Neural Kaleidoscopic Space Sculpting

Byeongjoo Ahn, Michael De Zeeuw, Ioannis Gkioulekas, Aswin C. Sankaranarayanan
Carnegie Mellon University

Abstract

We introduce a method that recovers full-surround 3D reconstructions from a single kaleidoscopic image using a neural surface representation. Full-surround 3D reconstruction is critical for many applications, such as augmented and virtual reality. A kaleidoscope, which uses a single camera and multiple mirrors, is a convenient way of achieving full-surround coverage, as it redistributes light directions and thus captures multiple viewpoints in a single image. This enables single-shot and dynamic full-surround 3D reconstruction. However, using a kaleidoscopic image for multi-view stereo is challenging, as we need to decompose the image into multi-view images by identifying which pixel corresponds to which virtual camera, a process we call labeling. To address this challenge, our approach avoids the need to explicitly estimate labels, but instead "sculpts" a neural surface representation through the careful use of silhouette, background, foreground, and texture information present in the kaleidoscopic image. We demonstrate the advantages of our method in a range of simulated and real experiments, on both static and dynamic scenes.

1. Introduction

Generating digital replicas of real-world objects from image measurements is a hard problem. Multi-view reconstruction approaches require a diverse set of viewpoints that provide full-surround coverage. A single camera, even if it is moving, can be insufficient for this problem, for example when the object under consideration undergoes dynamic motion and has complex shape. To capture the shape of a dynamic object, we would need simultaneous captures from multiple viewpoints. While a multi-camera system can provide such information, its cost and complexity can be prohibitive when we need to acquire objects with very complex appearance, geometry, and self-occlusions, and thus requiring very large numbers of viewpoints.

We use a kaleidoscope [6] to achieve single-shot full-surround 3D reconstruction for general dynamic objects. A kaleidoscope is a configuration of multiple interreflecting mirrors imaged by a camera, and dramatically increases the number of viewpoints, thereby enabling a virtual *time-synchronized* multi-view system. However, 3D reconstruction with a kaleidoscope requires identifying the specific sequence of mirrors encountered by light reaching each camera pixel; this is equivalent to identifying the specific virtual view corresponding to the pixel, commonly referred to as the *labeling problem.* This labeling problem can be solved using time-of-flight cameras [36] or structured light systems [3], but such active techniques require long scan times that make them unsuitable for dynamic objects. On the other hand, prior art with passive illumination first constructs the visual hull of the object, then uses it to estimate its label [29]. This two-stage process often produces erroneous results, especially when the visual hull differs significantly from the true shape.

Figure 1. **3D printing of shape reconstructions.** The proposed neural kaleidoscopic space sculpting can generate replicas of real objects with a range of shapes and reflectances. Reconstructed meshes are available on the project webpage [1].

Contributions. We propose a technique for full-surround 3D reconstruction with a single kaleidoscopic image. Our key insight is that a single pixel in a kaleidoscopic image is equivalent to multiple such pixels in its multi-camera counterpart. For example, the pre-image of the background pixel, which is the collection of 3D points that map to that pixel, in a kaleidoscope does not intersect with the object; this implies that is also a background pixel in all virtual views associated with it. Similarly, even a foreground pixel that intersects with the object can be used to carve out space since all of the light path prior to a ray's intersection with

the object can be considered as background.

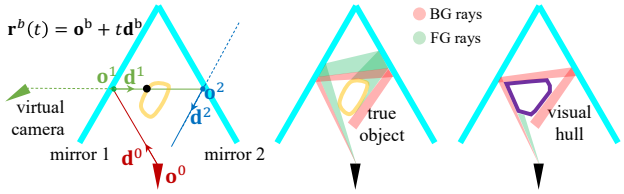
Armed with these insights on the nature of information encoded in a kaleidoscopic image, we propose a technique that we call *kaleidoscopic space sculpting*; sculpting sets up an optimization problem that updates a neural implicit surface [37] using a collection of cost functions that encode background information (to remove regions) and foreground regions (to add regions), as well as the texture of the object. Interestingly, our technique does not explicitly calculate the label information. Despite this, it provides robust single-shot full-surround 3D reconstructions. For dynamic objects, we apply our technique separately on each frame of a kaleidoscopic video, to obtain full-surround 3D videos. Figure 1 shows a gallery of objects placed beside their 3D printed counterparts, obtained using neural kaleidoscopic space sculpting.

Limitations. Our technique has a number of limitations, some of which are inherent in the use of a kaleidoscope. First, the size of objects we can scan is restricted by the kaleidoscope; for our lab setup, this constrains our technique to objects that fit in a sphere of diameter 4 inches. Second, the total number of pixels that we have at our disposal is limited to that of a single image sensor; divvying this pixel budget across the many (virtual) views results in lower resolution imagery, especially when we consider multi-view alternatives where the total pixel count grows linearly with the number of cameras. Third, our proposed technique is sensitive to foreground-background masking. We observed that automatic masking techniques produce erroneous masks that significantly reduce the quality of the final result. For this reason, we manually correct such mistakes prior to shape estimation.

2. Related Work

Full-surround 3D reconstruction. We refer to full-surround 3D reconstruction as reconstructing a 3D object in all its complexity. To achieve this, various approaches exist to obtain multiple viewpoints including turntable [21], single camera [7, 11, 13, 16, 19, 20, 22, 33, 34], multiple cameras [8, 12, 30], and non-optical methods (e.g., dip transform [2]). However, with the exception of multi-camera systems, these techniques cannot handle dynamic scenes.

Neural rendering. The proposed technique relies on recent advances in neural rendering and representations, which have found immense success in shape estimation [37] and novel viewpoint synthesis [5, 18, 31]), especially for scenes that are largely static. There have been some recent works towards extending these techniques for dynamic scenes [24, 25, 28]. The most successful among these work use class-specific priors for faces and the human body



(a) Labeling (b) Kaleidoscopic visual hull

Figure 2. **Background.** (a) Labeling is needed to utilize a kaleidoscopic imaging system as a virtual multi-view imaging system. (b) The kaleidoscopic visual hull [29] has the same silhouette as the true object but may have different labels because of the lack of label information in the silhouette.

[17, 26, 27, 32, 35]. In contrast, our technique provides a low-cost (vs. multi-view) solution for *general* objects (non-class specific). As our technique can reconstruct the scene frame by frame from a video, it can handle topology and color changes that cannot be modeled as rigid deformations.

