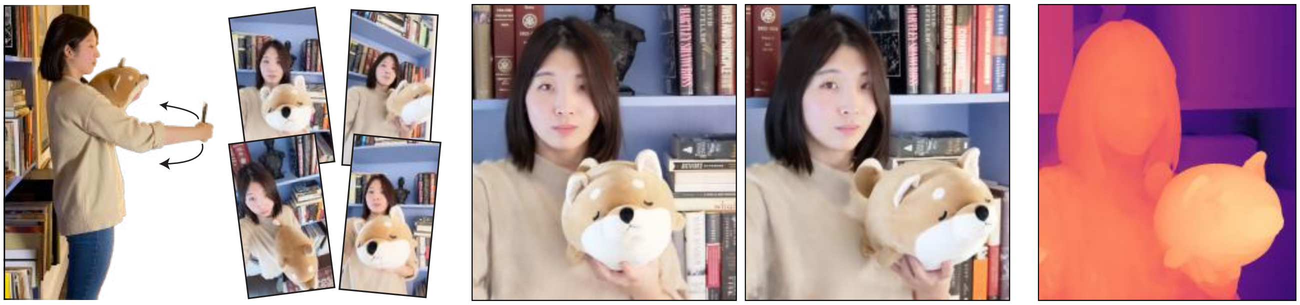
Nerfies: Deformable Neural Radiance Fields

Keunhong Park[1]* Utkarsh Sinha[2] Jonathan T. Barron[2] Sofien Bouaziz[2]
Dan B Goldman[2] Steven M. Seitz[1,2] Ricardo Martin-Brualla[2]

[1]University of Washington [2]Google Research
nerfies.github.io



(a) casual capture (b) input images (c) nerfie novel views (d) novel view depth

Figure 1: We reconstruct photo-realistic *nerfies* from a user casually waving a mobile phone (a). Our system uses selfie photos/videos (b) to produce a free-viewpoint representation with accurate renders (c) and geometry (d). Please see video.

Abstract

We present the first method capable of photorealistically reconstructing deformable scenes using photos/videos captured casually from mobile phones. Our approach augments neural radiance fields (NeRF) by optimizing an additional continuous volumetric deformation field that warps each observed point into a canonical 5D NeRF. We observe that these NeRF-like deformation fields are prone to local minima, and propose a coarse-to-fine optimization method for coordinate-based models that allows for more robust optimization. By adapting principles from geometry processing and physical simulation to NeRF-like models, we propose an elastic regularization of the deformation field that further improves robustness. We show that our method can turn casually captured selfie photos/videos into deformable NeRF models that allow for photorealistic renderings of the subject from arbitrary viewpoints, which we dub "nerfies." We evaluate our method by collecting time-synchronized data using a rig with two mobile phones, yielding train/validation images of the same pose at different viewpoints. We show that our method faithfully reconstructs non-rigidly deforming scenes and reproduces unseen views with high fidelity.

1. Introduction

High quality 3D human scanning has come a long way – but the best results currently require a specialized lab with many synchronized lights and cameras, e.g., [17, 19, 23]. What if you could capture a photorealistic model of yourself (or someone else) just by waving your mobile phone camera? Such a capability would dramatically increase accessibility and applications of 3D modeling technology.

Modeling people with hand-held cameras is especially challenging due both to 1) nonrigidity – our inability to stay perfectly still, and 2) challenging materials like hair, glasses, and earrings that violate assumptions used in most reconstruction methods. In this paper we introduce an approach to address both of these challenges, by generalizing Neural Radiance Fields (NeRF) [39] to model shape deformations. Our technique recovers high fidelity 3D reconstructions from short videos, providing free-viewpoint visualizations while accurately capturing hair, glasses, and other complex, view-dependent materials, as shown in Figure 1. A special case of particular interest is capturing a 3D self-portrait – we call such casual 3D selfie reconstructions *nerfies*.

Rather than represent shape explicitly, NeRF [39] uses a neural network to encode color and density as a function of location and viewing angle, and generates novel views using volume rendering. Their approach produces 3D visualizations of unprecedented quality, faithfully representing thin

[*] Work done while the author was an intern at Google.

structures, semi-transparent materials, and view-dependent effects. To model non-rigidly deforming scenes, we generalize NeRF by introducing an additional component: A canonical NeRF model serves as a template for all the observations, supplemented by a deformation field for each observation that warps 3D points in the frame of reference of an observation into the frame of reference of the canonical model. We represent this deformation field as a multi-layer perceptron (MLP), similar to the radiance field in NeRF. This deformation field is conditioned on a per-image learned latent code, allowing it to vary between observations.

Without constraints, the deformation fields are prone to distortions and over-fitting. We employ a similar approach to the elastic energy formulations that have seen success for mesh fitting [7, 14, 53, 54]. However, our volumetric deformation field formulation greatly simplifies such regularization, because we can easily compute the Jacobian of the deformation field through automatic differentiation, and directly regularize its singular values.

To robustly optimize the deformation field, we propose a novel coarse-to-fine optimization scheme that modulates the components of the input positional encoding of the deformation field network by frequency. By zeroing out the high frequencies at the start of optimization, the network is limited to learn smooth deformations, which are later refined as higher frequencies are introduced into the optimization.

For evaluation, we capture image sequences from a rig of two synchronized, rigidly attached, calibrated cameras, and use the reconstruction from one camera to predict views from the other. We plan to release the code and data.

In summary, our contributions are: ① an extension to NeRF to handle non-rigidly deforming objects that optimizes a deformation field per observation; ② rigidity priors suitable for deformation fields defined by neural networks; ③ a coarse-to-fine regularization approach that modulates the capacity of the deformation field to model high frequencies during optimization; ④ a system to reconstruct free-viewpoint selfies from casual mobile phone captures.

2. Related Work

Non-Rigid Reconstruction: Non-rigid reconstruction decomposes a scene into a geometric model and a deformation model that deforms the geometric model for each observation. Earlier works focused on sparse representations such as keypoints projected onto 2D images [11, 57], making the problem highly ambiguous. Multi-view captures [17, 19] simplify the problem to one of registering and fusing 3D scans [28]. DynamicFusion [40] uses a single RGBD camera moving in space, solving jointly for a canonical model, a deformation, and camera pose. More recently, learning-based methods have been used to find correspondences useful for non-rigid reconstruction [9, 47]. Unlike prior work, our method does not require depth nor multi-view capture systems and works on monocular RGB inputs. Most similar to our work, Neural Volumes [31] learns a 3D representation of a deformable scene using a voxel grid and warp field regressed from a 3D CNN. However, their method requires dozens of synchronized cameras and our evaluation shows that it does not extend to sequences captured from a single camera. Yoon *et al*. [62] reconstruct dynamic scenes from moving camera trajectories, but their method relies on strong semantic priors, in the form of monocular depth estimation, which are combined with multi-view cues. OFlow [41] solves for temporal flow-fields using ODEs, and thus requires temporal information. ShapeFlow [25] learns 3D shapes a divergence-free deformations of a learned template. Instead, we propose an elastic energy regularization.

Domain-Specific Modeling: Many reconstruction methods use domain-specific knowledge to model the shape and appearance of categories with limited topological variation, such as faces [4, 6, 8], human bodies [32, 61], and animals [13, 67]. Although some methods show impressive results in monocular face reconstruction from color and RGBD cameras [66], such models often lack detail (e.g., hair), or do not model certain aspects of a category (e.g., eyewear or garments). Recently, image translation networks have been applied to improve the realism of composited facial edits [20, 26]. In contrast, our work does not rely on domain-specific knowledge, enabling us to model the whole scene, including eyeglasses and hair for human subjects.

Coordinate-based Models: Our method builds on the recent success of coordinate-based models, which encode a spatial field in the weights of a multilayer perceptron (MLP) and require significantly less memory compared to discrete representations. These methods have been used to represent shapes [16, 38, 42] and scenes [39, 52]. Of particular interest are NeRFs [39], that use periodic positional encoding layers [51, 55] to increase resolution, and whose formulation has be extended to handle different lighting conditions [3, 36], transient objects [36], large scenes [30, 63] and to model object categories [49]. Our work extends NeRFs to handle non-rigid scenes.

Concurrent Work: Two concurrent works [44, 58] propose to represent deformable scenes using a translation field in conjunction with a template. This is similar to our framework with the following differences: ① we condition the deformation with a per-example latent [5] instead of time [44]; ② propose an as-rigid-as-possible regularization of the deformation field while NR-NeRF [58] penalizes the divergence of the translation field; ③ propose a coarse-to-fine regularization to prevent getting stuck in local minima; and ④ propose an improved SE(3) parameterization of the deformation field. Other concurrent works [29, 60] reconstruct space-time videos by recovering time-varying NeRFs while leveraging external supervision such as monocular depth estimation and flow-estimation to resolve ambiguities.

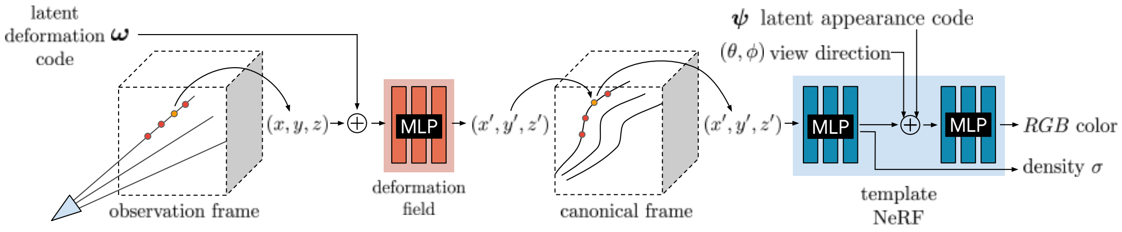


Figure 2: We associate a latent deformation code ($\boldsymbol{\omega}$) and an appearance code ($\boldsymbol{\psi}$) to each image. We trace the camera rays in the observation frame and transform samples along the ray to the canonical frame using a deformation field encoded as an MLP that is conditioned on the deformation code $\boldsymbol{\omega}$. We query the template NeRF [39] using the transformed sample (x', y', z'), viewing direction (θ, ϕ) and appearance code $\boldsymbol{\psi}$ as inputs to the MLP and integrate samples along the ray following NeRF.

3. Deformable Neural Radiance Fields

Here we describe our method for modeling non-rigidly deforming scenes given a set of casually captured images of the scene. We decompose a non-rigidly deforming scene into a template volume represented as a neural radiance field (NeRF) [39] (§3.1) and a per-observation deformation field (§3.2) that associates a point in observation coordinates to a point on the template (overview in Fig. 2). The deformation field is our key extension to NeRF and allows us to represent moving subjects. Jointly optimizing a NeRF together with a deformation field leads to an under-constrained optimization problem. We therefore introduce an elastic regularization on the deformation (§3.3), a background regularization (§3.4), and a continuous, coarse-to-fine annealing technique that avoids bad local minima (§3.5).

