mathbf{r}$ intersects a triangle with vertices $\mathbf{p}_a$, $\mathbf{p}_b$, and $\mathbf{p}_c$. The normal of this triangle can be computed by

$$\mathbf{n} = (\mathbf{p}_c - \mathbf{p}_b) \times (\mathbf{p}_a - \mathbf{p}_c). \tag{A4}$$

Using the normal, we can write t as

$$t = \frac{(\mathbf{p}_b - \mathbf{o}) \cdot \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}. \tag{A5}$$

Using the chain-rule, we obtain that

$$\begin{aligned} \frac{\partial \widetilde{X}_c^{i,j}}{\partial \mathbf{p}_a} &= \frac{\partial \widetilde{X}_c^{i,j}}{\partial t} \frac{\partial t}{\partial \mathbf{p}_a} && \text{(A6)} \\ &= \left(\sum_{i=1}^{3} \frac{\partial \widetilde{X}_c^{i,j}}{\partial p_i} \frac{\partial p_i}{\partial t} \right) \frac{\partial t}{\partial \mathbf{p}_a} && \text{(A7)} \\ &= \left(\frac{\partial \widetilde{X}_c^{i,j}}{\partial \mathbf{r}} \cdot \mathbf{d} \right) \frac{\partial t}{\partial \mathbf{p}_a}. && \text{(A8)} \end{aligned}$$

Here,

$$\begin{aligned} \frac{\partial \widetilde{X}_c^{i,j}}{\partial \mathbf{r}} &= \frac{\frac{\partial q_1}{\partial \mathbf{r}} q_3 - q_1 \frac{\partial q_3}{\partial \mathbf{r}}}{q_3{}^2} && \text{(A9)} \\ &= \frac{\frac{\partial q_1}{\partial \mathbf{r}} - \widetilde{X}_c^{i,j} \frac{\partial q_3}{\partial \mathbf{r}}}{q_3} && \text{(A10)} \\ &= \frac{\mathbf{p}_1 - \widetilde{X}_c^{i,j} \mathbf{p}_3}{q_3}. && \text{(A11)} \end{aligned}$$

The last missing term in Equation (A8) is

$$\begin{aligned} \frac{\partial t}{\partial \mathbf{p}_a} &= \frac{\partial}{\partial \mathbf{p}_a} \frac{(\mathbf{p}_b - \mathbf{o}) \cdot \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}} && \text{(A12)} \\ &= \frac{\frac{\partial}{\partial \mathbf{p}_a} [(\mathbf{p}_b - \mathbf{o}) \cdot \mathbf{n}](\mathbf{d} \cdot \mathbf{n}) - [(\mathbf{p}_b - \mathbf{o}) \cdot \mathbf{n}] \frac{\partial \mathbf{d} \cdot \mathbf{n}}{\partial \mathbf{p}_a}}{(\mathbf{d} \cdot \mathbf{n})^2} && \text{(A13)} \\ &= \frac{\frac{\partial \mathbf{n}}{\partial \mathbf{p}_a}^\top (\mathbf{p}_b - \mathbf{o}) - t \frac{\partial \mathbf{n}}{\partial \mathbf{p}_a}^\top \mathbf{d}}{\mathbf{d} \cdot \mathbf{n}} && \text{(A14)} \\ &= \frac{\frac{\partial \mathbf{n}}{\partial \mathbf{p}_a}^\top (\mathbf{p}_b - \mathbf{o} - t\mathbf{d})}{\mathbf{d} \cdot \mathbf{n}} && \text{(A15)} \\ &= \frac{\frac{\partial \mathbf{n}}{\partial \mathbf{p}_a}^\top (\mathbf{p}_b - \mathbf{r})}{\mathbf{d} \cdot \mathbf{n}}, && \text{(A16)} \end{aligned}$$

with

$$\begin{aligned} \frac{\partial \mathbf{n}}{\partial \mathbf{p}_a} &= \frac{\partial}{\partial \mathbf{p}_a} (\mathbf{p}_c - \mathbf{p}_b) \times (\mathbf{p}_a - \mathbf{p}_c) && \text{(A17)} \\ &= \frac{\partial}{\partial \mathbf{p}_a} (\mathbf{p}_c - \mathbf{p}_b) \times \mathbf{p}_a && \text{(A18)} \end{aligned}$$

