CVPR JUNE 17-21, 2024 SEATTLE, WA

Multi-level Neural Scene Graphs for Dynamic Urban Environments

Tobias Fischer[1] Lorenzo Porzi[2] Samuel Rota Bulò[2] Marc Pollefeys[1] Peter Kontschieder[2]

[1]ETH Zürich [2]Meta Reality Labs

tobiasfshr.github.io/pub/ml-nsg

ETH zürich · Meta

TL;DR: We present the ***first benchmark*** and a ***novel method*** for **radiance field** reconstruction of **dynamic urban areas** from ***heterogeneous, multi-sequence*** **data**.

What?

We estimate the **radiance field** of **large-scale dynamic areas** from **multiple** vehicle **captures** under varying **environmental conditions**.

Why?

Today, driving data is available at unprecedented scale.

Opportunity:
Up-to-date **digital twins** of entire **cities**!

Example of different driving captures at the same intersection in Pittsburgh

Challenge:
Increasingly **heterogeneous data** - different lighting, weather, season, dynamic objects, …

Previous works…

- are restricted to **static** environments
- **do not scale** to more than a single short video
- struggle to **separately represent dynamic object** instances

→ Make *dynamic* radiance fields scale!

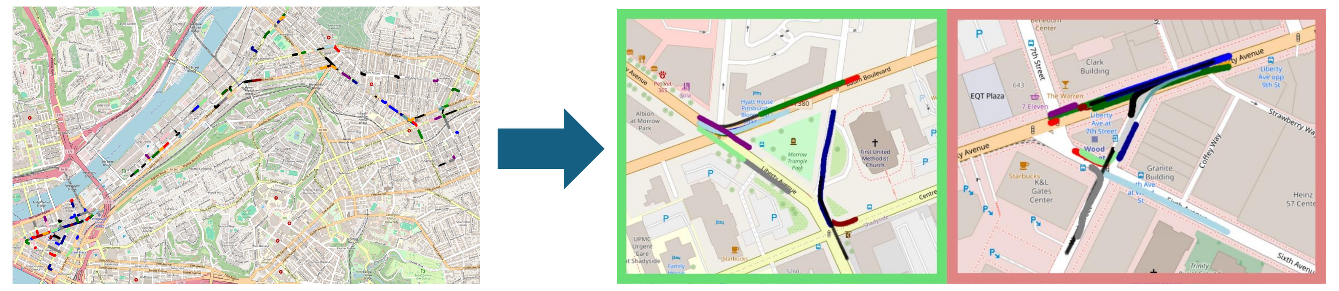
How?

1. **Multi-level** neural scene **graph representation** that **scales** to 10,000+ images with 1,000+ objects
2. **Fast** training and rendering via **composite ray sampling**
3. **Benchmark** for radiance field reconstruction in **dynamic urban** environments from **heterogeneous** vehicle captures

Benchmark

Data source: Argoverse 2 vehicle fleet

- Captures in **different weather, season, time of day**
- Calibrated, synchronized **sensors**: 7 global shutter cameras, LiDAR, GPS



Selected **37 vehicle captures**

- **2** geographic **regions**
  - Residential
  - Downtown
- >10K images per region
- >1K dynamic objects per region
- 14+ sequences in different conditions

Initialization via GPS → Offline **ICP alignment** across sequences

Problem Setup

Given *multiple, heterogeneous* input sequences S

- For each sequence $s \in S$
  - Timesteps $t \in T_s$
  - Ego-vehicle poses $\mathbf{P}_s^t$
  - Calibrated cameras C_s via extrinsics w.r.t. ego-vehicle $\mathbf{T}_c$ and intrinsics $\mathbf{K}_c$
- For each object $o \in O_s$
  - Poses w.r.t. ego-vehicle ξ_o^t
  - Object dimensions s_o