3. Overview

Our problem formulation is as follows: given a kaleidoscopic image $\mathbf{I}$ and its silhouette mask $\mathbf{M}$, we seek to reconstruct a 3D surface that is photo- and silhouette-consistent. There are many solutions to the multi-view version of this problem, where the input is multiple images instead of a single kaleidoscopic image, including neural-rendering approaches [18, 37] that have recently found great success. However, multi-view approaches do not directly apply to our problem, because the kaleidoscopic image has an additional challenge—the *labeling* problem. We discuss labeling and other concepts related to our approach.

Labels. We define a label as the sequence of mirrors that the ray backprojected from a pixel encounters before intersecting the object [3, 29]. With the label information, and given the mirror geometry, we can decompose the kaleidoscopic image into the corresponding virtual camera images, by unfolding backprojected rays and assign the virtual viewpoints for each pixel. In particular, given the mirror geometry, we can determine the label from the number of bounces b for each pixel $\mathbf{x}$. As the label of an empty kaleidoscope (*i.e.*, without object) can be precomputed, $b \in \{0..B^{\emptyset}_{\mathbf{x}}\}$ can determine the label and thereby the pose of the virtual camera, where $B^{\emptyset}_{\mathbf{x}}$ is the length of the empty label [3]. Figure 2(a) shows a labeling example where the empty label $(1, 2)$ is the sequence of mirror numbers. If we can figure out the number of bounces before hitting the object being $b = 1 \in \{0, 1, 2\}$ ($B^{\emptyset}_{\mathbf{x}} = 2$), then the resulting label becomes (1), implying that this ray is equivalent to the ray from the virtual camera generated by the label (1). Thus, labeling allows kaleidoscopic imaging to serve as a virtual

multi-view imaging system and to reconstruct the object geometry via multi-view stereo (MVS). However, it is challenging to obtain the label since it itself is a function of the unknown object shape.

Kaleidoscopic visual hull. Visual hull [14, 15], initially used for multi-view images, is obtained by "carving" the voxels that are not consistent with the silhouette of the input images. A visual hull can provide a good approximation of 3D shape, if we have ample multi-view images and if the object under consideration is not overly complex.

Visual hulls have been utilized for kaleidoscopic images [29] where the background rays are reflected multiple times. To be specific, this method backprojects a ray from a pixel $\mathbf{x}$ and reflects it with mirrors, and computes the rays $\mathbf{r}_{\mathbf{x}}^b(t) = \mathbf{o}_{\mathbf{x}}^b + t\mathbf{d}_{\mathbf{x}}^b$ for each bounce b. $\mathbf{o}_{\mathbf{x}}^0$ is the real camera location, $\mathbf{d}_{\mathbf{x}}^0$ is the backprojected direction, $\mathbf{o}_{\mathbf{x}}^{b+1}$ is the intersection point between $\mathbf{r}_{\mathbf{x}}^b(t)$ and the mirrors, and $\mathbf{d}_{\mathbf{x}}^{b+1}$ is the reflected direction. Once the visual hull is obtained, the label can also be obtained simply by ray tracing to the visual hull inside the kaleidoscope. The key idea underlying this approach is that it requires only background pixels, which are easy to label as the background labels are independent of the object shape.

A useful property of the visual hull is that it is the maximal shape among silhouette-consistent shapes [14]. Based on this property, the possible labels can be narrowed down by ignoring the bounces that do not intersect with the visual hull [29]. To be specific, for a pixel $\mathbf{x}$, if we define the set of bounces hitting the visual hull as

$$\mathcal{H}_{\mathbf{x}}^{\text{vh}} = \{b \mid \mathbf{r}_{\mathbf{x}}^b(t) \text{ intersects with the visual hull}\}, \quad (1)$$

then the true bounce should satisfy $b^{\text{true}} \in \mathcal{H}_{\mathbf{x}}^{\text{vh}}$. Furthermore, when $|\mathcal{H}_{\mathbf{x}}^{\text{vh}}| = 1$, implying there is only one bounce intersecting with the visual hull, the label of the pixel can be uniquely determined, which is refered to as *a reliable pixel*.

However, the visual hull cannot resolve ambiguities inherent in silhouette information. First, it cannot reconstruct concave objects, as concavity cannot be captured by silhouette. Second, kaleidoscopic visual hull cannot disambiguate shapes with the same silhouette but different label maps. We refer to this effect as *virtual occlusion*, as it is caused by the visual hull not being carved when the background is occluded by a virtual object. The ambiguity causes errors in decomposing the images into multi-view images, which results in an incorrect shape (Figure 2(b)). To resolve these ambiguities and obtain better labels, it is essential to use texture information from foreground pixels, which requires labeling, setting up a chicken-and-egg problem.

Neural surface representation. We represent the surface using neural signed distance function (SDF) [23,37]. In particular, we optimize for the surface using IDR [37], which we briefly review here. In IDR, shape and texture are represented by two separate neural networks θ and ϕ, which are updated to achieve a silhouette-consistent and photo-consistent shape based on input mask images $\{\mathbf{M}\}$ and RGB images $\{\mathbf{I}\}$. To update these networks, IDR backprojects a ray $\mathbf{r}_{\mathbf{x}}(t)$ from a pixel $\mathbf{x}$, checks if it intersects with the current shape, and computes a rendered mask $\widehat{\mathbf{M}}(\mathbf{x}, \theta)$. The intersection is computed using the sphere tracing algorithm [10] since the surface is represented as SDF $f(\cdot; \theta)$. If the ray hits the surface (*i.e.*, $\widehat{\mathbf{M}}(\mathbf{x}) = 1$), it computes an RGB value $\widehat{\mathbf{I}}(\mathbf{x}; \theta, \phi)$ by giving the intersection point, surface normal, and view direction to the texture network. To be specific, it computes the point along the ray that has the minimum SDF, which is represented by

$$\mathbf{p}_{\mathbf{x}}(\theta) = \text{argmin}_{\mathbf{p} \in \mathbf{r}_{\mathbf{x}}(t)} f(\mathbf{p}; \theta). \quad (2)$$

IDR updates the shape and texture with the following loss:

$$\text{loss}(\theta, \phi) = \text{loss}_{\text{tex}}(\theta, \phi) + \text{loss}_{\text{mask}}(\theta) + \text{loss}_{\text{eik}}(\theta). \quad (3)$$