$$= \frac{\partial}{\partial \mathbf{p}_a} [\mathbf{p}_c - \mathbf{p}_b]_\times \mathbf{p}_a \tag{A19}$$

$$= [\mathbf{p}_c - \mathbf{p}_b]_\times, \tag{A20}$$

where we utilize that a cross-product $\mathbf{a} \times \mathbf{b}$ can be written as a matrix-vector product $[\mathbf{a}]_\times \mathbf{b}$ using a skew-symmetric matrix. Inserting this into Equation (A16), we get

$$\frac{\partial t}{\partial \mathbf{p}_a} = \frac{(\mathbf{p}_b - \mathbf{r}) \times (\mathbf{p}_c - \mathbf{p}_b)}{\mathbf{d} \cdot \mathbf{n}}. \tag{A21}$$

If we write $\mathbf{r}$ using barycentric coordinates, $\mathbf{r} = \lambda_a \mathbf{p}_a + \lambda_b \mathbf{p}_b + \lambda_c \mathbf{p}_c$ with $\lambda_a + \lambda_b + \lambda_c = 1$, then Equation (A21) reduces to

$$\begin{aligned}\frac{\partial t}{\partial \mathbf{p}_a} &= \frac{(\mathbf{p}_b - (\lambda_a \mathbf{p}_a + \lambda_b \mathbf{p}_b + \lambda_c \mathbf{p}_c)) \times (\mathbf{p}_c - \mathbf{p}_b)}{\mathbf{d} \cdot \mathbf{n}} \\ &= \frac{(\lambda_a(\mathbf{p}_c - \mathbf{p}_a) + (1 - \lambda_b)(\mathbf{p}_b - \mathbf{p}_c)) \times (\mathbf{p}_c - \mathbf{p}_b)}{\mathbf{d} \cdot \mathbf{n}} \\ &= \frac{\lambda_a(\mathbf{p}_c - \mathbf{p}_a) \times (\mathbf{p}_c - \mathbf{p}_b)}{\mathbf{d} \cdot \mathbf{n}} \\ &= \frac{\lambda_a \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}. \end{aligned} \tag{A22}$$

Finally, we obtain (Equation (6))

$$\frac{\partial \widetilde{X}_c^{i,j}}{\partial \mathbf{p}_a} = \left(\frac{\mathbf{p}_1 - \widetilde{X}_c^{i,j} \mathbf{p}_3}{q_3} \cdot \mathbf{d} \right) \frac{\lambda_a \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}. \tag{A23}$$

The derivatives with respect to $\mathbf{p}_b$, and $\mathbf{p}_c$ can be found with similar derivations.

References

1. Gomes, L.; Bellon, O.R.P.; Silva, L. 3D reconstruction methods for digital preservation of cultural heritage: A survey. *Pattern Recognit. Lett.* **2014**, *50*, 3–14. [CrossRef]
2. Nocerino, E.; Stathopoulou, E.K.; Rigon, S.; Remondino, F. Surface reconstruction assessment in photogrammetric applications. *Sensors* **2020**, *20*, 5863. [CrossRef] [PubMed]
3. Turk, G. The Stanford Bunny. 2000. Available online: https://www.cc.gatech.edu/~turk/bunny/bunny.html (accessed on 21 December 2020).
4. Berger, M.; Tagliasacchi, A.; Seversky, L.M.; Alliez, P.; Guennebaud, G.; Levine, J.A.; Sharf, A.; Silva, C.T. A survey of surface reconstruction from point clouds. *Comput. Graph. Forum* **2017**, *36*, 301–329. [CrossRef]
5. Horn, B.K.P. Shape from Shading: A Method for Obtaining the Shape of a Smooth Opaque Object from One View. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1970. MAC TR-79.
6. Horn, B.K.P. Obtaining Shape from Shading Information. In *The Psychology of Computer Vision*; Winston, P.H., Ed.; McGraw-Hill: New York, NY, USA, 1975; Chapter 4, pp. 115–155.
7. Rockwood, A.P.; Winget, J. Three-dimensional object reconstruction from two-dimensional images. *Comput.-Aided Des.* **1997**, *29*, 279–285. [CrossRef]
8. Zhang, L.; Seitz, S.M. Image-based multiresolution shape recovery by surface deformation. *Proc. SPIE* **2000**, *4309*, 51–61.
9. Inagaki, A.; Tagawa, N.; Minagawa, A.; Moriya, T. Computation of shape and reflectance of 3D object using 3-D object using moiré phase and reflection model. In Proceedings of the International Conference on Image Processing (ICIP 2001), Thessaloniki, Greece, 7–10 October 2001; Volume 2, pp. 177–180.
10. Naganuma, S.; Tagawa, N.; Minagawa, A. Estimation of 3D shape and reflectance using multiple moiré images and shading model. In Proceedings of the ACM Symposium on Applied Computing (SAC 2003), Melbourne, FL, USA, 9–12 March 2003; pp. 943–950.
11. Naganuma, S.; Tagawa, N.; Minagawa, A.; Moriya, T. Simultaneous determination of object shape and color by Moire analysis using a reflection model. In Proceedings of the International Conference on Pattern Recognition (ICPR 2004), Cambridge, UK, 26 August 2004; Volume 3, pp. 202–205.
12. Isidoro, J.; Sclaroff, S. Stochastic mesh-based multiview reconstruction. In Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission, Padova, Italy, 19–21 June 2002; pp. 568–577.

