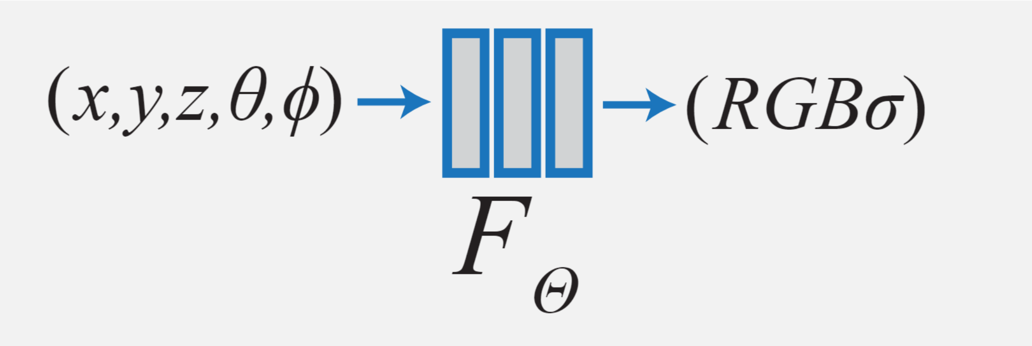
Dynamic Nerf

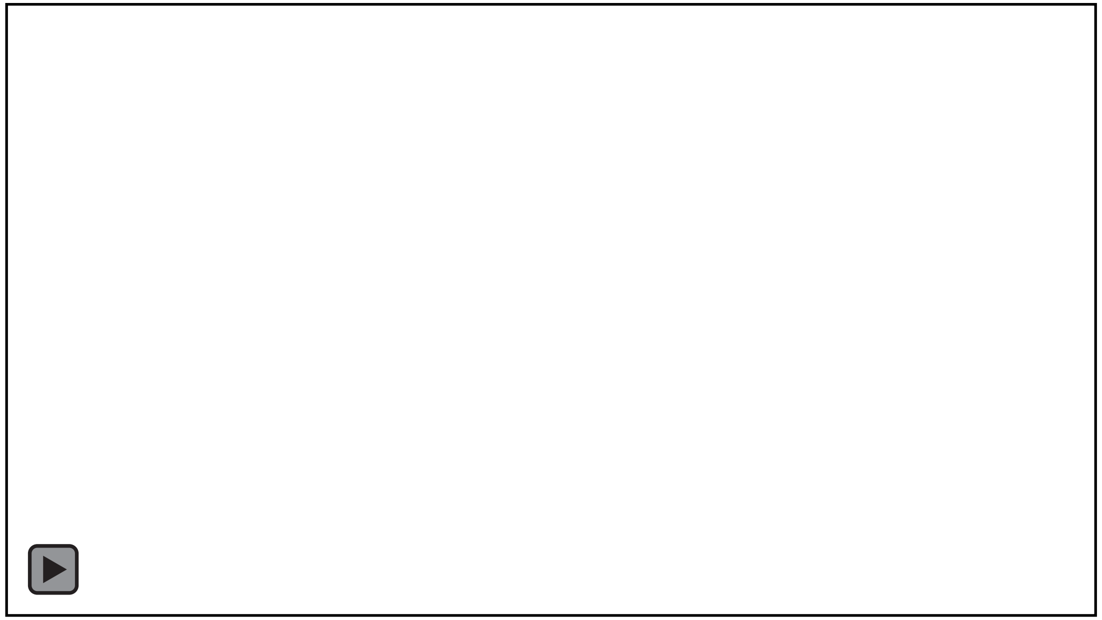
Xinxin Zuo

05/31/2022

Static Nerf

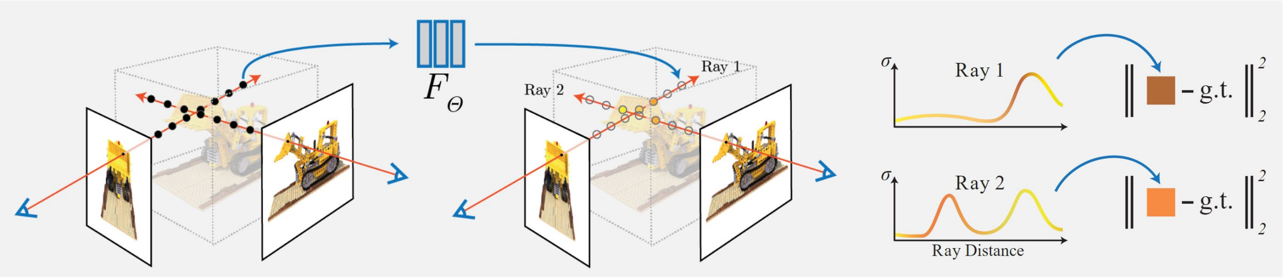
- Neural Radiance Field





$$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right) \qquad \mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