The texture loss $\text{loss}_{\text{tex}}(\theta, \phi)$ compares the rendered image and the input image as

$$\text{loss}_{\text{tex}}(\theta, \phi) = {}^{\lambda_{\text{tex}}}\!/_{|\mathcal{X}|} \sum\nolimits_{\mathbf{x} \in \mathcal{X}_{\text{in}}} |\mathbf{I}(\mathbf{x}) - \widehat{\mathbf{I}}(\mathbf{x}; \theta, \phi)|, \quad (4)$$

where λ_{tex} is the weight for texture loss, $\mathcal{X}$ is a set of pixels in the image, and $\mathcal{X}_{\text{in}} = \left\{\mathbf{x} \mid \mathbf{M}(\mathbf{x}) = 1 \text{ and } \widehat{\mathbf{M}}(\mathbf{x}) = 1\right\}$. The mask loss $\text{loss}_{\text{mask}}(\theta)$ compares the rendered mask and the input mask as

$$\text{loss}_{\text{mask}}(\theta) = {}^{\lambda_{\text{mask}}}\!/_{|\mathcal{X}|} \sum\nolimits_{\mathbf{x} \in \mathcal{X}_{\text{out}}} \text{BCE}\left(m(\mathbf{p}_{\mathbf{x}}(\theta)), \mathbf{M}(\mathbf{x})\right), \quad (5)$$

where λ_{mask} is the weight for mask loss, BCE is the binary cross-entropy function, $m(\cdot)$ is the soft binarization function for SDF defined as

$$m(\mathbf{p}) = \text{sigmoid}\left(-\alpha f(\mathbf{p})\right), \quad (6)$$

to make the loss differentiable, α is a parameter that controls the softness, and $\mathcal{X}_{\text{out}} = \mathcal{X} - \mathcal{X}_{\text{in}}$. The eikonal loss $\text{loss}_{\text{eik}}(\theta)$ enforces the gradient of the SDF to have a unit norm as

$$\text{loss}_{\text{eik}}(\theta) = \lambda_{\text{eik}} \mathbb{E}\left(\|\nabla_{\mathbf{p}} f(\mathbf{p}; \theta)\| - 1\right)^2, \quad (7)$$

where λ_{eik} is the weight for eikonal loss, and $\mathbf{p}$ is uniformly distributed in a bounding box of the scene.

4. Neural Kaleidoscopic Space Sculpting

Our goal is to jointly solve the labeling and geometry reconstruction from a kaleidoscopic image. To solve this problem, we propose a method that we call *kaleidoscopic space sculpting*, which updates the shape both in additive and subtractive ways without the necessity of labels.

4.1. Method

Our method uses two observations, which follow from the definitions of background and foreground pixels.

Observation 1 (Background rays)**.** *Background rays do not intersect with the object for all bounces, that is, for a background pixel* $\mathbf{x}$*,*

$$\forall b \forall t,\ f(\mathbf{r}_{\mathbf{x}}^{b}(t)) > 0, \tag{8}$$

where f *is the object SDF, and* $\mathbf{r}_{\mathbf{x}}^{b}(t)$ *is the pre-image of the pixel including mirror reflections.*

Observation 2 (Foreground rays)**.** *Foreground rays intersect with the object for at least one bounce, that is, for a foreground pixel* $\mathbf{x}$*,*

$$\exists b,\ \min_t |f(\mathbf{r}_{\mathbf{x}}^{b}(t))| = 0, \tag{9}$$

where $\mathbf{r}_{\mathbf{x}}^{b}(t)$ *is a ray bounced* b *times in a kaleidoscope after being backprojected from pixel* $\mathbf{x}$*, and* f *is the object SDF.*

From these observations, we can update the 3D shape by *carving* the points on the background rays, as they will never intersect the object; and *modeling* a point on the foreground rays, as there will be at least one intersection with the object. To do so, we need to answer the following questions: (1) How do we select points on the rays to carve and model? (2) how do we represent the shape and update it?

Point selection and labeling. The point selection for carving is straightforward, as all bounces can be used for carving in background rays. However, this is problematic for foreground rays, as we do not know which foreground ray bounce $\mathbf{r}_{\mathbf{x}}^{b}(t)$ will intersect with the object among all the possible bounces $b \in \{0..B_{\mathbf{x}}^{\emptyset}\}$. We need to select a point on the ray with the correct bounce b to avoid erroneous solutions. We can expect the current shape to be close to the true shape as the iteration proceeds, especially since an initial shape estimates can be determined by the background rays. Thus, we pick the foreground bounce with the minimum distance value in each iteration for selecting the modeling points. This turns out to be a simple yet effective approach. We now proceed to introduce how to update the shape with the silhouette constraint and texture constraint.

4.1.1 Silhouette constraint

To obtain a silhouette-consistent shape from a kaleidoscopic silhouette mask $\mathbf{M}$, we use the following loss function to update the neural SDF $f(\cdot;\theta)$ inspired by the IDR [37]:

$$\begin{aligned}\text{loss}_{\text{sil}}(\theta) =& \text{loss}_{\text{carve}}(\theta; \mathcal{P}_{\text{carve}}(\theta)) + \\ & \text{loss}_{\text{model}}(\theta; \mathcal{P}_{\text{model}}(\theta)) + \text{loss}_{\text{eik}}(\theta).\end{aligned} \tag{10}$$

$\mathcal{P}_{\text{carve}}(\theta)$ is the set of points for carving, $\mathcal{P}_{\text{model}}(\theta)$ is the set of points for modeling, $\text{loss}_{\text{carve}}(\theta)$ is the *carving loss*,

$$\text{loss}_{\text{carve}}(\theta) \equiv {}^{\lambda_{\text{carve}}}\!/_{|\mathcal{X}|} \sum\nolimits_{\mathbf{p} \in \mathcal{P}_{\text{carve}}(\theta)} \text{BCE}(m(\mathbf{p}), 0), \tag{11}$$

and $\text{loss}_{\text{model}}(\theta)$ is the *modeling loss*,

$$\text{loss}_{\text{model}}(\theta) \equiv {}^{\lambda_{\text{model}}}\!/_{|\mathcal{X}|} \sum\nolimits_{\mathbf{p} \in \mathcal{P}_{\text{model}}(\theta)} \text{BCE}(m(\mathbf{p}), 1), \tag{12}$$

where $m(\mathbf{p})$ is the soft binarization function defined in Equation (6), and λ_{carve} and λ_{model} are the weights for the carving loss and modeling loss, respectively. The difference in our method compared to IDR is that we use different point selection methods because rays are multiply reflected in the kaleidoscope with unknown labels. Next, we describe the point selection methods in detail for carving and modeling. Figure 3 illustrates the point selection methods.