3.1. Neural Radiance Fields

A neural radiance field (NeRF) is a continuous, volumetric representation. It is a function $F : (\mathbf{x}, \mathbf{d}, \boldsymbol{\psi}_i) \rightarrow (\mathbf{c}, \sigma)$ which maps a 3D position $\mathbf{x} = (x, y, z)$ and viewing direction $\mathbf{d} = (\phi, \theta)$ to a color $\mathbf{c} = (r, g, b)$ and density σ. In practice, NeRF maps the inputs $\mathbf{x}$ and $\mathbf{d}$ using a sinusoidal positional encoding $\gamma : \mathbb{R}^3 \rightarrow \mathbb{R}^{3+6m}$ defined as $\gamma(\mathbf{x}) = \left(\mathbf{x}, \cdots, \sin\left(2^k \pi \mathbf{x}\right), \cos\left(2^k \pi \mathbf{x}\right), \cdots\right)$, where m is a hyper-parameter that controls the total number of frequency bands and $k \in \{0, \ldots, m-1\}$. This function projects a coordinate vector $\mathbf{x} \in \mathbb{R}^3$ to a high dimensional space using a set of sine and cosine functions of increasing frequencies. This allows the MLP to model high-frequency signals in low-frequency domains as shown in [55]. Coupled with volume rendering techniques, NeRFs can represent scenes with photo-realistic quality. We build upon NeRF to tackle the problem of capturing deformable scenes.

Similar to NeRF-W [36], we also provide an appearance latent code $\boldsymbol{\psi}_i$ for each observed frame $i \in \{1, \ldots, n\}$ that modulates the color output to handle appearance variations between input frames, e.g., exposure and white balance.

The NeRF training procedure relies on the fact that given a 3D scene, two intersecting rays from two different cameras should yield the same color. Disregarding specular reflection and transmission, this assumption is true for all static scenes. Unfortunately, many scenes are not completely static; e.g., it is hard for people to stay completely still when posing for a photo, or worse, when waving a phone when capturing themselves in a selfie video.

3.2. Neural Deformation Fields

With the understanding of this limitation, we extend NeRF to allow the reconstruction of non-rigidly deforming scenes. Instead of directly casting rays through a NeRF, we use it as a canonical template of the scene. This template contains the relative structure and appearance of the scene while a rendering will use a non-rigidly deformed version of the template (see Fig. 3 for an example). DynamicFusion [40] and Neural Volumes [31] also model a template and a per-frame deformation, but the deformation is defined on mesh points and on a voxel grid respectively, whereas we model it as a continuous function using an MLP.

We employ an observation-to-canonical deformation for every frame $i \in \{1, \ldots, n\}$, where n is the number of observed frames. This defines a mapping $T_i : \mathbf{x} \rightarrow \mathbf{x}'$ that maps all observation-space coordinates $\mathbf{x}$ to a canonical-space coordinate $\mathbf{x}'$. We model the deformation fields for all time steps using a mapping $T : (\mathbf{x}, \boldsymbol{\omega}_i) \rightarrow \mathbf{x}'$, which is conditioned on a per-frame learned latent deformation code $\boldsymbol{\omega}_i$. Each latent code encodes the state of the scene in frame i. Given a canonical-space radiance field F and a observation-to-canonical mapping T, the observation-space radiance field can be evaluated as:

$$G(\mathbf{x}, \mathbf{d}, \boldsymbol{\psi}_i, \boldsymbol{\omega}_i) = F\left(T(\mathbf{x}, \boldsymbol{\omega}_i), \mathbf{d}, \boldsymbol{\psi}_i\right) . \tag{1}$$

When rendering, we simply cast rays and sample points in the observation frame and then use the deformation field to map the sampled points to the template, see Fig. 2.

A simple model of deformation is a displacement field $V : (\mathbf{x}, \boldsymbol{\omega}_i) \rightarrow \mathbf{t}$, defining the transformation as $T(\mathbf{x}, \boldsymbol{\omega}_i) = \mathbf{x} + V(\mathbf{x}, \boldsymbol{\omega}_i)$. This formulation is sufficient to represent all continuous deformations; however, rotating a group of points with a translation field requires a different translation for each point, making it difficult to rotate regions of the scene

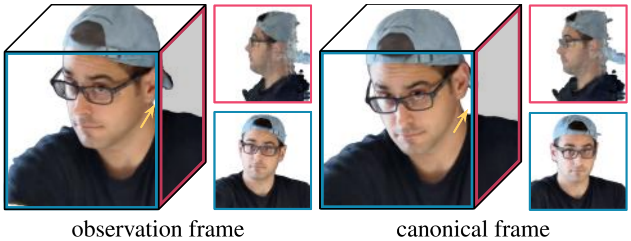


Figure 3: Visualizations of the recovered 3D model in the observation and canonical frames of reference, with insets showing orthographic views in the forward and left directions. Note the right-to-left and front-to-back displacements between the observation and canonical model, which are modeled by the deformation field for this observation.

simultaneously. We therefore formulate the deformation using a dense SE(3) field $W : (\mathbf{x}, \boldsymbol{\omega}_i) \rightarrow \mathrm{SE}(3)$. An SE(3) transform encodes rigid motion, allowing us to rotate a set of distant points with the same parameters.

We encode a rigid transform as a screw axis [35] $\mathcal{S} = (\mathbf{r}; \mathbf{v}) \in \mathbb{R}^6$. Note that $\mathbf{r} \in \mathfrak{so}(3)$ encodes a rotation where $\hat{\mathbf{r}} = \mathbf{r}/\|\mathbf{r}\|$ is the axis of rotation and $\theta = \|\mathbf{r}\|$ is the angle of rotation. The exponential of $\mathbf{r}$ (also known as Rodrigues' formula [46]) yields a rotation matrix $e^{\mathbf{r}} \in \mathrm{SO}(3)$:

$$e^{\mathbf{r}} \equiv e^{[\mathbf{r}]_\times} = \mathbf{I} + \frac{\sin\theta}{\theta}[\mathbf{r}]_\times + \frac{1-\cos\theta}{\theta^2}[\mathbf{r}]_\times^2 \,, \qquad (2)$$

where $[\mathbf{x}]_\times$ denotes the cross-product matrix of a vector $\mathbf{x}$.

Similarly, the translation encoded by the screw motion $\mathcal{S}$ can be recovered as $\mathbf{p} = \mathbf{G}\mathbf{v}$ where

$$\mathbf{G} = \mathbf{I} + \frac{1-\cos\theta}{\theta^2}[\mathbf{r}]_\times + \frac{\theta - \sin\theta}{\theta^3}[\mathbf{r}]_\times^2 \,. \qquad (3)$$

Combining these formulas and using the exponential map, we get the transformed point as $\mathbf{x}' = e^{\mathcal{S}}\mathbf{x} = e^{\mathbf{r}}\mathbf{x} + \mathbf{p}$.

As mentioned before, we encode the transformation field in an MLP $W : (\mathbf{x}, \boldsymbol{\omega}_i) \rightarrow (\mathbf{r}, \mathbf{v})$ using a NeRF-like architecture, and represent the transformation of every frame i by conditioning on a latent code $\boldsymbol{\omega}_i$. We optimize the latent code through an embedding layer [5]. Like with the template, we map the input $\mathbf{x}$ using positional encoding γ_α (see §3.5). An important property of the $\mathfrak{se}(3)$ representation is that $e^{\mathcal{S}}$ is the identity when $\mathcal{S} = \mathbf{0}$. We therefore initialize the weights of the last layer of the MLP from $\mathcal{U}(-10^{-5}, 10^{-5})$ to initialize the deformation near the identity.

3.3. Elastic Regularization

The deformation field adds ambiguities that make optimization more challenging. For example, an object moving backwards is visually equivalent to it shrinking in size, with many solutions in between. These ambiguities lead to under-constrained optimization problems which yield implausible results and artifacts (see Fig. 6). It is therefore crucial to introduce priors that lead to a more plausible solution.

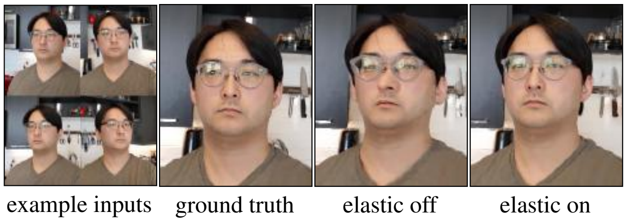


Figure 4: Our elastic regularization helps when the scene is under-constrained. This capture only contains 20 input images with the cameras biased towards one side of the face resulting in an under constrained problem. Elastic regularization helps resolve the ambiguity and leads to less distortion.

It is common in geometry processing and physics simulation to model non-rigid deformations using elastic energies measuring the deviation of local deformations from a rigid motion [7, 14, 53, 54]. In the vision community, these energies have been extensively used for the reconstruction and tracking of non-rigid scenes and objects [19, 40, 65] making them good candidates for our approach. While they have been most commonly used for discretized surfaces, e.g., meshes, we can apply a similar concept in the context of our continuous deformation field.

Elastic Energy: For a fixed latent code $\boldsymbol{\omega}_i$, our deformation field T is a non-linear mapping from observation-coordinates in $\mathbb{R}^3$ to canonical coordinates in $\mathbb{R}^3$. The Jacobian $\mathbf{J}_T(\mathbf{x})$ of this mapping at a point $\mathbf{x} \in \mathbb{R}^3$ describes the best linear approximation of the transformation at that point. We can therefore control the local behavior of the deformation through $\mathbf{J}_T$ [50]. Note that unlike other approaches using discretized surfaces, our continuous formulation allows us to directly compute $\mathbf{J}_T$ through automatic differentiation of the MLP. There are several ways to penalize the deviation of the Jacobian $\mathbf{J}_T$ from a rigid transformation. Considering the singular-value decomposition of the Jacobian $\mathbf{J}_T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, multiple approaches [7, 14] penalize the deviation from the closest rotation as $\|\mathbf{J}_T - \mathbf{R}\|_F^2$, where $\mathbf{R} = \mathbf{V}\mathbf{U}^T$ and $\|\cdot\|_F$ is the Frobenius norm. We opt to directly work with the singular values of $\mathbf{J}_T$ and measure its deviation from the identity. The log of the singular values gives equal weight to a contraction and expansion of the same factor, and we found it to perform better. We therefore penalize the deviation of the log singular values from zero:

$$L_{\text{elastic}}(\mathbf{x}) = \|\log\boldsymbol{\Sigma} - \log\mathbf{I}\|_F^2 = \|\log\boldsymbol{\Sigma}\|_F^2 \,, \qquad (4)$$

where log here is the matrix logarithm.

