This CVPR 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

STaR: Self-supervised Tracking and Reconstruction of Rigid Objects in Motion with Neural Rendering

Wentao Yuan* Zhaoyang Lv Tanner Schmidt Steven Lovegrove
University of Washington Facebook Reality Labs Research

https://wentaoyuan.github.io/star

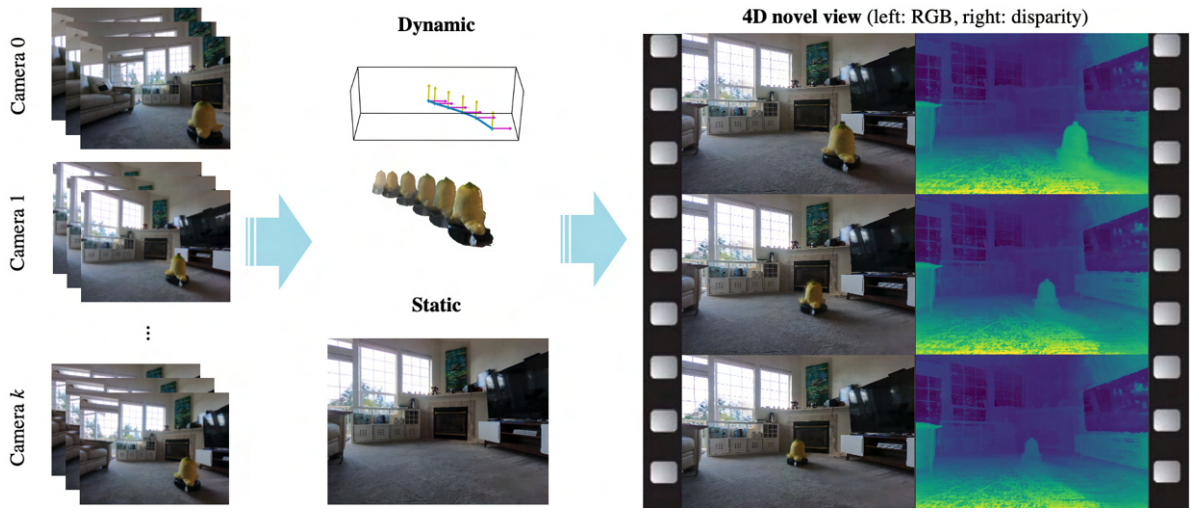

Figure 1: **An overview of our method.** Given multi-view RGB videos of a dynamic scene, STaR learns a decoupled 3D representation of the static and dynamic scene components without any human annotation, which allows it to synthesize the scene from new viewpoints at new time photorealisticly, or even animate the scene with novel trajectories.

Abstract

We present STaR, a novel method that performs Self-supervised Tracking and Reconstruction of dynamic scenes with rigid motion from multi-view RGB videos without any manual annotation. Recent work has shown that neural networks are surprisingly effective at the task of compressing many views of a scene into a learned function which maps from a viewing ray to an observed radiance value via volume rendering. Unfortunately, these methods lose all their predictive power once any object in the scene has moved. In this work, we explicitly model rigid motion of objects in the context of neural representations of radiance fields. We show that without any additional human specified supervision, we can reconstruct a dynamic scene with a single rigid object in motion by simultaneously decomposing it into its two constituent parts and encoding each with its own neural representation. We achieve this by jointly optimizing the parameters of two neural radiance fields and a set of rigid poses which align the two fields at each frame. On both synthetic and real world datasets, we demonstrate that our method can render photorealistic novel views, where novelty is measured on both spatial and temporal axes. Our factored representation furthermore enables animation of unseen object motion.

1. Introduction

Recent years have seen an explosion of novel scene representations which leverage multi-layer perceptrons (MLPs) as an implicit representation to encode spatially-varying properties of a scene. While these implicit representations are optimized via stochastic gradient descent, they are not really "learning" in the traditional sense. Instead, they exploit MLPs as a compressed representation of scene content. They can act as a substitute for traditional representations (e.g. explicit volumetric grids), but can

*Work done while the author was an intern at FRL Research.

adaptively distribute their limited capacity over the scene to enable high-fidelity scene representation. Representative examples include DeepSDF [16], Scene Representation Networks [20] and Neural Radiance Fields (NeRF) [13].

Among all of these representations, NeRF [13] and its variants [8, 24, 11] show enormous potential in their ability to photorealistically reconstruct a scene from only a sparse set of images. However, these works assume the scene is static, or at least that the dynamic content of the scene is uninteresting and can be discarded as in the work by Martin-Brualla et al. [11]. When objects in a scene move, these methods can no longer correctly render novel views. It is possible to represent time-varying scenes by dedicating one NeRF volume per frame or by extending the input to four dimensions by including time. The former is unnecessarily expensive, and neither can be used to render the object in novel poses (or remove it entirely) because they have no object-level understanding of the scene.