Carving. Selecting the points for the carving $\mathcal{P}_{\text{carve}}(\theta)$ from background rays is straightforward as all the points $\mathbf{p}$ on the background rays should have positive SDF values (*i.e.*, Observation 1) and thereby $m(\mathbf{p})$ should be zero. However, instead of using all the points on the ray (*e.g.*, stratified sampling), we use a single point per each ray that has the minimum SDF value following IDR, which is more memory efficient and provides promising results. For a background pixel $\mathbf{x}$, these points can be expressed as

$$\mathcal{P}_{\mathbf{x}}^{\text{bg}}(\theta) = \left\{ \mathbf{p} \mid \operatorname*{argmin}_{\mathbf{p} \in \mathbf{r}_{\mathbf{x}}^{b}(t)} f(\mathbf{p};\theta),\ \forall b \in \{0..B_{\mathbf{x}}^{\emptyset}\} \right\}. \tag{13}$$

We collect $\mathcal{P}_{\mathbf{x}}^{\text{bg}}(\theta)$ for all background pixels $\mathbf{x} \in \mathcal{X}_{\text{bg}}$ and obtain $\{\mathcal{P}_{\mathbf{x}}^{\text{bg}}(\theta)\}_{\mathbf{x} \in \mathcal{X}_{\text{bg}}}$. We use $\mathcal{X}_{\text{bg}}$, which is different from $\mathcal{X}_{\text{in}}$ in IDR [29]. Then, the point set for carving becomes

$$\mathcal{P}_{\text{carve}}(\theta) = \{\mathcal{P}_{\mathbf{x}}^{\text{bg}}(\theta)\}_{\mathbf{x} \in \mathcal{X}_{\text{bg}}}. \tag{14}$$

Figure 3(a-b) illustrates the point selection for background pixels for carving the shape.

Modeling. For the modeling, we select a point on the foreground ray with the minimum distance to the current shape without knowing the label. As we will select the point with the minimum SDF value again on this ray, the selected point will have the minimum value on all foreground rays $\{\mathbf{r}_{\mathbf{x}}^{b}(t)\}_{b=0}^{B_{\mathbf{x}}^{\emptyset}}$. We select points only when the ray does *not hit* the object (*i.e.*, $\widehat{\mathbf{M}}(\mathbf{x}) = 0$) because the ray already hitting the object should not model the space further. Thus, the points for modeling $\mathcal{P}_{\mathbf{x}}^{\text{fg,nh}}(\theta)$ from a foreground pixel $\mathbf{x}$ is

$$\mathcal{P}_{\mathbf{x}}^{\text{fg,nh}}(\theta) = \begin{cases} \operatorname*{argmin}_{\mathbf{p} \in \{\mathbf{r}_{\mathbf{x}}^{b}(t)\}_{b=0}^{B_{\mathbf{x}}^{\emptyset}}} f(\mathbf{p};\theta) & \text{if } \widehat{\mathbf{M}}(\mathbf{x}) = 0, \\ \emptyset & \text{if } \widehat{\mathbf{M}}(\mathbf{x}) = 1. \end{cases} \tag{15}$$

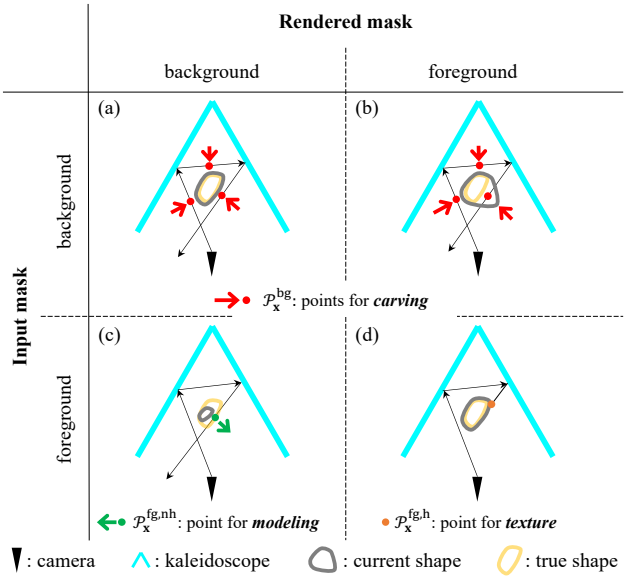


Figure 3. **Point selection for sculpting.** We select the points for sculpting differently in each case: (a-b) Background pixel (Equation (13)). (c) Foreground pixel not hitting the current shape (Equation (15)). (d) Foreground pixel hitting the current shape.

We collect $\mathcal{P}_{\mathbf{x}}^{\text{fg,nh}}(\theta)$ for all foreground pixels $\mathbf{x} \in \mathcal{X}_{\text{fg}}$ and compute $\{\mathcal{P}_{\mathbf{x}}^{\text{fg,nh}}(\theta)\}_{\mathbf{x}\in\mathcal{X}_{\text{fg}}}$. Then,

$$\mathcal{P}_{\text{model}}(\theta) = \{\mathcal{P}_{\mathbf{x}}^{\text{fg,nh}}(\theta)\}_{\mathbf{x}\in\mathcal{X}_{\text{fg}}}. \tag{16}$$

Figure 3(c) shows the point selection for foreground pixels for modeling the shape.