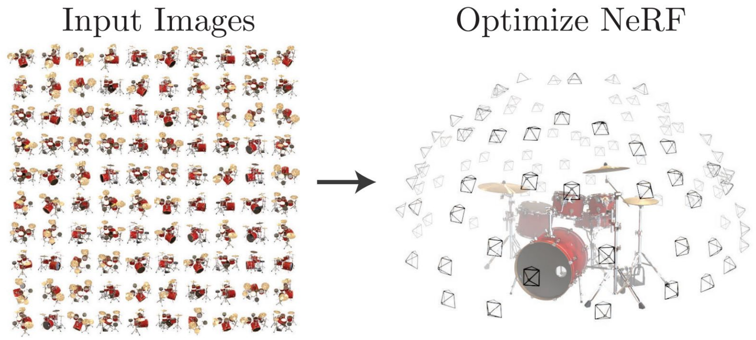
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$



Space-time/Dynamic Nerf

Dynamic Nerf

$$(x,y,z,\theta,\phi) \rightarrow F_\Theta \rightarrow (RGB\sigma)$$



$$(x,y,z,\theta,\phi) \rightarrow F_\Theta \rightarrow (RGB\sigma)$$

t

An under-constrained problem

- only monocular video
- Camera?
- Limited view directions
- Dynamic and diverse motion

Concurrent works

- D-NeRF: Neural Radiance Fields for Dynamic Scenes. CVPR 2021.
- Space-time Neural Irradiance Fields for Free-Viewpoint Video. CVPR 2021.
- Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. CVPR 2021.

D-NeRF: Neural Radiance Fields for Dynamic Scenes

Albert Pumarola[1] Enric Corona[1] Gerard Pons-Moll[2,3] Francesc Moreno-Noguer[1]
[1]Institut de Robòtica i Informàtica Industrial, CSIC-UPC
[2]University of Tübingen
[3]Max Planck Institute for Informatics

Approach

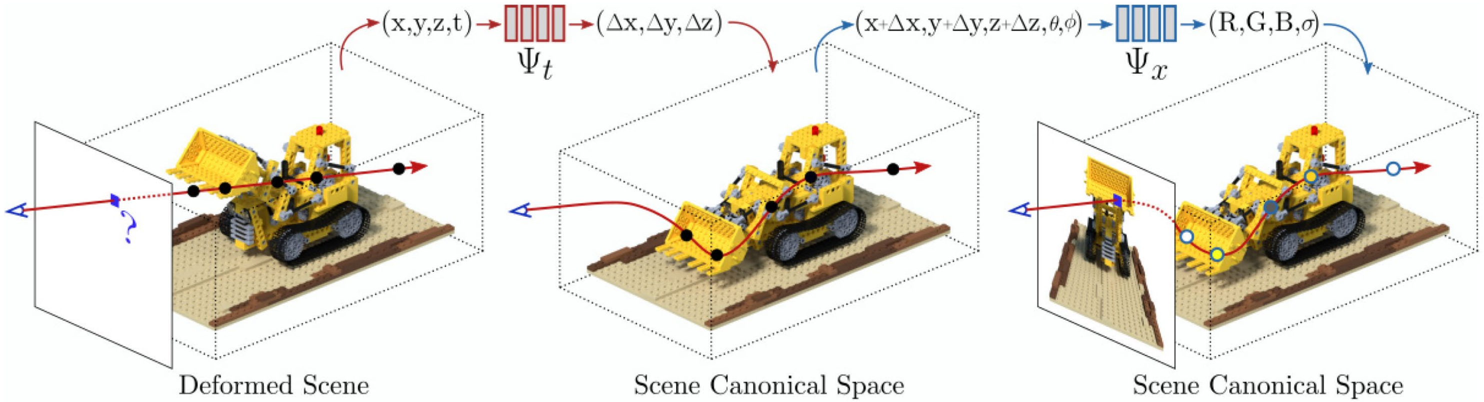


Figure 3: **D-NeRF Model**. The proposed architecture consists of two main blocks: a deformation network Ψ_t mapping all scene deformations to a common canonical configuration; and a canonical network Ψ_x regressing volume density and view-dependent RGB color from every camera ray.