In this work, we aim to learn an interpretable and editable representation of a dynamic scene by simply observing an object in motion from multiple viewpoints. As an initial effort to tackle this challenge, we start from a simplified setting that assumes the scene contains only one moving object and the motion is fully rigid. We accomplish this by rendering a compositional neural radiance field with density and radiance defined via a composition of a static and a dynamic neural radiance field. Under this model, the only way all observations in a video can accurately be predicted is by segmenting the scene into the two volumes and correctly estimating the pose of the object in each frame.

To achieve this goal, our paper provides two main technical contributions. First, we present the first self-supervised neural rendering based representation that can simultaneously reconstruct a rigid moving scene as well as its background from videos only. Our approach enables photorealistic spatial-temporal novel view rendering as well as novel scene animation. Second, we present an optimization scheme that can effectively tackle the ambiguity and local optima during training.

Our experiments show that it is possible to recover the segmentation of static and dynamic contents as well as motion trajectory without any supervision other than multi-view RGB observations. Compared to NeRF and its extensions, our approach achieves more photorealistic reconstruction in complex synthetic as well as real world scenes. Further, our factorized representation can be edited to position the object in novel locations never observed during training, which no existing method can achieve without 3D ground truth or supervision.

2. Related Work

Our work is inspired and related to the recent progress in learning-based representations, and particular in novel differentiable rendering from images, e.g. NeRF [13] and its variants [11, 24]. However, no prior work has successfully demonstrated photorealistic reconstruction and understanding of dynamic scenes using only real-world natural images. We believe we are the first to achieve self-supervised tracking and reconstruction of dynamic scenes with a single rigidly-moving object using neural rendering.

Neural 3D representation Recently deep 3D representations using MLPs have demonstrated a promising ability to learn high quality geometry and appearance representations. Existing works use MLPs to represent the scene as a continuous 3D function which maps from scene coordinates to scene properties, e.g. signed distance functions (SDFs) [16], occupancy [12], occupancy flow [14], volumetric density and radiance [13], or an implicit feature [20] which can be further decoded into pixel appearance. Several approaches show high quality reconstruction results by pairing such deep 3D representations with differentiable ray marching and some manual input such as object masks [15] [22]. Sitzmann *et al*. [19] and Tancik *et al*. [21] shows that neural representation can compress videos, but has yet to demonstrate its ability to reconstruct dynamic 3D scenes.

NeRF [13] demonstrated that a continuous 5D volume rendering function can be learned from only sparse sets of images, achieving state-of-the-art novel view synthesis results without relying on any manually specified inputs. Recent work has extended this representation to be trained with web images under different lighting conditions [11], with hierarchical spatial latent priors [8], or with a latent embedding that can support generative modeling [18]. However, existing works all assume the input scene coordinate is quasi time-invariant while we assume the scene is rigidly moving. Among all of these work, only NeRF-W [11] consider the real-world that contains dynamic information. They incorporated a latent embedding in addition to the scene coordinate to model the photometric variation across web images and treat the moving scene as transient objects. This method discards these objects and only reconstructs the background scene, while we can simultaneously reconstruct dynamic objects and recover their trajectory.

Dynamic scene novel view synthesis Recently a few systems demonstrate novel view synthesis in dynamic scenes using multi-view videos, with a neural image-based rendering representation [2], neural volume rendering [9], or a multi-spherical image representation [3]. However, these approaches can only support dynamic video playback; they cannot understand scene dynamics, and therefore cannot be used for interactive animation. Yoon *et al*. [23] propose a method that blends multi-view depth estimation in background regions, learned monocular depth estimation in the foreground, and a convolutional network. They demonstrated novel view synthesis as well as animation. However, this system requires precise manual specifica-

tion of the foreground mask as part of training and rendering. This is a significant drawback, especially when it comes to video input. In contrast, we demonstrate that our approach can achieve simultaneous reconstruction of background and foreground completely self-supervised and demonstrate that it achieves high-quality reconstruction using real-world videos.

Self-supervised learning in dynamic scenes There is also a surge in self-supervised learning approaches using videos, particularly in the autonomous driving domain, including 3D object detection [1], joint camera poses and depth estimation [6, 4], with motion segmentation [17, 10] and scene flow [7]. Due to the sensor limitation on autonomous driving domain, these approaches focus on monocular videos or narrow baseline stereo videos where the change in viewpoint is small. As a result, these methods use 2D or 2.5D representations which limits their ability to render or animate the scene in a photorealistic fashion. In contrast, our method builds a implicit 3D representation that can be photorealistically rendered and animated. We demonstrate applications of our model on natural wide-baseline multi-view videos, which shows promise for applications in virtual and augmented reality.

3. Method

We introduce STaR, a differentiable rendering-based framework for self-supervised tracking and reconstruction of rigidly moving objects and scenes. Given only multi-view passive video observations of an unknown object which rigidly moves in a novel environment, STaR can simultaneously reconstruct a 3D model of the object (including both geometry and appearance) and track its 6DoF motion relative to a canonical frame without any human annotation. STaR can enable high-quality novel viewpoint rendering of both static and dynamic components of the original scene independently at any time, and can also be animated and re-rendered with a novel object trajectory.