4.1.2 Visual hull constraint

Although the point selection method described above works promisingly, we introduce another constraint that can be naturally incorporated into our method—the visual hull constraint based on Equation (1). As the true bounce b^{true} satisfies $b^{\text{true}} \in \mathcal{H}_{\mathbf{x}}^{\text{vh}}$, we can exclude the bounces that are not in $\mathcal{H}_{\mathbf{x}}^{\text{vh}}$ (*i.e.*, outside the visual hull) from the modeling. The points on these bounces are represented as

$$\mathcal{P}_{\mathbf{x}}^{\text{ovh}} = \{\mathbf{p} \mid \underset{\mathbf{p}\in\mathbf{r}_{\mathbf{x}}^{b}(t)}{\text{argmin}}\, f(\mathbf{p};\theta),\ \forall b \notin \mathcal{H}_{\mathbf{x}}^{\text{vh}}\}, \tag{17}$$

and these points should be carved rather than modeled. We compute $\{\mathcal{P}_{\mathbf{x}}^{\text{ovh}}\}_{\mathbf{x}\in\mathcal{X}_{\text{fg}}}$ for all foreground pixels, then the point sets for carving and modeling becomes

$$\mathcal{P}_{\text{carve}}(\theta) = \{\mathcal{P}_{\mathbf{x}}^{\text{bg}}(\theta)\}_{\mathbf{x}\in\mathcal{X}_{\text{bg}}} \cup \{\mathcal{P}_{\mathbf{x}}^{\text{ovh}}(\theta)\}_{\mathbf{x}\in\mathcal{X}_{\text{fg}}}, \tag{18}$$

and

$$\mathcal{P}_{\text{model}}(\theta) = \{\mathcal{P}_{\mathbf{x}}^{\text{fg,nh}}(\theta) - \mathcal{P}_{\mathbf{x}}^{\text{ovh}}(\theta)\}_{\mathbf{x}\in\mathcal{X}_{\text{fg}}}. \tag{19}$$

Figure 4 illustrates the visual hull constraint. Note that our kaleidoscopic sculpting technique naturally handles unreliable pixels as well as reliable pixels.

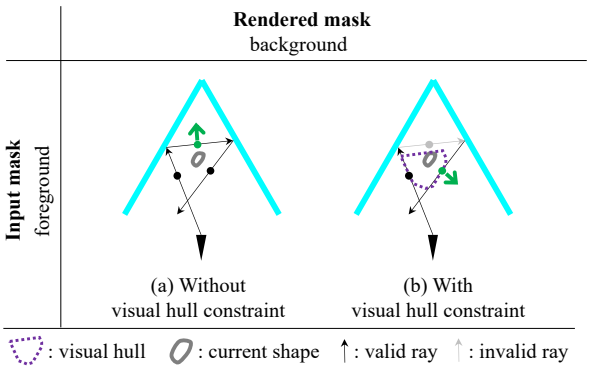


Figure 4. **Visual hull constraint.** The rays not intersecting with the visual hull never intersect with the true shape, so we can exclude the points on these rays from modeling.

4.1.3 Texture constraint

One of the main advantages of our technique compared to the visual hull technique [29] is that ours is designed to utilize foreground pixels as well as background pixels, and thereby can take advantage of texture information on the foreground pixels. On the other hand, the visual hull technique relies only on the background pixels, ignoring foreground pixels that provide the information of correspondence, and thus it cannot capture concavity, virtual occlusion, or high-frequency details on the surface. The texture loss is the same as the texture loss in IDR (Equation (4)), but we use the first intersection points along all the bounces as there can be multiple bounces intersecting with the object. We collect these points from all foreground pixels and use them for the texture loss as

$$\mathcal{P}_{\text{tex}} = \{\mathcal{P}_{\mathbf{x}}^{\text{fg,h}}(\theta)\}_{\mathbf{x}\in\mathcal{X}_{\text{fg}}}. \tag{20}$$

Figure 3(d) shows the point selection for foreground pixels for the texture constraint.

4.2. Information in a kaleidoscopic image

A kaleidoscopic image encodes information in a more complicated way compared to an image without any reflection—a kaleidoscopic pixel $\mathbf{x}$ is associated with multiple rays $\{\mathbf{r}_{\mathbf{x}}^{b}(t)\}_{b=0}^{B_{\mathbf{x}}^{\emptyset}}$, whereas a regular pixel is associated with just a single ray (*i.e.*, backprojected ray).

Conventional kaleidoscopic imaging methods [3, 36] chose the ray out of all the bounces that is intersecting with the object (*i.e.*, labeling) and it could achieve the benefit of the redistribution of the ray in the kaleidoscope. Ours not only achieves this benefit but also achieves another benefit of using multiple rays $\{\mathbf{r}_{\mathbf{x}}^{b}(t)\}_{b=0}^{B_{\mathbf{x}}^{\emptyset}}$ per pixel either for carving or modeling. Interestingly, not only the background rays but also the foreground pixels can be multiply used by making use of the rays not hitting the visual hull for carving.

Table 1. **Comparison of the kaleidoscopic visual hull [29] and the proposed neural kaleidoscopic space sculpting.**

	Visual hull [29]	Space sculpting (ours)
Background pixels	✓	✓
Foreground pixels	✗	✓
Texture	✗	✓
Shape representation	voxel	neural SDF
Shape update	subtract	add & subtract

Comparison to kaleidoscopic visual hull [29]. Table 1 summarizes the differences between the visual hull and ours. They use the same number of rays for background pixels but ours additionally uses foreground pixels and thereby texture information. Also, we have more rays to carve as we collect points for carving $\mathcal{P}_{\mathbf{x}}^{\text{ovh}}$ from foreground pixels.

5. Implementation

Hardware and calibration. We developed a hardware prototype for kaleidoscopic imaging, comprising an RGB camera (FLIR Blackfly S BFS-U3-200S6C) and four triangular mirrors (Edmund optics 46-656 customized). The image size is 5472×3648. We calibrated the kaleidoscope following [3], which uses a reference sphere object of diameter $40\,\text{mm}$. The calibration error is $346\,\mu\text{m}$ in sphere fitting error and 1.327 pixels in reprojection error.

Implementation detail. We obtain the input mask by computing the difference between the images with and without the object, and manually refining the mask using Adobe Photoshop for some pixels. We evaluate the effect of mask refinement in the supplement. For the shape and texture network, we use the same architecture as IDR [37]. The shape network is initialized to be an SDF of an approximate unit sphere [4]. We provide the results with a different initialization in the supplemental where the network is pretrained to represent the visual hull using [9]. For the final mesh results, we ran a marching cube from $f(\cdot;\theta)$ and extracted the largest connected part for the final results [37]. The weights used for the loss are $\lambda_{\text{tex}} = 1.0$ $\lambda_{\text{eik}} = 0.1$ $\lambda_{\text{model}} = 100.0$ $\lambda_{\text{carve}} = 20.0$. Note that λ_{carve} is smaller than λ_{model} by 5, which is about the average number of rays per pixel.