$$C(p,t) = \int_{h_n}^{h_f} \mathcal{T}(h,t)\sigma(\mathbf{p}(h,t))\mathbf{c}(\mathbf{p}(h,t),\mathbf{d})dh,$$

$$\text{where} \quad \mathbf{p}(h,t) = \mathbf{x}(h) + \Psi_t(\mathbf{x}(h),t),$$

$$[\mathbf{c}(\mathbf{p}(h,t),\mathbf{d}),\sigma(\mathbf{p}(h,t))] = \Psi_x(\mathbf{p}(h,t),\mathbf{d}),$$

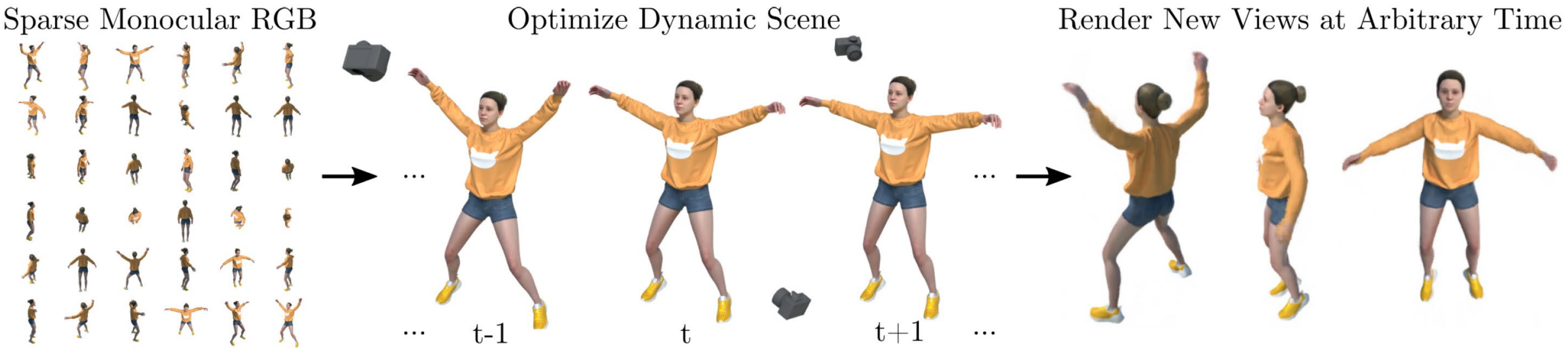
$$\text{and} \quad \mathcal{T}(h,t) = \exp\left(-\int_{h_n}^{h} \sigma(\mathbf{p}(s,t))ds\right).$$

$$\boxed{\mathcal{L} = \frac{1}{N_s}\sum_{i=1}^{N_s}\left\|\hat{C}(p,t) - C'(p,t)\right\|_2^2}$$

Experiments

Issues

- GT camera
- 100-200 frames, quite small motion
- Not applied into real scenes
- Difficult to train/converge



Concurrent works

- D-NeRF: Neural Radiance Fields for Dynamic Scenes. CVPR 2021.
- Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. CVPR 2021.
- Space-time Neural Irradiance Fields for Free-Viewpoint Video. CVPR 2021.

Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes

Zhengqi Li[1] Simon Niklaus[2] Noah Snavely[1] Oliver Wang[2]

[1] Cornell Tech [2] Adobe Research

Figure 1: Our method can synthesize novel views in both space and time from a single monocular video of a dynamic scene. Here we show **video** results with various configurations of fixing and interpolating view and time (left), as well as a visualization of the recovered scene geometry (right). Please view with Adobe Acrobat or KDE Okular to see animations.

Motivations

- To deal with real world videos
  - ➢ Camera? → standard SFM pipeline, COLMAP
  - ➢ Arbitrary motion and scene → Incorporate several regularization terms & data-driven loss

Models and Losses

$$(\mathbf{c}_i, \sigma_i, \mathcal{F}_i, \mathcal{W}_i) = F_\Theta^{\mathrm{dy}}(\mathbf{x}, \mathbf{d}, i).$$

scene flow $\mathcal{F}_i = (\mathbf{f}_{i\rightarrow i+1}, \mathbf{f}_{i\rightarrow i-1})$

disocclusion weights $\mathcal{W}_i = (w_{i\rightarrow i+1}, w_{i\rightarrow i-1})$

