Flow supervision for Deformable NeRF

Chaoyang Wang[1] Lachlan Ewen MacDonald[2] Laszlo A. Jeni[1] Simon Lucey[12]
[1]Carnegie Mellon University [2]University of Adelaide
{chaoyanw,laszlojeni}@cs.cmu.edu {lachlan.macdonald, simon.lucey}@adelaide.edu.au

https://mightychaos.github.io/projects/fsdnerf

Abstract

In this paper we present a new method for deformable NeRF that can directly use optical flow as supervision. We overcome the major challenge with respect to the computationally inefficiency of enforcing the flow constraints to the backward deformation field, used by deformable NeRFs. Specifically, we show that inverting the backward deformation function is actually not needed for computing scene flows between frames. This insight dramatically simplifies the problem, as one is no longer constrained to deformation functions that can be analytically inverted. Instead, thanks to the weak assumptions required by our derivation based on the inverse function theorem, our approach can be extended to a broad class of commonly used backward deformation field. We present results on monocular novel view synthesis with rapid object motion, and demonstrate significant improvements over baselines without flow supervision.

1. Introduction

Reconstructing dynamic scenes from monocular videos is a significantly more challenging task compared to its static-scene counterparts, due to lack of epipolar constraints for finding correspondences and ambiguities between motion and structure. Recent advances in differentiable rendering have lead to various solutions using an analysis-by-synthesis strategy – solving the non-rigid deformation and structure by minimizing the difference between synthesized images and input video frames. Among those, deformable neural radiance fields [14, 21, 25, 31] has been a notable technique to represent dynamic scenes and shows plausible space-time view synthesis results. However, the current implementations only warrant success on teleporting-like videos whose camera motions are significantly more rapid than object motions. Quality of their results significantly decrease on videos with more rapid object motions [6].

In this work, we conjecture the deficiency of these deformable NeRF-based methods is mainly due to lack of temporal regularization. As they represent deformation as *backward* warping from the sampled frames to some canonical space, the motions or scene flows between temporally adjacent frames is not directly modeled nor supervised. Another deficiency is that these methods minimize photometric error alone, which is insufficient for gradient descent to overcome poor local minima when the canonical and other frames has little spatial overlap. This deficiency is severe for non-object-centric videos with large translations.

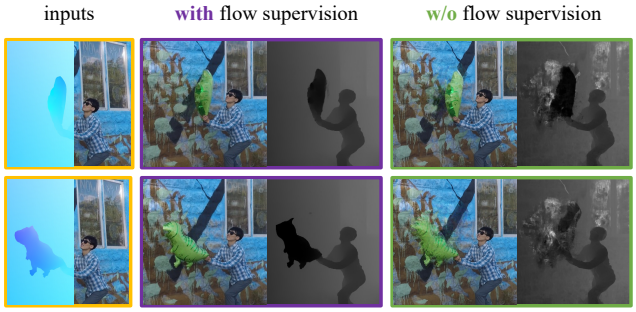


Figure 1. We propose a method to use optical flow supervision for deformable NeRF. It noticeably improves novel view synthesis for monocular videos with rapid object motions. In the figure, we visualize rendered novel view images and depth maps for the first and last frame of the input video.

The community has explored optical flow as an additional cue to help supervise the temporal transitions of other motion representations, such as scene flow fields [4, 5, 13, 43] and blend skinning fields [40]. However, enforcing flow constraints with respect to a generic *backward* warping field as in Nerfies [21] is non-trivial. Intuitively, to compute scene flows, it requires inverting the *backward* warp by having a *forward* warp which maps points from canonical space to other frames. Then scene flows can be computed either by taking time derivative of the forward warp or predicting the next position of a point through evaluating the forward warp. But this can be problematic since analytical inverse of a complicated non-bijective function (*e.g.* neural networks) is impossible, and an approximate solution by

having an auxiliary network to represent the forward warp will introduce computational overhead and is theoretically ill-posed. Counter to this intuition, we will show that inverting the backward warping function is actually not needed for computing scene flows between frames.

The main **contribution** of this paper is: we derive an analytical solution to compute velocities of objects directly from the *backward* warping field of the deformable NeRF. The velocities are then used to compute scene flows through temporal integration, which allows us to supervise the deformable NeRF through optical flow. This leads to significant improvement on videos with more rapid motions, as shown in Fig. 1.

The advantage of our approach is twofold: (i) Our method applies to all kinds of backward warping function, thanks to the weak assumptions required by the inverse function theorem which our derivation is based on. Thus our method is more general compared to other works using invertible normalizing flows [11] or blend skinning [3, 40]. (ii) Our method is also computationally more tractable compared to neural scene flow fields [4, 12, 13], which would require integrating flows over long period of time to reach some canonical frame.

2. Related works

Deformable NeRF. One common approach to model the dynamic scene is to represent it as the deformation of a static unknown template, and reconstruction is then done by fitting the model to the input 2D observations. Such approach has been implemented using techniques from recent advances in differentiable rendering. Most noticeably is deformable neural radiance field [14, 21, 25, 31], which models the deformation and the template radiance field using coordinate-based neural networks, and employs volumetric rendering to synthesis images under different viewing directions.