Goal

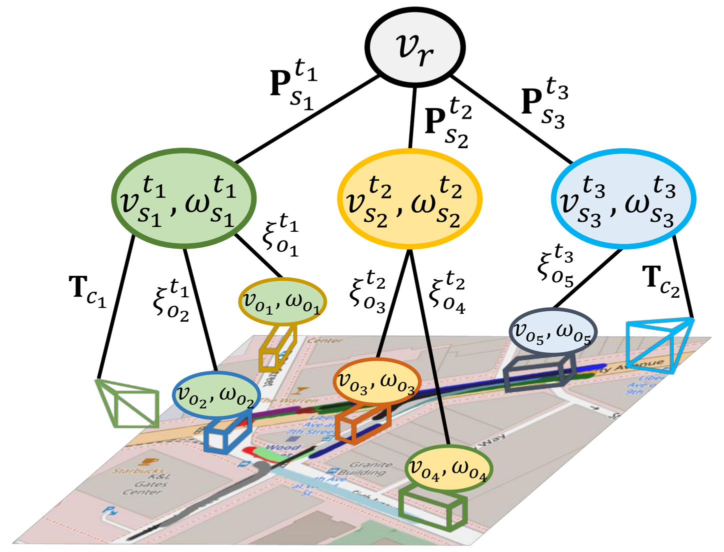
Estimate the radiance field f with parameters θ

$$f_\theta(\mathbf{x}, \mathbf{d}, t, s) = (\sigma(\mathbf{x}, t, s), \mathbf{c}(\mathbf{x}, \mathbf{d}, t, s))$$

Method

1. We create a **scene graph** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

- Nodes $\mathcal{V}$
  - Sequence nodes v_s^t
    - Latent code ω_s^t
  - Object nodes v_o
    - Latent code ω_o
  - Camera nodes v_c
- Edges $\mathcal{E}$
  - Rigid transformations
  - $e_{v_s^t \to v_r} = \mathbf{P}_s^t, \ldots$



2. Given $\mathcal{G}$, we **model** $\boldsymbol{f_\theta}$ with

- **Static** radiance field: $\phi(\mathbf{x}, \mathbf{d}, \omega_s^t)$
- **Dynamic** radiance field: $\psi(\mathbf{x}, \mathbf{d}, \omega_s^t, \omega_o)$

3. Given rays $(\mathbf{r}, t, s) \in \mathcal{R}$, we apply **volume rendering**

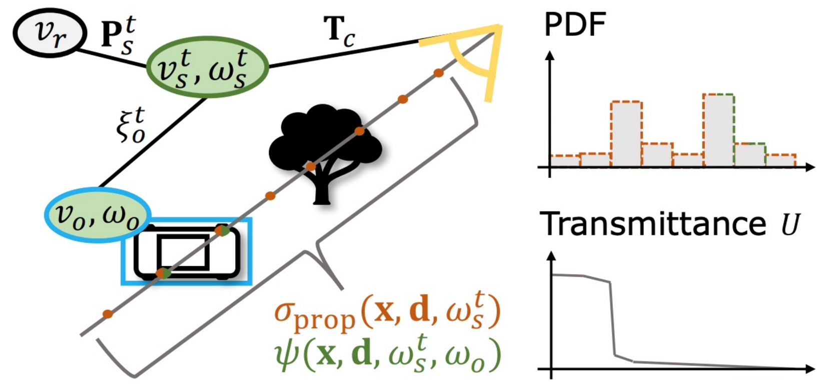
$$\hat{\mathbf{C}}(\mathbf{r}, t, s) = \int_{u_n}^{u_f} U(u)\sigma(\mathbf{r}(u), t, s)\mathbf{c}(\mathbf{r}(u), \mathbf{d}, t, s)du$$

where $\sigma = \sigma_\phi + \sigma_\psi$; $\mathbf{c} = \frac{\sigma_\phi}{\sigma_\phi + \sigma_\psi}\mathbf{c}_\phi + \frac{\sigma_\psi}{\sigma_\phi + \sigma_\psi}\mathbf{c}_\psi$

Composite Ray Sampling

1. CUDA ray-box intersection
→ $[u_{\text{in}}, u_{\text{out}}]$
2. Proposal sampling

- Proposal networks σ_{prop}
- Composite density:
  $\sigma_{\text{prop}} + \psi$



Optimization

RGB/depth losses: $\left\|\mathbf{C}(\mathbf{r}, t, s) - \hat{\mathbf{C}}(\mathbf{r}, t, s)\right\| + \left\|\mathbf{D}(\mathbf{r}, t, s) - \hat{\mathbf{D}}(\mathbf{r}, t, s)\right\|$