We will first describe the 3D representation of STaR, which consists of a static and a rigidly-moving NeRF model. Next, we elaborate how to optimize STaR over a multi-view RGB video, specifically including optimization over a set of 3D transformations between two neural radiance fields, a regularized loss function, and an sequential training scheme that works with videos of arbitrary length.

3.1. STaR as Dynamic Neural Radiance Fields

Preliminaries NeRF [13] represents the scene as a continuous function $\mathcal{F}_\theta$ using a MLP with parameters θ

$$\mathcal{F}_\theta : \mathbf{r}(s), \mathbf{d} \rightarrow \sigma, \mathbf{c} \tag{1}$$

which maps a 3D scene coordinate $\mathbf{r}(s) \in \mathbb{R}^3$ and view direction $\mathbf{d} \in S^2$ to volumetric density $\sigma \in \mathbb{R}$ and color $\mathbf{c} \in \mathbb{R}^3$, where $\mathbf{r}(s) = \mathbf{o} + s\mathbf{d}$ is the point on the ray from camera origin $\mathbf{o}$ with a depth of s. The color of a pixel $\hat{C}(\mathbf{r})$ can be obtained via an integral of accumulated radiance along the ray $\mathbf{r}$. We can numerically estimate the integral using quadrature:

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i \alpha_i \mathbf{c}_i \ , \tag{2}$$

$$\text{where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j (s_{j+1} - s_j)\right) \tag{3}$$

$$\text{and } \alpha_i = 1 - \exp(-\sigma_i (s_{i+1} - s_i)). \tag{4}$$

Here, $\{s_i\}_{i=1}^N$ is a set of samples from near bound s_n to far bound s_f and $\sigma_i = \sigma(\mathbf{r}(s_i))$, $\mathbf{c}_i = \mathbf{c}(\mathbf{r}(s_i))$ are evaluations of volume density and color at sample points $\mathbf{r}(s_i)$ along the ray.

Equation (2) can only represent time-invariant scenes. Similar to methods using a MLP to model time-dependent functions [19], a straightforward extension to represent dynamic scenes using NeRF is to concatenate time $t \in \mathbb{R}$ to the input of the implicit function $\mathcal{F}_\theta$ as

$$\mathcal{F}_\theta^t : \mathbf{r}(s), \mathbf{d}, t \rightarrow \sigma, \mathbf{c} \tag{5}$$

However, this time-dependent extension neither accurately reconstructs the complex time-varying scene using real-world images nor provides dynamic scene factorization to support animation, as we will show this in Sec. 4.

Rigidly composed dynamic radiance fields Instead, we represent the scene using two time-invariant implicit volumetric models: a static NeRF $\mathcal{F}_\theta^S$ for static components and a dynamic NeRF $\mathcal{F}_\theta^D$ for dynamic components.

$$F_\theta^S : \mathbf{r}(s_i), \mathbf{d} \rightarrow \sigma_i^S, \mathbf{c}_i^S \tag{6}$$

$$F_\theta^D : \mathbf{r}(s_i), \mathbf{d}, \xi(t) \rightarrow \sigma_i^D, \mathbf{c}_i^D \tag{7}$$

Here, $\xi(t) \in \mathfrak{se}(3)$ is a set of time-dependent rigid poses that defines the transformation $T_{ow}(t) = \exp(\xi(t)) \in \mathbf{SE}(3)$ in world coordinate from time t to time 0, which aligns the dynamic volume and the static volume under a single time-invariant canonical frame.

To compute the color of a pixel at a particular time t, we compose the transformed dynamic NeRF with the static NeRF using $\xi(t)$ and alpha blending. Specifically, given a set of samples $\{s_i\}_{i=1}^N$, we evaluate the static density and color $\sigma_i^S, \mathbf{c}_i^S$ at the points $\mathbf{r}(s_i)$, as well as the dynamic density and color $\sigma_i^D, \mathbf{c}_i^D$ at the *transformed* points $T_{ow}(t)\mathbf{r}(s_i)$. We derive the compositional radiance field

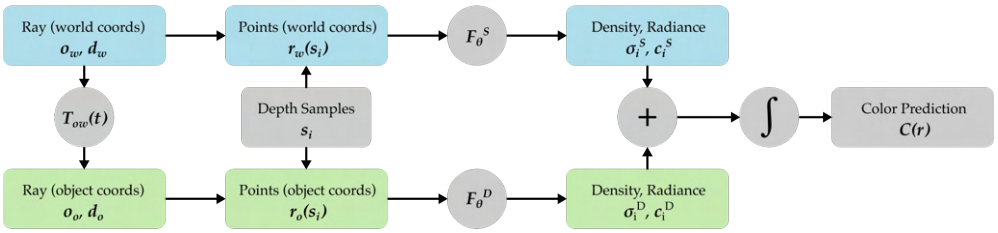

Figure 2: Illustration of the architecture of STaR. In order to cast a ray through the composed dynamic volume, we sample points on the original ray as well as the equivalent ray in the object's canonical frame of reference. Samples are passed to the corresponding neural networks, and resulting radiance and density values (now coordinate-free) are combined by addition.