More concretely, to synthesize the color of a pixel at time t, it first samples points along the line of sight. The sampled points $\mathbf{p}$'s are then fed into a *backward* deformation field , *i.e*.

$$w_{c\leftarrow}(\mathbf{p}; t) \longrightarrow \mathbf{p}_c, \tag{1}$$

which maps the input spacetime point $(\mathbf{p}, t)$ to its corresponding 3D position $\mathbf{p}_c \in \mathbb{R}^3$ in the canonical space. Colors $\mathbf{c}$ and densities σ are then queried from the radiance field network, *i.e*.

$$f(\mathbf{p}_c, \mathbf{d}, \boldsymbol{\lambda}) \longrightarrow \mathbf{c}, \sigma, \tag{2}$$

where $\mathbf{d} \in S^2$ is the viewing direction and $\boldsymbol{\lambda}$ is an additional frame-wise code to allow the template to vary per frame so as to cope with topological and appearance changes [17, 21]. $\boldsymbol{\lambda}$ can also be extended to vary spatially as ambient embeddings to enable greater flexibility to topological changes [22]. With the computed colors and densities of points along the viewing ray, RGB intensities of a pixel are computed using the volumetric rendering equations proposed in NeRF [18]. Finally, the optimization objective is to minimize the difference between rendered pixels and the input observations.

Backward deformation fields. Different ways exist for representing the backward deformation field. The most straightforward design is to use a neural network to output displacement between the input point and its canonical position [14, 27, 31]. Park *et al*. shows that having the neural network outputting SE(3) transformations leads to improvement in reconstructing rotational motions [21].

For articulated objects such as animals and humans, blend skinning is widely used in literature [9, 15, 44, 45]. However, it is mainly designed for *forward* deformation, thus adjustment is needed to adapt it for *backward* warping field. Yang *et al*. [39,40] uses mixture of gaussians to model the blending weights. This enables them to approximate the inverse of a forward blend skinning by simply inverting the SE(3) transformation of each deformation nodes. On the other hand, instead of explicitly define a deformation function, Chen *et al*. [3] solves for canonical correspondences of any deformed point using iterative root finding.

For homeomorphic deformation where the mappings between any frames are bijective and continuous, recent works explored the use of invertible normalizing flows [2, 11] where the forward and backward deformation are computed with the same network parameters. The downside is the network architecture is restrictive, and in practice requires more compute due to having more coupling layers to enumerate different axis partitions.

Other dynamic NeRF representations. Instead of having a static template NeRF, other works [5,13,32] choose to use a time-modulated NeRF to directly represent warped radiance fields. To enforce temporal consistency, they optimize neural scene flow fields to regularize pairwise motion between adjacent frames. This is only suitable for enforcing short-term consistency, but intractable for long-term consistency due to the expensive computational cost for performing scene flow integration. To improve computational efficiency, Wang *et al*. [32, 33] propose neural trajectory field which directly outputs trajectories for all space-time locations. This allows enforcing long-term consistency without the need for scene flow integration.

Optical flow supervision. Using optical flows to assist view interpolation [1, 8, 35] and 3D reconstruction [10, 16, 29] has been a common practise. Several recent dynamic NeRF works [4, 5, 13, 32] also use optical flow supervision, but none of them is based on backward deformation fields. Yang *et al*. [40] apply optical flow to supervise a blend skinning deformation field for object-centric reconstruction. Their result focuses on articulated objects and requires amodel segmentation mask as input. Under the

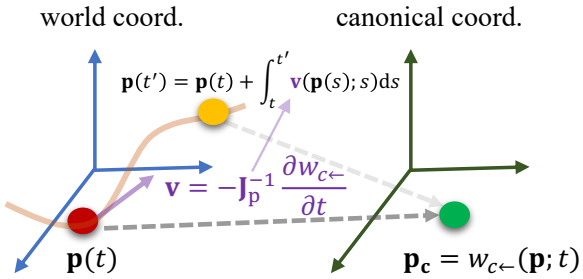


Figure 2. Given a point at $\mathbf{p}(t)$ and a *backward* warping field $w_{c\leftarrow}$, we want to compute the 3D scene flow by predicting the next position of the point at time t'. We achieve this by deriving the velocity field v as a differentiable function of $w_{c\leftarrow}$, and then perform time integration.

context of view synthesis for dynamic scenes, we are unaware of any deformable NeRF-based approach using flow supervision. Thus our result provides a useful tool for future related research.

3. Supervise deformable NeRF by flow

Problem setup. We concern the problem of fitting the deformable NeRF given a monocular video input. We precomputed the camera intrinsics and extrinsics using off-the-shelf structure-from-motion methods *e.g.* Colmap [28]. Optical flows between neighboring frames $o_{t\rightarrow t\pm\nabla t}$ are also computed using RAFT [30]. Then we want to find optimal parameters for the backward deformation field $w_{c\leftarrow}$ and radiance field f such that the synthesised images c'_t and optical flow maps $o'_{t\pm t'}$ match the input video frames c_t and precomputed optical flow maps $o_{t\pm t'}$,

$$\min \sum_t \Big[\underbrace{\|c'_t - c_t\|_2^2}_{\text{image loss}} + \beta \sum_{t'\in\{t\pm\Delta t\}} \underbrace{\|M_{t\rightarrow t'} \odot (o'_{t\rightarrow t'} - o_{t\rightarrow t'})\|_1}_{\text{optical flow loss}} \Big] \quad (3)$$

To prevent errors in the precomputed optical flow maps from misleading the reconstruction, we use binary masks $M_{t\rightarrow t'}$ to turn off losses for pixels which fail the forward-backward flow consistency test. Moreover, we follow the trick proposed by Li *et al.* [13] to gradually decay the weighting β during optimization, so that the reconstruction is able to correct mistakes of the input optical flows.