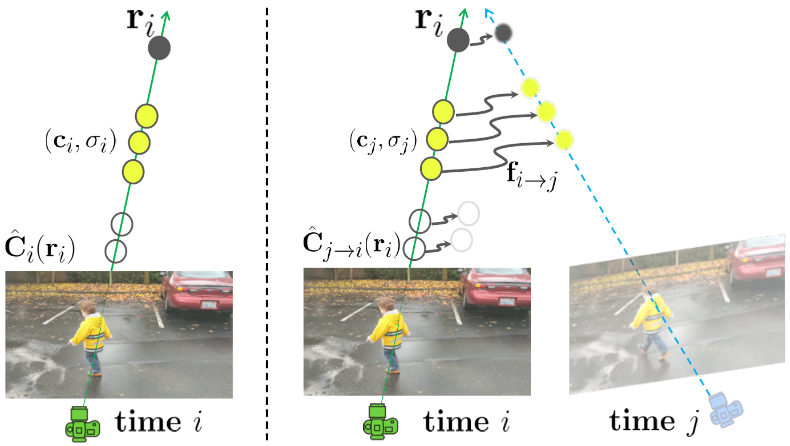


Figure 2: **Scene flow fields warping.** To render a frame at time i, we perform volume rendering along ray $\mathbf{r}_i$ with RGBσ at time i, giving us the pixel color $\hat{\mathbf{C}}_i(\mathbf{r}_i)$ (left). To warp the scene from time j to i, we offset each step along $\mathbf{r}_i$ using scene flow $\mathbf{f}_{i\rightarrow j}$ and volume render with the associated color and opacity $(\mathbf{c}_j, \sigma_j)$ (right).

Losses

- **Temporal photometric consistency.**

$$\hat{\mathbf{C}}_{j\rightarrow i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\,\sigma_j(\mathbf{r}_{i\rightarrow j}(t))\,\mathbf{c}_j(\mathbf{r}_{i\rightarrow j}(t), \mathbf{d}_i)dt$$

$$\text{where } \mathbf{r}_{i\rightarrow j}(t) = \mathbf{r}_i(t) + \mathbf{f}_{i\rightarrow j}(\mathbf{r}_i(t)). \qquad (5)$$

$$\hat{W}_{j\rightarrow i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\,\sigma_j(\mathbf{r}_{i\rightarrow j}(t))\,w_{i\rightarrow j}(\mathbf{r}_i(t))\;dt \qquad (7)$$

$$\boxed{\begin{aligned}\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i}\sum_{j\in\mathcal{N}(i)} \hat{W}_{j\rightarrow i}(\mathbf{r}_i)||\hat{\mathbf{C}}_{j\rightarrow i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2 \\ + \beta_w \sum_{\mathbf{x}_i} ||w_{i\rightarrow j}(\mathbf{x}_i) - 1||_1, \qquad (8)\end{aligned}}$$