6. Results

We evaluate the proposed method through both simulated and real experiments. Our implementation and data are available on our project page [1].

6.1. Simulated experiments

We simulate a kaleidoscopic image of an armadillo of height $60\,\text{mm}$, and Figure 5 and Table 2 show the qualitative and quantitative results for the simulated experiment.

Table 2. **Quantitative results for simulated experiments.**

Method	PSNR↑ [dB]	chamfer↓ [μm]	mask err↓ [%]	label err↓ [%]
visual hull [29]	-	9.11	0.93	4.12
IDR w/ VH label (first ray)	6.92	2.3×10^3	26.8	37.71
IDR w/ VH label (last ray)	12.21	1.8×10^3	21.63	20.67
IDR w/ VH label (all rays)	21.84	4.42	0.80	2.74
IDR w/ VH label (reliable)	21.89	4.39	0.78	2.64
IDR w/ GT label	22.53	1.87	0.62	1.42
ours, SIL	13.73	5.53	0.95	3.11
ours, SIL+VH	13.95	6.22	0.80	2.45
ours, SIL+TEX	23.04	2.56	0.59	1.66
ours, SIL+VH+TEX	22.85	1.87	0.57	1.39

Comparisons to baseline methods. We compare our method with other kaleidoscopic imaging methods: kaleidoscopic visual hull [29], and IDR after decomposing the kaleidoscopic image into virtual multi-view images. For the virtual multi-view decomposition, we test several variations in the use of background rays while using the foreground label from the visual hull: (1) use only first background rays, (2) use only final background rays, and (3) use all background rays. Also, we have: (4) use only reliable pixels, and (5) use the ground-truth labels with all background rays. Figure 5(a-e) shows the results for the baselines (1-5), and Figure 5(f) shows the kaleidoscopic visual hull.

Ours with all constraints performs best overall qualitatively and quantitatively, with the exception of PSNR where it ranks second. There are artifacts near the left ear of the armadillo in (Figure 5(c-d)) because of the incorrect label from the visual hull. This is caused by the virtual occlusion as the background near the left ear is occluded by another virtual object and observed as a foreground, and thereby the space is not carved. Our method does not have this problem as we are jointly solving the labeling and 3D reconstruction.

Ablation study. We conduct an ablation study with the silhouette constraint, visual hull constraint, and texture constraint as shown in Figure 5(g-j) and Table 2. Silhouette constraint provides the result without details, and texture constraint captures the detail on the surface. Adding the visual hull constraint additionally improves the result.

6.2. Real experiments

We capture a kaleidoscopic image with our hardware prototype and reconstruct the 3D shape. The objects are placed inside the kaleidoscope either directly on the mirror or hung with strings. The size of the object is within $100\,\text{mm}$ (*e.g.*, *Treble clef* has the height $100\,\text{mm}$).

Scanned objects. Figure 9 shows the reconstruction results on the real objects exhibiting a range of shapes with different visibilities and reflectances with different textures and materials. The label map is visualized by sorting the la-

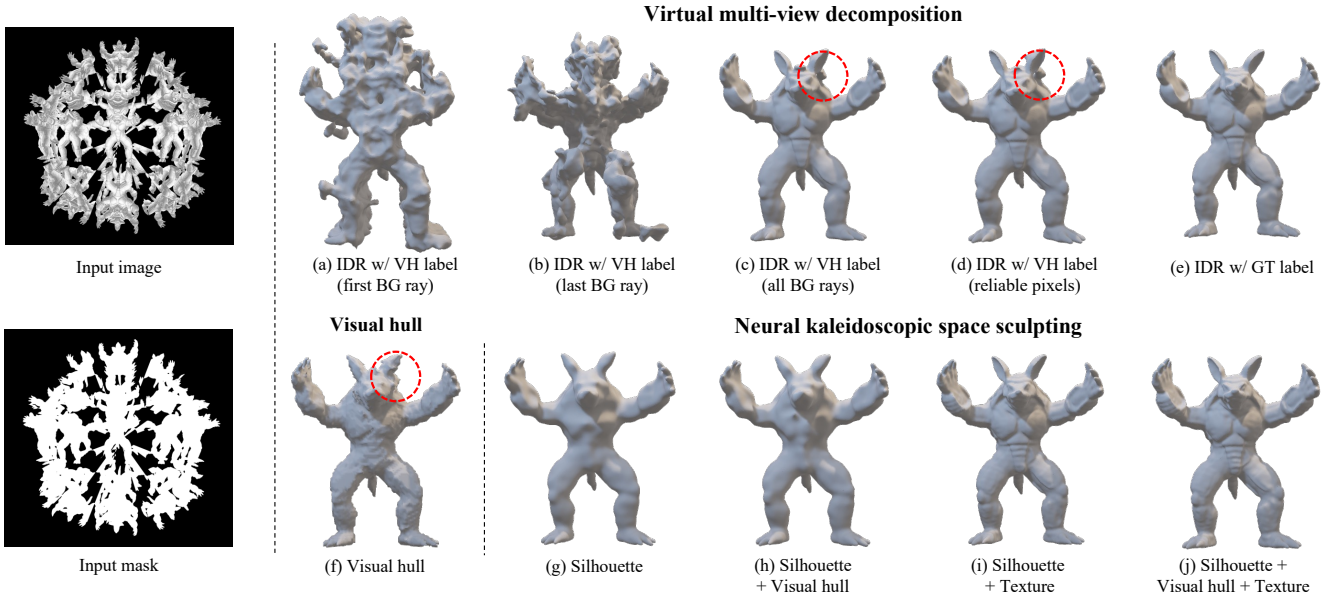


Figure 5. **Simulated experiments.** We compare our method with other kaleidoscopic imaging methods on the simulated image of the armadillo. (a-e) Virtual multi-view decomposition followed by IDR, where the kaleidoscopic image is decomposed into virtual multi-view images either with the visual hull label or ground-truth labels. (f) Kaleidoscopic visual hull. (g-j) Ablation study of our method.