13. Isidoro, J.; Sclaroff, S. Stochastic Refinement of the Visual Hull to Satisfy Photometric and Silhouette Consistency Constraints. In Proceedings of the International Conference on Computer Vision (ICCV 2003), Nice, France, 13–16 October 2003; Volume 2, pp. 1335–1342.
14. Yu, T.; Xu, N.; Ahuja, N. Shape and view independent reflectance map from multiple views. In Proceedings of the European Conference on Computer Vision (ECCV 2004), Prague, Czech Republic, 11–14 May 2004; pp. 602–615.
15. Yu, T.; Xu, N.; Ahuja, N. Shape and view independent reflectance map from multiple views. *Int. J. Comput. Vis.* **2007**, *73*, 123–138. [CrossRef]
16. Wu, C.; Wilburn, B.; Matsushita, Y.; Theobalt, C. High-quality shape from multi-view stereo and shading under general illumination. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR 2011), Providence, RI, USA, 20–25 June 2011; pp. 969–976.
17. Ha, H.; Oh, T.H.; Kweon, I.S. A multi-view structured-light system for highly accurate 3D modeling. In Proceedings of the International Conference on 3D Vision (3DV 2015), Lyon, France, 19–22 October 2015; pp. 118–126.
18. Loubet, G.; Holzschuch, N.; Jakob, W. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Trans. Graph.* **2019**, *38*, 228:1–228:14. [CrossRef]
19. Liu, S.; Li, T.; Chen, W.; Li, H. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In Proceedings of the International Conference on Computer Vision (ICCV 2019), Seoul, Korea, 27 October–2 November 2019; pp. 7708–7717.
20. Cazals, F.; Giesen, J. Delaunay triangulation based surface reconstruction. In *Effective Computational Geometry for Curves and Surfaces*; Springer: Berlin, Germany, 2006; pp. 231–276.
21. Kazhdan, M.; Bolitho, M.; Hoppe, H. Poisson surface reconstruction. In Proceedings of the Fourth Eurographics Symposium on Geometry Processing (SGP 2006), Cagliari, Italy, 26–28 June 2006; Volume 7.
22. Kazhdan, M.; Chuang, M.; Rusinkiewicz, S.; Hoppe, H. Poisson Surface Reconstruction with Envelope Constraints. *Comput. Graph. Forum* **2020**, *39*, 173–182. [CrossRef]
23. Lorensen, W.E.; Cline, H.E. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Comput. Graph.* **1987**, *21*, 163–169. [CrossRef]
24. Kazhdan, M.; Hoppe, H. Screened poisson surface reconstruction. *ACM Trans. Graph.* **2013**, *32*, 1–13. [CrossRef]
25. Shen, Y.; Barner, K.E. Fuzzy vector median-based surface smoothing. *IEEE Trans. Vis. Comput. Graph.* **2004**, *10*, 252–265. [CrossRef] [PubMed]
26. Diebel, J.R.; Thrun, S.; Brünig, M. A Bayesian method for probable surface reconstruction and decimation. *ACM Trans. Graph.* **2006**, *25*, 39–59. [CrossRef]
27. Zheng, Y.; Fu, H.; Au, O.K.C.; Tai, C.L. Bilateral normal filtering for mesh denoising. *IEEE Trans. Vis. Comput. Graph.* **2010**, *17*, 1521–1530. [CrossRef] [PubMed]
