VortSDF: 3D Modeling with Centroidal Voronoi Tesselation on Signed Distance Field

Diego Thomas, Briac Toussaint, Jean-Sébastien Franco, Edmond Boyer

VortSDF: 3D Modeling with Centroidal Voronoi Tessellation on Signed Distance Field

Diego Thomas[1], Briac Toussaint[2], Jean-Sebastien Franco[2], Edmond Boyer[3]
[1]Kyushu University, [2]INRIA Grenoble Rhone-Alpes-LJK, [3]Meta Reality Labs

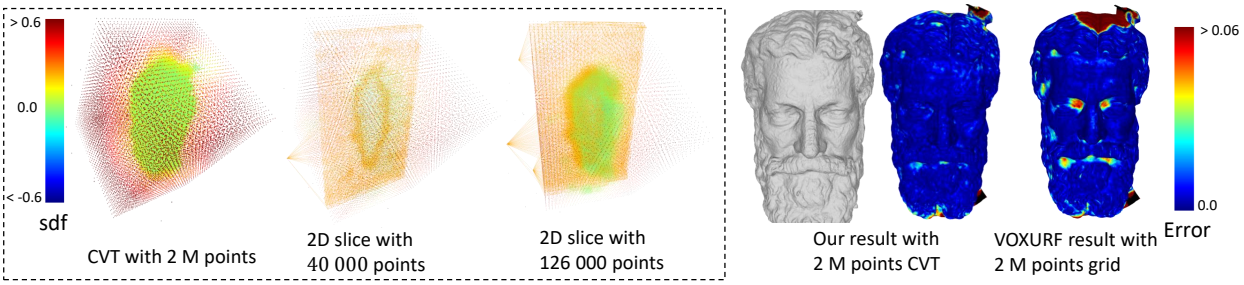

Figure 1. Our proposed method reconstructs detailed 3D surfaces using CVT that adapts to the reconstructed geometry.

Abstract

Volumetric shape representations have become ubiquitous in multi-view reconstruction tasks. They often build on regular voxel grids as discrete representations of 3D shape functions, such as SDF or radiance fields, either as the full shape model or as sampled instantiations of continuous representations, as with neural networks. Despite their proven efficiency, voxel representations come with the precision versus complexity trade-off. This inherent limitation can significantly impact performance when moving away from simple and uncluttered scenes. In this paper we investigate an alternative discretization strategy with the Centroidal Voronoi Tessellation (CVT). CVTs allow to better partition the observation space with respect to shape occupancy and to focus the discretization around shape surfaces. To leverage this discretization strategy for multi-view reconstruction, we introduce a volumetric optimization framework that combines explicit SDF fields with a shallow color network, in order to estimate 3D shape properties over tetrahedral grids. Experimental results with Chamfer statistics validate this approach with unprecedented reconstruction quality on various scenarios such as objects, open scenes or human.

1. Introduction

The 3D digitization of real-world objects is a foundational element for future technologies, that has motivated extensive research in recent decades. Among the primary solutions, multi-view capture systems have arisen as key tools to generate high-quality shape and appearance models of 3D scenes. Despite their effectiveness, the reconstruction of detailed geometry from multiple high-resolution images remains a challenging task due to the inherent ambiguities and complexity in the visual observations.

In multi-view 3D reconstruction, volumetric shape representations are increasingly prevalent, e.g. [15, 21, 33, 35]. This is in part due to their ability to relate shape properties to image observations through differential rendering. This has been extensively leveraged, particularly with networks, typically MLPs, which are trained to model shape geometry or appearance that best explain the image observations under photometric losses.

In the seminal work NeRF [21], volumetric radiance fields are estimated by integrating color and opacity along pixel rays. While this method produces highly realistic new viewpoints, the associated geometry extracted from opacity boundaries is artifact-ridden. Subsequent works, such as NeuS [33], address this by explicitly parametrizing the surface and its properties with signed distance fields, though they struggle to achieve the same image quality as volumetric radiance fields. Hybrid methods that combine explicit SDF grids with shallow networks for color offer benefits in both geometry and appearance modelling [29, 35]. Yet, the majority of such methods rely on some form of regular axis-aligned grids to discretize 3D observation spaces and are therefore sensitive to the inherently poor quality-to-parsimony trade-off of these representations. Moreover,

regular grids result in sub-optimal meshes when paired with the Marching Cubes algorithm [19] ubiquitously used for explicit mesh surface conversion.

Voxel grids uniformly discretize the observation space, regardless of the shape's location. Consequently, increasing resolution specifically near the shape surface requires non-trivial specializations. Octrees and HashMaps [15, 34] offer hierarchical space discretization, but their dynamic update during optimization and raymarching is cumbersome as they cannot straightforwardly follow the deformation during surface updates. Instead, we explore a hierarchical tetrahedral discretization guided by the Centroidal Voronoi Tesselation (CVT) algorithm. CVTs yield provably optimal discretizations and exhibit noteworthy advantages in our context: (i) Efficient ray marching along rays through tetrahedra; (ii) The ability to hierarchically up-sample and deform tetrahedral grids in adaptive fashion w.r.t the encoded shape surface.

As conventional 3D convolution or automatic differentiation do not easily extend to such non-uniform cell complexes, we here develop a complete representation and methodology to encode and optimise 3D fields over them. Specifically, given multiple images of a scene, our approach jointly optimizes a hierarchical CVT discretization with an associated neural field. The dual of the CVT defines a tetrahedral grid over which SDF and color feature values are stored. Images are rendered by sampling along pixel ray, with SDF and feature values interpolated at the sampled points. A color network is trained to predict the colors at sample points based on the interpolated features. We show that feature extraction and gradient back-propagation along rays can be efficiently performed over the tetrahedral dual of the CVT. Our hierarchical approach up-samples the CVT, after neural field convergence, at increasing levels of details, with a tenfold difference in grid resolution between subsequent levels.