Table 3. **Statistics for real experiments.**

Object	#views	FG #rays /#pixels	BG #rays /#pixels	PSNR↑ [dB]	mask err↓ [%]
Toy	107	5.68	6.08	20.47	1.55
Chair	112	5.77	6.34	22.19	1.38
Treble clef	123	5.72	5.97	15.91	3.73
Venus	88	5.72	6.02	25.83	1.07
Monkey	84	5.80	5.99	21.77	2.42
Spinner	111	5.67	6.06	21.62	0.75

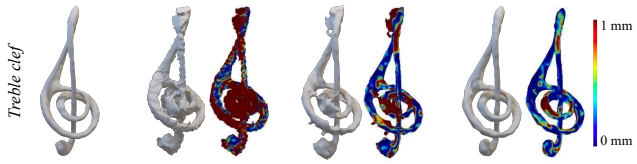


Figure 6. **Comparison to kaleidoscopic structured light [3].** Our method produces a comparable result to kaleidoscopic structured light that uses an additional projector for active illumination.

bel in ascending order and using the "rainbow" Matplotlib colormap. We observe that our technique produces high-quality results for this variety of objects.

Statistics. Table 3 shows the statistics of the results for each object. The number of viewpoints (*i.e.*, the number of different labels) is about 100 viewpoints for all objects, which shows the light direction is redistributed well by the kaleidoscope. The number of rays per pixel is $5.67 - 5.80$ for foreground pixels and $5.97 - 6.34$ for background pixels, which shows each pixel in a kaleidoscope produces about 6 times more information compared to a regular pixel without any reflection. PSNR and mask error shows the reconstructed shape is photo-consistent and silhouette-consistent.

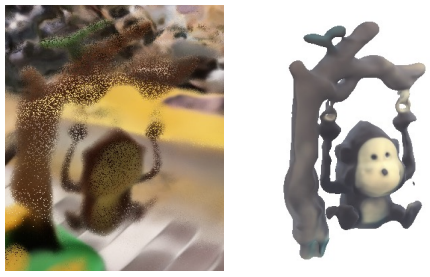


Figure 7. **Comparison to Nerfies [24].** Nerfies fail to model the large deformation between the different views of the swinging monkey, whereas ours does not suffer from the dynamic movement since we capture multiple viewpoints in a single shot.

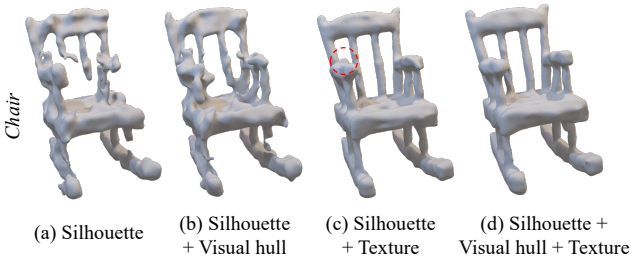


Figure 8. **Ablation study on *Chair* data.**

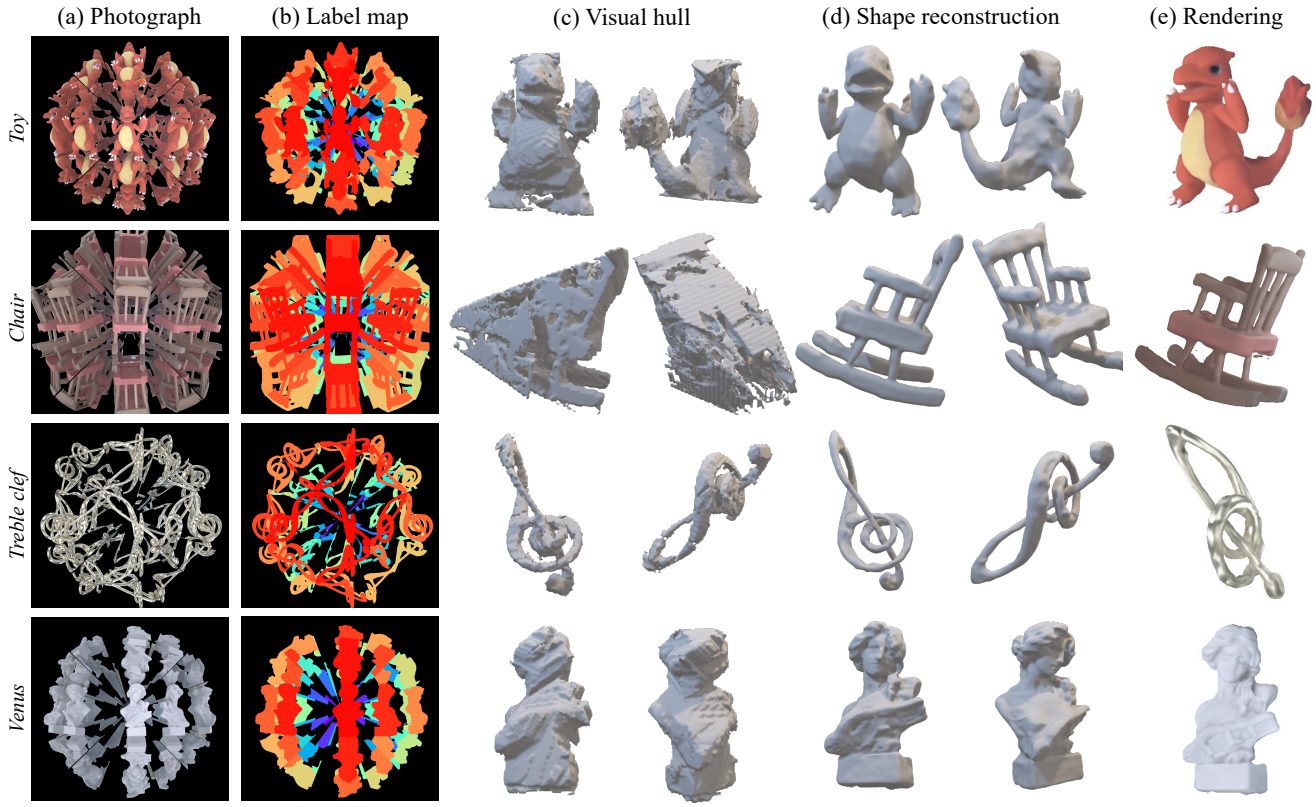


Figure 9. **Real object reconstructions from neural kaleidoscopic space sculpting.** We provide more results in the supplement.