28. Lu, X.; Chen, W.; Schaefer, S. Robust mesh denoising via vertex pre-filtering and L_1-median normal filtering. *Comput. Aided Geom. Des.* **2017**, *54*, 49–60. [CrossRef]
29. Zollhöfer, M.; Stotko, P.; Görlitz, A.; Theobalt, C.; Nießner, M.; Klein, R.; Kolb, A. State of the Art on 3D Reconstruction with RGB-D Cameras. *Comput. Graph. Forum* **2018**, *37*, 625–652. [CrossRef]
30. Chen, W.; Ling, H.; Gao, J.; Smith, E.; Lehtinen, J.; Jacobson, A.; Fidler, S. Learning to Predict 3D Objects with an Interpolation-based Differentiable Renderer. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 9609–9619.
31. Fu, K.; Peng, J.; He, Q.; Zhang, H. Single image 3D object reconstruction based on deep learning: A review. *Multimed. Tools Appl.* **2021**, *80*, 463–498. [CrossRef]
32. Nocedal, J.; Wright, S. *Numerical Optimization*; Springer Science & Business Media: Berlin, Germany, 2006.
33. Reich, C.; Ritter, R.; Thesing, J. White light heterodyne principle for 3D-measurement. In *Sensors, Sensor Systems, and Sensor Data Processing*; Loffeld, O., Ed.; SPIE: Bellingham, WA, USA, 1997; Volume 3100, pp. 236–244. [CrossRef]
34. Zhou, Q. PyMesh—Geometry Processing Library for Python. 2020. Available online: https://github.com/PyMesh/PyMesh (accessed on 21 December 2020).
35. Adobe Systems Incorporated. *Digital Negative (DNG) Specification*; Version 1.5.0.0; Adobe Inc.: San Jose, CA, USA, 2019.
36. Xu, J.; Ma, Y.; He, S.; Zhu, J. 3D-GIoU: 3D Generalized Intersection over Union for Object Detection in Point Cloud. *Sensors* **2019**, *19*, 4093. [CrossRef]
37. Jack, D.; Pontes, J.K.; Sridharan, S.; Fookes, C.; Shirazi, S.; Maire, F.; Eriksson, A. Learning free-form deformations for 3D object reconstruction. In Proceedings of Asian Conference on Computer Vision (ACCV 2018), Perth, Australia, 2–6 December 2018; pp. 317–333.
38. Varol, G.; Ceylan, D.; Russell, B.; Yang, J.; Yumer, E.; Laptev, I.; Schmid, C. BodyNet: Volumetric inference of 3D human body shapes. In Proceedings of the European Conference on Computer Vision (ECCV 2018), Munich, Germany, 8–14 September 2018; pp. 20–36.
39. Shin, D.; Fowlkes, C.C.; Hoiem, D. Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR 2018), Salt Lake City, UT, USA, 18–22 June 2018; pp. 3061–3069.
40. Kato, H.; Ushiku, Y.; Harada, T. Neural 3D mesh renderer. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR 2018), Salt Lake City, UT, USA, 18–22 June 2018; pp. 3907–3916.
41. Steve. Dandelion Vase. 2013. Available online: www.thingiverse.com/thing:157102 (accessed on 21 December 2020).

42. Öztireli, A.C.; Guennebaud, G.; Gross, M. Feature preserving point set surfaces based on non-linear kernel regression. *Comput. Graph. Forum* **2009**, *28*, 493–501. [CrossRef]
43. Nayar, S.K.; Krishnan, G.; Grossberg, M.D.; Raskar, R. Fast separation of direct and global components of a scene using high frequency illumination. *ACM Trans. Graph.* **2006**, *25*, 935–944. [CrossRef]