The key question is then how to synthesize optical flows $o'_{t\rightarrow t'}$ from $w_{c\leftarrow}$ and f?

3.1. Velocity fields from $w_{c\leftarrow}(p; t)$

Velocity fields $v(\mathbf{p}; t) : \mathbb{R}^3\times\mathbb{R} \rightarrow \mathbb{R}^3$ describe the velocity of an object if placed at position $\mathbf{p}$ in the world coordinate at time t. Velocity fields is straightforward to compute if the *forward* deformation field $w_{c\rightarrow}(\mathbf{p_c}; t) : \mathbb{R}^3\times\mathbb{R} \rightarrow \mathbb{R}^3$

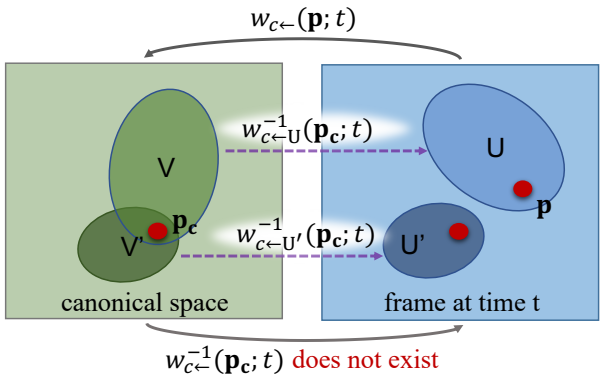


Figure 3. Illustration of the invertibility for *backward* deformation field $w_{c\leftarrow}$. Modeled by coordinate-based MLP, $w_{c\leftarrow}(\mathbf{p}; t)$ is not invertible on the whole input domain. But local homeomorphism exists. we can have a bijective mapping $w_{c\leftarrow U}(\mathbf{p}; t)$ from an open set U to another open set V in the canonical space. Although $w^{-1}_{c\leftarrow U}(\mathbf{p}_c; t)$ exists locally from V to U, but it is not possible to analytically derive it in closed form. Fortunately, inverting $w_{c\leftarrow U}$ is not needed for predicting scene flow due to equation (5).

exists, *i.e.*:

$$v(\mathbf{p}; t) = \frac{\partial w_{c\rightarrow}}{\partial t}\bigg|_{(w_{c\leftarrow}(\mathbf{p};t);t)}. \quad (4)$$

However, this intuitive solution is flawed if $w_{c\rightarrow}$ is not strictly an inverse function of $w_{c\leftarrow}$. Unfortunately having an invertible $w_{c\leftarrow}$ is theoretically unwarranted since most deformation representations such as neural networks and blend skinnings are not bijective.

Since seeking a *global* inverse function of $w_{c\leftarrow}$ on the whole domain is impractical, we propose to consider local regions of $w_{c\leftarrow}$'s domain where *local* inverse function may exist. And based on the inverse function theorem, we find that we can analytically compute $v(\mathbf{p}; t)$ for any positions $\mathbf{p}$ without actually inverting $w_{c\leftarrow}$, as long as $w_{c\leftarrow}$ is bijective in an open set includes $\mathbf{p}$. This leads to the main theoretical result of the paper.

Proposition. If the warp Jacobian matrix $\mathbf{J_p}(\mathbf{p}, t) = [\frac{\partial w_{c\leftarrow}(\mathbf{p};t)}{\partial \mathbf{p}}]$ is non-singular at some position $\mathbf{p}$ at time t, and there exists an open set including $\mathbf{p}$ where $w_{c\leftarrow}$ is continuously differentiable, then the velocity at $(\mathbf{p}, t)$ is computed as:

$$v(\mathbf{p}; t) = -\mathbf{J}_\mathbf{p}^{-1}(\mathbf{p}, t)\frac{\partial w_{c\leftarrow}(\mathbf{p}; t)}{\partial t}. \quad (5)$$

Proof. First, we upgrade $w_{c\leftarrow}$ to have the same input and output dimension, *i.e.* $\phi(\mathbf{p}, t) = (w_{c\leftarrow}(\mathbf{p}; t), t)$. Then given the assumptions made by the proposition, and according to the inverse function theorem [20], ϕ is a local diffeomorphism. In other words, there is some open set U containing $(\mathbf{p}, t)$ and an open set V containing $\phi(\mathbf{p}, t)$ such that