We target several applications representative of different reconstruction scenarios such as objects, open scenes or humans and evaluate our method with Chamfer error statistics. Our experiments on public datasets like BlendedMVS [36] and 4D Human Outfit [2] demonstrate that this strategy yields significantly more reconstruction detail when compared to SOTA techniques with equivalent time and primitive budgets. Our main contributions are: (1) introducing CVT discretization for neural fields in multi-view reconstruction; (2) an implicit CVT optimization method that adapts to the optimized SDF field; (3) proposing an associated fast optimization framework.

2. Related works

Neural radiance fields have been a highly active topic in recent research. Initially popularized for the problem of image-based rendering by NeRF [21], it has been immediately explored by the community as a way to approach the multi-view 3d reconstruction problem [37]. NeRF relies on an implicitly parameterized neural function to approximate the plenoptic function [12], and leverages volumetric differential rendering [18] to optimize the neural parameters. While providing volumetric continuity in the observation space, the neural optimization proved initially slow and inefficient for accurate and detailed surface geometry extraction, leading to several research threads of improvement.

More detailed volumes have been pursued by addressing limitations of neural field representations, *e.g*. in the frequency domain with Mip-Nerf [3,5], in the bounded restriction of the spatial domain for background inclusion with Nerf++ [39], or both [4]. They do not however trivially transpose to higher detail surface extraction as proposed.

Surface-based representations were introduced to address volumetric approaches' inherently limited ability to encode continuous shape surfaces. While UNISURF [23] centered its representation on occupancy to this goal, inference based on signed distance fields (SDF) [24] proved more successful, as initially explored by IDR [37], and later improved by Neus [33] by modeling shape uncertainty through a local volume integration around the main surface mode with a Gaussian opacity profile, which remains competitive to this day and serves as a key benchmark.

Computational efficiency has been an essential topic to bring Nerf and derivatives to a more usable realm than the initial 30+ hour optimization time. Parting with the implicit network representation in favor of an explicit uniform volumetric grid, either with a fully explicit non-neural radiance parametrization [11], or using shallow networks parametrized by grid features embedded in the volume [27, 30] or onto coordinate-projected hyperplanes for parameter dimensionality reduction [6, 7, 10], have been central ideas leading to computational time improvements of several orders of magnitude. Of particular interest to us are the attempts to spatially sparsify and hierarchize the volume to focus resources on shape boundaries, using *e.g*. an octree structure [16, 38], and simultaneously improve representation performance through multiscale features [22]. These order of magnitude improvements to volumetric methods were recently transposed to the realm of surface methods with predictive performance that is on par [34] or improved [35]. Recent methods leverage the representational advantage for surface extraction of extended scenes [15]. Yet, all these representations are fundamentally tied to an axis aligned grid core, which we show inhibits access to even higher surface estimation performance.

Adaptive non-uniform spatial discretization. A number of methods have explored non-uniform cell sampling or de-

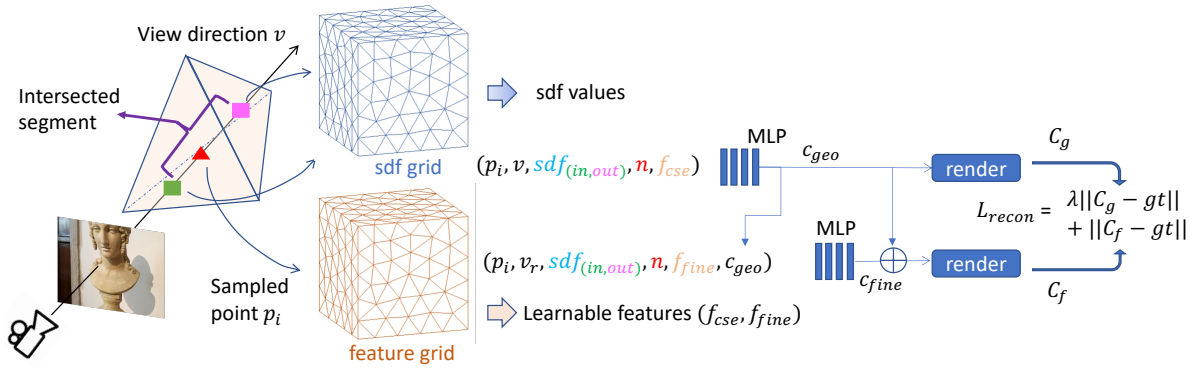

Figure 2. We propose a volumetric optimization framework that combines explicit SDF fields and learnable features with two shallow color networks, in order to estimate 3D shape properties over tetrahedral grids.

composition of space as a way to optimally sample and focus resources on the surface vicinity. The recent splatting approaches [8, 13] for example opt for blob primitives whose extent and positions are optimized jointly with adjoining opacities and colors, to explain input views. Of particular relevance to our work, DeRF [26] optimizes a Voronoi cell decomposition to improve overall performance in a volumetric setting, but does not explore coarse-to-fine and surface adaptive sampling as we propose. TetraNerf [14] uses a Delaunay marching structure to slightly accelerate volumetric radiance field inference down to 10-20 hours, but the decomposition is fixed, based on a precomputed set of COLMAP points. None of these approaches deal with surface extraction and leverage the cell structure to efficiently perform this task, as proposed.