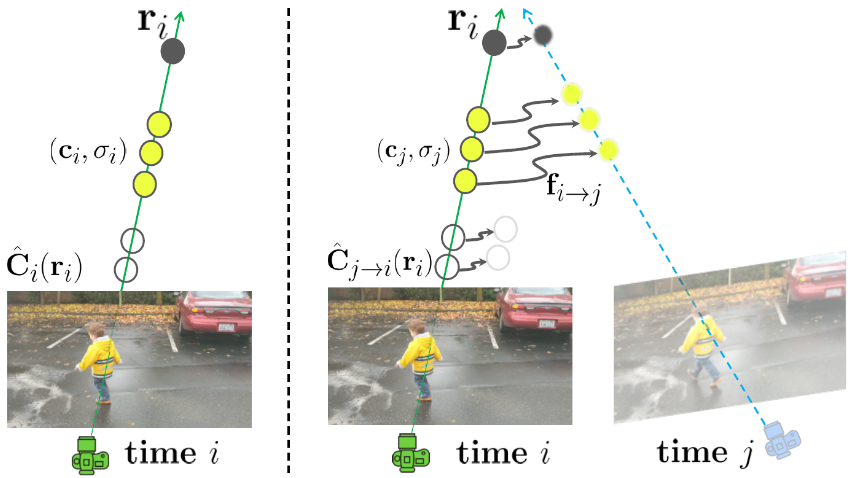


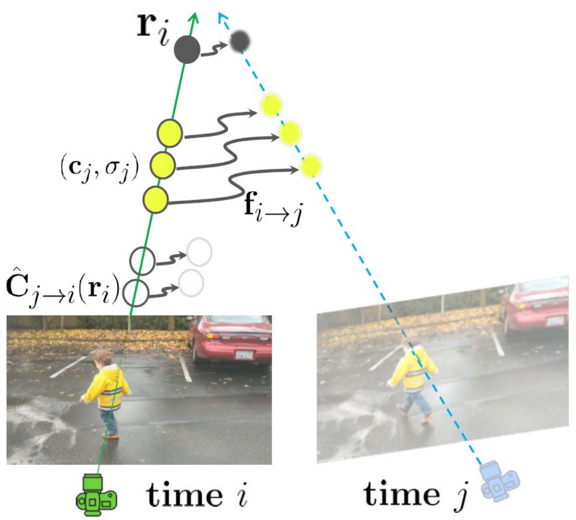
Figure 2: **Scene flow fields warping.** To render a frame at time i, we perform volume rendering along ray $\mathbf{r}_i$ with RGBσ at time i, giving us the pixel color $\hat{\mathbf{C}}_i(\mathbf{r}_i)$ (left). To warp the scene from time j to i, we offset each step along $\mathbf{r}_i$ using scene flow $\mathbf{f}_{i\rightarrow j}$ and volume render with the associated color and opacity $(\mathbf{c}_j, \sigma_j)$ (right).

Losses

However, as both of these data-driven priors are noisy (rely on inaccurate or incorrect predictions), we use these for initialization only, and linearly decay the weight to zero during training.

- **Data-driven priors.**
  - optical flow

$$\mathcal{L}_{\text{geo}} = \sum_{\mathbf{r}_i} \sum_{j \in \{i \pm 1\}} ||\hat{\mathbf{p}}_{i \to j}(\mathbf{r}_i) - \mathbf{p}_{i \to j}(\mathbf{r}_i))||_1 .$$



  - depth

$$\mathcal{L}_z = \sum_{\mathbf{r}_i} ||\hat{Z}_i^*(\mathbf{r}_i) - Z_i^*(\mathbf{r}_i)||_1$$

$$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right) \qquad \mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$$

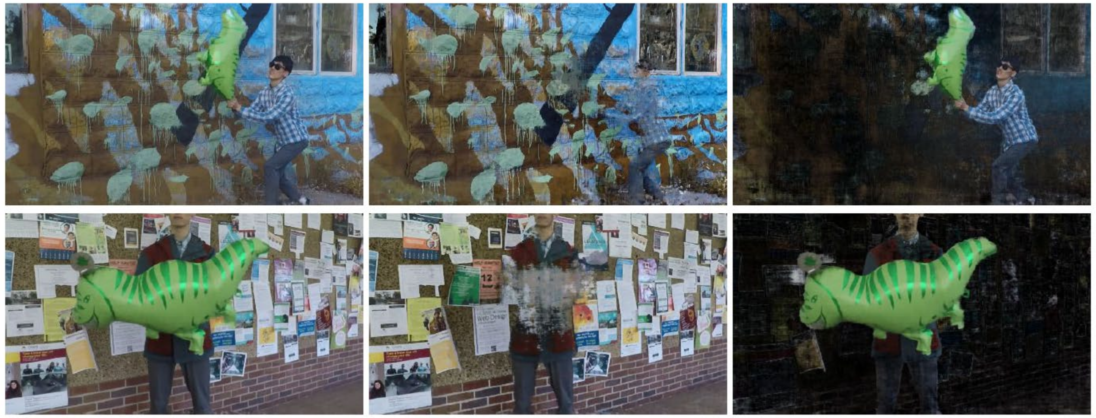
$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt$$

Combine static and dynamic

$$(\mathbf{c}_i, \sigma_i, \mathcal{F}_i, \mathcal{W}_i) = F_{\Theta}^{\text{dy}}(\mathbf{x}, \mathbf{d}, i).$$

$$(\mathbf{c}, \sigma, v) = F_{\Theta}^{\text{st}}(\mathbf{x}, \mathbf{d})$$

$$\sigma_i^{\text{cb}}(t)\,\mathbf{c}_i^{\text{cb}}(t) = v(t)\,\mathbf{c}(t)\,\sigma(t) + (1\text{-}v(t))\,\mathbf{c}_i(t)\,\sigma_i(t)$$



Combined render | Static only | Dynamic only

Figure 5: **Dynamic and static components.** Our method learns static and dynamic components in the combined representation. Note person is almost still in the bottom example.