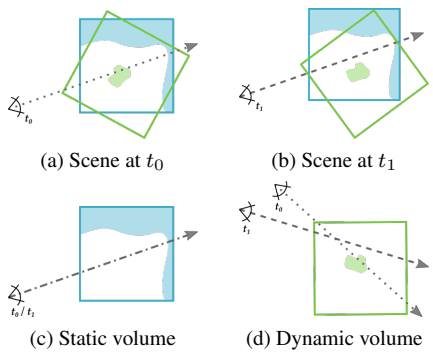

Figure 3: A visualization of the volumetric composition of the two volumes. (a) and (b) show a 2D scene observed at two different time t_0 and t_1 from a static viewpoint. To render the images observed at t_0 and t_1, we would cast the same ray through the static volume (shown in (c)), but two different rays through the dynamic volume (shown in (d)), which are transformed copies of the static ray based on the dynamic object's poses at t_0 and t_1.

based on (2) as

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i(\alpha_i^S \mathbf{c}_i^S + \alpha_i^D \mathbf{c}_i^D) \tag{8}$$

$$\text{where } T_i = \exp\left(-\sum_{j=1}^{i-1}(\sigma_j^S + \sigma_j^D)(s_{j+1} - s_j)\right) \tag{9}$$

$$\text{and } \alpha_i^S = 1 - \exp(-\sigma_i^S(s_{i+1} - s_i)). \tag{10}$$

Our rigidly composed NeRF representation decouples rigid motion from geometry and appearance, allowing it to have full control over the dynamics of the environment. As shown in Sec. 4.2, this enables applications such as removing the dynamic object and animating novel trajectories which cannot be achieved via a simple time-varying model.

Architecture We use the same MLP $\mathcal{F}_\theta$ model as [13] for both static and dynamic NeRF. We also use positional encoding with the same bandwidth for the inputs and hierarchical volume sampling strategy. We use two independent coarse and fine MLP models to represent static and dynamic NeRFs. The coarse models use stratified sampling along the ray while the fine model uses importance sampling where the importance weights are the composed density $\sigma_j^S + \sigma_j^D$ obtained by both coarse models. This ensures the same sample intervals $\{s_i\}_{i=1}^N$ for the static and dynamic NeRFs, which enables the field composition shown in (8).

While we choose NeRF as the underlying 3D representation due to its simplicity and excellent performance, it is worth noting that STaR as a differentiable renderer can generally subsume any 3D static scene representation as $\mathcal{F}_\theta$ which shares the same input and output mapping.

3.2. Optimizing STaR

During training, we optimize the following objectives

$$\mathcal{L} = \sum_{\mathbf{r}\in\mathcal{R}} \left(\|\hat{C}_c(\mathbf{r}) - C(\mathbf{r})\|^2 - \|\hat{C}_f(\mathbf{r}) - C(\mathbf{r})\|^2\right) \tag{11}$$

$$+ \beta \sum_{i=1}^{M} \left(\mathcal{H}(\alpha_i^S) + \mathcal{H}(\alpha_i^D) + \mathcal{H}(\alpha_i^S, \alpha_i^D)\right) \tag{12}$$

The first term in (11) is a MSE loss, where $\hat{C}_c(\mathbf{r})$ and $\hat{C}_f(\mathbf{r})$ are the pixel color (RGB) rendered by the coarse and fine model respectively, $C(\mathbf{r})$ is the ground truth color and $\mathcal{R}$ is the set of rays in a batch. Note that $C(\mathbf{r})$ is the only source of supervision. Given a limited number of camera views, minimizing the objective in (11) can suffer from local optima. We introduce the regularization term in (12) which we observe can converge better.

Entropy regularization (12) regularizes the entropy $\mathcal{H}$ of the rendered transparency values and is summed over all samples along all rays in a batch. It consists of two parts. The first part $\mathcal{H}(\alpha_i^S) + \mathcal{H}(\alpha_i^D)$ encourages the transparencies to be close to 0 or 1, where

$$\mathcal{H}(\alpha_i^S) = \alpha_i^S \log \alpha_i^S + (1 - \alpha_i^S)\log(1 - \alpha_i^S) \tag{13}$$

and $\mathcal{H}(\alpha_i^D)$ is computed likewise. This helps reducing fuzziness in the volume. The second part $\mathcal{H}(\alpha_i^S, \alpha_i^D)$ prevents the static and dynamic volume from both having large density at the same point, which helps the model to obtain a less entangled decomposition. Specifically,

$$\mathcal{H}(\alpha_i^S, \alpha_i^D) = \left(\overline{\alpha}_i^S \log \overline{\alpha}_i^S + \overline{\alpha}_i^D \log \overline{\alpha}_i^D\right)(\alpha_i^S + \alpha_i^D) \tag{14}$$

where $\overline{\alpha}_i^S = \alpha_i^S / (\alpha_i^S + \alpha_i^D)$ is the normalized transparency. The term is weighted by the total transparency $\alpha_i^S + \alpha_i^D$, so that a point is allowed to be empty but not occupied in both static and dynamic volume.