To provide this key improvement and still benefit from the associated computational performance boost, we note that two surface extraction algorithms based on CVTs [32] were shown to significantly outperform Marching Cubes and Delaunay Tessellation for exactly this task. The first such variant clips the cells by looking at intersection between edges and implicit surface; and the second one refines the tessellation by adding vertices at the intersection between surface and bisector of edges. The work demonstrates CVTs to be a regular tessellation guaranteed to be manifold, with excellent surface accuracy achieved with a predefined number of cells. It is also notable that efficient ray-marching algorithms exist for cell-based 3D structures [1]. Our novel framework builds on these properties to propose optimally adapted sampling of 3D space around the surface of interest, for the purpose of neural adaptive 3D surface reconstruction.

3. Method

Given a set of multiple images, our method jointly optimizes an SDF field, discretized on a hierarchical CVT, and two view-dependent shallow color networks to predict view-dependent color at any 3D location, inspired by recent works on direct SDF optimization with uniform voxel grids [30, 35]. Those color networks are queried at sampled points along pixel rays defined by the camera location and random pixel coordinates.

As illustrated in the method outline (Figure 2), the first color network takes as input a 3D location, a 3D direction vector, SDF values, a normal vector and 8-dimensional learnable features f_{cse}. The second network is a color refinement network that takes as additional input the coarse color and uses an additional 8-dimensional learnable feature set f_{fine}. Note that the refinement network takes the reflected vector v_r instead of the viewing direction v in order for the network to focus more on specular effects.

SDF and feature values are optimized by back-propagating a photometric loss from the sampled points to the CVT sites. After convergence, the CVT is iteratively refined and up-sampled non-uniformly with respect to the current estimate of the SDF field, therefore increasing the shape resolution at the vicinity of the shape surface.

3.1. Centroidal Voronoi Tesselation

A tessellation of a 3D space is a disjoint set of polyhedron that fills the 3D space of interest. CVTs are used in a wide range of applications, in Visual Computing and beyond, as it provides an elegant tool to compute a regular and optimized discretization of the 3D space [31]. Given a set of points, called sites, a Voronoi tessellation partitions the space into regions around the sites and is dual to the Delaunay triangulation of these sites.

A Voronoi cell V_i is associated to its site x_i and is composed of all points that are closest to x_i:

$$\{p \in R^3 \,/\, ||p - x_i|| < ||p - x_j||,\ j \in [1, K], j \neq i\}$$

Voronoi cells are delimited by segments in 2D and convex polygons in 3D that are the intersection of the bisectors between pairs of sites. When the sites coincide with the

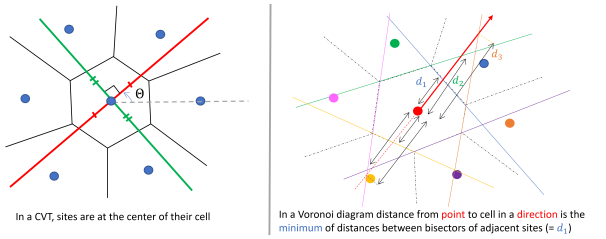

Figure 3. The site locations are optimized using an approximated CVT algorithm that does not explicitly identify the Voronoi cells (left) but consider the neighboring bisectors instead (right).

centroids of the associated cells, the tessellation is called a Centroidal Voronoi Tessellation. Intuitively, CVT cells optimally partition the input domain as k-means clusters minimizing a variance or quantification error [9].

3.2. Coarse-to-fine Centroidal Voronoi Tesselation

Central to our proposed method is densifying the continuous Signed Distance Function (SDF) discretization near the surface. Unlike previous work such as DMTet [28], we do not know the true SDF. Instead, we simultaneously optimize both the SDF values and their discretization, which presents a significantly greater challenge. Given a 3D region of interest, such as a bounding box, we begin with a coarse uniform grid, for example, $16 \times 16 \times 16$ in our experiments. From this grid, a coarse initial Centroidal Voronoi Tessellation (CVT) is generated by minimizing the CVT energy that moves the sites to the center of their cell. Additionally, we add the center of each camera as complementary sites to the CVT in order to speed-up ray-marching operations, as will be elaborated on in Section 3.3.

After the optimization of the color network parameters and of the SDF and feature values within the CVT, we upsample the discretization by adding a point at the center of each surface-crossing edge of the dual of the CVT, *i.e.* of the associated tetrahedral Delaunay mesh. Edges that present a SDF value smaller than 1.5 the edge length at one end point are also up-sampled. The CVT energy is then minimized, at each up-sampling iteration, to ensure that the vertices are locally uniformly distributed, which is crucial to reduce sampling artifacts. This process is repeated until the expected level of details is reached.

We keep a KD-tree for each level of discretization and compute the K-nearest neighbors of each site to enable propagation of gradients and smoothing.

3.2.1 SDF-aware implicit CVT optimization

High quality 3D reconstructions require very fine discretization, in practice millions of sites. At such scale, traditional CVT optimizations, even those based on the L-BFGS quasi-Newton method [17], become computationally prohibitive, since the explicit boundaries of each cell must be recomputed every time the sites are moved. While full GPU solutions have been proposed [25] they still struggle with millions of sites. In addition, we expect the CVT to adapt locally to the optimized SDF field so that the shape surface is materialized by the Voronoi cell faces. Inspired by [20], we propose to build an approximate CVT, with significant computational benefits yet providing nearly equivalent behaviour in cell spacing. With millions of sites, computing the exact Voronoi diagram with off-the-shelf libraries takes about 5 minutes. Then 30 iterations of the CVT optimization requires about 2.5 hours. With our approximated CVT, one iteration takes about 1 second and we can run 300 iterations in about 5 minutes.

