Differentiable Stereopsis: Meshes from multiple views using differentiable rendering

Shubham Goel
UC Berkeley
shubham-goel@berkeley.edu

Georgia Gkioxari
Meta AI
gkioxari@fb.com

Jitendra Malik
UC Berkeley
malik@eecs.berkeley.edu

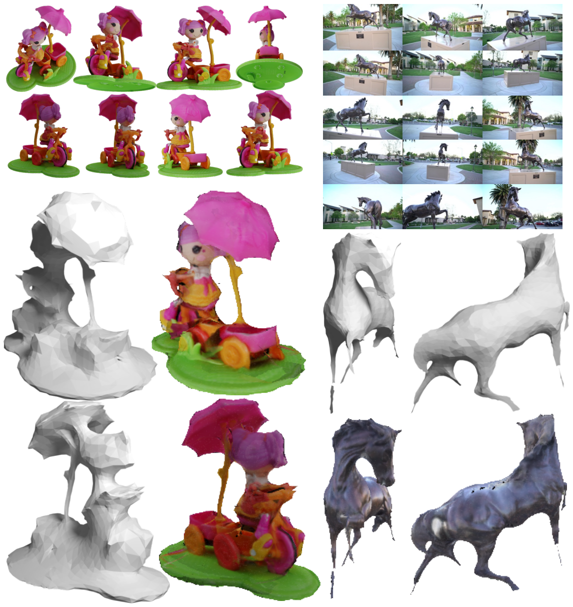

Figure 1. Reconstructions with Differentiable Stereopsis (DS) from few input views and noisy cameras. We show input views (top) and novel views of reconstructions (bottom).

Abstract

We propose Differentiable Stereopsis, a multi-view stereo approach that reconstructs shape and texture from few input views and noisy cameras. We pair traditional stereopsis and modern differentiable rendering to build an end-to-end model which predicts textured 3D meshes of objects with varying topologies and shape. We frame stereopsis as an optimization problem and simultaneously update shape and cameras via simple gradient descent. We run an extensive quantitative analysis and compare to traditional multi-view stereo techniques and state-of-the-art learning based methods. We show compelling reconstructions on challenging real-world scenes and for an abundance of object types with complex shape, topology and texture. [1]

1. Introduction

Binocular stereopsis [47], and its multi-view cousin, Structure from Motion [14, 44], has traditionally been formulated as a two stage process:

1. Find corresponding 2D points across views, which are the 2D projections of the *same* 3D scene point.
2. Recover relative orientations of cameras, and the depths of these points by triangulation.

In this work, we bypass the first stage of finding point correspondences across images and directly estimate 3D shape and cameras given multiple 2D views with noisy cameras. We formulate this as an optimization problem that we solve using newly developed differentiable rendering tools. We name our approach *Differentiable Stereopsis*.

Our approach is linked to old work in multi-view geometry, and in particular model-based stereopsis which was explored by Debevec *et al*. [6] and related ideas in plane plus parallax by Irani *et al*. [17]. The key observation in model-based stereo is simple: two images of the same scene which appear different become similar after projection onto an approximate 3D model of the scene. Projecting the texture from one image onto the 3D model produces a warped version of that view which when transformed from a second view is directly comparable to the second image. Initially, the 3D model and the estimated relative camera orientation are inaccurate. But as shape and camera predictions improve, the two images will start to look more similar and will eventually become identical – in the idealized case of Lambertian surfaces and no imaging noise. Upon convergence, the shape is expected to be an accurate representation of the scene.

An important step in traditional stereopsis is finding 2D correspondences across views. We bypass this and directly recover shape and cameras using modern optimization techniques. We frame stereopsis as an optimization problem by minimizing a differentiable objective with respect to shape and cameras. To this end, we exploit advances in *differentiable rendering* [4, 21, 30, 32, 40] to project shape and texture onto image planes which we compare to scene views.

[1] Project webpage: https://shubham-goel.github.io/ds/

Algorithm 1: Differentiable Stereopsis (2-view)

```
Input: $I_{1,2}$, $\pi_{1,2}$;
$S \leftarrow$ Sphere();
while not converged do
    points $\leftarrow$ rasterize($S$, $\pi_1$);
    texels $\leftarrow$ sample($I_2$, $\pi_2$(points));
    $I_1^r \leftarrow$ blend(texels);
    loss $\leftarrow$ compute_loss($I_1^r$, $I_1$);
    $S, \pi_{1,2} \leftarrow S, \pi_{1,2} -$ lr * gradient(loss);
end
Outputs $S$, $\pi_{1,2}$
```

We update shape and cameras via gradient descent. Algorithm 1 illustrates our proposed Differentiable Stereopsis (DS) for the case of 2 views. We rely on (i) object masks to isolate and refine object topology; and (ii) noisy camera pose initializations, which may still come from a correspondence matching algorithm. Fig. 1 shows shape reconstructions with DS when noisily posed views are given as input.

At the core of our approach is a novel and differentiable texture transfer method which pairs rendering with the key insight of texture warping via 3D unprojections. Our texture transfer learns to sample texture from the input views based on the shape estimate and noisy cameras. To allow for differentiation, it composites the final texture in a soft manner by weighing texture samples proportionally to their visibility and direction from each view.

We test our approach on challenging datasets and on a large variety of objects with complex and varying shapes. Unlike prior works that assume several dozens of object views, our experimental settings follow real-world practical scenarios where only a few views are available. For example, Amazon, eBay or Facebook Marketplace only contain a handful of views for each listed item, and any 3D reconstruction has to originate from 10 views or less. We emphasize on this harder, yet more realistic, setting and show empirical results with real product images from Amazon [5]. On Google's Scanned Objects [41] we perform an extensive quantitative and qualitative analysis and compare to competing approaches under settings similar to ours. We also show results on Tanks and Temples [24] which contains RGB views of complex scenes, as shown on the right in Fig. 1.