$\phi_U := \phi : U \to V$ has a continuous and differentiable inverse $\phi_U^{-1} : V \to U$; in particular, for $(\mathbf{p}_c, t) = \phi(\mathbf{p}, t)$, we have $(J\phi_U^{-1})(\mathbf{p}_c, t) = [(J\phi_U)(\mathbf{p}, t)]^{-1}$. Re-expressing $\phi_U = (w_{c\leftarrow}, t)$ and denoting the inverse of $w_{c\leftarrow}$ inside the open set U as $w_{c\leftarrow U}^{-1}$ (see Fig. 3) yields :

$$\left[\begin{matrix} J_{\mathbf{p}} w_{c\leftarrow U}^{-1} & \frac{\partial w_{c\leftarrow U}^{-1}}{\partial t} \\ \mathbf{0}_3^\top & 1 \end{matrix}\right]\Bigg|_{(\mathbf{p}_c,t)} = \left[\begin{matrix} J_{\mathbf{p}} w_{c\leftarrow} & \frac{\partial w_{c\leftarrow}}{\partial t} \\ \mathbf{0}_3^\top & 1 \end{matrix}\right]^{-1}\Bigg|_{(\mathbf{p},t)} \quad (6)$$

By moving the matrix inverse inside the block matrix on the righthand side of (6) through Schur complement, we have $\frac{\partial w_{c\leftarrow U}^{-1}(\mathbf{p}_c,t)}{\partial t} = -[(J_{\mathbf{p}} w_{c\leftarrow})(\mathbf{p}, t)]^{-1} \frac{\partial w_{c\leftarrow}(\mathbf{p},t)}{\partial t}$. Finally by noticing that $v(\mathbf{p}; t) = \frac{\partial w_{c\leftarrow U}^{-1}(\mathbf{p}_c,t)}{\partial t}$, we have the proof. □

We note that the sufficient condition of the proposition is weak and satisfied for most domain of deformation fields. For rare cases where $\det(\mathbf{J}_{\mathbf{p}}) < \epsilon$ for some space-time positions $(\mathbf{p}, t)$, we choose to exclude it from evaluating the loss so as to avoid numerical instability. Moreover, we implement equation (5) as a differentiable operator so that gradients can be back propagated during optimization.

3.2. Scene flow from time integration of velocity

With the velocity fields $v(\mathbf{p}; t)$ computed by equation (5), we estimate 3D scene flows *i.e.* the displacement between $\mathbf{p}(t)$ and $\mathbf{p}(t + \Delta t)$ by time integration (see Fig. 2),

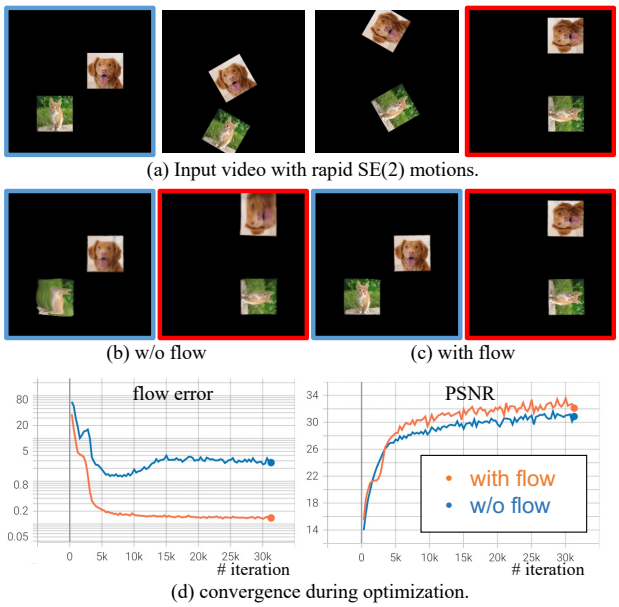
$$\mathbf{p}(t + \nabla t) - \mathbf{p}(t) = \int_t^{t+\Delta t} v(\mathbf{p}(s); s) ds. \quad (7)$$

We implement the time integration though differentiable numerical ODE solvers. Because in our problem we are dealing with scene flows between small time intervals ∇t, which usually is the duration of 1 or 2 frames, we find unrolling the fourth order Runge-Kutta method [7] for two iterations gives stable results and achieves good trade-off between accuracy and computational cost. To prevent overfitting to a fixed step size, we randomly perturbed the step size by adding a Gaussian noise.

Rendering optical flow. For every viewing ray at time t, we first calculate the next position $\mathbf{p}(t + \Delta t)$ for each sampled points along the ray by equation (7). Then the expected next position $\bar{\mathbf{p}}(t + \Delta t)$ for the visible points is estimated by weighted averaging $\mathbf{p}(t+\Delta t)$'s using NeRF's volumetric rendering equation. Finally, the optical flow is estimated by taking the difference between the 2D projection of $\bar{\mathbf{p}}(t+\Delta t)$ and the pixel location of the viewing ray.

3.3. Removing Gauge freedom

One issue for the deformable NeRF is the recovered background tends to be not static. Deformation of the apparent static scene regions severely affects viewing experience. We find that a main cause for the jittering background is due



(a) Input video with rapid SE(2) motions.
(b) w/o flow
(c) with flow
(d) convergence during optimization.