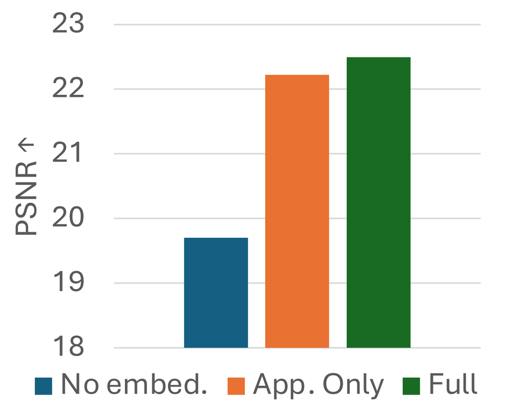
Entropy regularization: $\int_{u_n}^{u_f} \mathcal{H}\left(\frac{\sigma_\psi(\mathbf{r}(u), t, s)}{\sigma_\phi(\mathbf{r}(u), t, s) + \sigma_\psi(\mathbf{r}(u), t, s)}\right) du$

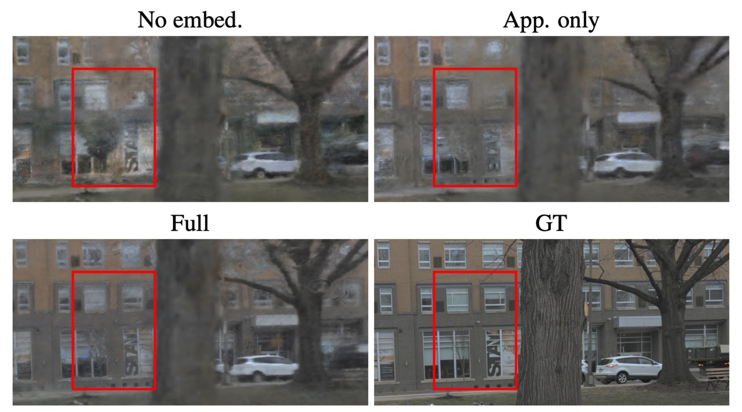
Hierarchical pose optimization:
Ego-vehicle poses $\delta\mathbf{P}_s^t \in \mathbf{SE}(3)$
Object poses $\delta\xi_o^t \in \mathbf{SE}(2)$

Experiments

Analysis

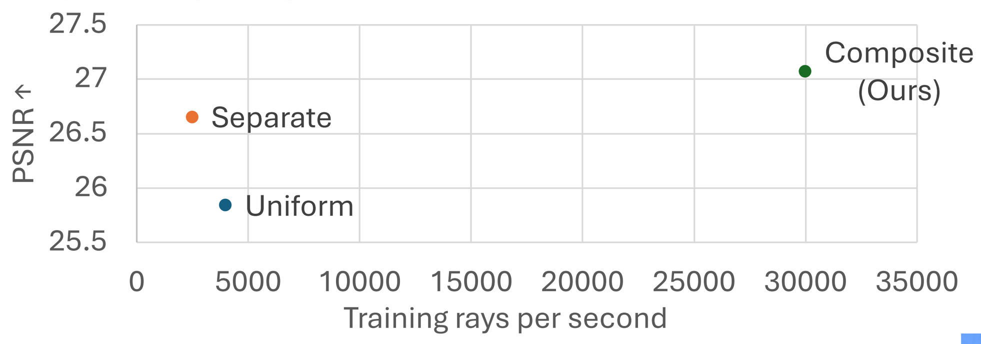
Importance of latent codes at nodes v_s^t



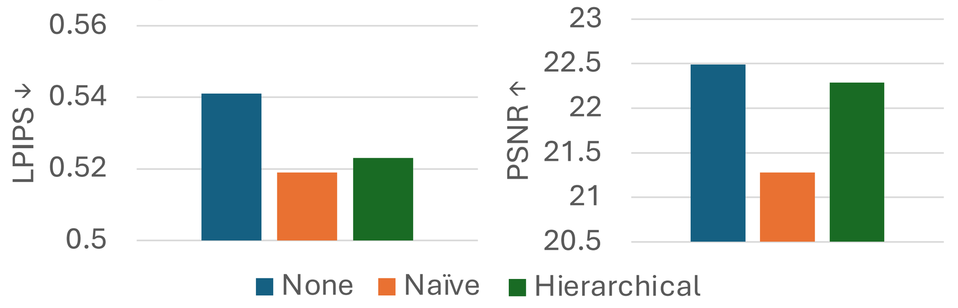


Exchange scene appearance ω_s^t
→ Car's appearance adapts to conditions of the scene