As mentioned earlier, a CVT is obtained when its sites are the centroids of the associated Voronoi diagram. Traditional algorithms iterate therefore between estimating the Voronoi diagram and moving the sites towards the cell centroids. With the aim to avoid the explicit estimation of the Voronoi diagram when optimizing the site positions, we observe that a CVT site should be equidistant from the border of its Voronoi cells in any direction around the site (see fig. 3). Such Voronoi cells are spanned by the bisectors between sites. Hence we propose to define our CVT loss using differences between distances to the neighbouring bisectors, instead of distances to identified Voronoi cells. To this purpose we randomly sample directions around a site using two angles (Θ, Φ) and sum the distance differences to the closest bisectors along each direction. In practice we use $N = 24$ nearest neighbours[1] around a site to estimate the bisectors and sample in 3 orthogonal directions obtained by rotating the cartesian basis with the random angles (Θ, Φ). Note that these angles are different for each site and change at each iteration. The CVT loss writes then

$$L_{CVT} = \frac{1}{2} \sum_{s_i} \sum_{j=0,1,2} (d(\mathbf{s_i}, \mathbf{e_j}(\Theta, \Phi)) - d(\mathbf{s_i}, -\mathbf{e_j}(\Theta, \Phi)))^2. \tag{1}$$

$\{\mathbf{e_0}(\Theta, \Phi), \mathbf{e_1}(\Theta, \Phi), \mathbf{e_2}(\Theta, \Phi)\}$ is the rotated cartesian basis and $d()$ is defined as

$$d(\mathbf{s_i}, \mathbf{r}) = min_{\mathbf{s_j} \in N(\mathbf{s_i})}(\|\mathbf{s_i} - b(\mathbf{s_i}, \mathbf{s_j}, \mathbf{r})\|_2), \tag{2}$$

where $b(\mathbf{s_i}, \mathbf{s_j}, \mathbf{r})$ is the distance from $\mathbf{s_i}$ to its bisector with $\mathbf{s_j}$ in the direction $\mathbf{r}$.

In addition, in case when the SDF values of $\mathbf{s_i}$ and $\mathbf{s_j}$ have opposite signs we move the bisector plane so that it lies on the 0 crossing. The intuition is that the bisectors of the CVT cells coincide with the middle of the dual tetrahedra. Thus the points that are sampled at the middle of the segments that intersect ray and tetrahedron will lie closer to

[1] We re-estimate the K-nearest neighbors every 100 iterations.

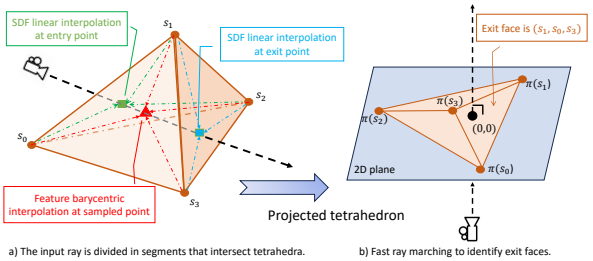

Figure 4. a) The SDF values at the intersecting segments extremities are obtained by linear interpolation of the SDF values at the entry and exit faces. Features at sampled points are linearly interpolated from the 4 vertices of the tetrahedron. b) The viewing ray exit face is the face to which camera center belongs after projecting the tetrahedron into the plane perpendicular to the viewing ray.

the 0 level set. As a consequence the color networks would be optimized closer to the surface.

We implement the losses on the GPU, compute the explicit gradients and use PyTorch Adam optimizer to minimise L_{CVT} with 300 iterations. After the CVT is optimized, the Delaunay tetrahedra are computed to restart the SDF and features optimization.

3.3. Differentiable rendering in a tetrahedral mesh

Differentiable rendering is a key component of volumetric reconstruction strategies. In such rendering, efficient sampling of points along pixel viewing lines is essential. However, doing efficient sampling on non-uniform tetrahedral grids requires specific algorithms.

We adapt the 32 bits tetrahedral structure proposed in [1] for fast ray-marching. We walk through the tetrahedral mesh and along a viewing line by: (i) finding the exit face in the current tetrahedron; (ii) identifying the next tetrahedron. To find the exit face we project all vertices in the plane centered at the camera center and which normal equals the ray direction vector. Then the exit face is the face that contains the origin in the projected coordinate system (the entry face is not counted). See figure 4 for illustration of the process.

In contrast to [1] and [14], we include the camera centers in the tetrahedral mesh. This way, the entry point of a ray in the tetrahedral mesh is easily obtained as we only need to check the tetrahedron that contains the camera center. The output of our ray marching algorithm is a list of visited tetrahedra in association with their entry and exit points. To speed up computations, we prune out all the tetrahedra that exhibit a null contribution to the rendered image.

3.3.1 Volumetric rendering

The 3D ray $(\mathbf{o}, \mathbf{v})$ originates from the camera center $\mathbf{o}$ in the direction $\mathbf{v}$. When intersecting the tetrahedral grid, it is split into n segments $\{s(t) = [\mathbf{in}(t) : \mathbf{out}(t)] | 0 \leq t \leq n\}$ with non-null contribution to the accumulated color using the output of our proposed ray-marching algorithm[2]. For each segment we sample a point $\mathbf{p}(t) = \frac{\mathbf{in}(t)+\mathbf{out}(t)}{2}$ at the middle of the segment and query the color networks g_{geo} and g_{fine} at each point $\mathbf{p}(t)$. The color of the ray is then:

$$\mathbf{C}(\mathbf{o}, \mathbf{v}) = \sum_{t=0}^{n} \omega(t) c(\mathbf{p}(t), \mathbf{v}), \tag{3}$$