Comparison to kaleidoscopic structured light. Figure 6 shows the comparison to the kaleidoscopic structured light [3], which uses a projector additionally for labeling and better correspondences. We compare the results from our method and baseline methods to the result of kaleidoscopic structured light for *Treble clef* object. For each result, we compute the distance of each vertex from the kaleidoscopic structured light mesh, and visualize it as the vertex color. Our technique produces high-quality results both qualitatively and quantitatively.

Comparison to Nerfies [24]. Figure 7 shows the comparison with Nerfies, which can handle dynamic objects. We captured the video of *Monkey* data where the monkey playing on the swings using a smartphone camera (iPhone SE 2) from surrounding views. Nerfies cannot capture the deformation of this dynamic object and fail to reconstruct the object. By contrast, ours can capture multiple viewpoints in a single shot and obtain the full-surround reconstruction. Note that both methods use different inputs, where our method captures multiple viewpoints of a dynamic object in *a single frame*, whereas Nerfies reconstructs dynamic objects from *multiple frames* by optimizing for a deformation field.

Ablation study Figure 8 shows the ablation study on the real chair data. The result using only silhouette constraint produces an over-carved result, partly because of the imperfect input mask. Adding the visual hull constraint improves the over-carved parts. Adding the texture constraint greatly improves the result, and applying the visual hull constraint additionally improves some artifacts on the armrest.

7. Conclusion

We introduce a single-shot full-surround 3D reconstruction method producing a silhouette-consistent and photo-consistent shape from a single kaleidoscopic image. Based on neural SDF representation, our method carves and models the space by selecting adequate points without the necessity of labels, and jointly solves the labeling and 3D reconstruction problems. We show that our method takes the advantage of the information in the kaleidoscope by reusing the pixels multiple times by the number of reflections.

Acknowledgments We thank Giljoo Nam for helpful discussions. This work was supported by the National Science Foundation (NSF) under awards 1652569, 1900849, and 2008464, a gift from AWS Cloud Credits for Research, as well as a Sloan Research Fellowship for Ioannis Gkioulekas.

References

[1] Project page: Neural kaleidoscopic space sculpting. https://imaging.cs.cmu.edu/neural_kaleidoscopic_space_sculpting, 2023. 1, 6

[2] Kfir Aberman, Oren Katzir, Qiang Zhou, Zegang Luo, Andrei Sharf, Chen Greif, Baoquan Chen, and Daniel Cohen-Or. Dip transform for 3d shape reconstruction. *ACM Transactions on Graphics (TOG)*, 36(4), 2017. 2

[3] Byeongjoo Ahn, Ioannis Gkioulekas, and Aswin C Sankaranarayanan. Kaleidoscopic structured light. *ACM Transactions on Graphics (TOG)*, 40(6), 2021. 1, 2, 5, 6, 7, 8

[4] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 6

[5] 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 *IEEE International Conference on Computer Vision (ICCV)*, 2021. 2

[6] David Brewster. *The kaleidoscope, its history, theory and construction: with its application to the fine and useful arts*. J. Murray, 1858. 1

[7] Yan Cui, Sebastian Schuon, Derek Chan, Sebastian Thrun, and Christian Theobalt. 3d shape scanning with a time-of-flight camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[8] Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. Multiview face capture using polarized spherical gradient illumination. *ACM Transactions on Graphics (TOG)*, 30(6), 2011. 2

[9] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *International Conference on Machine Learning*, 2020. 6

[10] John C Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10), 1996. 3

[11] Michael Holroyd, Jason Lawrence, and Todd Zickler. A coaxial optical scanner for synchronous acquisition of 3d geometry and surface reflectance. *ACM Transactions on Graphics (TOG)*, 29(4), 2010. 2

[12] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Godisart, Bart Nabbe, Iain Matthews, et al. Panoptic studio: A massively multiview system for social interaction capture. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 41(1), 2017. 2

[13] Kalin Kolev, Petri Tanskanen, Pablo Speciale, and Marc Pollefeys. Turning mobile phones into 3d scanners. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[14] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International Journal of Computer Vision (IJCV)*, 38(3), 2000. 3

[15] Aldo Laurentini. The visual hull concept for silhouette-based image understanding. 16(2), 1994. 3

[16] Daniel Lichy, Jiaye Wu, Soumyadip Sengupta, and David W Jacobs. Shape and material capture at home. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[17] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural actor: Neural free-view synthesis of human actors with pose control. *ACM Transactions on Graphics (TOG)*, 40(6), 2021. 2

[18] 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. 2

[19] Giljoo Nam, Joo Ho Lee, Diego Gutierrez, and Min H Kim. Practical svbrdf acquisition of 3d objects with unstructured flash photography. *ACM Transactions on Graphics (TOG)*, 37(6), 2018. 2

[20] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011. 2

[21] NextEngine, 2000. [Accessed Nov. 5, 2022]. 2

[22] Peter Ondrúška, Pushmeet Kohli, and Shahram Izadi. Mobilefusion: Real-time volumetric surface reconstruction and dense tracking on mobile phones. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 21(11), 2015. 2

[23] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[24] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 2, 7, 8

[25] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics*, 40(6), 2021. 2

[26] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[27] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[28] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[29] Ilya Reshetouski, Alkhazur Manakov, Hans-Peter Seidel, and Ivo Ihrke. Three-dimensional kaleidoscopic imaging. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 1, 2, 3, 4, 5, 6

[30] Christopher Schwartz, Ralf Sarlette, Michael Weinmann, and Reinhard Klein. Dome ii: a parallelized btf acquisition system. In *Eurographics Workshop on Material Appearance Modeling: Issues and Acquisition*, 2013. 2

[31] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[32] Ziyan Wang, Timur Bagautdinov, Stephen Lombardi, Tomas Simon, Jason Saragih, Jessica Hodgins, and Michael Zollhofer. Learning compositional radiance fields of dynamic human heads. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[33] Hongzhi Wu, Zhaotian Wang, and Kun Zhou. Simultaneous localization and appearance estimation with a consumer rgb-d camera. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 22(8), 2015. 2

[34] Hongzhi Wu and Kun Zhou. Appfusion: Interactive appearance acquisition using a kinect sensor. *Computer Graphics Forum*, 34(6), 2015. 2

[35] Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. In *Neural Information Processing Systems (NeurIPS)*, 2021. 2

[36] Ruilin Xu, Mohit Gupta, and Shree K Nayar. Trapping light for time of flight. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 5

[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. In *Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 4, 6