Figure 4. A 2D toy experiment showing flow supervision is crucial for reconstructing rapid object motions. (a) 4 frames evenly sampled from the synthetic video. Two image patches are moving with large linear and angular velocity and has no overlap between their starting and ending positions. (b) fitting a SE(2) deformation field fails to recover correct motions for the first and last frame. (c) adding flow supervision correctly reconstruct the video frames.(d) we monitor the flow error and PSNR of reconstructed image during optimization. The flows are estimated using equation (5,7). Our method (with flow) converges significantly faster than w/o flow, and is able to fit the input optical flows with low error.

to the Gauge ambiguity in deformable NeRF, *i.e.* the optimization objective in (3) does not specify which canonical coordinate the deformation field is defined at. In theory, any deformation of a canonical space can also be a valid one. This causes confusion for the deformable NeRF during optimization, which tends to trap it in the wrong factorization of motion and shapes.

Therefore, we propose to remove the Gauge freedom by specifying that the canonical coordinate is attached to a key frame t_0 from the input sequence. This is enforced through adding a loss to penalize the magnitude of deformation at time t_0, *i.e.*

$$\mathcal{L}_{\text{Gauge}} = \sum_{\mathbf{p}} \|w_{c\leftarrow}(\mathbf{p}; t_0) - \mathbf{p}\|_2. \quad (8)$$

Through our experiments, we choose the middle frame of the input sequence as the canonical frame. This is based on the assumption that the average deformation to other frames is likely to be minimal in the middle frame. Nevertheless, more sophisticated algorithms of choosing canonical frames may further improve the robustness of the method.

We note that prior works would introduce extra regular-

ization to enforce static background. Compared to training the density field to align with sparse points estimated by structure from motion [21, 38], our approach is self-contained without external modules. Compared to prior works having separate NeRFs for static and dynamic regions [5, 13, 17, 37], our implementation is computationally more friendly. Though all the above methods could be included as supplementary.

4. Experiment

4.1. Implementation details

Deformation field. We follow Nerfies [21] by using a 6 layer 128-width MLP which outputs SE(3) transformations. The deformation MLP_g takes input $\mathbf{p} \in \mathbb{R}^3$ and an 8-dim deformation code. Instead of treating the codes as independent variables, we use another 2 layer 32-width MLP_α to map time t to the deformation code. This helps improve convergence under smaller batch size. We also choose to use softplus as the activation function, since it produces smoother outputs than ReLU. With these, mathematically $w_{c\leftarrow}(\mathbf{p}; t) = \text{MLP}_g(\mathbf{p}, \text{MLP}_\alpha(t)) \circ \mathbf{p}$, where $\circ$ denotes the action of SE(3) transform.

Radiance field. We use the original architecture from Nefies with minor modification. Instead of having independent 8-dim appearance codes to optimize, we use a 2 layer 32-width MLP to produce the codes conditioned on time t.

Optimization hyperparameters. We set the initial weighting β for the optical flow loss as 0.04 and anneal it to 0.0001. We set a fixed weighting for $\mathcal{L}_{\text{Gauge}}$ as 1. The initial learning rate of Adam optimizer is 0.001, and decayed 1/10 every 50 epochs. We train a total of 120 epochs or roughly 150k iterations. Each batch samples 4096 rays from 8 frames, and samples 256 points per ray. The optimization takes 4 GTX-3090 GPUs approximately 13hrs.

4.2. Effectiveness of flow supervision

In experiments, we evaluate the effectiveness of flow supervision using the velocity fields derived in equation (5). We aim to answer two key qustions:

- *Does our method help improve convergence for rapid motions?*
- *Does the flow supervision help disambiguate dynamic motion and structure in monocular deformable NeRF?*

4.2.1 Flow supervision for fitting rapid motions

To clearly address the first question, we conduct a 2D toy experiment. As shown in Fig. 4, we created a 25-frame video, consists of two rapidly moving image patches. Unlike the synthetic experiment in Nerfies [21], the image patches have large translational motion in addition to rotations, and the motions are different for each patches. It turns out that fitting an SE(2) deformation field using only image intensity loss is slow to converge, and results in distorted images. Applying optical flow loss by our method significantly speeds up convergence, and gives correct image reconstruction. This result indicates that our flow calculation algorithm *i.e*. equation (5,7) is effective and also flow supervision is necessary for handling rapid motions.

4.2.2 Monocular dynamic view synthesis

Dataset. State-of-the-art deformable NeRF, *e.g*. Nerfies and HyperNeRF [21, 22] have shown satisfactory view synthesis result on the data they captured. However as discussed by Gao *et al*. [6], Nerfies and HyperNeRF's data have high effective multi-view factors (EMFs). In other words, the objects are either quasi static or the camera motions are significantly larger than object motions. In this work, we evaluate on datesets with less EMFs.

We first report results on the NVIDIA dynamic view synthesis dataset (NDVS) [41] which has significantly lower EMFs. We follow the preprocessing steps of NSFF [13] which extracted 24 frames per sequence from the raw multi-view videos in NDVS. Neighboring frames are extracted from different cameras to simulate a monocular moving camera. For fair comparison with NSFF, we also downsize the images to have 288 pixels in height. For evaluation, we compare synthesized images to all images captured by 12 cameras, and report metrics such as PSNR, SSIM [36] and LPIPS [42].

We next compare view synthesis results on sequences collected by the authors of Nerfies [21]. To ensure the inputs appear as if they were casually recorded in real life, we use video frames only from the left camera from the stereo rig, as opposed to teleporting between the left and right cameras in the original paper of Nerfies.

We also test our method on one DAVIS sequence [23] and two casual videos captured by Wang *et al*. [32].