2. Related Work

Extracting 3D structure from 2D views of a scene is a long standing goal of computer vision. Classical multi-view stereo methods and Structure from Motion techniques [6, 9, 14, 17, 44] find correspondences across images and triangulate them into points in 3D space. The resulting point clouds, if dense enough, can be meshed into surfaces [1, 22]. The culmination of a long line of classical SfM and stereo approaches is COLMAP [42, 43] – a widely used tool for estimating camera poses and reconstructing dense point clouds from 2D input views. All aforementioned techniques assume calibrated and accurate cameras and thus are not very robust to camera noise.

Finding point correspondences, the first stage of stereopsis, is challenging especially in the case of sparse widely-separated views. Debevec *et al.* [6] tackle this by proposing model-based stereopsis wherein a coarse scene geometry allows views to be placed in a common reference frame, making the correspondence problem easier. We draw inspiration from this work and pair it with new learning tools to reconstruct textured 3D meshes from sparse views. We frame stereopsis as an optimization problem and minimize a differentiable objective which allows both shape and cameras to self-correct. This increases robustness to camera noise, in antithesis to classical techniques.

Recent work on multi-view stereo [48, 49] train deep neural nets (DNNs) with depth supervision. As expected, these methods outperform COLMAP for point cloud reconstruction but are limited as they need ground truth. We rely solely on image re-projection losses and no true depth information.

Work on unsupervised depth prediction [28, 52, 54, 55] estimate depth via DNNs trained on monocular videos and without ground truth depth. They exploit photometric and depth consistency across multiple views, much like classical stereo. However, they focus on forward-facing scenes like KITTI [11] and do not reconstruct high-fidelity shape.

There is extensive work on recovering shape from images using differentiable rendering [4, 13, 19, 21, 25, 27, 30, 32, 40, 45, 51]. These approaches focus on extracting object priors by training on large datasets and test on images of seen categories. We also use differentiable rendering [4, 21, 30, 32, 40] to frame stereopsis as a differentiable optimization problem. Differentiating with respect to shape and camera allows for both to self-correct during optimization.

Most relevant to our work are methods that learn shape by fitting to a set of images. Early work on extracting shape from silhouettes used a visual hull [26]. Gadelha *et al.* [10] reconstruct voxels from silhouettes and noisy camera poses via differentiable projection but don't use any texture information. However, shape details such as concavities cannot be captured by silhouettes. We show in Fig. 5 in our experiments (Sec. 4) that optimizing for shape without texture information fails to reconstruct creases in shape. Some variational approaches for MVS [7, 8, 15, 39] exploit photometric consistency to refine shape via gradient descent but they require many images, initial shapes or accurate cameras. Recently, IDR [50] and DVR [37] recover shape from multiple posed images and masks using implicit volumetric representations. IDR shows superior results to DVR and claims to work with few input views and slightly noisy camera poses; a setting similar to ours. We compare to IDR in Sec. 4.

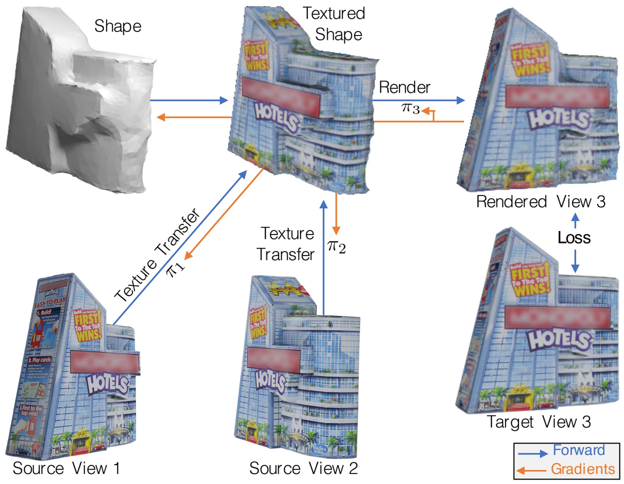

Figure 2. Overview of Differentiable Stereopsis (DS). The shape estimate is textured using source views, rendered from a target view's camera and compared against the target view. The loss is backpropagated to update shape and cameras.

Recent novel-view-synthesis approaches [31, 34, 56] encode volumetric occupancy information in their internal representations for the task of image synthesis from novel viewpoints. While they don't explicitly learn shape, their representation can be processed to extract geometry. NeRF [34] is one such approach which takes posed multiple views as input and encodes occupancy and color for points in 3D space as an implicit function. In Sec. 4, we compare to NeRF and extend it with a variant that optimizes for noisy cameras by enabling backpropagation to its parameters.

3. Approach

We tackle the problem of stereopsis using modern differentiable rendering techniques. Our approach takes N image views $I_{1..N}$ of an object with corresponding masks $A_{1..N}$ and *noisy* camera poses $\pi_{1..N}$ as input, and outputs the shape of the object as a textured mesh. We frame stereopsis as an optimization problem, outlined in Fig. 2. We iteratively render a shape estimate from multiple cameras using differentiable textured rendering and update the shape and cameras by minimizing image reprojection losses.

We first provide some background on differentiable rendering and then describe our approach in detail.

3.1. Background

We define a textured mesh $M = (V, F, T)$ as a set of vertices V, faces F and a texture map T. Under a camera viewpoint π, mesh M is rendered to image $I^r = R_T(M, \pi)$ and mask $A^r = R_S(M, \pi)$, where R_T denotes textured rendering and R_S silhouette rendering.

Both R_S and R_T perform mesh rasterization. Rasterization computes which parts of the mesh are projected to a pixel on the image plane. For each pixel p, we find the K nearest faces that intersect with a ray originated at p [40].

In the case of silhouette rendering R_S, rasterization is followed by a soft silhouette shader. This shader assigns each pixel an occupancy probability by blending the euclidean distance of the pixel to each of the K faces [30, 40].