Rigid pose optimization We optimize a set of pose parameters $\xi(t) \in \mathfrak{se}(3)$ using on manifold optimization [5] over the Lie algebra of $\mathrm{SE}(3)$. During optimization, we represent the transformation from dynamic to static volume at time t as an iterative update $\exp(\epsilon + \xi(t))$, where $\epsilon \in \mathfrak{se}(3)$ is a local perturbation on the manifold that is initialized to zero at the beginning of each iteration. We can compute the gradient of ϵ using the gradient of the transformed point $\exp(\epsilon + \xi(t))\mathbf{p}$ and an analytical Jacobian that can be computed from $\xi(t)$ in the forward pass (see Sec. F of the supplement for the full form). Then, we update $\xi(t)$ as $\xi(t) = a_\xi \nabla \epsilon + \xi(t)$. Intuitively, at each iteration, we project the $\mathrm{SE}(3)$ manifold to $\mathbb{R}^6$ locally around the current pose estimate $\xi(t)$ and take a gradient step in this projected 6D coordinates. Note that we use different learning rate a_ξ for the pose parameters and a_θ for the NeRF parameters.

Appearance initialization We observe there are many local optima in the optimization of STaR due to entanglement of geometry, appearance and pose, which requires careful initialization. To initialize, we first train a static NeRF $\mathcal{F}_\theta^S$ following (2) only using the images from the first frame. The initialization is terminated when the average MSE of the fine model $\frac{1}{|\mathcal{R}_1|}\sum_{r\in\mathcal{R}_1} \|\hat{C}_f(\mathbf{r}) - C(\mathbf{r})\|$ over all images from the first frame $\mathcal{R}_1$ is below the threshold m_1. Note that at this stage, the static volume can (and likely does) contain information from the dynamic object, but we observe that this initialization provides the model with a good initial estimate of the geometry and appearance from which to begin disentangling the scene.

Sequential training We train STaR in a sequential fashion and can handle video sequences of arbitrary length. After appearance initialization, the static NeRF $\mathcal{F}_\theta^S$, dynamic NeRF $\mathcal{F}_\theta^D$ and pose parameters $\{\xi_t\}_{t=1}^k$ are optimized jointly on the first k frames of the video. k is set to k_0 initially and incremented when the average MSE of the fine model $\frac{1}{|\mathcal{R}_k|}\sum_{r\in\mathcal{R}_k} \|\hat{C}_f(\mathbf{r}) - C(\mathbf{r})\|$ over all rays in the first k frames $\mathcal{R}_k$ is below the threshold m_2. For $t \leq k_0$, $\xi(t)$ is initialized to identity and for $t > k_0$, $\xi(k)$ is set to the pose of the previous frame $\xi(k-1)$ when frame k is added. In Sec. 4, we show that both appearance initialization and sequential training is crucial for the optimization of STaR.

4. Experiments

Our experiments demonstrate that STaR is able to decouple static and dynamic components of a dynamic scene in challenging synthetic as well as real-world scenarios (see Fig. 5). As a result, STaR is not only capable of synthesizing a dynamic scene from novel views and time with superior quality (see Sec. 4.1) but also animating an imaginary trajectory in a photorealistic fashion (see Sec. 4.2).

Sequence	B	β	a_θ	γ	a_ξ	m_1	m_2	k_0	GPU	Time
Lamp and desk	3200	2e-3	5e-4	0.5	5e-5	4e-4	2e-4	5	2	12hr
Kitchen table	3200	2e-3	5e-4	0.5	5e-5	5e-4	5e-4	5	2	24hr
Moving banana	10816	2e-3	5e-4	0.8	5e-4	2e-3	1e-3	5	8	55hr

Table 1: Hyperparameters and training time.

We implement STaR in PyTorch. We summarize the set of hyperparameters and training time/GPU used in Tab. 1, which includes the batch size B, the entropy regularization weight β, learning rates a_θ, a_ξ for NeRF and pose parameters, MSE thresholds m_1, m_2 for appearance initialization and sequential training, and number of bootstrapping frames k_0. We optimize the objective using ADAM. Following [13], we exponentially decay a_θ every 250K steps with decay rate γ, but we do not decay a_ξ.

Datasets We created two synthetic and one real world multi-view RGB videos to evaluate STaR. The videos are highly challenging, containing complex geometry, large object motion, significant view-dependent and time-dependent visual effects such as specular highlight and shadows, all of which are not present in any existing public dataset. Please refer to supplementary materials for more details.