$$\omega(t) = \alpha(t) T(t), \tag{4}$$

$$T(t) = \prod_{i<t} (1 - \alpha_i), \tag{5}$$

where $\mathbf{C}(\mathbf{o}, \mathbf{v})$ is the estimated color for this ray, $\omega(t)$ a weight for the point $\mathbf{p}(t)$, and $\mathbf{c}(\mathbf{p}(\mathbf{t}), \mathbf{v})$ the color at the point $\mathbf{p}$ along the viewing direction $\mathbf{v}$ that is the output of either g_{geo} or g_{fine}. $\alpha(t)$ is the transmittance at t^{th} point and $T(t)$ is the accumulated transmittance. Different strategies exist to compute the transmittance, a standard one being to use a volume rendering formulation [21]. A more efficient strategy is to use the normalized S-density as weights [33]:

$$\alpha_t = clip\left(\frac{1 + e^{-\beta sdf(\mathbf{in}(t))}}{1 + e^{-\beta sdf(\mathbf{out}(t)}}, 0, 1\right), \tag{6}$$

where $sdf(\mathbf{in}(t))$ is the SDF at the entry point in the t^{th} segment and β is a scale factor that is gradually increased during optimization. The $clip$ function clamps the transmittance value between 0 and 1. When β becomes large, long segments may generate too small gradients as the differences in SDF values between the entry and exit points become too large. Consequently, we further subdivide segments that cross the surface.

The SDF values for each entry and exit points are obtained by linearly interpolating the SDF values at the three vertices of the entry and exit faces (respectively). These two SDF values form the geometric feature f_{geo} of the coarse color network. Similarly we compute the normal vectors of the entry and exit points and add these normal vectors as input of the refinement network.

We render colors for both the coarse and fine color networks and obtain the corresponding colors $\mathbf{C_i^{geo}}$ and $\mathbf{C_i^{fine}}$, respectively, at pixel i. Given the ground truth color $\mathbf{C_i}$ at this pixel, we get the following photometric data term for the SDF optimization:

$$E_{rgb} = \sum_{i \in [1:N]} \frac{\lambda \|(\mathbf{C_i} - \mathbf{C_i^{geo}})\|_2^2 + \|(\mathbf{C_i} - \mathbf{C_i^{fine}})\|_2^2}{(\|\mathbf{C_i}\|_2 + \epsilon)},$$

where λ is a weight (1 at the coarser stage then 0.5) and ϵ is a small value (0.1 in our experiments).

3.4. SDF field regularization

While fully implicit surface representations are naturally regularized by the weights of the neural network, special attention is required to regularize discrete SDF fields.

[2] We use a maximum of $n = 1024$ segments per ray in our experiments.

3.4.1 Normal Smoothing

Since SDF values are linearly interpolated within a tetrahedron, we can express the gradient of the SDF function within a tetrahedron as a function of the values at the tetrahedron's vertices and of the spatial gradients $\nabla \mathbf{w_i}$ of the interpolation weights w_i. By solving a linear system in each tetrahedron we compute the gradient vector associated to each tetrahedron. We can then express the spatial gradient of the SDF field inside each tetrahedron as a function of the SDF values at the four summits.

$$\nabla \mathbf{sdf_t} = \sum_i \nabla \mathbf{w_i} sdf(i). \tag{7}$$

Each gradient $\nabla \mathbf{sdf_t}$ linearly depends on the SDF values at the 4 vertex of a tetrahedron.

We use a smoothing regulator that aligns the gradients of the sdf and the gradients of the smoothed sdf values.

$$L_{reg} = 0.5 \sum_t (1 - (\frac{\nabla \mathbf{sdf_t} \cdot \nabla \mathbf{sdf_t}^{smooth}}{\|\nabla \mathbf{sdf_t}\|_2 \|\nabla \mathbf{sdf_t}^{smooth}\|_2})^2).$$

3.4.2 Smoothing with K nearest neighbors.

One key difficulty in using tetrahedral grid is that given a 3D point in space we cannot directly access the tetrahedra that contains the point. As a consequence it is not possible to average SDF values sampled uniformly around a summit of the tetrahedral grid. In addition, the tetrahedral grid is non uniform so simply averaging SDF values at summits of adjacent tetrahedra creates unwanted artifacts. Therefore, we compute the smoothed SDF values on the CVT by using weighted average of SDF values of K-nearest sites in the current CVT. Note that the K-nearest neighbors are computed only once at each up-sampling step using the corresponding KD-trees.

We also use the total variation loss defined on the edges of the tetrahedral mesh.The final SDF gradient writes:

$$\frac{\partial sdf}{\partial t} = \frac{\partial E_{rgb}}{\partial t} + w_{reg} \frac{\partial L_{reg}}{\partial t} + w_{tv} \frac{\partial L_{TV}}{\partial t},$$

where w_{reg} and w_{tv} are weight factors and L_{TV} is a total variation loss.

4. Experiments

We evaluate the ability of our approach to reconstruct detailed 3D surfaces compared with the state-of-the-art methods NeuS [33], Neus2 [34] and Voxurf [35]. We use the code provided by the authors for both NeuS, Neus2 and Voxurf with recommended parameters and run our experiments on an RTX3090 GPU. We qualitatively and quantitatively evaluate our method on a subset of the BlendedMVS dataset [36] and a subset of the 4D Human Outfit dataset [2].

4.1. Metrics

We evaluate the quality of the estimated geometry using the available ground truth 3D meshes and a point to mesh Chamfer distance. We compute these errors from the ground truth mesh to the predicted meshes to obtain accuracy Acc[3] and from the predicted mesh to the ground truth mesh to obtain completeness $Compl$. We clip the errors to a maximum distance of 0.1 meter. Note that smaller values are better for these metrics.