Combine static and dynamic

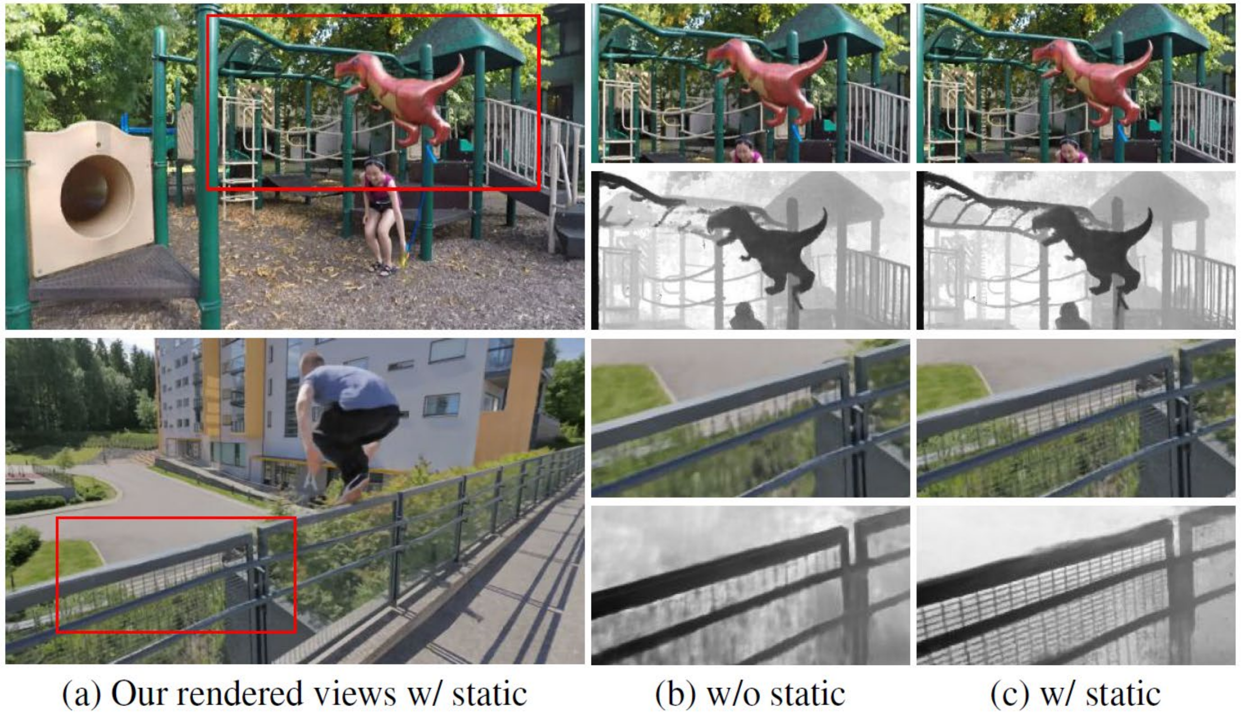
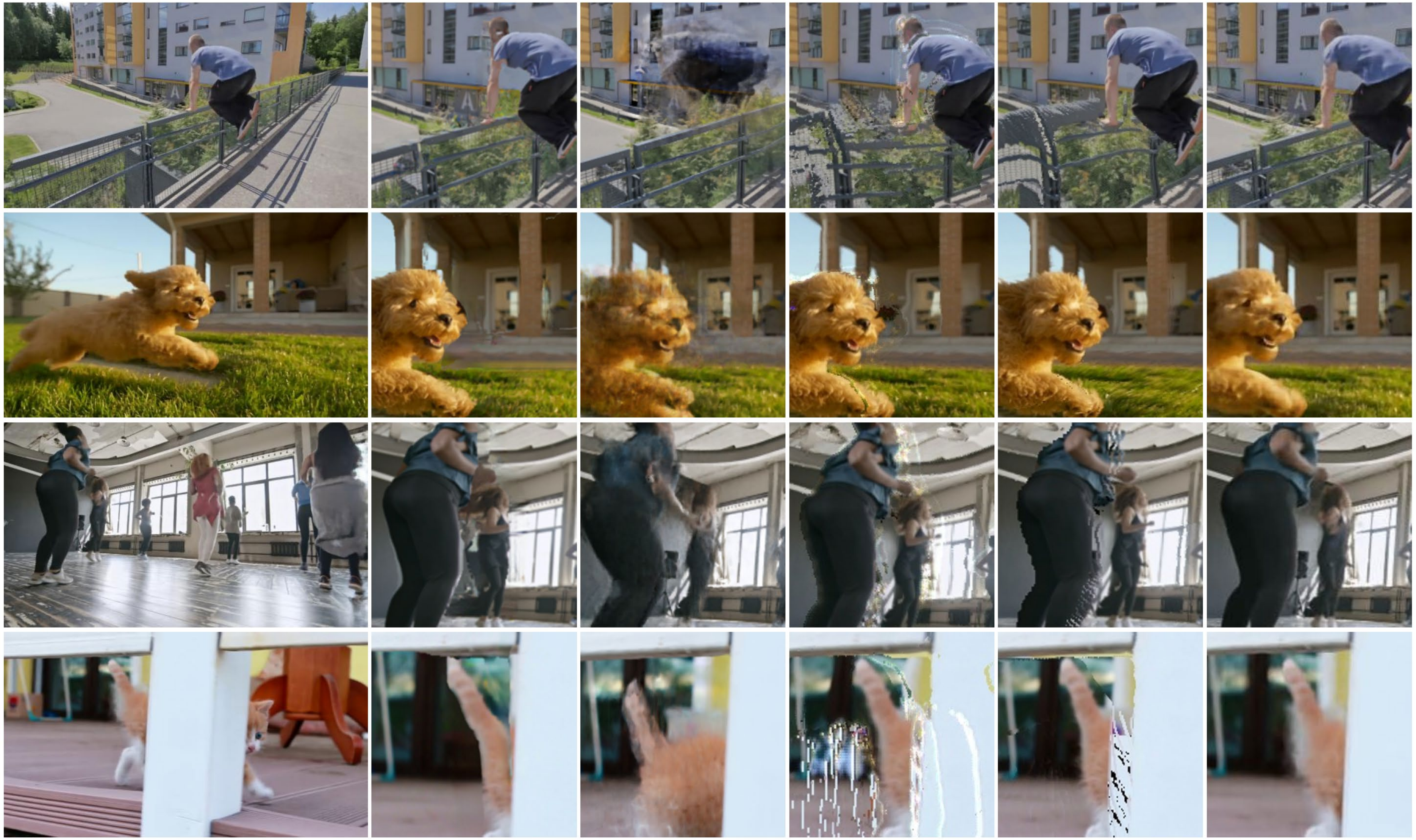