Robustness: Although humans are mostly rigid, there are some movements which can break our assumption of local rigidity, e.g., facial expressions which locally stretch and compress our skin. We therefore remap the elastic energy

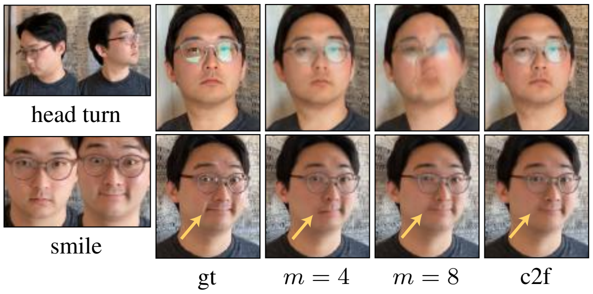


Figure 5: In this capture, the subject rotates their head (top) and smiles (bottom). With $m = 4$ positional encoding frequencies, the deformation model does not capture the smile, while it fails to rotate the head with $m = 8$ frequencies. With coarse-to-fine regularization (c2f) the model captures both.

defined above using a robust loss:

$$L_{\text{elastic-r}}(\mathbf{x}) = \rho\left(\left\|\log \mathbf{\Sigma}\right\|_F, c\right), \tag{5}$$

$$\rho(x, c) = \frac{2(x/c)^2}{(x/c)^2 + 4}. \tag{6}$$

where $\rho(\cdot)$ is the Geman-McClure robust error function [21] parameterized with hyperparameter $c = 0.03$ as per Barron [2]. This robust error function causes the gradients of the loss to fall off to zero for large values of the argument, thereby reducing the influence of outliers during training.

Weighting: We allow the deformation field to behave freely in empty space, since the subject moving relative to the background requires a non-rigid deformation somewhere in space. We therefore weight the elastic penalty at each sample along the ray by its contribution to the rendered view, *i.e.* w_i in Eqn. 5 of NeRF [39].

3.4. Background Regularization

The deformation field is unconstrained and therefore everything is free to move around. We optionally add a regularization term which prevents the background from moving. Given a set of 3D points in the scene which we know should be static, we can penalize any deformations at these points. For example, camera registration using structure from motion produces a set of 3D feature points that behave rigidly across at least some set of observations. Given these static 3D points $\{\mathbf{x}_1 \ldots, \mathbf{x}_K\}$, we penalize movement as:

$$L_{\text{bg}} = \frac{1}{K}\sum_{k=1}^{K}\left\|T(\mathbf{x}_k) - \mathbf{x}_k\right\|_2. \tag{7}$$

In addition to keeping the background points from moving, this regularization also has the benefit of aligning the observation coordinate frame to the canonical coordinate frame.

3.5. Coarse-to-Fine Deformation Regularization

A common trade-off that arises during registration and flow estimation is the choice between modeling minute versus large motions, that can lead to overly smooth results or incorrect registration (local minima). Coarse-to-fine strategies circumvent the issue by first solving the problem in low-resolution, where motion is small, and iteratively upscaling the solution and refining it [33]. We observe that our deformation model suffers from similar issues, and propose a coarse-to-fine regularization to mitigate them.

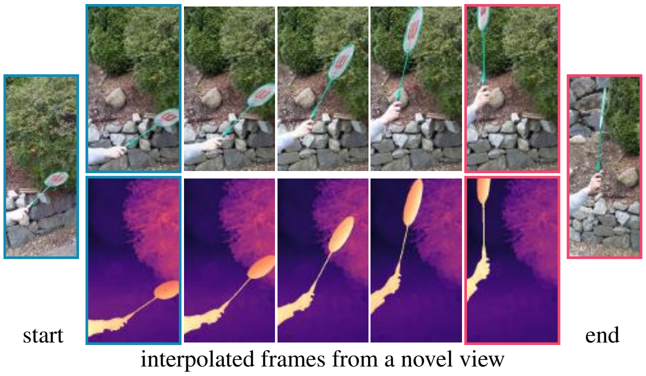


Figure 6: Novel views synthesized by linearly interpolating the deformation latent codes of two frames of the BADMINTON (left and right) show a smooth racquet motion.

Recall the positional encoding parameter m introduced in §3.1 that controls the number of frequency bands used in the encoding. Tancik *et al.* [55] show that controls it the smoothness of the network: a low value of m results in a low-frequency bias (low resolution) while a higher value of m results in a higher-frequency bias (high resolution).

Consider a motion like in Fig. 5, where subject rotates their head and smiles. With a small m for the deformation field, the model cannot capture the minute motion of the smile; conversely, with a larger m, the model fails to correctly rotate the head because the template overfits to an underoptimized deformation field. To overcome this tradeoff, we propose a coarse-to-fine approach that starts with a low-frequency bias and ends with a high-frequency bias.

Tancik *et al.* [55] show that positional encoding can be interpreted in terms of the Neural Tangent Kernel (NTK) [24] of NeRF's MLP: a stationary interpolating kernel where m controls a tunable "bandwidth" of that kernel. A small number of frequencies induces a wide kernel which causes under-fitting of the data, while a large number of frequencies induces a narrow kernel causing over-fitting. With this in mind, we propose a method to smoothly anneal the bandwidth of the NTK by introducing a parameter α that windows the frequency bands of the positional encoding, akin to how coarse-to-fine optimization schemes solve for coarse solutions that are subsequently refined at higher resolutions. We define the weight for each frequency band j as:

$$w_j(\alpha) = \frac{(1 - \cos(\pi\,\text{clamp}(\alpha - j, 0, 1))}{2}, \tag{8}$$

where linearly annealing the parameter $\alpha \in [0, m]$

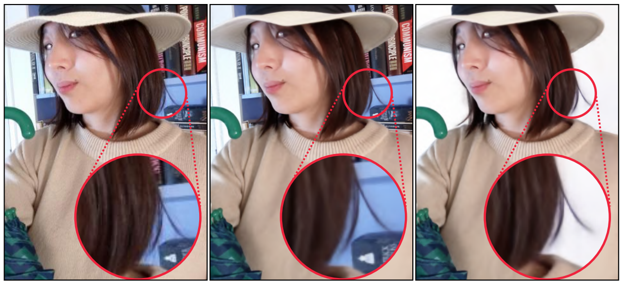


Figure 7: Our method recovers thin hair strands. By adjusting the camera's far plane, we can render the subject against a flat white background.

can be interpreted as sliding a truncated Hann window (where the left side is clamped to 1 and the right side is clamped to 0) across the frequency bands. The positional encoding is then defined as $\gamma_\alpha(\mathbf{x}) = \big(\mathbf{x}, \cdots, w_k(\alpha)\sin(2^k\pi\mathbf{x}), w_k(\alpha)\cos(2^k\pi\mathbf{x}), \cdots\big)$. During training, we set $\alpha(t) = \frac{mt}{N}$ where t is the current training iteration, and N is a hyper-parameter for when α should reach the maximum number of frequencies m. We provide further analysis in the supplementary materials.

4. Nerfies: Casual Free-Viewpoint Selfies

So far we have presented a generic method of reconstructing non-rigidly deforming scenes. We now present a key application of our system – reconstructing high quality models of human subjects from casually captured selfies, which we dub *"nerfies"*. Our system takes as input a sequence of selfie photos or a selfie video in which the user is standing mostly still. Users are instructed to wave the camera around their face, covering viewpoints within a 45° cone. We observe that 20 second captures are sufficient. In our method, we assume that the subject stands against a static background to enable a consistent geometric registration of the cameras. We filter blurry frames using the variance of the Laplacian [43], keeping about 600 frames per capture.

Camera Registration: We seek a registration of the cameras with respect to the static background. We use COLMAP [48] to compute pose for each image and camera intrinsics. This step assumes that enough features are present in the background to register the sequence.

Foreground Segmentation: In some cases, SfM will match features on the moving subject, causing significant misalignment in the background. This is problematic in video captures with correlated frames. In those cases, we found it helpful to discard image features on the subject, which can be detected using a foreground segmentation network.

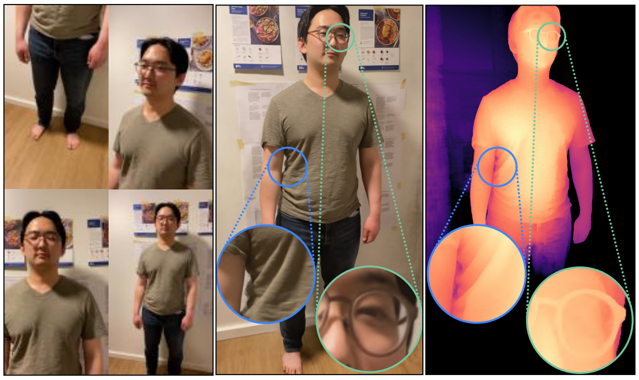


Figure 8: Our method reconstructs full body scenes captured by a second user with high quality details.

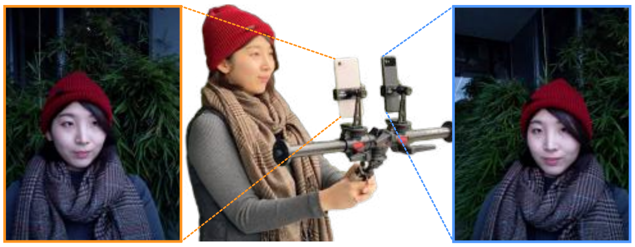


Figure 9: Validation rig used only for evaluation.

5. Experiments

5.1. Implementation Details

Our NeRF template implementation closely follows the original [39], except we use a Softplus activation $\ln(1 + e^x)$ for the density. We use a deformation network with depth 6, hidden size 128, and a skip connection at the 4th layer. We use 256 coarse and fine ray samples for full HD (1920×1080) models and half that for the half resolution models. We use 8 dimensions for the latent deformation and appearance codes. For coarse-to-fine optimization we use 8 frequency bands and linearly anneal α from 0 to 8 over 80K iterations. We use the same MSE photometric loss as in NeRF [39] and weight the losses as $L_{\text{total}} = L_{\text{rgb}} + \lambda L_{\text{elastic-r}} + \mu L_{\text{bg}}$ where we use $\lambda = \mu = 10^{-3}$ for all experiments except when mentioned. We train on 8 V100 GPUs for a week for full HD models, and for 16 hours for the half resolution models used for the comparisons in Tab. 1, Fig. 10, and Fig. 11. We provide more details in the Section A of the appendix and plan to release code and data.