4.2. Experiments on BlendedMVS dataset

We used 7 scenes of the blendedMVS dataset, to evaluate the ability of our method to reconstruct detailed geometry by comparing to the state of the art. Two different scenarios occur: uniform scenes where objects occupy most of the bounding box, which favors in principle uniform discretizations. Second, more open scenes with significant amount of empty space in the bounding box (Stone or Durian) to confirm the advantage of our adaptive discretization.

We evaluate our method at the last 2 levels of densification. Our method usually terminates with about 2M points at lvl 5 and about 500K points at lvl 4[4]. In comparison VOXURF uses a uniform grid of $256 \times 256 \times 256$ voxels, which corresponds to about 16M points.

Table 1 shows the quantitative evaluation compared to NeUS, NeuS2 and Voxurf. Our method almost always improves the accuracy or ranks second to best of the reconstructed 3D mesh compared to these SOTA methods. As expected, we observe more significant gains against VOXURF with large scenes for which uniform space discretizations are not well suited. This confirms that using an adaptive CVT to support the SDF field optimization is an effective solution that yields higher frequency details in the reconstructed geometry and appearance.

Our experimental results also show that, our method retrieves accurate reconstructed meshes already at lvl 4, which only contains about 500K points (30 times less than VOXURF). Our method even obtains better results than VOXURF at lvl 4 for the data Sculpt. Our method is also able to produce significantly fewer artifacts than other methods as shown by the completeness results. Our method converges in about 30mn at level 4 and about 50mn at level 5. In comparison, NeuS converges in about 8h30mn, NeuS2 in 5mn and VOXURF in 45mn.

Figure 5 shows qualitative comparisons. They demonstrate that denser discretizations around the surface can effectively yield higher frequency details and less outliers than other methods. Note in particular that Neus2 has some discretization artifacts in the reconstructed meshes.

[3] When computing Acc, cluttered regions are naturally removed.

[4] In our experiments we performed up-sampling every 10000 iterations.

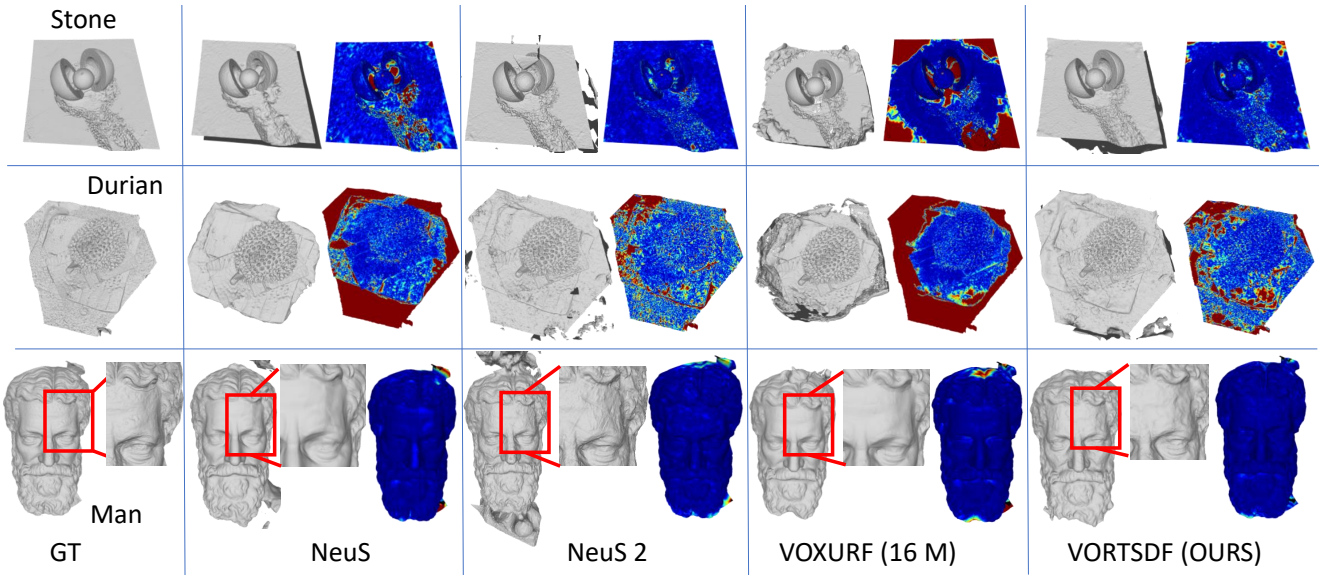

Figure 5. Comparative results we obtained with our method, NeuS and Voxurf on data "Stone", "Durian" and "Man" of BlendedMVS. We output the final 3D meshes using Marching Cubes (MC) for NeuS and Voxurf and Marching Tetrahedra for our method. We also show errors from ground truth meshes to predicted meshes as heatmaps.

Table 1. Average geometric accuracy *Acc* (mm) (lower is better) and completeness *compl*(mm) (lower is better) obtained with our method, NeuS2 and Voxurf, for each of the 7 test scenes. We highlight the **best** and second values.