For textured rendering R_T, we use a texture shader which computes the RGB color for each pixel p in the image. This shader blends the colors from the top K faces for each pixel, as computed by the rasterizer. For the k-th face, the color $c_k = T(x)$ is computed by sampling the texture map T at the point of intersection x of the ray originating at p and the k-th face. The set of colors $c_{1..K}$, also known as *texels*, are composited to get the final color for the pixel.

3.2. Texture Transfer

The goal of our approach is to find $M = (V, F, T)$ that represents the object as seen from the noisily posed input views. For each shape hypothesis (V, F) we need to find the optimal texture T. We introduce a *novel* texture shader which relies on texture transfer from the inputs $I_{1...N}$.

Our shader computes the texture map T as a function of the shape hypothesis (V, F) and the posed input views $I_{1...N}$. The texture map $T : x \rightarrow (r, g, b)$ assigns an RGB color for each point x on the mesh surface. The color is directly sampled from one or more input views. We build on a key insight: for a correct shape (V, F) and correct cameras $\pi_{1...N}$, there exists one (or many) view i in which x is unoccluded, or in other words, there is a clear line-of-sight to x. For all such views, the projections $\pi_i(x)$ in the images correspond to the same 3D point x and for Lambertian surfaces, all these points will share the same color $I_i(\pi_i(x))$. The color $T(x)$ assigned to point x is composited from the colors $I_i(\pi_i(x))$ for all views with a clear line-of-sight to x. Formally, we define the texture transfer as follows:

$$T(x) = \sum_i w_i I_i(\pi_i(x)) \tag{1}$$

where weights are unit-normalized and defined as $w_i = \sigma_i \gamma_i$.

σ encodes whether x has a clear line-of-sight from the corresponding view. Formally, we compare the z-distance of the camera transformed point $\pi_i(x)$ to the rendered depth map D_i at $\pi_i(x)$ as follows

$$\sigma_i = \exp(-(\pi_i(x)_z - D_i(\pi_i(x)))/\tau_{\text{vis}}) \tag{2}$$

If there is a clear line-of-sight to x, then $\pi_i(x)_z \approx D_i(\pi_i(x))$ and thus $\sigma_i \approx 1.0$. If x is obstructed by other parts of the shape, then $\pi_i(x)_z > D_i(\pi_i(x))$ and $\sigma_i < 1.0$. We set the temperature τ_{vis} to 10^{-4}.

γ is a heuristic that favours views that look at x fronto-parallel with minimal foreshortening. If $\hat{n}_i(x)$ is the outward surface-normal at x in i-th view coordinates, then

$$\gamma_i = \mathbb{1}[\hat{n}_i(x)_z < 0] \exp(-(1 + \hat{n}_i(x)_z)/\tau_{\cos}) \tag{3}$$

γ_i is highest when the normal points opposite the camera's z-axis, or $\hat{n}_i(x)_z = -1$. γ_i decreases exponentially as $\hat{n}_i(x)_z$ increases. We set the temperature τ_{cos} to 0.1. In addition, we cull backward-facing normals ($\hat{n}_i(x)_z > 0$) to correctly sample texture in thin surfaces where σ fails to capture visibility information of points on the two sides of the surface.

Texture Rendering We described how to sample texture for a point x on the mesh surface. To render the texture under a viewpoint π, for each pixel p we sample texels $c_{1..K}$ for all points $x_{1..K}$ with $c_k = T(x_k)$, where x_k is the point on the k-th face that intersects the ray originating at p. We use softmax blending [30] to composite the final color at p.

3.3. Optimization

We have explained how to define the texture map T for an object shape (V, F) given posed input views $I_{1..N}$ and we have described how to render $M = (V, F, T)$ to images and silhouettes. We now describe our objective and how we optimize it w.r.t. vertices V and cameras $\pi_{1..N}$.

Parametrization We parametrize a camera $\pi = (r, t, f)$ as rotation via an axis-angle representation $r \in \mathbb{R}^3$ (magnitude $|\mathbf{r}|$ is angle, normalization $\mathbf{r}/|\mathbf{r}|$ is axis), translation as $t \in \mathbb{R}^3$ and focal length f as half field-of-view.

We parametrize geometry as $V = V_0 + \Delta V$ where $\Delta V \in \mathbb{R}^{|V| \times 3}$ is the deformation being optimized and V_0 are initial mesh vertices which remain constant.

Objective Given a shape hypothesis $M = (V, F, T)$, cameras $\pi_{1..N}$ and input views $I_{1..N}$, we render silhouette $A_i^r = R_S(M, \pi_i)$ and image $I_i^r = R_T(M, \pi_i)$ for each view $i = 1, ..., N$. We define our total loss to be

$$L_{\text{total}} = L_{\text{tex}} + L_{\text{mask}} + L_{\text{edge}} + L_{\text{lap}} \tag{4}$$

The texture reconstructions loss L_{tex} is defined as the sum of an L_1 loss and perceptual distance metric L_{perc} [53]:

$$L_{\text{tex}} = \sum_i |I_i^r - I_i| + L_{\text{perc}}(I_i^r, I_i) \tag{5}$$

The mask reconstruction loss combines an MSE loss and a bi-directional distance transform loss (see details in Appendix).

$$L_{\text{mask}} = \sum_i ||A_i^r - A_i||_2^2 + L_{\text{bi-dt}}(A_i^r, A_i) \tag{6}$$

In addition to reprojection losses in Eq. 5 & 6, we employ smoothness reguralizers on the mesh: $L_{\text{edge}} = ||E - l||_2^2$ is an MSE loss penalizing edge lengths that deviate from the mean initial edge length l, while $L_{\text{lap}} = ||L_{\text{cot}} V||_2$ is a cotangent-laplacian loss that minimizes mean curvature [36].

Initialization and Warmup We initialize cameras with the noisy input cameras, V_0 with an ico-sphere and ΔV with zeros. During an initial warmup phase of 500 iterations, we freeze cameras and optimize shape without the texture loss. We start with a very low-resolution sphere and subdivide it twice during warmup, at 100 and 300 iterations respectively.