5.2. Evaluation Dataset

In order to evaluate the quality of our reconstruction, we must be able to measure how faithfully we can recreate the scene from a viewpoint unseen during training. Since we are reconstructing non-rigidly deforming scenes, we cannot simply hold out views from an input capture, as the structure of the scene will be slightly different in every image. We sound

	Quasi-Static																		Dynamic									
	Glasses (78 images)		Beanie (74 images)		Curls (57 images)		Kitchen (40 images)		Lamp (55 images)		Toby Sit (308 images)		Mean		Drinking (193 images)		Tail (238 images)		Badminton (356 images)		Broom (197 images)		Mean	
	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓
NeRF [39]	18.1	.347	16.8	.509	14.4	.525	19.2	.329	17.5	.354	21.2	.578	17.9	.440	18.1	.415	21.9	.675	17.5	.490	19.8	.800	19.3	.595
NeRF + latent	19.6	.319	19.5	.426	17.3	.434	20.2	.293	18.9	.281	18.6	.492	19.0	.374	20.9	.251	23.4	.497	18.5	.401	20.4	.711	20.8	.465
Neural Volumes [31]	15.6	.576	15.8	.563	15.3	.551	14.8	.572	13.6	.556	24.8	.512	16.7	.555	22.3	.255	25.6	.626	18.2	.694	22.4	.717	22.1	.573
NSFF† [29]	18.0	.443	18.0	.531	15.1	.543	20.1	.320	18.3	.348	22.8	.488	18.7	.446	24.0	.237	19.8	.683	17.8	.541	21.0	.746	20.7	.552
$\gamma(t)$ + trans†	22.3	.201	20.9	.358	20.8	.298	22.7	.230	22.1	.164	22.9	.539	21.9	.298	22.4	.168	24.9	.498	20.5	.313	21.5	.764	22.4	.436
Ours ($\lambda = 0.01$)	23.6	.146	22.3	.266	24.7	.155	24.2	.155	23.8	.112	20.7	.186	23.2	.170	21.1	.0967	22.1	.182	20.0	.172	19.8	.283	20.7	.183
Ours ($\lambda = 0.001$)	24.4	.149	23.3	.268	25.0	.147	23.7	.153	24.0	.109	20.7	.206	23.5	.172	20.7	.106	22.0	.197	19.8	.174	19.6	.360	20.5	.210
No elastic	23.2	.158	24.4	.262	24.2	.157	23.1	.166	24.0	.110	20.9	.333	23.3	.198	20.9	.0957	22.0	.197	19.8	.175	19.3	.329	20.5	.199
No coarse-to-fine	24.0	.158	22.0	.279	24.6	.157	24.3	.154	23.0	.121	20.7	.311	23.1	.197	21.0	.107	22.6	.297	19.7	.199	20.2	.510	20.8	.278
No SE(3)	23.7	.156	22.1	.275	24.7	.151	23.9	.159	22.9	.116	20.8	.258	23.0	.186	21.0	.0968	21.9	.217	19.3	.202	19.3	.314	20.4	.207
No BG loss	22.4	.160	21.6	.275	20.2	.197	22.7	.165	20.5	.135	20.3	.163	21.3	.182	21.0	.0951	22.0	.242	18.3	.209	19.4	.394	20.2	.235
Ours (bare)	24.2	.167	21.0	.325	23.7	.183	22.6	.199	22.2	.131	20.6	.218	22.4	.204	21.3	.142	22.7	.370	19.1	.233	20.4	.627	20.9	.343

Table 1: Quantitative evaluation on validation captures against baselines and ablations of our system, we color code each row as **best**, **second best**, and **third best**. †denotes use of temporal information. Please see Sec. 5.3 for more details.

that existing datasets were not representative of many capture scenarios. For example, the multi-view dataset of Yoon *et al*. [62] contains too few viewpoints (12) and exaggerated frame-to-frame motions due to temporal subsampling. We therefore build a simple multi-view data capture rig for the sole purpose of evaluation.

Our rig (Fig. 9) is a pole with two Pixel 3's rigidly attached. We have two methods for data capture: (a) for selfies we use the front-facing camera and capture time-synchronized photos using the method of Ansari *et al*. [1], which achieves sub millisecond synchronization; or (b) we use the back-facing camera and record two videos which we manually synchronize based on the audio; we then subsample to 5 fps. We register the images using COLMAP [48] with rigid relative camera pose constraints. Sequences captured with (a) contain fewer frames (40~78) but the focus, exposure, and time are precisely synchronized. Sequences captured with (b) have denser samples in time (between 193 and 356 frames) but the synchronization is less precise and exposure and focus may vary between the cameras. We split each capture into a training set and a validation set. We alternate assigning the left view to the training set, and right to the validation, and vice versa. This avoids having regions of the scene that one camera has not seen.

Quasi-static scenes: We capture 5 human subjects using method (a), that attempt to stay as still as possible during capture, and a mostly still dog using method (b).

Dynamic scenes: We capture 4 dynamic scenes containing deliberate motions of a human subject, a dog wagging its tail, and two moving objects using method (b).

5.3. Evaluation

Here we provide quantitative and qualitative evaluations of our model. However, to best appreciate the quality of the reconstructed *nerfies*, we encourage the reader to watch the supplementary video that contains many example results.

Quantitative Evaluation: We compare against NeRF and a NeRF + latent baseline, where NeRF is conditioned on a per-image learned latent code [5] to modulate density and color. We also compare with a variant of our system similar to the concurrent work of D-NeRF [44], which conditions a translational deformation field with a position encoded time variable $\gamma(t)$ instead of a latent code ($\gamma(t)$+trans in Tab. 1). We also compare with the high quality model of Neural Volumes (NV) [31] using a single view as input to the encoder, and Neural Scene Flow Fields (NSFF) [29]. We do not evaluate the method of Yoon et al. [62] due to the lack of available code (note that NSFF outperforms it). NSFF and the $\gamma(t) + trans$ baseline use temporal information while other baselines and our method do not. NSFF also uses auxiliary supervision such as estimated flow and relative depth maps; we do not. Photometric differences between the two rig cameras may exist due to different exposure/white balance settings and camera response curves. We therefore swap the per-frame appearance code ψ_i for a per-camera $\{\psi_L, \psi_R\} \in \mathbb{R}^2$ instead for validation rig captures.

Tab. 1 reports LPIPS [64] and PSNR metrics for the unseen validation views. PSNR favors blurry images and is therefore not an ideal metric for dynamic scene reconstruction; we find that LPIPS is more representative of visual quality. See Fig. 10 and Fig. 11 for side-by-side images with associated PSNR/LPIPS metrics. Our method struggles with PSNR due to slight misalignments resulting from factors such as gauge ambiguity [37] while we outperform all baselines in terms of LPIPS for all sequences.

Ablation Study: We evaluate each of our contributions: SE(3) deformations, elastic regularization, background regularization, and coarse-to-fine optimization. We ablate them one at a time, and all at once (Ours (bare) in Tab. 1). As expected, a stronger elastic regularization ($\lambda = 0.01$) improves results for dynamic scenes compared to the baseline ($\lambda = 0.001$) while minimally impacting quasi-static scenes. Removing the elastic loss hurts performance for quasi-static scenes while having minimal effect on the dynamic scene; this may be due to the larger influence of other losses in the presence of larger motion. Elastic regularization fixes distortion artifacts when the scene is under-constrained (e.g., Fig. 4). Disabling coarse-to-fine regularization mildly drops

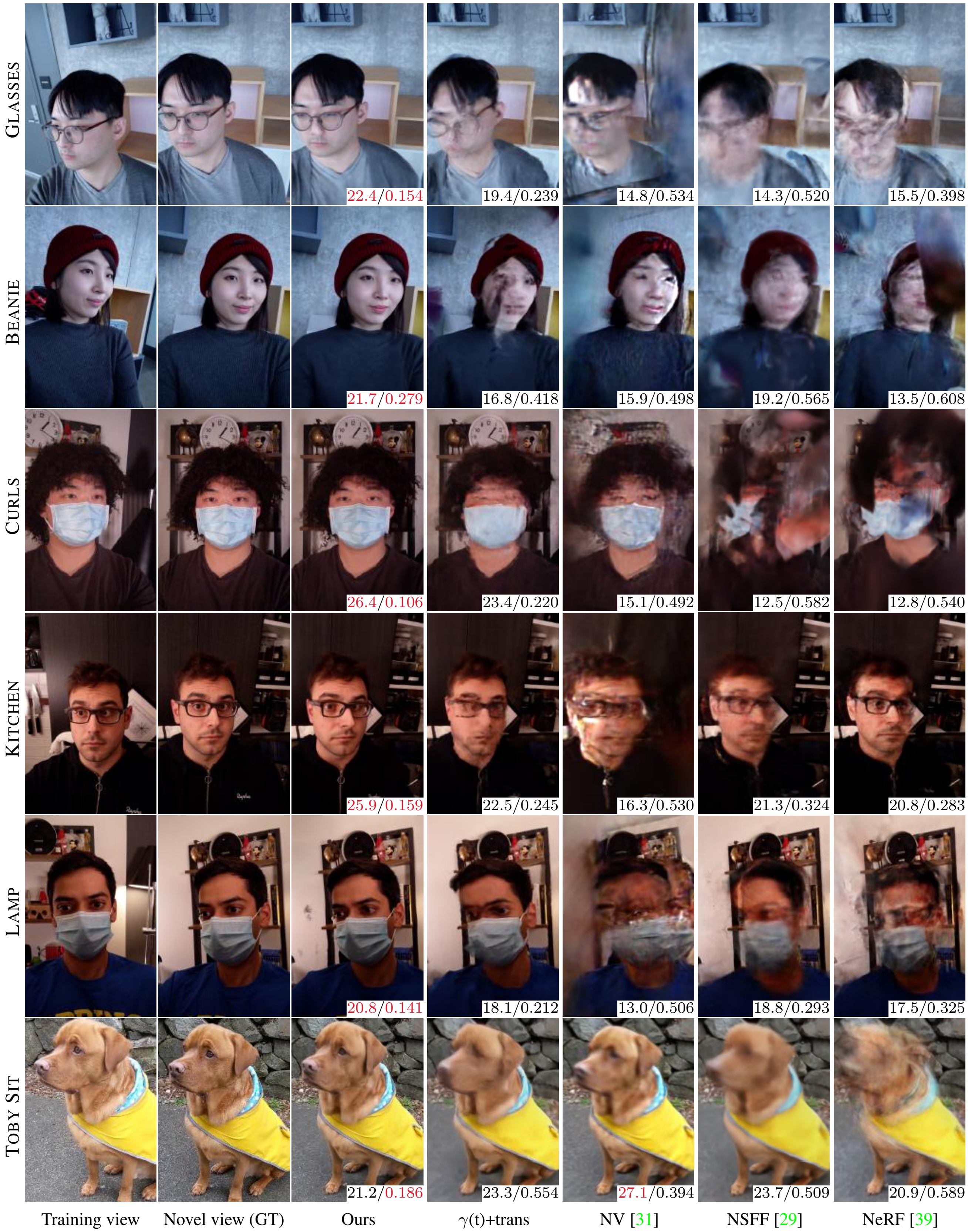


Figure 10: Comparisons of baselines and our method on quasi-static scenes. PSNR / LPIPS metrics on bottom right with best colored red.