Trajectories by velocity integration. To visually inspect the quality of our optimized deformation field and the derived velocity fields, we sparsely sample points on the surface of the reconstructed scene, and perform time integration with the velocity fields to create trajectories. As visualized in Fig. 5, the recovered trajectories are smooth and closely follow the object motions.

Foreground background separation. Since we removed Gauge freedom by picking one video frame as the canonical frame, the distance of a point $\mathbf{p}$ to its canonical correspondence $\mathbf{p}_c$ now directly indicates whether the point is static or moving. In Fig. 6, we visualized the distance $\|\mathbf{p} - \mathbf{p}_c\|_2$ for each frame in a video. To visualize 3D volumes of distances in 2D, we project the distances along a ray by the

Figure 5. 3D visualizing of trajectories computed by integrating velocity fields $v(\mathbf{p}; t)$ from equation (5). Trajectories are colored to represent temporal order, and overlayed with colored point clouds extracted from the optimized deformable NeRF. Bottom row shows two frames from each of the input videos.

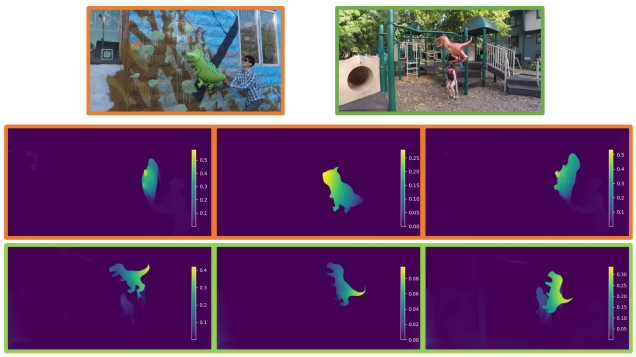


Figure 6. Visualization the distance $\|\mathbf{p} - \mathbf{p}_c\|_2$ by volumetric rendering. Since we removed the Gauge freedom by picking the canonical frame to be one of the input frame, large distance only happens on the moving objects. Our result shows clean separation between the moving forground objects and the static background.

volumetric rendering equation in NeRF. Thus brightness of the color corresponds to the distance of the visible area. We only observe large distances for the moving balloons and some small distances for the human subjects. The distances on the static background region are correctly rendered as close to 0. This indicates our success on the proposed removal of Gauge freedom.

Baselines. We formed a close comparison with the state-of-the-art deformable NeRFs *e.g*. Nerfies [21] on the NDVS dataset. Due to the official code from the authors do not work out of the box for the NDVS dataset, we adapt it by changing its Euclidean coordinates to the NDC coordinates, so as to automatically deal with the increased scene depth in some of the NDVS sequences. Given all the aforementioned implementation and design choices, our own method is essentially applying the proposed flow supervision to the adapted Nerfies implementation. Thus comparison to Nerfies also serves as an ablation showing the effectiveness of the proposed flow supervision.

We also compared with another deformable NeRF approach, *i.e*. NR-NeRF [31], whose deformation network outputs translation rather than SE(3) transformation. Finally, as a reference, we compared to NSFF [13], which optimizes a time-modulated NeRF and scene flow fields, and is supervised not only by optical flows but also by the depth maps from the state-of-the-art monocular depth estimation network [26]. Thus NSFF serves as a strong reference to check the status of other methods without depth supervision. We summarize the compared methods in Table. 1.

	representation		supervision		
method	motion	NeRF	image	flow	depth
NSFF [13]	scene flow	dynamic	✓	✓	✓
NR-NeRF [31]	translation	static	✓		
Nerfies [21]	SE(3)	static	✓		
HyperNeRF [22]	SE(3) + ambient	semi-static	✓		
ours	SE(3)	static	✓	✓	

Table 1. Summary of the compared deformable NeRF methods and neural scene flow field (NSFF).

With vs. w/o flow supervision. In Fig. 7 we form a side-by-side comparison with Nerfies, which does not use optical flow supervision. We notice that Nerfies constantly make structural mistakes as indicated by its rendered noisy depth maps. As a result, it has noticeable artifacts concentrated around the dynamic objects. In contrast, with the help from flow supervision, our method renders smoother depth maps and visually more pleasing view synthesis images. These improvements are reflected quantitatively in Tab. 2, where we show consistent improvement across all metrics. Our method also produces more plausible results for quasi-static scenes as shown in Fig. 8.

Compare with NSFF. In table 2 we show competitive results on Balloon1 and Umbrella compared to NSFF which is supervised using depth. However we fall behind on the Playground and Balloon2 sequence. Closer diagnoses show that this is due to wrong depth scales which our method assigned to the fast moving balloons. As shown in Fig. 9, our method actually produces smoother depth maps compared to NSFF, thanks to our stronger temporal constraint due to having a single static template NeRF. However as high-

Figure 7. Results on NVIDIA dyanamic view synthesis dataset (NDVS). With flow supervision, our method produces smooth depth map and sharp novel view images. Without flow supervision leads to severe artifacts and noisy depth maps. We note that *Nerfies is our own adaptation of the official code for NDVS dataset.