	NeuS		NeuS2		Voxurf		Ours (lvl 4/ lvl 5))	
	Acc ↓	Compl ↓	Acc ↓	Compl ↓	Acc ↓	Compl ↓	Acc ↓	Compl ↓
Dog (31 images)	1.54	19.1	1.79	**4.81**	0.98	11.6	1.83/ **0.96**	18.72/ 9.88
Bear (123 images)	6.10	**11.25**	2.07	70.8	3.17	44.08	2.20/ **1.41**	33.3/ 26.3
Clock (143 images)	1.34	11.8	1.11	3.29	0.95	**0.82**	0.99/ **0.79**	1.67/ 1.35
Durian (124 images)	23.85	44.3	12.48	70.2	23.34	63.7	17.6/ **11.43**	42.8 / **41.1**
Man (24 images)	2.52	49.0	**2.18**	50.5	2.89	34.33	2.51/ 2.31	27.2 / **19.10**
Sculpt (79 images)	1.95	10.0	1.45	4.54	1.34	3.63	1.14 / **1.03**	4.41/ **3.40**
Stone (56 images)	10.30	43.28	4.35	46.99	14.53	44.26	8.48/ **3.93**	27.10 / **24.73**
Avg	6.31	26.37	3.75	31.38	6.79	26.54	4.96/ **3.30**	22.17 / **20.57**
Time	8h30mn		5mn		45mn		30mn /50mn	

4.3. Experiments on 4D Human Outfit dataset

We used 8 scenes of the 4D Human Outfit dataset. This dataset contains sets of 63 high resolution images with calibrated cameras and ground truth 3D mesh captured with a millimeter precision laser scanner. With this dataset we evaluate the ability of VortSDF to reconstruct detailed geometry such as clothing wrinkles. Note that the mannequins have arms close to the body, which makes it quite favorable to uniform discretization methods. Yet Table 2 shows that VortSDF obtained better results in most scenes. VortSDF always obtained better results than NeuS2. The advantage VortSDF is more evident on Figure 6 where we can see significantly better level of details in the face and wrinkles.

4.4. Ablation study

We evaluate the advantage of using our proposed CVT regularization on the Bear scene of Blended MVS. We run our proposed method with and without applying the CVT regularization after each up-sampling. Figure 7 shows that regularizing the discretization around the surface significantly improves the reconstruction accuracy.

4.5. Limitations

Our proposed method has some limitations that can be addressed in future works. Mainly, at each level of up-

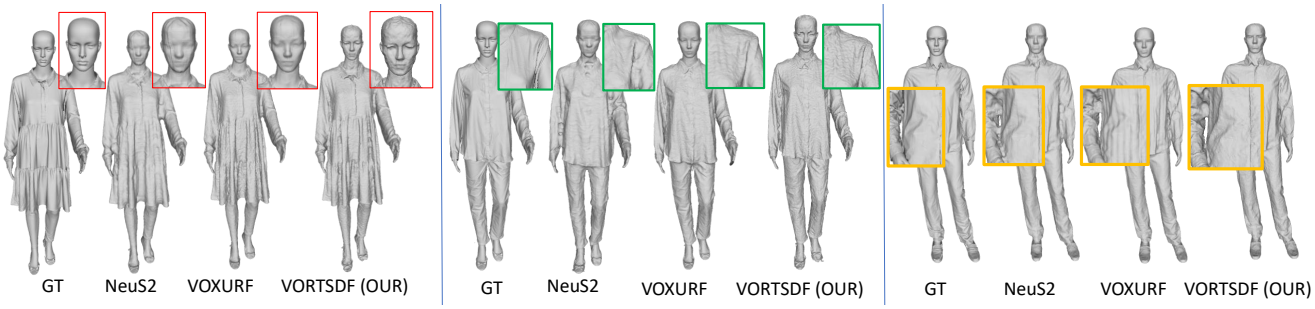

Figure 6. Comparative results we obtained with our method, NeuS and Voxurf on the 4D Human Outfit dataset. We output the final 3D meshes using Marching Cubes (MC) for NeuS and Voxurf and Marching Tetrahedra for our method.

Table 2. Average geometric accuracy *Acc* (mm) (lower is better) and completeness *Compl* (mm) (lower is better) obtained with our method, NeuS2 and Voxurf, for each of the 3 test scenes of 4D Human Outfit dataset. We highlight the **best** and second values.

	NeuS2		Voxurf		Ours (lvl 3/ lvl 4/ lvl 5)	
	Acc ↓	Compl ↓	Acc ↓	Compl ↓	Acc ↓	Compl ↓
f-cos-hx	4.64	3.17	1.76	2.32	2.54/ 1.65 / **1.56**	2.68/ 1.99 / **1.91**
f-jea-hx	5.35	4.89	2.25	2.54	2.20/ **1.73** / 1.78	2.25/ **1.91** / 1.94
f-opt1-hx	2.34	2.25	**1.57**	**2.09**	1.85/1.70/ 1.69	2.91/2.33/ 2.24
f-opt2-hx	3.23	2.77	1.87	**2.33**	2.36/1.89/ **1.84**	3.27/2.87/2.79
f-opt3-hx	2.27	2.21	1.39	1.91	1.51/ 1.33 / **1.29**	2.14/1.93/ **1.86**
f-sho-hx	2.26	2.11	**1.38**	1.84	1.60/1.45/ 1.39	2.03/ 1.77 / **1.73**
m-jea-hx	1.79	1.54	1.27	1.32	1.25/ 1.22 / **1.10**	2.16/2.09/ **1.09**
m-opt-hx	2.89	2.37	3.35	2.94	2.35/ 2.32 / **1.89**	2.49/ 2.35 / **2.15**
Avg	3.09	2.66	1.68	2.16	1.96/ 1.66 / **1.57**	2.49/ 2.15 / **1.96**

sampling the tetrahedral mesh must be computed to do ray marching. We used an off-the-shelf software to compute the tetrahedral mesh (open3D) that is a CPU version of Delaunay triangulation and takes significant amount of time for millions of input points. Using more advanced GPU implementation of the Delaunay tetrahedralization construction algorithm would provide significant speedups.