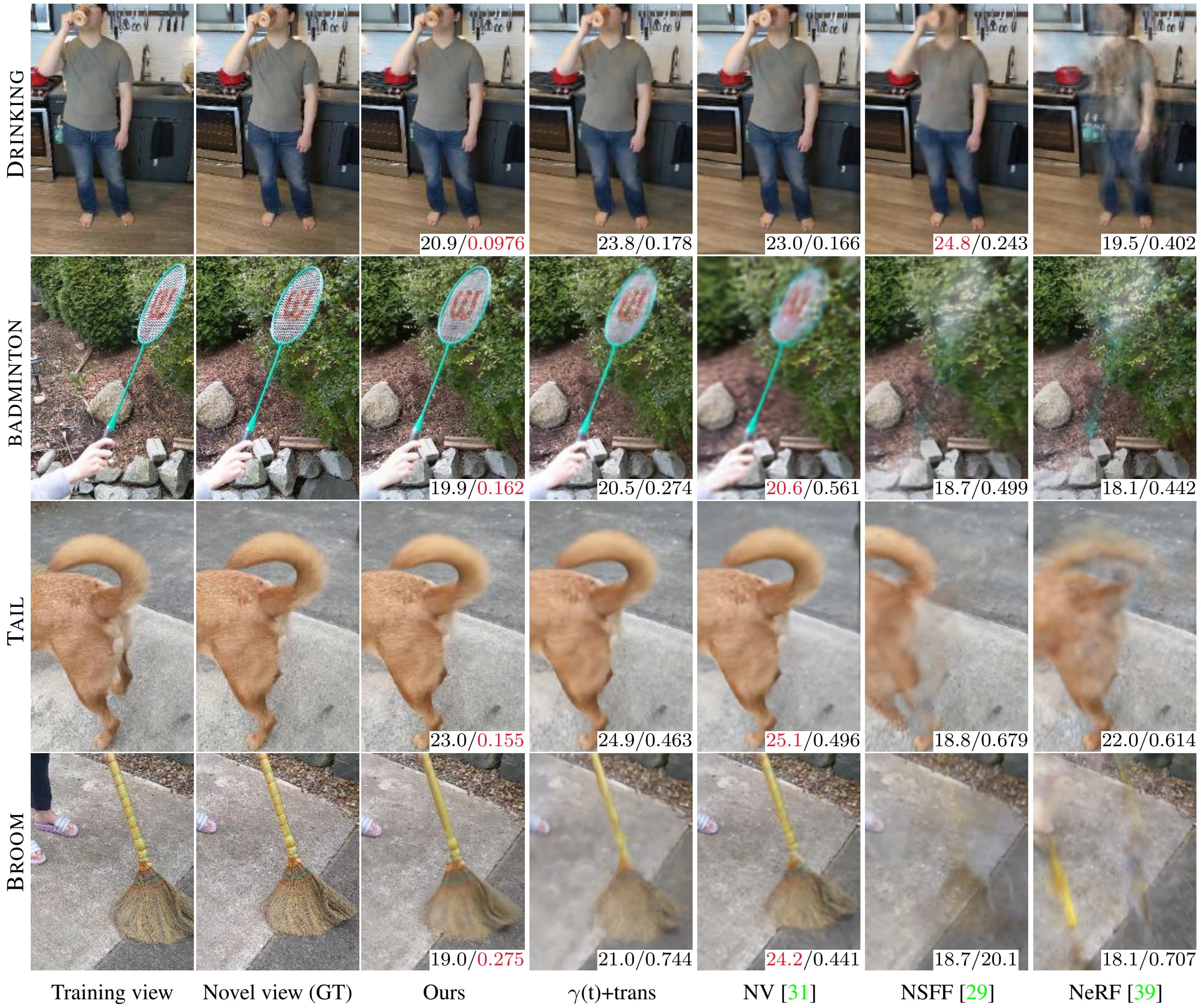


Figure 11: Comparisons of baselines and our method on dynamic scenes. PSNR / LPIPS metrics on bottom right with best colored red.

performance for quasi-static scenes while causing a significant drop for dynamic scenes. This is expected since large motions are a main source of local minima (e.g., Fig. 5). Our SE(3) deformations also quantitatively outperform translational deformations. Background regularization helps PSNR by reducing shifts in static regions and removing it results in worse performance. Finally, removing all of our contributions performs the worst in terms of LPIPS.

Qualitative Results: We show results for the captures used in the quantitative evaluation in Fig. 10 and Fig. 11. Our method can reconstruct fine details such as strands of hair (e.g., in CURLS of Tab. 1 and Fig. 7), shirt wrinkles, and glasses (Fig. 8). Our method works on general scenes beyond human subjects as shown in Fig. 10 and Fig. 11. In addition, we can create smooth animations by interpolating the deformation latent codes of any input state as shown in Fig. 6.

Elastic Regularization: Fig. 4 shows an example where the user only captured 20 images mostly from one side of their face, while their head tracked the camera. This results in ambiguous geometry. Elastic regularization helps in such under-constrained cases, reducing distortion significantly.

Depth Visualizations: We visualize the quality of our reconstruction using depth renders of the density field. Unlike NeRF[39] that visualizes the expected ray termination distance, we use the *median* depth termination distance, which we found to be less biased by residual density in free space (see Fig. 7). We define it as the depth of the first sample with accumulated transmittance $T_i \geq 0.5$ (Eqn. 3 of NeRF [39]).

Limitations: Our method struggles with topological changes e.g., opening/closing of the mouth (see Fig. 12) and may fail for certain frames in the presence of rapid motion (see supplementary). As mentioned in §3.4, our deformations are unconstrained so static regions may shift; this contributes to the disjunction between PSNR and LPIPS in Tab. 1. Future work may address this by modeling static regions separately as in [29, 36]. Finally, the quality of our method depends on camera registration, and when SfM fails so do we.

6. Conclusion

Deformable Neural Radiance Fields extend NeRF by modeling non-rigidly deforming scenes. We show that our as-rigid-as-possible deformation prior, and coarse-to-fine deformation regularization are the key to obtaining high-quality results. We showcase the application of casual selfie captures (*nerfies*), and enable high-fidelity reconstructions of human subjects using a cellphone capture. Future work may tackle larger/faster motion, topological variations, and enhance the speed of training/inference.

Acknowledgments

We thank Peter Hedman and Daniel Duckworth for providing feedback in early drafts, and all our capture subjects for their patience, including Toby who was a good boy.

A. Details of SE(3) Field Formulation

As mentioned in the main text, we encode a rigid transform as a screw axis [35] $\mathcal{S} = (\mathbf{r}; \mathbf{v}) \in \mathbb{R}^6$ where

$$e^{\mathbf{r}} \equiv e^{[\mathbf{r}]_\times} = \mathbf{I} + \frac{\sin\theta}{\theta}[\mathbf{r}]_\times + \frac{1-\cos\theta}{\theta^2}[\mathbf{r}]_\times^2 . \tag{9}$$

$[\mathbf{x}]_\times$ is a skew-symmetric matrix also known as the cross-product matrix of a vector $\mathbf{x}$ since given two 3-vectors $\mathbf{a}$ and $\mathbf{b}$, $[\mathbf{a}]_\times \mathbf{b}$ gives the cross product $\mathbf{a} \times \mathbf{b}$.

$$[\mathbf{x}]_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix} . \tag{10}$$

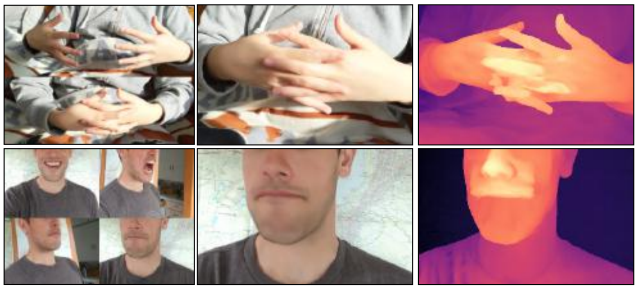


Figure 12: Our method fails in the presence of topological changes. Although the rendered color views look good from some viewpoints, the recovered geometry is incorrect.

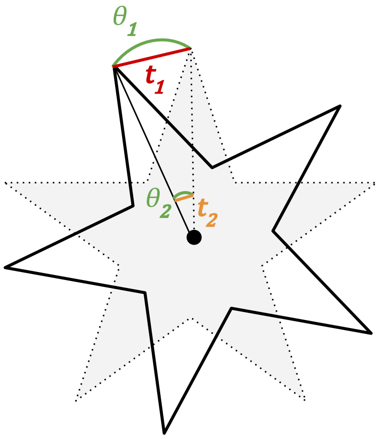


Figure 13: Here we illustrate why a rigid transformation field works better than a translation field with a simple toy example where a star is rotated counter-clockwise around its center. A translation field requires different parameters for every point to encode the rotation (e.g., $\|\mathbf{t_1}\| \gg \|\mathbf{t_2}\|$) whereas a rotation field only needs a single parameter to encode the rotation (e.g., $\theta_1 = \theta_2$). More details in §*A*.

The translation encoded by the screw motion $\mathcal{S}$ can be recovered as $\mathbf{p} = \mathbf{G}\mathbf{v}$ where

$$\mathbf{G} = \mathbf{I} + \frac{1-\cos\theta}{\theta^2}[\mathbf{r}]_\times + \frac{\theta - \sin\theta}{\theta^3}[\mathbf{r}]_\times^2 . \tag{11}$$

The exponential of $\mathcal{S}$ can also be expressed in homogeneous matrix form $e^{\mathcal{S}} \in \mathrm{SE}(3)$:

$$e^{\mathcal{S}} = \begin{pmatrix} e^{\mathbf{r}} & \mathbf{p} \\ 0 & 1 \end{pmatrix} . \tag{12}$$

The deformed point is then given by $\mathbf{x}' = e^{\mathcal{S}}\mathbf{x}$.

Why does an SE(3) field work better? Consider the example in Fig. 13 where a star has been rotated counter-clockwise along its center. Now consider what transformation would be required at every point on the star to encode this rotation. With a translation field, points towards the center (e.g., $\mathbf{t_2}$) need translations of small magnitude while points towards the outside (e.g., $\mathbf{t_1}$) need translations of larger magnitude. Every point on the star requires a different parameter to encode a simple rotation. On the otherhand, with a rotation, every point on the star can be parameterized by a single angle which is the angle of rotation $\theta = \theta_1 = \theta_2$. This makes optimization much easier since the deformation field MLP only needs to predict a single parameter across space. We illustrate this further in §H.

B. Details of Coarse-to-Fine Optimization

Window Function: Our coarse-to-fine deformation regularization is implemented by windowing the frequency bands

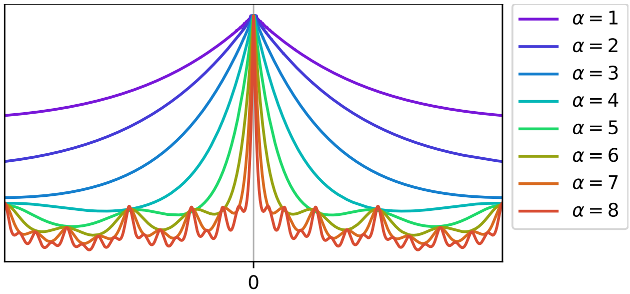


Figure 14: Visualizations of the neural-tangent kernel (NTK) [24] of our annealed positional encoding for different values of α. Our coarse-to-fine optimization scheme works by easing in the influence of each positional encoding frequency through a parameter α. This has the effect of shrinking the bandwidth of the NTK corresponding to the deformation MLP as α is increased, thereby allowing higher frequency deformations.

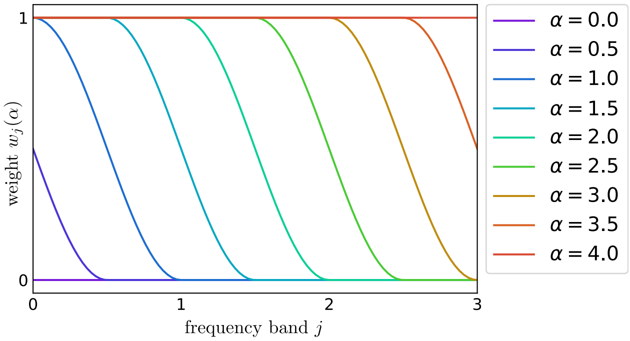


Figure 15: A visualization of the window function $w_j(\alpha)$ for the annealed positional encoding. We show an example with a maximum number of frequency bands of $m = 4$ where $j \in \{0, \ldots, m-1\}$. $\alpha = 0$ sets the weight of all frequency bands to zero leaving only the identity mapping, while an $\alpha = 4$ sets the weight of all frequency bands to one. Increasing the value of α is equivalent to sliding the window to the right across the frequency bands.

of the positional encoding. Eqn. 8 of the main paper defines this windowing function as a weight applied to each frequency band. We visualize our windowing function for different values of α in Fig. 15.