method		Playground			Balloon1			Balloon2			Umbrella		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
NSFF [13]	full	24.69	0.889	0.065	24.36	0.891	0.061	30.59	0.953	0.030	24.40	0.847	0.088
	dyn.	19.02	0.715	0.123	18.49	0.619	0.174	24.46	0.843	0.065	16.82	0.546	0.156
NR-NeRF [31]	full	14.16	0.337	0.363	15.98	0.444	0.277	20.49	0.731	0.348	20.20	0.526	0.315
	dyn.	11.78	0.221	0.466	16.94	0.548	0.398	12.65	0.353	0.575	16.20	0.435	0.321
w/o flow	full	22.18	0.802	0.133	23.36	0.852	0.102	24.91	0.864	0.089	24.29	0.803	0.169
(*Nerfies)	dyn.	16.33	0.535	0.244	18.66	0.613	0.215	20.50	0.717	0.141	17.57	0.581	0.202
w flow	full	22.39	0.812	**0.109**	24.36	0.865	0.107	25.82	0.899	**0.081**	24.25	0.813	**0.123**
(ours)	dyn.	16.70	0.597	**0.168**	19.53	0.654	**0.175**	20.13	0.719	**0.113**	18.00	0.597	**0.148**

Table 2. Comparing deformable NeRFs and NSFF on the NVIDIA dynamic view synthesis (NDVS) dataset. Metrics are reported for the full image as well as only for the masked regions containing dynamic motions. Our method shows significant improvement over deformable NeRF methods without flow supervision. Comparison with NSFF is mixed, mainly due to the scale ambiguity without depth supervision. See Fig. 9 for further discussion.

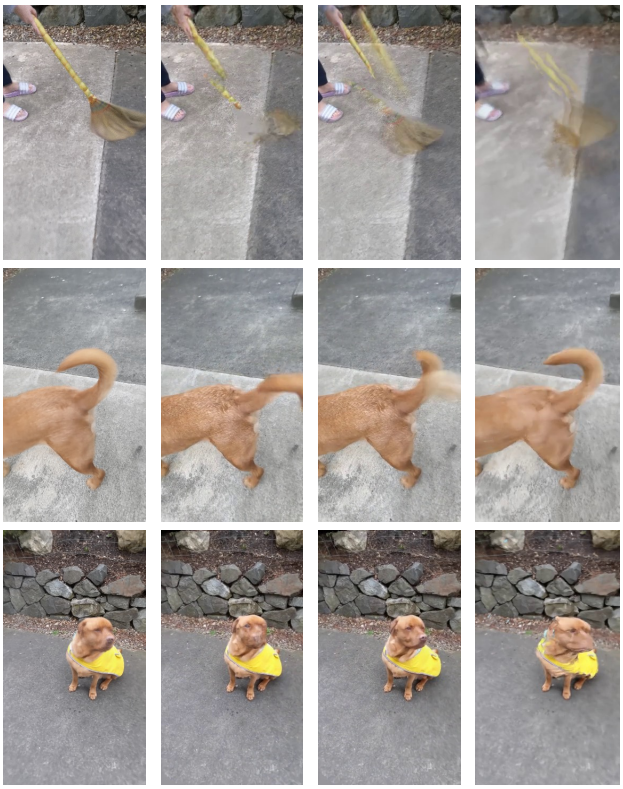
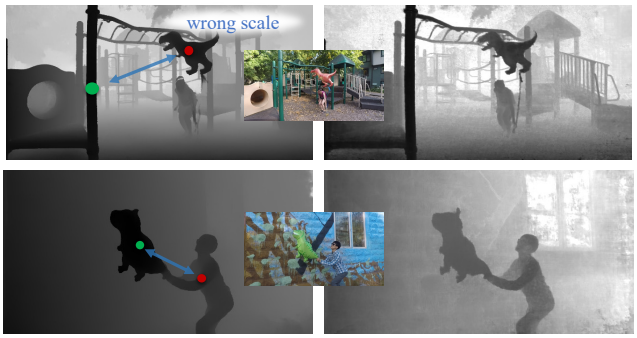


Figure 8. View synthesis results on sequences from Nerfies [21]. We only use frames from the left camera for training, instead of teleportation between left and right cameras as in the original paper of Nefies. We find the compared methods make noticeable mistakes in the "broom" sequence (1st row) and some frames in the "tail" sequence (2nd row). In the "toby-sit" sequence (3rd row), HyperNeRF and NSFF produce blurry or distorted dog faces in some frames. In contrast, our method consistently yields a more plausible view synthesis result. This indicates that adding flow supervision by our method is also helpful for quasi-static videos.

lighted by the blue arrows, the depth value of the moving objects are too small in comparison to the reference points on the background. We hypothesize this is due to the small motion bias of the deformation field which tends to explain 2D motions with smaller 3D motions. This causes the rendered foreground objects have significant offsets compared to the groundtruth, and consequently receives large penalties in terms of the image similarity metrics used in Table 2, even though our method produces equivalent if not sharper view synthesis result compared to NSFF. This scale ambiguity issue is inherent from the single camera problem setup and should not blame the methods supervised without depth. To recover plausible relative scales of different moving parts of a dynamic scene, mid or higher level reasoning (*e.g.* learning-based depth estimation [26, 34], 2D supervision from image generative models [24]) is required.