Figure 6: **Static scene representation ablation.** Adding a static scene representation yields higher fidelity renderings, especially in static regions (a,c) when compared to the pure dynamic model (b).

Experiments

Figure 8: **Qualitative comparisons on the Dynamic Scenes dataset.** Compared with prior methods, our rendered images more closely match the ground truth, and include fewer artifacts, as shown in the highlighted regions.

Experiments



Our rendered views 3D Photo [64] NeRF [46] Yoon *et al*. [81] Luo *et al*. [40] Ours

Concurrent works

- D-NeRF: Neural Radiance Fields for Dynamic Scenes. CVPR 2021.
- Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. CVPR 2021.
- Space-time Neural Irradiance Fields for Free-Viewpoint Video. CVPR 2021.

Space-time Neural Irradiance Fields for Free-Viewpoint Video

Wenqi Xian* Cornell Tech
Jia-Bin Huang Virginia Tech
Johannes Kopf Facebook
Changil Kim Facebook

https://video-nerf.github.io

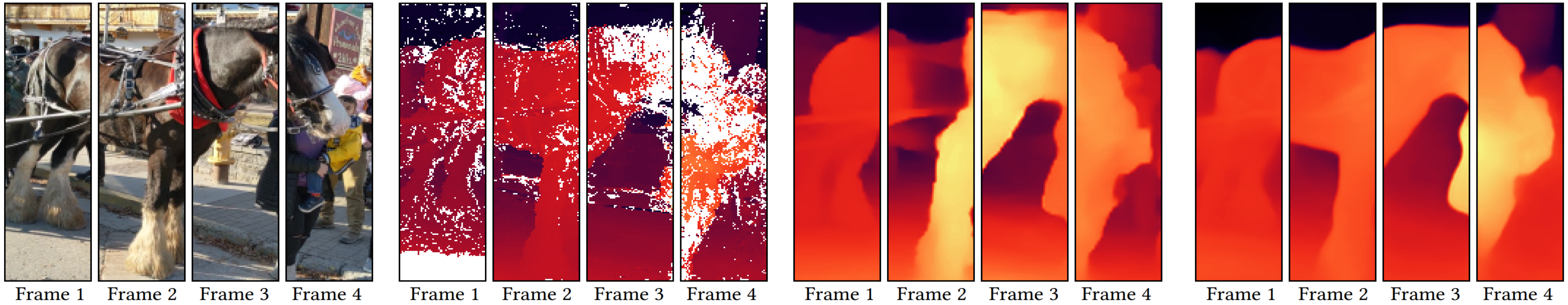
Figure 1. Our method takes a *single* casually captured video as input and learns a space-time neural irradiance field. (*Top*) Sample frames from the input video. (*Middle*) Novel view images rendered from textured meshes constructed from depth maps. (*Bottom*) Our results rendered from the proposed space-time neural irradiance field.