NTK: We also show a visualization of the neural tangent kernel (NTK) induced by our annealed positional encoding in Fig. 14. This figure shows the normalized NTK for an 8 layer MLP of width 256. Note how the bandwidth of the interpolation kernel gets narrower as the value of α increases.

C. Additional Illustrations

Unintentional Movement: In Fig. 16 we show an example of how a person can move even when trying to sit still.

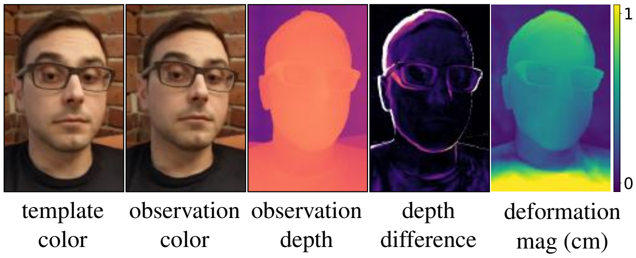


Figure 16: Users move even when trying not to. Here we visualize the depth difference and deformation magnitude between the template and an observation.

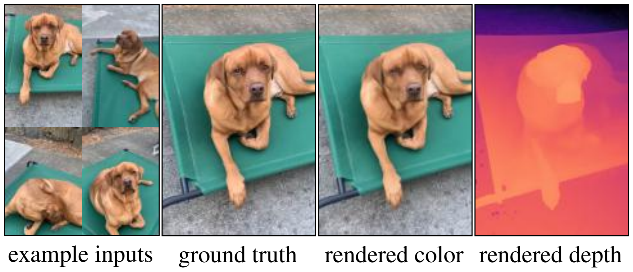


Figure 17: Not relying on domain specific priors enables our method to reconstruct any deformable object. In this case, the dog fails to stay still, yet we recover an accurate model.

We visualize the degree of movement by showing the difference in predicted depth as well as by showing a direct plot of the magnitude of the deformation field at the predicted depth point.

Domain Agnostic: In Fig. 17, Fig. 10, and Fig. 11. we show that our method works agnostic of the type of subject.

D. Additional Implementation Details

Architecture Details: We provide architecture details of the deformation field network and canonical NeRF networks in Figures 18 and 19 respectively.

Training: We train our network using the Adam optimizer [27] with a learning rate exponentially decayed by a factor 0.1 until the maximum number of iterations is reached. The exact hyper-parameters for each configuration are provided in Tab. 3.

Background Regularization: Since the total number of background points varies per scene, we sample 16384 points for each iteration when computing the background regularization loss in order to avoid memory issues. We additionally jitter each input point using Gaussian noise $\varepsilon \sim \mathcal{N}(0, 0.001)$ and use a robust Geman-McClure loss function [21] with $\alpha = -2$ and $c = 0.001$ implemented as per Barron [2].

Implementation: We extend the JAX [10] implementation of NeRF [18] for our method.

	Quasi-Static														Dynamic									
	GLASSES (78 images)		BEANIE (74 images)		CURLS (57 images)		KITCHEN (40 images)		LAMP (55 images)		TOBY SIT (308 images)		MEAN		DRINKING (193 images)		TAIL (238 images)		BADMINTON (356 images)		BROOM (197 images)		MEAN	
	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓	MS-SSIM↑	LPIPS↓
NeRF [39]	.638	.347	.581	.509	.445	.525	.702	.329	.651	.354	.765	.578	.630	.440	.612	.415	.655	.675	.757	.490	.613	.800	.659	.595
NeRF + latent	.741	.319	.727	.426	.623	.434	.770	.293	.753	.281	.767	.492	.730	.374	.846	.251	.776	.497	.835	.401	.653	.711	.777	.465
Neural Volumes [31]	.522	.576	.581	.563	.542	.551	.553	.572	.519	.556	.870	.512	.598	.555	.832	.255	.785	.626	.617	.694	.700	.717	.733	.573
NSFF†	.617	.443	.630	.531	.471	.543	.749	.320	.686	.348	.871	.488	.671	.446	.911	.237	.538	.683	.762	.541	.698	.746	.727	.552
$\gamma(t)$ + Trans† [29]	.830	.201	.785	.358	.768	.298	.858	.230	.870	.164	.841	.539	.825	.298	.901	.168	.858	.498	.913	.313	.714	.764	.847	.436
Ours ($\lambda = 0.01$)	.879	.146	.845	.266	.888	.155	.915	.155	.922	.112	.773	.186	.870	.170	.884	.0967	.722	.182	.913	.172	.634	.283	.788	.183
Ours ($\lambda = 0.001$)	.890	.149	.861	.268	.891	.147	.902	.153	.921	.109	.772	.206	.873	.172	.870	.106	.700	.197	.909	.174	.576	.360	.764	.210
No elastic	.862	.158	.882	.262	.875	.157	.890	.166	.924	.110	.790	.333	.870	.198	.880	.0957	.705	.197	.906	.175	.555	.329	.761	.199
No coarse-to-fine	.879	.158	.825	.279	.887	.157	.913	.154	.903	.121	.775	.311	.864	.197	.882	.107	.737	.297	.899	.199	.658	.510	.794	.278
No SE3	.874	.156	.839	.275	.884	.151	.907	.159	.905	.116	.778	.258	.865	.186	.885	.0968	.685	.217	.886	.202	.560	.314	.754	.207
Ours (base)	.882	.167	.795	.325	.868	.183	.874	.199	.889	.131	.758	.218	.845	.204	.885	.142	.740	.370	.880	.233	.656	.627	.790	.343
No BG Loss	.833	.160	.823	.275	.753	.197	.889	.165	.856	.135	.740	.163	.816	.182	.882	.0951	.691	.242	.857	.209	.555	.394	.746	.235

Table 2: SSIM and LPIPS metrics on validation captures against baselines and ablations of our system, we color code each row as **best**, **second best**, and **third best**. Please see the main text for PSNR.

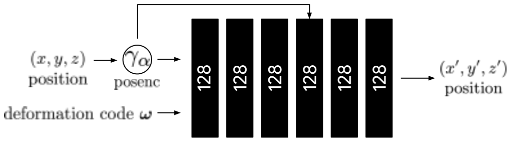


Figure 18: A diagram of our deformation network. The deformation network takes a position encoded position $\gamma_\alpha(\mathbf{x})$ using our coarse-to-fine annealing parameterized by α, along with a deformation code $\boldsymbol{\omega}$ and outputs a deformed position $\mathbf{x}'$. The architecture is identical for all of our experiments.

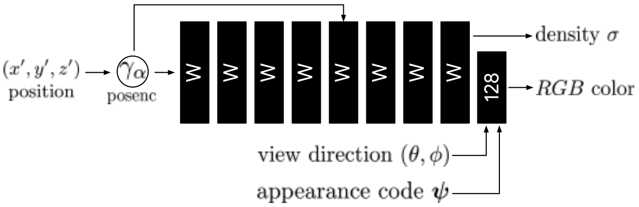


Figure 19: A diagram of the canonical NeRF network. Our network is identical to the original NeRF MLP, except we provide an appearance latent code ψ along with the view direction to allow modulating the appearance as in the NeRF-A model of [36]. The width W of the network is defined according to Tab. 3.

E. Experiment Details

E.1. Dataset Processing

Blurry Frame Filtering: For video captures, we filter blurry frames using the variance of the Laplacian [43]. To compute the blur score for an image, we apply the Laplace operator with kernel size 3 and compute the variance of the resulting image. We then filter the images based on this score to leave around 600 frames for each capture.

Camera Registration: For camera registration, we first compute a foreground mask using a semantic segmentation network such as DeepLabV3 [15]. We then use COLMAP [48] to compute the camera registration while using the mask to ignore foreground pixels when computing features. We found that this step can improve the quality of the camera registration in the presence of a moving foreground. We skip this step for captures for which we cannot obtain a segmentation mask such as for BADMINTON and BROOM.

Facial Landmarks: Although not necessary for our method, we use facial landmarks for selfie and full body captures to estimate a canonical frame of reference. Using this canonical frame of reference, we automatically generate visually appealing novel view trajectories of our reconstructed *nerfies*, like figure-eight camera paths in front of the user. We compute the 2D facial landmarks using MediaPipe's face mesh [34], and triangulate them in 3D using the Structure-from-Motion camera poses. We then set our canonicalized coordinate frame that is centered at the facemesh, with a standard orientation ($+y$ up, $+x$ right, $-z$ into the face), and with approximately metric units, by setting the scale so that the distance between the eyes matches the average interpupillary distance of 6 cm. Note that the 3D triangulation of facial landmarks is only correct if the subject is static, which is not guaranteed in our method, but in practice we observed that the triangulation result is sufficiently good to define the coordinate frame even when the subject rotates the head side-to-side. For the animal captures, we manually generate virtual camera paths.

E.2. Baselines

Comparison to Neural Volumes: Neural Volumes [31] reconstructs a deformable model of a subject captured by dozens of time-synchronized cameras. To apply it to our setting, where only one camera sees the subject at each time instance, we modify the encoder to take a single input image instead of three, as in the original method. We disable the background estimation branch and learn instead the complete scene centered around the face and scaled to a unit cube. For each frame, we render the volume from the viewpoint of the second camera of the validation rig and compute image comparison metrics. We provide quantitative comparisons in Table 1 in the main paper, and qualitative comparisons in Fig. 10 and Fig. 11.

We use a 128^3 voxel grid, a 32^3 warp field and train the network for 100k iterations for each of the five sequences.

We evaluate all results using the same camera parameters and spatial resolution. We show some renderings when interpolating the camera position between training and validation views in the supplementary video.

Comparison to NSFF: Concurrently to our work, Neural-Scene Flow Fields (NSFF) [29] proposes to model dynamic scenes by directly conditioning the NeRF with a position-encoded time variable $\gamma(t)$, modulating color, density, and a scene-flow prediction. Differences from our method are: (a) NSFF directly modulates the density of the NeRF by conditioning it with $\gamma(t)$ while our method uses a deformation field; (b) NSFF uses a position-encoded time variable ($\gamma(t)$) to condition each observation whereas our method uses a per-example latent code [5]; (c) NSFF uses depth from MIDAS [45] and optical flow from RAFT [56] as supervision whereas our method only uses a photometric loss.

We quantitatively compare with NSFF in Tab. 1 and Tab. 2, and show corresponding qualitative results in Fig. 10 and Fig. 11. We use the official code released by the NSFF authors. We use the default parameters provided by the authors on all datasets. We found that the motion segmentation algorithm provided by the authors fails in many cases, and therefore instead provide a "ground-truth" foreground mask obtained through DeepLab [15] when possible.