Texture Sampling We compute the texture map T after each shape update during optimization. For each training view i, and for each pixel p, we find the K closest faces intersecting a ray originating at p and the corresponding points of intersection $x_{1..K}$. We compute texels $c_k = T(x_k)$, described in Sec. 3.2, and set $w_i = 0$ in Eq. 1 so that image I_i does not contribute to the texture for pixel p in the rendered i-th view. This ensures that image I_i^r for camera π_i is generated by sampling colors from all images $I_{1..N}$ but I_i to encourage photometric consistency.

Handling Variable Topology Each gradient descent step updates the vertex positions of the mesh and the camera parameters. However, the topology of the shape is left intact. To handle objects with varying topology and to deviate from shapes homeomorphic to spheres, we update the topology of our shapes during optimization. At intermediate steps during training, we voxelize our mesh [35, 38], project voxels onto the view plans and check for occupancy by comparing to the ground truth silhouettes $A_{1...N}$. We remove voxels that project to an unoccupied area in any mask. We re-mesh the remaining voxels using marching cubes, reset all shape-optimization parameters and resume optimization.

4. Experiments

We test our differentiable stereopsis approach, which we call DS, on three datasets: Google's Scanned Objects [41], Tanks and Temples [24] and the Amazon-Berkeley Objects [5]. We additionally evaluate on DTU MVS [18] in the Appendix. We run extensive quantitative analysis on objects of varying topology and shapes, for which 3D ground truth is available. We also show qualitative results on real objects and challenging real-world scenes.

4.1. Experiments on Google's Scanned Objects

Google's Scanned Objects (CC-BY 4.0) [41] consists of 1032 common household objects that have been 3D scanned to produce high-resolution Lambertian textured 3D meshes. From these, we pick 50 object instances with varying shape, topology and texture for quantitative analysis including toys, electronics, instruments, appliances, cutlery and many more. For each object, we render 2048×2048 RGBA images from 12 random camera viewpoints. Camera rotation Euler angles and field-of-view are uniformly sampled in $[0°, 360°]$ and $[20°, 50°]$ respectively. To the cameras, we add rotation noise $\theta \sim \mathcal{N}(0, \sigma^2)$ about a uniformly sampled axis with varying $\sigma = \{10°, 20°, 30°\}$.

Metrics We report a variety of metrics to quantitatively compare the predicted with the ground-truth meshes. We use L_2-Chamfer distance, normal consistency and F1 score at different thresholds, following [12]. Since predicted shapes

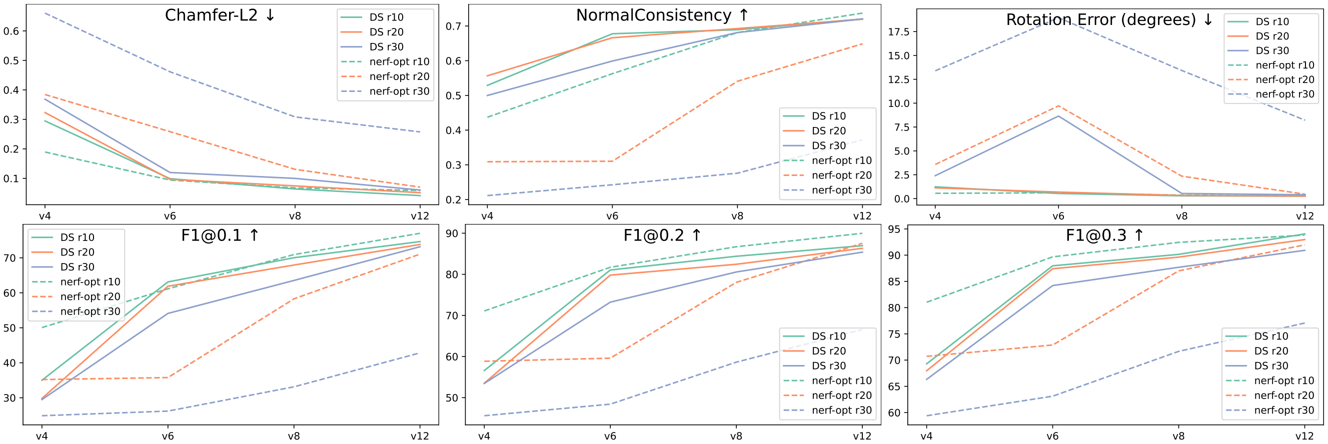

Figure 3. Performance of DS and nerf-opt on Google's Scanned Objects with varying number of views $\{4, 6, 8, 12\}$ (x-axis) and camera noise $\{10°, 20°, 30°\}$. Each plot reports the median across the 50 objects in our evaluation set. We report shape reconstruction metrics (Chamfer, F1), normal consistency, and camera error. For Chamfer and camera error, lower is better. For everything else, higher is better.

don't lie in the same coordinate frame as the ground-truth, we align predictions to ground truth before benchmarking via the iterative-closest-point (ICP) algorithm [2]. See Appendix for more details. Lastly, we report the rotation error (in degrees) between the ground truth and the output cameras from DS optimization.

Comparison with baselines We extensively compare to NeRF [34], as the state-of-the-art volumetric method, which learns an implicit function from accurately posed input views. While NeRF doesn't explicitly output shape, we extract geometry from its implicit representation via voxelization and run marching cubes to get a mesh. We compare to two NeRF variants which use additional mask information: (a) *nerf* - the original NeRF approach with an additional MSE loss on rendered masks, and (b) *nerf-opt* - which is the same as (a) but optimizes camera poses with gradients from the reprojection loss. nerf-opt uses the same camera parametrization as our approach. To prevent NeRF from collapsing due to large areas of white background in the input views, all NeRF baselines sample 50% of their points inside the mask in every iteration. We also compare qualitatively to IDR [50], a volumetric method with an implicit representation that learns geometry and appearance from sparse wide-baseline images and masks with noisy camera poses. In the Appendix, we also compare to COLMAP [42, 43] as the state-of-the-art photogrammetry approach. Finally, we compare to variants of our approach: (a) *DS-notex*, which does not use any texture information removing L_{tex} from Eq. 4; and (b) *DS-naive*, which naively optimizes a UV texture image in addition to shape/camera instead of using our texture-transfer. For texturing, the texture image is mapped to the mesh surface using a fixed UV map [16] that is automatically computed with Blender [3]. Whenever the mesh topology changes, the texture image is re-initialized and the UV-map recomputed.