- **Synthetic data:** We rendered two synthetic videos using Blender: *lamp and desk*, a 15-frame video showing a chair pulled away from the desk in a study room, and *kitchen table*, a 20-frame video showing a vase sliding across a reflective breakfast bar in a kitchen. The motion is created by modifying the pose of an object in a photorealistic scene. We leave the material and lighting produced by human designers untouched. Both videos have 8 fixed camera views for training and 1 held-out view for evaluation with an image resolution of 400×400.
- **Real world data:** We captured a multi-view video, *moving banana*, in real natural indoor environment. In the scene, a moving banana toy is put on a robot vacuum which sweeps across a living room. The data consists of 17 time-synchronized videos with a total of 792 temporal frames. To demonstrate our algorithm on relatively large motion, we uniformly subsample the video to 38 key frames and use 16 views for training and 1 held-out view for evaluation. The image resolution is 676×507.

4.1. 4D Novel View Synthesis Evaluation

We first evaluate our approach on its ability to photorealisticly render the dynamic scene from a novel 4D view (3D camera pose + time). More specifically, we render a 4x slow motion of the original video from a novel camera view by linearly interpolating the estimated object poses. We compare our method to the following baselines (see supplementary for the detailed architecture):

	Sequence	Lamp and desk (synthetic)			Kitchen table (synthetic)			Moving banana (real)		
	Metric	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Composition	NeRF [13]	22.09	0.873	0.182	20.39	0.657	0.374	24.48	0.777	0.261
	NeRF-time	25.86	0.873	0.101	21.95	0.686	0.319	24.82	0.776	0.259
	NeRF-W [11]	27.68	0.931	0.048	27.99	0.749	0.220	**27.26**	0.791	0.257
	STaR (ours)	**32.95**	**0.957**	**0.023**	**29.51**	**0.767**	**0.195**	27.19	**0.803**	**0.209**
	Sequence	Lamp and desk (synthetic)			Kitchen table (synthetic)			Moving banana (real)		
	Metric	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Static	NeRF [13]	25.82	0.921	0.092	21.02	0.667	0.353	25.54	0.793	0.235
	NeRF-time	30.00	0.925	0.064	22.56	0.695	0.309	25.51	0.790	0.244
	NeRF-W [11]	34.12	0.971	0.025	29.25	0.760	0.203	**27.68**	0.801	0.243
	STaR (ours)	**34.57**	**0.967**	**0.018**	**29.75**	**0.771**	**0.193**	27.44	**0.812**	**0.198**
	Sequence	Lamp and desk (synthetic)			Kitchen table (synthetic)			Moving banana (real)		
	Metric	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Dynamic	NeRF [13]	14.33	0.498	0.464	10.83	0.339	0.319	16.87	0.336	0.567
	NeRF-time	18.48	0.555	0.212	13.80	0.469	0.243	20.68	0.477	0.341
	NeRF-W [11]	19.04	0.597	0.142	15.54	0.490	0.230	24.71	**0.687**	0.298
	STaR (ours)	**28.11**	**0.903**	**0.027**	**23.97**	**0.837**	**0.036**	**25.14**	**0.687**	**0.228**

Table 2: **Quantitative comparison of our method to baselines in photorealistic interpolation of motion.** The models are evaluated on a 4x slow motion of the original video from a fixed novel viewpoint. The synthetic data are followed with a notation (synthetic), and the real world data is notated as (real). We use a ground truth bounding box as a reference to evaluate static and dynamic regions separately.

- **NeRF [13]:** The original NeRF assumes the scene is static. This baseline cannot properly reconstruct the dynamic object, but can still provide a reference to the quality in terms of the static part of the scenes.
- **NeRF-time:** This model takes positional-encoded time as an additional input, thus creating a 4D representation using a strategy similar to [19, 21].
- **NeRF-W [11]:** This model takes an additional latent code input as proposed by Martin-Brualla *et al*. [11]. We did not include the appearance embedding as did NeRF-W in [11] because our video data does not contain strong appearance variations. This is the most related existing work that can handle dynamic variation based on NeRF.

We evaluate the novel view synthesis quality in terms of photorealism following the standard metrics in [13]: the average PSNR, SSIM and LPIPS [25] across all test frames. Since our baselines do not provide separate static and dynamic rendering, we use object bounding boxes to divide the image into static background and dynamic foreground (see Fig. 4). We compute metrics on the original image, the static background and the dynamic foreground.

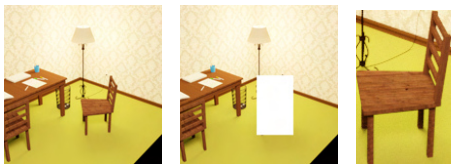

Figure 4: A visual illustration of our evaluation protocol (from left to right: Composition, Static, Dynamic).

	PSNR ↑	SSIM ↑	LPIPS ↓
No initialization*	20.78	0.820	0.188
No sequential	25.98	0.925	0.079
No entropy	32.88	**0.959**	0.032
Full model	**32.95**	0.957	**0.023**

Table 3: **Ablation study of optimization strategy discussed in Sec. 3.2 on the *lamp and desk* sequence**. No initialization removes appearance initialization. No sequential trains over entire video instead of sequentially. No entropy uses the MSE loss without entropy regularization. *We use a higher MSE threshold $m_2 = 0.001$ due to the model's difficulty in convergence.