Additional Metrics: MS-SSIM metrics are in Tab. 2.

F. Additional Results

We show qualitative results from each of the sequences presented in our quantative evaluation (Tab. 1, Tab. 2) for quasi-static scenes (Fig. 10) and dynamic scenes (Fig. 11).

G. Limitations

Topological Variation: Our method struggles when the scene has motion which varies the topology of the scene. For example, when a person opens their mouth (as in Fig. 11 of the main text), the effective topology of their head changes. This is problematic for our method since we use a continuous deformation field parameterized by an MLP. In order to understand this, consider the mouth example and suppose that the template contains the person with their mouth open. Suppose that $\mathbf{x}_U$ and $\mathbf{x}_L$ are two adjacent points near the seam of the lips, and the $\mathbf{x}_U$ is on the upper lip and $\mathbf{x}_L$ is on the lower lip. It is then evident that a sharp discontinuity in the deformation is required to map both points to their appropriate positions on the template. Such a discontinuity is difficult for our continuous MLP to predict. We find that instead the optimization will often yield an incorrect but valid solution e.g., it will explain a closing mouth by protruding the lip and pulling it down as in Figure 11 of the paper.

Config	Resolution	Steps	Learning Rate	Batch Size	# Samples Fine	Coarse	Width W
FULL	1080p	1M	7.5e-4	3072	256	256	256
HALF	540p	100K	1e-3	8096	128	128	128

Table 3: Here we provide the hyper-parameters used for each configuration. FULL is the full resolution configuration used in our qualitative results. HALF is half the resolution of FULL and is used for our quantitative evaluation and ablation studies.

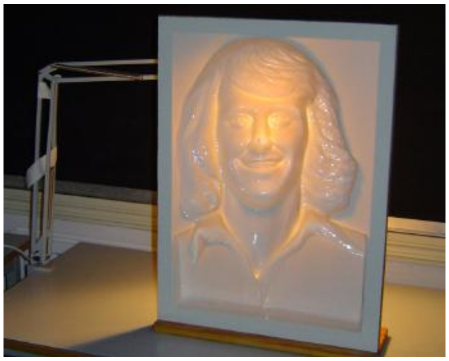
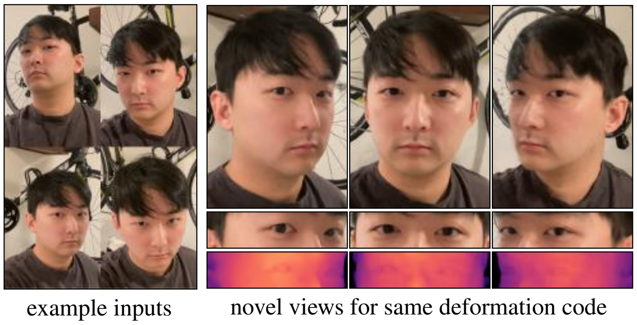


Figure 20: This concave mask of the Swedish tennis player Bjorn Borg appears to be convex due to the hollow-face illusion. By Skagedal, 2004 (Public Domain).



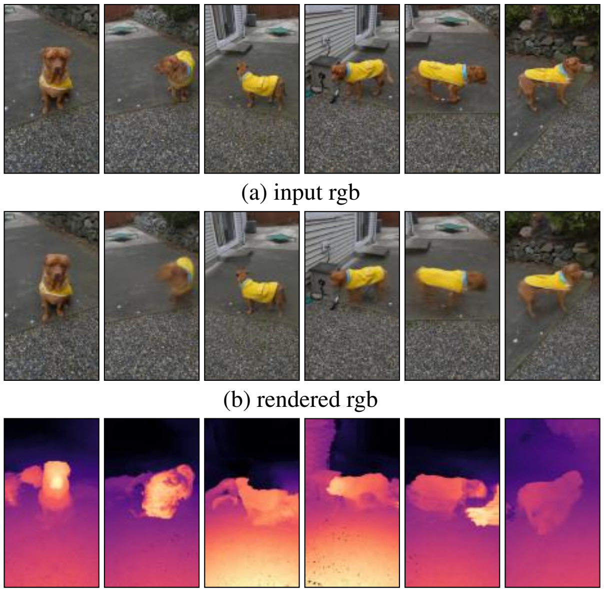
example inputs

novel views for same deformation code

Figure 21: If the user's gaze consistently follows the camera, the reconstructed *nerfie* represents the user's gaze as geometry, akin to the Hollow-Face illusion [22]. This is apparent in the depth map and makes the reconstructed model appear as if they are looking at the camera even when the geometry is fixed.

Rapid Motion: NeRF relies on seeing multiple observations to constrain where density lies in the volumes. In the presence of rapid motion, such as in Fig. 22, certain states of the scene may only be visible for a short period of time making it harder to reconstruct.

Orientation flips: Optimizations solving for any parameterization of rotations are known to be non-convex due to



(a) input rgb

(b) rendered rgb

(c) rendered depth

Figure 22: This example shows Toby the dog moving around freely, showcasing two limitations of our method. (1) *Rapid motion*: Because Toby moves quite fast, the camera only sees him in certain poses for a short amount of time, resulting in a sparse set of observations for certain poses. This can make those poses under-constrained. (2) *Orientation flips*: Toby wanders back and forth, showing different sides of his body. Depending on which orientation Toby is modeled as in the template, it is difficult for the deformation field to predict a flipped orientation.

both Gauge ambiguity and the inherent “twistedness” of the space of SO(3) [59]. As a simple example to illustrate this, imagine trying to align two coins in 3D. If the coin is initialized in a flipped orientation where heads faces the tail side of the other coin, then the ‘fit’ of the two coins must get worse before getting better when rotating towards the global minimum.

We encounter the same issue when optimizing for our deformation fields. If the template of a scene is in a certain orientation, but the deformation field for an observation is initialized in the wrong orientation the method will get stuck in a local minima and result in sub-optimal alignment. We show an example of this in Fig. 22 where frames with Toby’s left side visible are reconstructed better than when Toby’s right side is visible.

Hollow Face Illusion: The hollow-face illusion is an optical illusion where a concave (pushed in) imprint of an object appears to be convex (pushed out) instead. A feature of this illusion is that the convex illusion appears to follow the viewer’s eye. This illusion has been purposefully used in the Disneyland haunted mansion to create face busts which appear to follow you and in the popular T-Rex illusion [12]. We show an example of this illusion in Fig. 20.

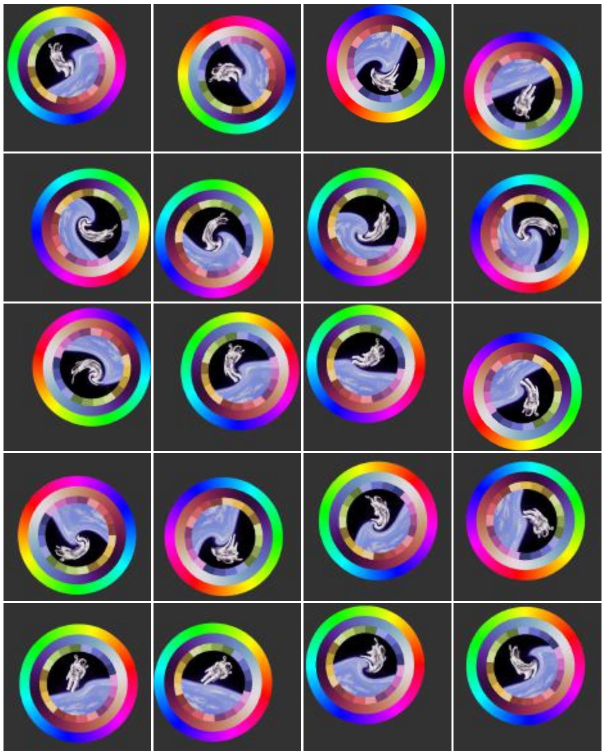


Figure 23: Our 2D toy dataset comprised of an image with a random translation, a random rotation, and a random nonlinear distortion near the center. Astronaut photo by Robert Gibson (1984, Public Domain).

We observe that the ambiguity which causes this illusion can also be a failure more for our method. In Fig. 21, we show an example where a user fixes their gaze in the direction of the camera while capturing themselves. Instead of modeling the eye motion as a deformation, our method models the eyes concavities as can be seen in the geometry.

H. 2D Deformation Experiment

Here we analyze the behavior of a deformation field in a 2D toy setting. In this 2D setting, a "scene" is comprised of a single image which is randomly translated, rotated, and nonlinearly distorted near the center. We show the full dataset in Fig. 23. Akin to our deformable NeRF setting, the task is to reconstruct each image by using a 2D deformation field which references a single template. The template is an MLP $F : (x, y) \rightarrow (r, g, b)$ which maps normalized image coordinates $x, y \in [-1, 1]$ to color values. The deformation field (i.e., 2D flow) is represented as an MLP $T : (x, y) \rightarrow (t_x, t_y)$ for a translation field or $T : (x, y) \rightarrow (\theta, p_x, p_y, t_x, t_y)$ for a rigid SE(2) transformation field. These are 2D analogs to the 3D translation field and SE(3) field described in Sec. 3.2.

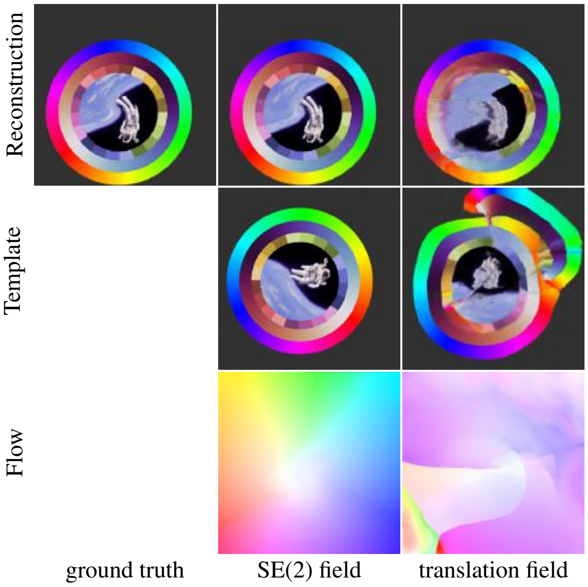


Figure 24: A comparison of our SE(2) field and a translation field. The translation parameterization has difficulty rotating groups of distant pixels whereas the rigid transformation successfully finds the correct orientation.

Deformation Formulation: Fig. 24 shows how an SE(2) rigid transformation field outperforms a translation field. An SE(2) field is able to faithfully reconstruct each image with a reasonable template and smooth deformation field. On the other hand, a translation field is not able to recover a reasonable template, and as a result the reconstruction has many artifacts and the flow field is messy.

Positional Encoding Frequencies: We show how changing the number of frequency bands changes the convergence behavior in Fig. 25. With a small number of frequencies ($m = 1$) we are able to converge to the correct orientation in the template, but cannot fully model the non-linear 'swirl' towards the center of the image. If we increase the number of frequencies ($m = 2 \ldots 6$) then while we can reconstruct the high swirl better, we start introducing artifacts due to early overfitting of the template and deformation field. With our coarse-to-fine approach we are able to both get the correct orientation without artifacts and also model the swirl.