(a) ours (b) NSFF

Figure 9. Our method suffers from scale ambiguity due to not using depth supervision. This is highlighted on the top left by comparing the points on the balloon and the pole, where the balloon should be behind the pole rather than having similar depth. Similarly, on the bottom left, the arm of the person should be close to the balloon, not behind it. Although we produce much smoother depth maps compared to NSFF, we make more error in the scale of depth, resulting in lower metrics in Table 2.

(a) selected canonical frame (b) noisy depth

Figure 10. For highly deformable objects such as umbrella, the choice of the canonical frame is sensitive. In this example, we choose the first frame instead of mid frame as the canonical frame, which has a very different topogy compared to other frames when the umbrella is open. This leads to degenerated results as highlighted by the red box.

5. Discussion

We presented a method to apply flow supervision for deformable NeRF. We demonstrated that our method significantly improves view synthesis quality of deformable NeRF on videos with lower effective multi-view factors. However, due to the ambiguities of the monocular 3D reconstruction problem, our method has following **limitations:** (i) our method is not able to recover correct relative scale of moving objects (see Fig. 9); (ii) our method can be sensitive to the selection of canonical frame if there is large object deformations (see Fig. 10); (iii) it requires sufficient motion parallax and does not work for fixed or small camera motion; (iv) Long optimization time is required, but could be sped up using more efficient implementation [19].

Acknowledgement. This work was partially supported by Apple and by NSF award No. IIS-1925281. We thank Tim Clifford and Ian R Fasel from Apple for the helpful discussions, Orazio Gallo for collecting data and Hang Gao for helping with the baselines.

References

[1] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3703–3712, 2019. 2

[2] Hongrui Cai, Wanquan Feng, Xuetao Feng, Yan Wang, and Juyong Zhang. Neural surface reconstruction of dynamic scenes with monocular rgb-d camera. *arXiv preprint arXiv:2206.15258*, 2022. 2

[3] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11594–11604, 2021. 2

[4] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 1, 2

[5] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5712–5721, 2021. 1, 2, 5

[6] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. Monocular dynamic view synthesis: A reality check. *arXiv preprint arXiv:2210.13445*, 2022. 1, 5

[7] Richard Hamming. *Numerical methods for scientists and engineers*. Courier Corporation, 2012. 4

[8] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Rife: Real-time intermediate flow estimation for video frame interpolation. *arXiv preprint arXiv:2011.06294*, 2020. 2

[9] Ladislav Kavan. Part i: direct skinning methods and deformation primitives. In *ACM SIGGRAPH*, volume 2014, pages 1–11, 2014. 2

[10] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 2

[11] Jiahui Lei and Kostas Daniilidis. Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6624–6634, 2022. 2

[12] Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural scene flow prior. *Advances in Neural Information Processing Systems*, 34:7838–7851, 2021. 2

[13] 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. 1, 2, 3, 5, 6, 7

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

[15] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015. 2

[16] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM TOG*, 2020. 2

[17] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. 2020. 2, 5

[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 *ECCV*, 2020. 2

[19] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv:2201.05989*, Jan. 2022. 8

[20] James R Munkres. *Analysis on manifolds*. CRC Press, 2018. 3

[21] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020. 1, 2, 5, 6, 8

[22] 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 (TOG)*, 40(6):1–12, 2021. 2, 5, 6

[23] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016. 5

[24] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 8

[25] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1, 2

[26] 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 TPAMI*, 2020. 6, 8

[27] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. DeRF: Decomposed radiance fields. *arXiv preprint arXiv:2011.12490*, 2020. 2

[28] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 3

[29] Zachary Teed and Jia Deng. Deepv2d: Video to depth with differentiable structure from motion. *arXiv preprint arXiv:1812.04605*, 2018. 2

[30] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 3

[31] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 1, 2, 6, 7

[32] Chaoyang Wang, Ben Eckart, Simon Lucey, and Orazio Gallo. Neural trajectory fields for dynamic novel view synthesis. *arXiv preprint arXiv:2105.05994*, 2021. 2, 5

[33] Chaoyang Wang, Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural prior for trajectory estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6532–6542, June 2022. 2

[34] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. Web stereo video supervision for depth prediction from dynamic scenes. 2019. 8

[35] Qianqian Wang, Zhengqi Li, David Salesin, Noah Snavely, Brian Curless, and Janne Kontkanen. 3d moments from near-duplicate photos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3906–3915, June 2022. 2

[36] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[37] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. D2 nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *arXiv preprint arXiv:2205.15838*, 2022. 5

[38] 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. 5

[39] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Ce Liu, and Deva Ramanan. Viser: Video-specific surface embeddings for articulated 3d shape reconstruction. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[40] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. *arXiv preprint arXiv:2112.12761*, 2021. 1, 2

[41] 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. 5

[42] 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. 5

[43] Zhoutong Zhang, Forrester Cole, Richard Tucker, William T Freeman, and Tali Dekel. Consistent depth of moving objects in video. *ACM Transactions on Graphics (TOG)*, 40(4):1–12, 2021. 1

[44] Tiancheng Zhi, Christoph Lassner, Tony Tung, Carsten Stoll, Srinivasa G Narasimhan, and Minh Vo. Texmesh: Reconstructing detailed human texture and geometry from rgb-d video. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 492–509. Springer, 2020. 2

[45] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6365–6373, 2017. 2