Motivations

$$F : (\mathbf{x}, t) \rightarrow (\mathbf{c}, \sigma).$$

- Explicitly model the scene flow is difficult
- a stream of RGB-D images

Consistent Video Depth Estimation



(a) Input video (b) COLMAP depth (c) Mannequin Challenge depth (d) Our result

Losses

$$\mathcal{L}_{\text{color}} = \sum_{(\mathbf{r},t)\in\mathcal{R}} \left\| \hat{C}(\mathbf{r},t) - C(\mathbf{r},t) \right\|_2^2 ,$$

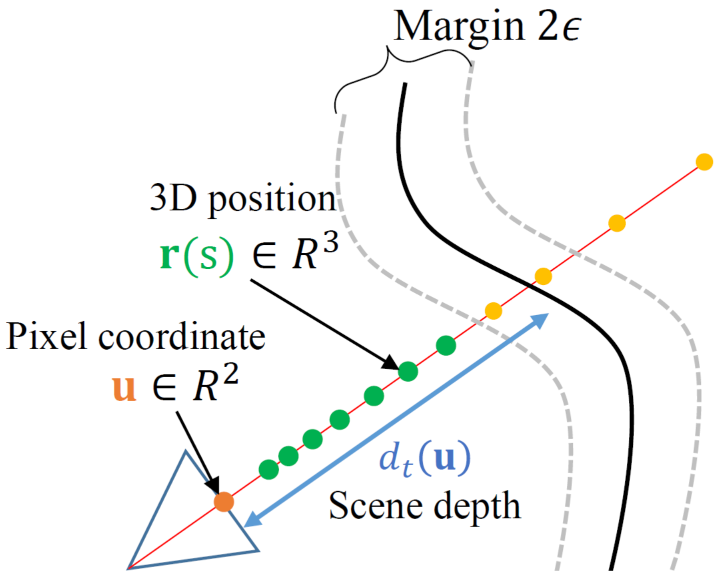
Losses

$$\mathcal{L}_{\text{depth}} = \sum_{(\mathbf{r},t)\in\mathcal{R}} \left\| \frac{1}{\hat{D}(\mathbf{r},t)} - \frac{1}{D(\mathbf{r},t)} \right\|_2^2,$$

$$\hat{D}(\mathbf{r},t) = \int_{s_{\text{n}}}^{s_{\text{f}}} T(s,t)\,\sigma(\mathbf{r}(s),t)\,s\,\mathrm{d}s,$$

(a) w/o depth loss (b) w/ depth loss

$$\mathcal{L}_{\text{empty}} = \sum_{(\mathbf{r},t)\in\mathcal{R}} \int_{s_{\text{n}}}^{d_t(\mathbf{u})-\varepsilon} \sigma(\mathbf{r}(s),t)\,\mathrm{d}s,$$



Losses

- **Static scene loss.**

$$\mathcal{L}_{\text{static}} = \sum_{(\mathbf{x},t)\in\mathcal{X}} \left\| F(\mathbf{x},t) - F(\mathbf{x},t') \right\|_2^2 ,$$

$(\mathbf{x},t)$ and $(\mathbf{x},t')$ are *not* close to any visible

(a) Input frame (b) w/o static loss (c) w/ static loss

Experiments

(a) Input (b) Mesh (c) Inpainted (d) NeRF-T (e) Ours

Summary

- From static to dynamic Nerf
  - ➢ Extra variables, i.e. timestamp
  - ➢ Monocular video
- Constraints/Losses
  - ➢ Depth/Scene flow
  - ➢ Static + Dynamic
- Issues
  - ➢ Per-instance/video training
  - ➢ Heavily rely on depth & flow estimation & accurate camera