References

[1] Sameer Ansari, Neal Wadhwa, Rahul Garg, and Jiawen Chen. Wireless software synchronization of multiple distributed cameras. *ICCP*, 2019. 7

[2] Jonathan T. Barron. A general and adaptive robust loss function. *CVPR*, 2019. 5, 11

[3] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824*, 2020. 2

[4] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. *SIGGRAPH*, 1999. 2

[5] Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the latent space of generative networks. *ICML*, 2018. 2, 4, 7, 13

[6] James Booth, Anastasios Roussos, Allan Ponniah, David Dunaway, and Stefanos Zafeiriou. Large scale 3D morphable models. *IJCV*, 2018. 2

[7] Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. Projective dynamics: Fusing constraint projections for fast simulation. *ACM TOG*, 2014. 2, 4

[8] Sofien Bouaziz, Yangang Wang, and Mark Pauly. Online modeling for realtime facial animation. *ACM TOG*, 2013. 2

[9] Aljaž Božič, Pablo Palafox, Michael Zollhöfer, Angela Dai, Justus Thies, and Matthias Nießner. Neural non-rigid tracking. *arXiv preprint arXiv:2006.13240*, 2020. 2

[10] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 11

[11] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. *CVPR*, 2000. 2

[12] brusspup. Amazing t-rex illusion!, Dec 2013. 14

[13] Thomas J Cashman and Andrew W Fitzgibbon. What shape are dolphins? building 3D morphable models from 2D images. *TPAMI*, 2012. 2

[14] Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. A simple geometric model for elastic deformations. *ACM Trans. Graph.*, 2010. 2, 4

[15] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. 12, 13

[16] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 2

[17] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM ToG*, 2015. 1, 2

[18] Boyang Deng, Jonathan T. Barron, and Pratul P. Srinivasan. JaxNeRF: an efficient JAX implementation of NeRF, 2020. 11

[19] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, et al. Fusion4D: Real-time performance capture of challenging scenes. *ACM ToG*, 2016. 1, 2, 4

[20] Ohad Fried, Ayush Tewari, Michael Zollhöfer, Adam Finkelstein, Eli Shechtman, Dan B Goldman, Kyle Genova, Zeyu Jin, Christian Theobalt, and Maneesh Agrawala. Text-based editing of talking-head video. *ACM TOG*, 2019. 2

[21] Stuart Geman and Donald E. McClure. Bayesian image analysis: An application to single photon emission tomography. *Proceedings of the American Statistical Association*, 1985. 5, 11

[22] Richard Langton Gregory. The intelligent eye. 1970. 13

[23] Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch,

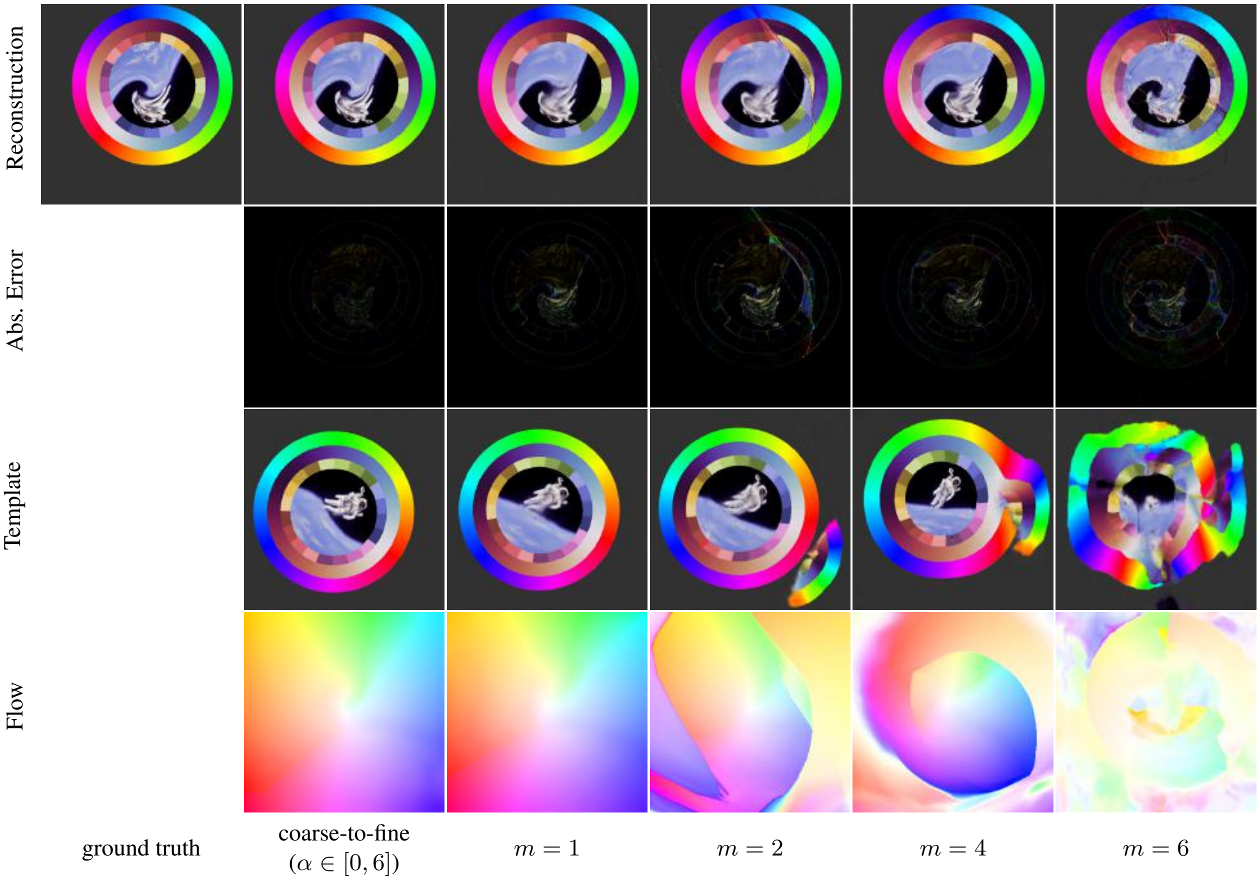


Figure 25: We show how the optimization changes depending on the number of frequency bands in the positional encoding of the deformation field. With 1 frequency, the model find the correct orientation of all images but is unable to model the high frequency distortion near the center. If we increase the frequencies then the templates overfits early and gets stuck in bad local minima. With our coarse-to-fine technique we are less prone to local minima while also modeling high frequency details.

Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, Danhang Tang, Anastasia Tkach, Adarsh Kowdle, Emily Cooper, Mingsong Dou, Sean Fanello, Graham Fyffe, Christoph Rhemann, Jonathan Taylor, Paul Debevec, and Shahram Izadi. The relightables: Volumetric performance capture of humans with realistic relighting. *ACM ToG*, 2019. 1

[24] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *NeurIPS*, 2018. 5, 11

[25] Chiyu Jiang, Jingwei Huang, Andrea Tagliasacchi, Leonidas Guibas, et al. Shapeflow: Learnable deformations among 3d shapes. *arXiv preprint arXiv:2006.07982*, 2020. 2

[26] Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Nießner, Patrick Pérez, Christian Richardt, Michael Zollöfer, and Christian Theobalt. Deep video portraits. *ACM ToG*, 2018. 2

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 11

[28] Hao Li, Linjie Luo, Daniel Vlasic, Pieter Peers, Jovan Popović, Mark Pauly, and Szymon Rusinkiewicz. Temporally coherent completion of dynamic shapes. *ACM TOG*, 2012. 2

[29] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv preprint arXiv:2011.13084*, 2020. 2, 7, 8, 9, 10, 12, 13

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

[31] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM ToG*, 2019. 2, 3, 7, 8, 9, 12

[32] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graph.*, 2015. 2

[33] BD LUCAS. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop, 1981*, 1981. 5

[34] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. MediaPipe: A framework for building perception pipelines. *arXiv preprint*

arXiv:1906.08172, 2019. 12

[35] Kevin M Lynch and Frank C Park. *Modern Robotics*. Cambridge University Press, 2017. 4, 10

[36] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. *CVPR*, 2021. 2, 3, 10, 12

[37] Philip F McLauchlan. Gauge invariance in projective 3d reconstruction. In *Proceedings IEEE Workshop on Multi-View Modeling and Analysis of Visual Scenes (MVIEW'99)*, pages 37–44. IEEE, 1999. 7

[38] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019. 2

[39] 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. *ECCV*, 2020. 1, 2, 3, 5, 6, 7, 8, 9, 12

[40] Richard A Newcombe, Dieter Fox, and Steven M Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. *CVPR*, 2015. 2, 3, 4

[41] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. *ICCV*, 2019. 2

[42] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[43] José Luis Pech-Pacheco, Gabriel Cristóbal, Jesús Chamorro-Martinez, and Joaquín Fernández-Valdivia. Diatom autofocusing in brightfield microscopy: a comparative study. In *ICPR*, volume 3, pages 314–317. IEEE, 2000. 6, 12

[44] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. *arXiv preprint arXiv:2011.13961*, 2020. 2, 7

[45] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 13

[46] Olinde Rodrigues. De l'attraction des sphéroïdes. In *Correspondence Sur l'École Impériale Polytechnique*, pages 361–385, 1816. 4

[47] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Dart: dense articulated real-time tracking with consumer depth cameras. *Autonomous Robots*, 2015. 2

[48] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. *CVPR*, 2016. 6, 7, 12

[49] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3D-aware image synthesis. *NeurIPS*, 2020. 2

[50] Eftychios Sifakis and Jernej Barbic. Fem simulation of 3D deformable solids: A practitioner's guide to theory, discretization and model reduction. *ACM SIGGRAPH 2012 Courses*, 2012. 4

[51] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. 2

[52] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *NeurIPS*, 2019. 2

[53] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. *EUROGRAPHICS*, 2007. 2, 4

[54] Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. *ACM TOG*, 2007. 2, 4

[55] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 2, 3, 5

[56] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, pages 402–419. Springer, 2020. 13

[57] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *TPAMI*, 2008. 2

[58] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video, 2021. 2

[59] Kyle Wilson, David Bindel, and Noah Snavely. When is rotations averaging hard? In *European Conference on Computer Vision*, pages 255–270. Springer, 2016. 14

[60] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. *arXiv preprint arXiv:2011.12950*, 2020. 2

[61] Hongyi Xu, Eduard Gabriel Bazavan, Andrei Zanfir, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. GHUM & GHUML: Generative 3D human shape and articulated pose models. *CVPR*, 2020. 2

[62] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. In *CVPR*, 2020. 2, 7

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

[64] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 7

[65] Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Trans. Graph.*, 2014. 4

[66] Michael Zollhöfer, Justus Thies, Pablo Garrido, Derek Bradley, Thabo Beeler, Patrick Pérez, Marc Stamminger, Matthias Nießner, and Christian Theobalt. State of the art on monocular 3D face reconstruction, tracking, and applications. *Computer Graphics Forum*, 2018. 2

[67] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3D menagerie: Modeling the 3D shape and pose of animals. *CVPR*, 2017. 2