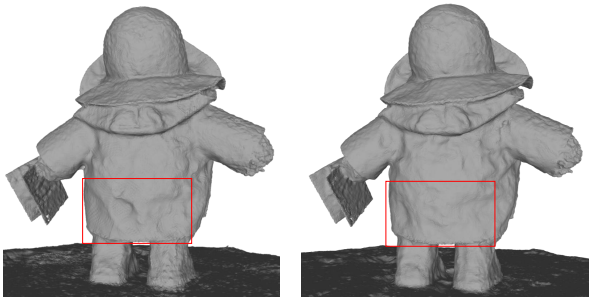

Figure 7. Regularizing the 3D discretization with our proposed approximated CVT loss significantly improve reconstruction quality.

5. Conclusion

We propose a novel method to reconstruct 3D geometry of a target scene from a set of multi-view images by optimizing a SDF field on a Centroidal Voronoi Tessellation. We formulate the optimization framework over the CVT and its dual tetrahedral mesh, designing an efficient framework to output detailed 3D shapes with competitive computation times. Our experimental results validate the key ideas in our proposed method and demonstrate at equivalent discretization level we can achieve a significantly higher level of extracted detail in a majority of situations, compared to competitive approaches. In a number of occurrences our method outperforms or is competitive with SOTA methods while using a magnitude lower level of discretization. Our work opens new promising directions toward detailed 3D reconstruction of large scale scenes under a contained computational time and GPU memory budget.

6. Acknowledgement

This work was in part supported by JSPS/KAKENHI JP23H03439 in Japan.

References

[1] Aytek Aman, Serkan Demirci, and Uğur Güdükbay. Compact tetrahedralization-based acceleration structures for ray tracing. *Journal of Visualization*, 25(5):1103–1115, 2022. 3, 5

[2] Matthieu Armando, Laurence Boissieux, Edmond Boyer, Jean-Sebastien Franco, Martin Humenberger, Christophe Legras, Vincent Leroy, Mathieu Marsot, Julien Pansiot, Sergi Pujades, Rim Rekik, Gregory Rogez, Anilkumar Swamy, and Stefanie Wuhrer. 4dhumanoutfit: a multi-subject 4d dataset of human motion sequences in varying outfits exhibiting large displacements. *Computer Vision and Image Understanding*. 2, 6

[3] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5855–5864, 2021. 2

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, 2022. 2

[5] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, pages 333–350. Springer, 2022. 2

[7] Anpei Chen, Zexiang Xu, Xinyue Wei, Siyu Tang, Hao Su, and Andreas Geiger. Factor fields: A unified framework for neural fields and beyond. *arXiv preprint arXiv:2302.01226*, 2023. 2

[8] Zhang Chen, Zhong Li, Liangchen Song, Lele Chen, Jingyi Yu, Junsong Yuan, and Yi Xu. Neurbf: A neural fields representation with adaptive radial basis functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4182–4194, 2023. 3

[9] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal voronoi tessellations: applications and algorithms. *SIAM review 41(4)*, 1999. 4

[10] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12479–12488, 2023. 2

[11] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5501–5510, 2022. 2

[12] Petr Kellnhofer, Lars C Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4287–4297, 2021. 2

[13] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):1–14, 2023. 3

[14] Jonas Kulhanek and Torsten Sattler. Tetra-nerf: Representing neural radiance fields using tetrahedra. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 3, 5

[15] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2

[16] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 2

[17] Yang Liu, Wenping Wang, Bruno Lévy, Feng Sun, Dong-Ming Yan, Lin Lu, and Chenglei Yang. On centroidal voronoi tessellation—energy smoothness and fast computation. *ACM Transactions on Graphics (ToG)*, 28(4):1–17, 2009. 4

[18] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019. 2

[19] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pages 347–353. 1998. 2

[20] Nissim Maruani, Roman Klokov, Maks Ovsjanikov, Pierre Alliez, and Mathieu Desbrun. Voromesh: Learning watertight surface meshes with voronoi diagrams. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14565–14574, 2023. 4

[21] 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 *European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 5

[22] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989*, Jan. 2022. 2

[23] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5589–5599, 2021. 2

[24] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 165–174, 2019. 2

[25] Nicolas Ray, Dmitry Sokolov, Sylvain Lefebvre, and Bruno Lévy. Meshless voronoi on the gpu. *ACM Transactions on Graphics (TOG)*, 37(6):1–12, 2018. 4

[26] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. Derf: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14153–14161, 2021. 3

[27] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14335–14345, 2021. 2

[28] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 4

[29] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[30] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5459–5469, 2022. 2, 3

[31] Li Wang, Franck Hétroy-Wheeler, and Edmond Boyer. A Hierarchical Approach for Regular Centroidal Voronoi Tessellations. *Computer Graphics Forum*, 35(1):152–165, Feb. 2016. 3

[32] Li Wang, Franck Hétroy-Wheeler, and Edmond Boyer. On volumetric shape reconstruction from implicit forms. In *European Conference on Computer Vision (ECCV)*, pages 173–188. Springer, 2016. 3

[33] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 1, 2, 5, 6

[34] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2, 6

[35] Tong Wu, Jiaqi Wang, Xingang Pan, Xudong Xu, Christian Theobalt, Ziwei Liu, and Dahua Lin. Voxurf: Voxel-based efficient and accurate neural surface reconstruction. In *International Conference on Learning Representations (ICLR)*, 2023. 1, 2, 3, 6

[36] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. *Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 6

[37] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020. 2

[38] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5752–5761, 2021. 2

[39] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2020. 2