Fig. 3 quantitatively compares DS to nerf-opt, the best performing NeRF variant of the two. We train with varying number of views $N = 4, 6, 8, 12$ (x-axis) and varying camera noise $\{10°, 20°, 30°\}$. Each plot reports the median across the 50 instances selected from the dataset. For small camera noise (10°), DS and nerf-opt (green lines) achieve comparable Chamfer and F1, except for $N = 4$ views where nerf-opt achieves higher F1. Undoubtedly, predicting shape from 4 views is challenging for all methods, as indicated by the absolute performance and is the only setting where nerf-opt performs better than DS. For larger camera noise (20°), DS performs better than nerf-opt (orange lines) under all metrics for $N \geq 6$ and on par for $N = 4$. For even larger camera noise (30°), DS leads by a significant margin (blue lines) for all N and all metrics. We note that as the number of views N increases, both methods converge roughly to the same performance for 10° & 20° noise. For 30° noise, DS also converges to the above optimum with increasing views N. On the other hand, nerf-opt is unable to recover shape or cameras for 30° noise and achieves much lower reconstruction quality. These results prove that DS can learn better shapes and recover cameras even under larger camera noise and fewer views. When given slightly more views, DS reaches the same reconstruction quality as with little camera noise proving its robustness to errors in cameras. See Appendix for quantitative comparisons to IDR, COLMAP, and DS-naive.

Fig. 4 qualitatively compares nerf, nerf-opt, IDR and DS with 8 views and 30° noise. IDR and both NeRF variants produce shapes with cloudy artifacts, with nerf-opt visibly outperforming nerf. DS captures better shape under the same settings proving its robustness to noisy cameras and few views. We observe that in a few-view wide-baseline setting, like ours, implicit volumetric approaches attempt to explain the few input views without relying on accurate shape geometry and appearance. However, meshes, which

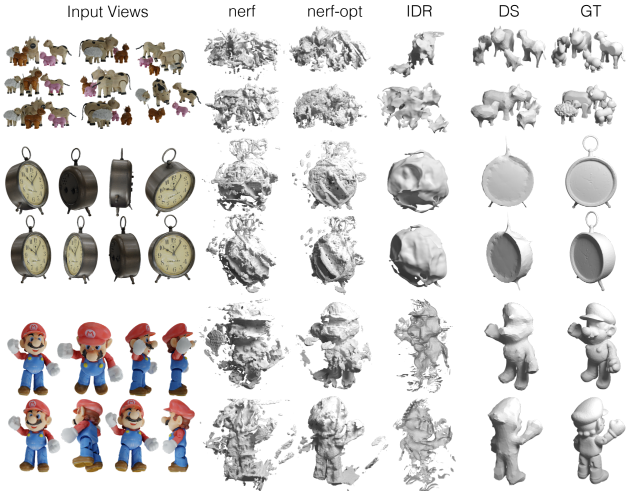

Figure 4. Results of nerf, nerf-opt, IDR and DS with 8 views and 30° camera noise. nerf-opt and IDR outperform nerf but they fail to capture good shape. DS captures better geometry illustrating its robustness to high levels of camera noise.

explicitly represent surfaces, offer stronger surface regularization and predict more precise geometry. Also, in the first row of Fig. 4 we observe that DS is able to reconstruct different animals as disconnected components despite having been initialized to a single sphere.

Fig. 5 compares DS to DS without texture (*DS-notex*) and naive texture map optimization (*DS-naive*) with 8 input views and 20° camera noise. DS-notex fails to capture shape concavities, which are impossible to capture via just silhouettes. DS-naive results in shapes with some concavities but in the wrong place and of shape quality similar to DS-notex. With naive texture optimization as in DS-naive, texture converges to the mean texture from different images, providing unreliable gradients to improve shape/cameras and leading to suboptimal geometry. In contrast, DS accurately captures creases in object shapes by exploiting texture.

Fig. 9 shows qualitative results on Google's Scanned Objects. For each object, we train with 8 input views and 20° camera noise. We show the input views (left) and the output shape and texture for two novel views (right).

Fig. 7 shows the evolution of shape with time for two examples from Google's Scanned Objects with 8 input views and 20° camera noise. We remesh the shape by updating its topology at three intermediate steps during optimization. This brings the final shape close to the ground truth both in terms of geometry and topology.

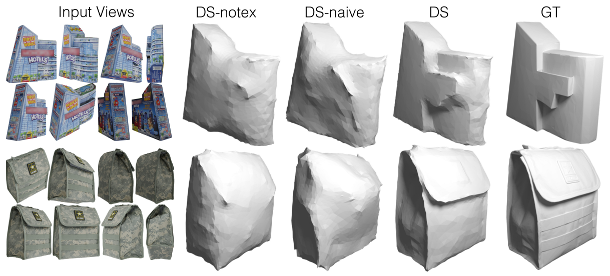

Figure 5. DS without texture (DS-notex), DS-naive and DS with 8 views and 20° camera noise. DS-notex fails to capture shape concavities, while DS-naive fails to recover accurate shape and cameras. The ground truth shape (GT) is shown in the last column.

We also show failure modes in the Appendix.

4.2. Results on Amazon Products

We show results from images of 6 real-world objects from the ABO dataset (CC-BY-NC 4.0) [5]. Pixel-thresholding the white-background images gives masks. Camera poses for these images are unknown and COLMAP fails to give sensible estimates. We get rough initial cameras by manually annotating a set of 40 keypoint correspondences across all images of an object. We estimate the parameters for a weak-perspective camera for each image using a orthographic rigid-body factorization formulation [33] adopted in [19, 20, 46]. We initialize our perspective cameras using the computed weak-perspective cameras and assume a 30° field-of-view.