Results Tab. 2 shows the comparison to all baselines on unseen novel views. Overall, our method provides significantly better reconstruction quality over the all baselines, in all factorized regions using both synthetic and real world data. In real-world data, NeRF-time fails to reconstruct dynamic regions well and cannot provide reconstructions as good as NeRF in static regions. In contrast, our method can handle both regions well. NeRF-W can provide competitive reconstruction quality in pixel accuracy, but we still outperform it significantly in perceptual metric LPIPS. This demonstrates the benefits of using a factorized representation. Fig. 5 shows the visual comparisons in synthetic and real world data respectively. Compared to all baselines, our method is the only the method that can reconstruct both static background and dynamic objects in details from novel views. We encourage the readers to watch the supplementary video which best demonstrates the perceptual quality of the rendering on novel spatial-temporal views.

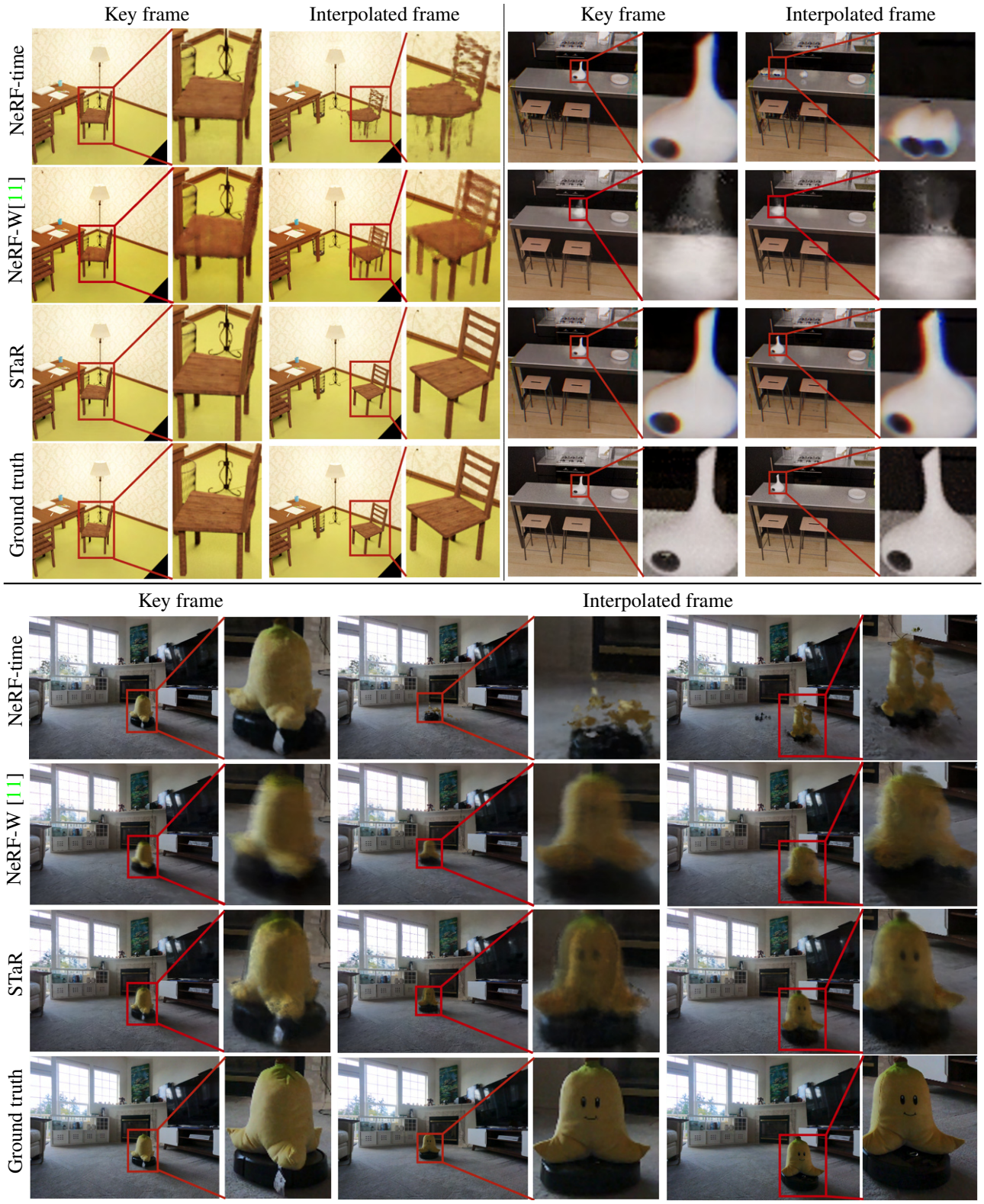

Figure 5: **Qualitative comparison of 4D novel view synthesis on both synthetic and real world data.** We compare synthesized novel views of the dynamic scene on key frames which share the same time stamps as training images, and on interpolated frames from time stamps that are not included in training (i.e. spatial-temporal novel views).