Fig. 6 shows shape and texture reconstructions. Despite very noisy cameras and few views, ranging from 4 to 9, our approach reconstructs shape and texture reasonably well even for challenging shape topologies like the lawnmower. We also note that DS is able to reconstruct more specular surfaces like the wristwatch in the last row.

4.3. Results on Tanks and Temples

Tanks and Temples (CC-BY-NC-SA 3.0) [24] is a 3D reconstruction benchmark consisting of RGB videos of indoor and outdoor scenes with corresponding laser-scanned ground-truth 3D point clouds. The dataset comes with cameras computed by COLMAP's SfM pipeline [42]. We evaluate on 7 scenes using *only 15* input images and corresponding SfM-reconstructed cameras as initialization. For *Barn*, *Ignatius*, *Caterpillar* and *Truck*, we generate masks by rendering the 3D point clouds from SfM-reconstructed cameras. To stress test our approach without relying on 3D point

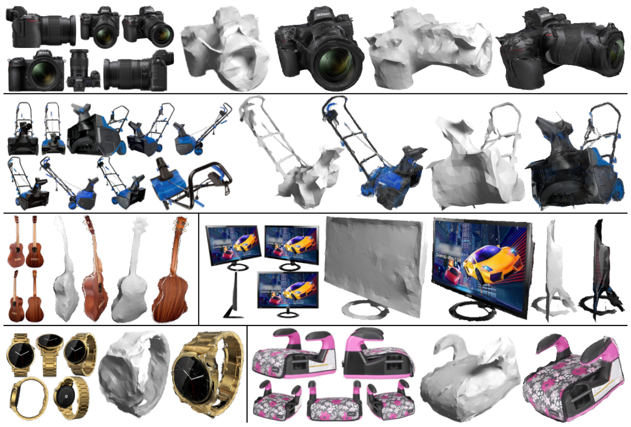

Figure 6. DS evaluated on real-world product images from Amazon [5]. For each example, we show input views (left) and reconstructed shape and texture for novel views (right).

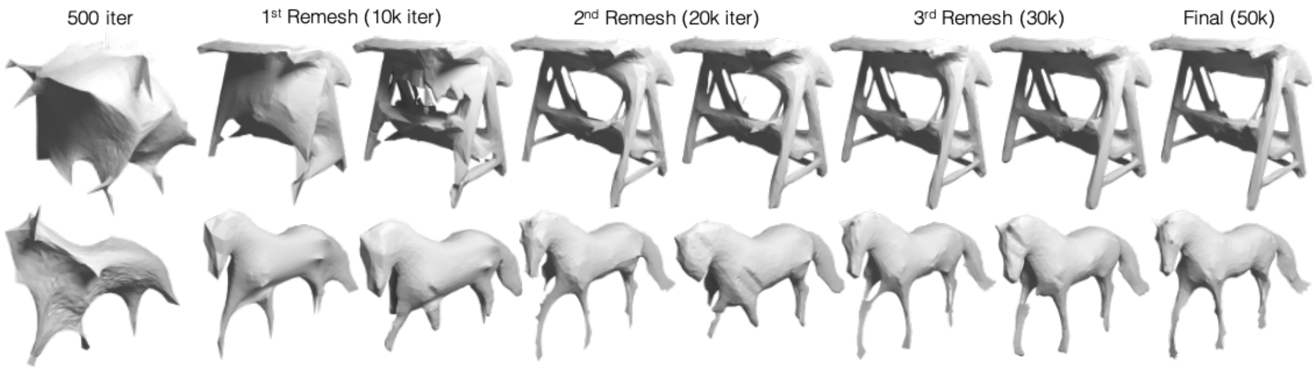

Figure 7. Evolution of shape over time for *Garden Swing* (top) and *Breyer Horse* (bottom) from Google's Scanned Objects with 8 views and 20° camera noise. We visualize shape at key optimization steps: at the end of warmup (at 500 iterations), before and after the $1^{\text{st}}/2^{\text{nd}}/3^{\text{rd}}$ remesh (at 10k/20k/30k iterations) and final shape (at 50k iterations).

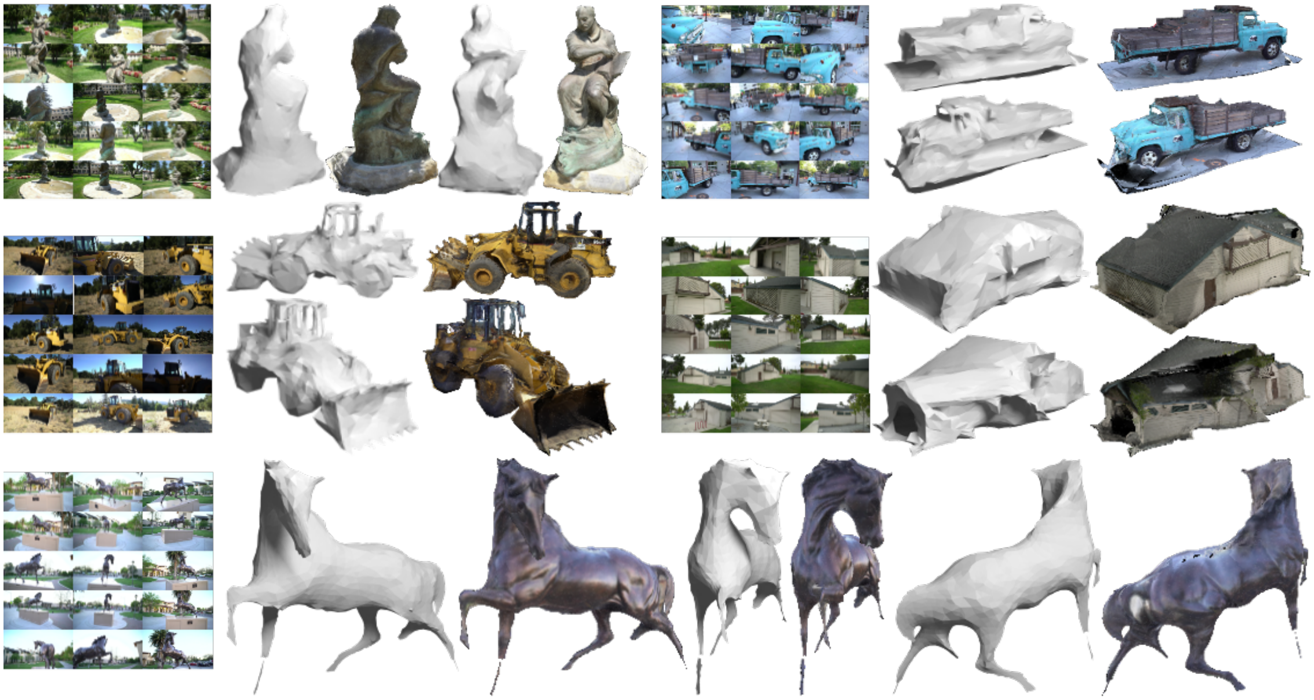

Figure 8. Reconstructions of DS on *Ignatius*, *Truck*, *Caterpillar*, *Barn* and *Horse* from Tanks and Temples with 15 input views and SfM-generated camera poses. For each example, we show input views (left), shape and texture reconstructions from two novel views (right). Silhouettes for *Horse* were generated by a pretrained off-the-shelf 2D object detector.

clouds to get masks, for *Horse*, *Family* and *Train* we use an off-the-shelf object detector [23] pretrained on COCO [29].

Fig. 8 shows reconstructions for scenes from Tanks and Temples with 15 input views and SfM-reconstructed cameras. DS is able to produce good reconstructions and undoubtedly has a harder time for *Barn* due to occlusions by trees. For *Family* and *Train*, detected masks are poor leading to bad reconstructions. In the Appendix, we compare to IDR, NeRF-opt, and COLMAP.

5. Discussion

We propose Differentiable Stereopsis (DS) by pairing traditional model-based stereopsis with modern differentiable rendering. We show results on a diverse set of object shapes with noisy cameras and few input views. Even though DS performs well, it has limitations. It assumes Lambertian surfaces and consistent lighting. DS works for objects – extending to complex scenes is future work. While DS is robust to noisy masks (*e.g*. predictions from Mask R-CNN) and inaccurate cameras provided at input, eliminating them from the input alltogether is important future work.

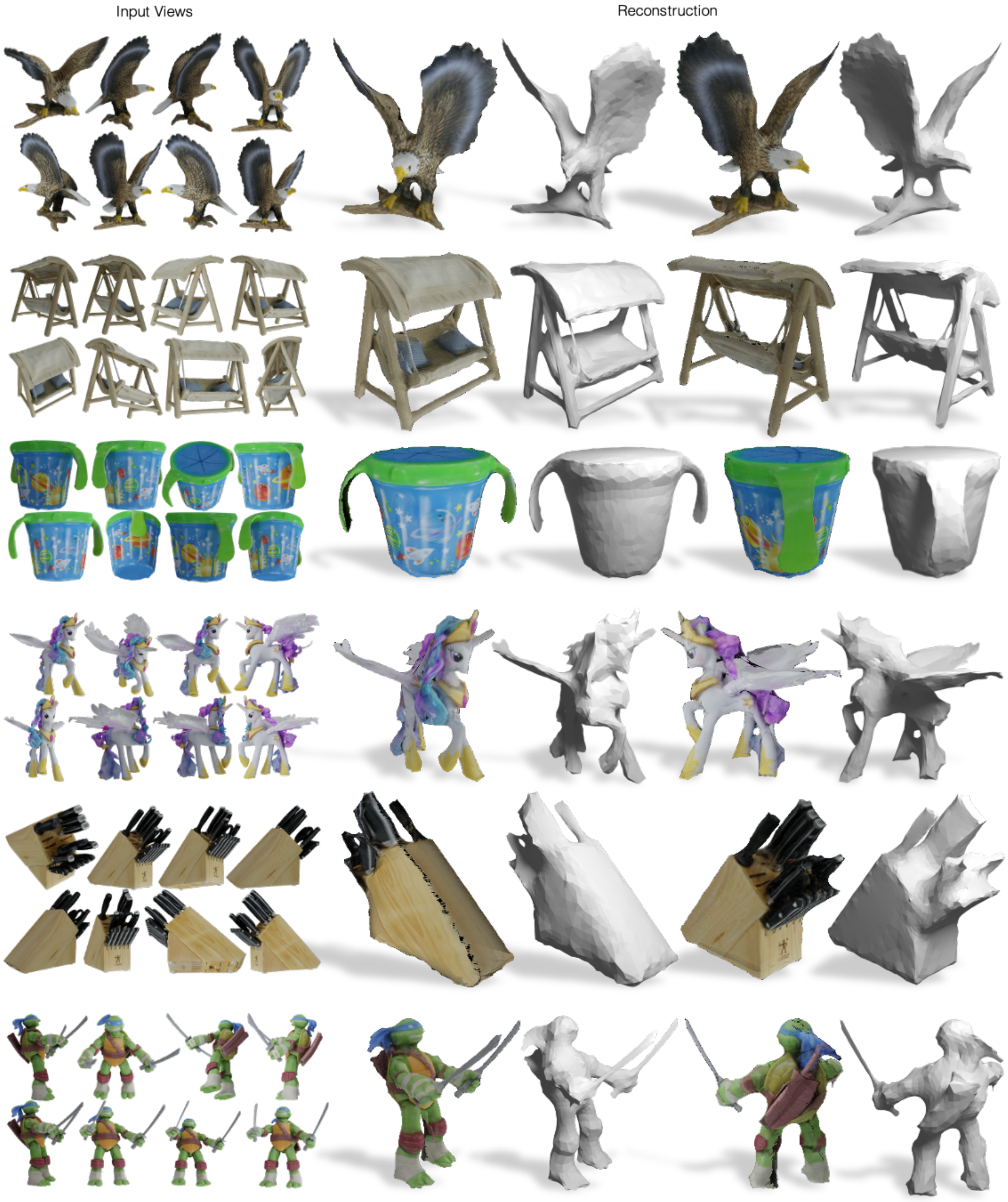

Figure 9. Qualitative results of DS on Google's Scanned Objects with 8 input views and 20° camera noise. We show input views (left) and reconstructed shape and texture from two novel views (right).