Figure 6: **Synthesizing novel trajectories**. We visualize synthesized novel views of the dynamic object moving along an imaginary trajectory unseen during training (rightmost column: training trajectory in blue and novel trajectory in red)

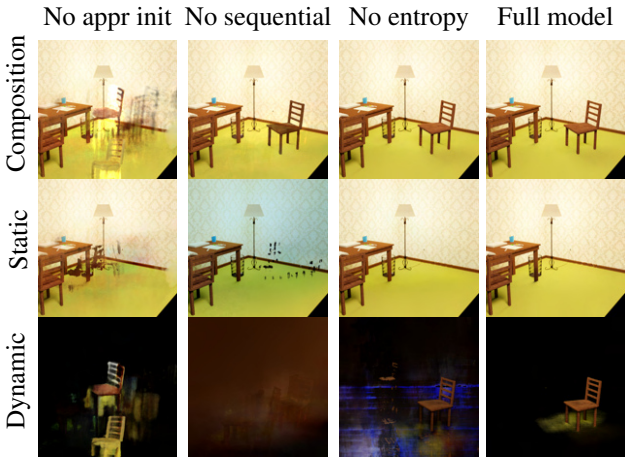

Figure 7: A visual comparison of the decomposition learned by our full model compared to ablated models.

Ablation We perform an ablation study on our optimization strategies discussed in Sec. 3.2 using the *lamp and desk* sequence. See quantitative results in Tab. 3 and qualitative results in Fig. 7. The results indicate our full model with regularization and sequential training performs the best. Quantitatively, our model exhibits dramatically worse performance without the appearance initialization stage or the sequential training strategy. With additional entropy regularization over the volume density, we can further improve the results. Fig. 7 highlights the significant perceptual difference which may not be properly reflected in the quantitative numbers.

4.2. Rendering on Animated Trajectories

Our factorized representation of motion and appearance allows STaR to synthesize novel views of animated trajectories of the dynamic object which have not been seen during training. Fig. 6 shows synthesized novel views of a trajectory dramatically different from the training trajectory in both synthetic and real world scenes. It is worth noting that no existing method we know of is able to synthesize motion so different from observed data and re-render it in a photorealistic fashion without any 3D ground truth or supervision, including NeRF-time and NeRF-W, which can only interpolate object poses from the observed trajectory.

5. Conclusion

Our method demonstrates a novel direction towards reconstructing dynamic scenes using only video observations. It should be noted here that this system is a proof of concept and has not completely solved the problem of fully decomposing dynamic scenes into their constituent parts. First, we assume only one object in motion. Extending the model to multiple objects is trivial but estimating the number of moving objects when it is not known a priori is an interesting direction for further research. Second, we cannot represent non-rigid motion in the presented model. This could probably be accomplished by combining our insights with the orthogonal work on deforming neural representations by Niemeyer et al. [14]. Finally, we effectively remove geometric dynamism from both NeRF volumes by factorizing all motion into an explicit rigid transform, but we cannot do the same for appearance due to the mutual influence of each volume on the lighting conditions of the other. This could be solved by further factoring the appearance into material and lighting conditions, but these explorations are very much outside the scope of this paper and we leave it for future work.

References

[1] Deniz Beker, Hiroharu Kato, Mihai Adrian Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. Monocular differentiable rendering for self-supervised 3d object detection. In *European Conference on Computer Vision*, 2020.

[2] Mojtaba Bemana, Karol Myszkowski, Hans-Peter Seidel, and Tobias Ritschel. X-fields: Implicit neural view-, light- and time-image interpolation. *ACM Transactions on Graphics (TOG)*, 39(6), 2020.

[3] Michael Broxton, John Flynn, Ryan Overbeck, Daniel Erickson, Peter Hedman, Matthew DuVall, Jason Dourgarian, Jay Busch, Matt Whalen, and Paul Debevec. Immersive light field video with a layered mesh representation. *ACM Transactions on Graphics (TOG)*, 39(4):86:1–86:15, 2020.

[4] Yuhua Chen, Cordelia Schmid, and Cristian Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *IEEE/CVF International Conference on Computer Vision*, pages 7063–7072, 2019.

[5] Daniel Gabay. Minimizing a differentiable function over a differential manifold. *Journal of Optimization Theory and Applications*, 37(2):177–219, 1982.

[6] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *IEEE/CVF International Conference on Computer Vision*, pages 3828–3838, 2019.

[7] Junhwa Hur and Stefan Roth. Self-supervised monocular scene flow estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7396–7405, 2020.

[8] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33, 2020.

[9] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[10] Chenxu Luo, Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, Ram Nevatia, and Alan Yuille. Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2624–2641, 2019.

[11] 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. *arXiv preprint arXiv:2008.02268*, 2020.

[12] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[13] 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. pages 405–421, 2020.

[14] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *IEEE/CVF International Conference on Computer Vision*, pages 5379–5389, 2019.

[15] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020.

[16] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019.

[17] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12240–12249, 2019.

[18] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33, 2020.

[19] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[20] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, pages 1121–1132, 2019.

[21] 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. *Advances in Neural Information Processing Systems*, 2020.

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

[23] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5336–5345, 2020.

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

[25] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.