References

[1] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Claudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 2

[2] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 5

[3] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Institute, Amsterdam, 2019. 5

[4] Wenzheng Chen, Huan Ling, Jun Gao, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. In *NeurIPS*, 2019. 1, 2

[5] Jasmine Collins, Shubham Goel, Matthieu Guillaumin, Thomas Dideriksen, Kenan Deng, Himanshu Arora, Arnab Dhua, and Jitendra Malik. Amazon berkeley objects (abo) dataset. `https://amazon-berkeley-objects.s3.amazonaws.com/index.html`, 2021. 2, 4, 6

[6] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *CGIT*, 1996. 1, 2

[7] Amaël Delaunoy and Marc Pollefeys. Photometric bundle adjustment for dense multi-view 3d modeling. In *CVPR*, pages 1486–1493, 2014. 2

[8] Olivier Faugeras and Renaud Keriven. Complete dense stereovision using level set methods. In *ECCV*, pages 379–393. Springer, 1998. 2

[9] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 2009. 2

[10] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *3DV*, 2017. 2

[11] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 2

[12] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *ICCV*, 2019. 4

[13] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoint without keypoints. In *ECCV*, pages 88–104. Springer, 2020. 2

[14] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 1, 2

[15] Vu Hoang Hiep, Renaud Keriven, Patrick Labatut, and Jean-Philippe Pons. Towards high-resolution large-scale multiview stereo. In *CVPR*, pages 1430–1437. IEEE, 2009. 2

[16] John F Hughes and James D Foley. *Computer graphics: principles and practice*. Pearson Education, 2014. 5

[17] Michal Irani, P Anandan, and Meir Cohen. Direct recovery of planar-parallax from multiple frames. *TPAMI*, 2002. 1, 2

[18] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *CVPR*, pages 406–413. IEEE, 2014. 4

[19] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 2, 6

[20] Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *CVPR*, 2015. 6

[21] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *CVPR*, 2018. 1, 2

[22] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 2

[23] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. PointRend: Image segmentation as rendering. In *CVPR*, 2020. 7

[24] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 2017. 2, 4, 6

[25] Nilesh Kulkarni, Abhinav Gupta, David F Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *CVPR*, pages 452–461, 2020. 2

[26] Aldo Laurentini. The visual hull concept for silhouette-based image understanding. *TPAMI*, 16(2):150–162, 1994. 2

[27] Xueting Li, Sifei Liu, Kihwan Kim, Shalini De Mello, Varun Jampani, Ming-Hsuan Yang, and Jan Kautz. Self-supervised single-view 3d reconstruction via semantic consistency. In *ECCV*, pages 677–693. Springer, 2020. 2

[28] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, pages 2041–2050, 2018. 2

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 7

[30] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *ICCV*, 2019. 1, 2, 3, 4

[31] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *CVPR*, 2019. 3

[32] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *ECCV*, 2014. 1, 2

[33] Manuel Marques and João Costeira. Estimating 3d shape from degenerate sequences with missing data. *CVIU*, 2009. 6

[34] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 3, 5

[35] Patrick Min. Binvox. `http://www.patrickmin.com/binvox`, 2004 - 2019. 4

[36] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *CGIT*, 2006. 4

[37] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, June 2020. 2

[38] Fakir S. Nooruddin and Greg Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics*, 2003. 4

[39] Jean-Philippe Pons, Renaud Keriven, and Olivier Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *IJCV*, 72(2):179–193, 2007. 2

[40] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 1, 2, 3

[41] Google Research. Scanned objects dataset. `http://goo.gle/scanned-objects`, 2020. 2, 4

[42] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2, 5, 6

[43] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 2, 5

[44] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science and Business Media, 2010. 1, 2

[45] Shubham Tulsiani, Nilesh Kulkarni, and Abhinav Gupta. Implicit mesh reconstruction from unannotated image collections. *arXiv preprint arXiv:2007.08504*, 2020. 2

[46] Sara Vicente, Joao Carreira, Lourdes Agapito, and Jorge Batista. Reconstructing pascal voc. In *CVPR*, 2014. 6

[47] Charles Wheatstone. Contributions to the physiology of vision. —part the first. on some remarkable, and hitherto unobserved, phenomena of binocular vision. 1838. 1

[48] Jianfeng Yan, Zizhuang Wei, Hongwei Yi, Mingyu Ding, Runze Zhang, Yisong Chen, Guoping Wang, and Yu-Wing Tai. Dense hybrid recurrent multi-view stereo net with dynamic consistency checking. In *ECCV*, 2020. 2

[49] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, pages 767–783, 2018. 2

[50] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *NeurIPS*, 33, 2020. 2, 5

[51] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. Shelf-supervised mesh prediction in the wild. *arXiv preprint arXiv:2102.06195*, 2021. 2

[52] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, pages 1983–1992, 2018. 2

[53] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep networks as a perceptual metric. In *CVPR*, 2018. 4

[54] Yuyang Zhang, Shibiao Xu, Baoyuan Wu, Jian Shi, Weiliang Meng, and Xiaopeng Zhang. Unsupervised multi-view constrained convolutional network for accurate depth estimation. *IEEE Transactions on Image Processing*, 29:7019–7031, 2020. 2

[55] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, pages 1851–1858, 2017. 2

[56] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *SIGGRAPH*, 2018. 3