Ray sampling schemes

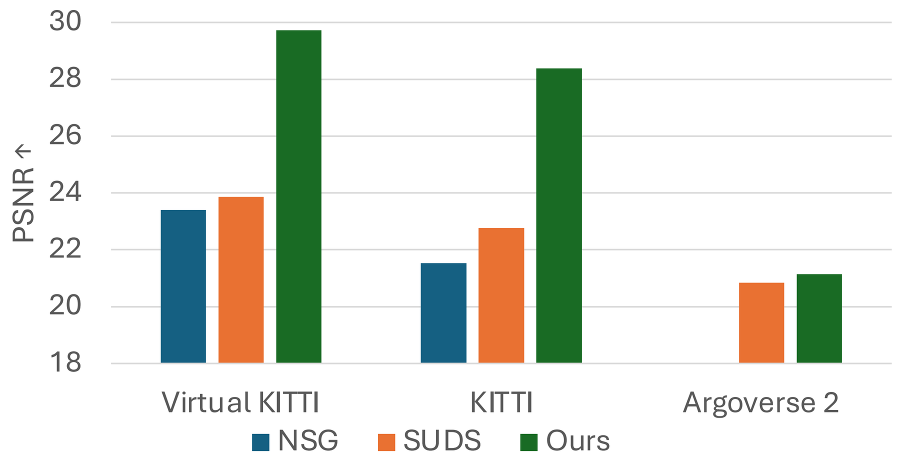


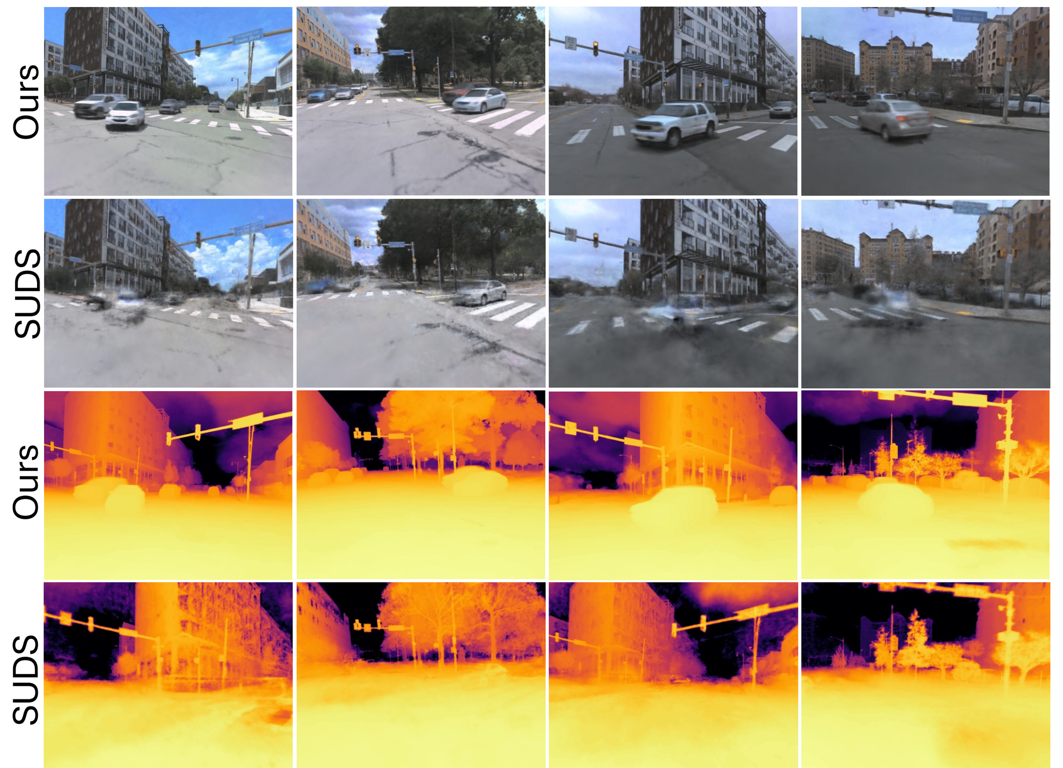
Pose optimization



Comparison to state-of-the-art

We use KITTI, Virtual KITTI, Argoverse 2





Take aways

1. Modeling **transient** geometry ***and*** sequence **appearance** via latent codes
2. Dynamic **objects** can **change** their **appearance according to** the **scene**
3. Composite **ray sampling** is **key for efficiency**
4. Leveraging **multi-camera constraints** in **pose optimization** improves quality

Also check out our follow-up work!

Dynamic 3D Gaussian Fields for Urban Areas
tobiasfshr.github.io/pub/4dgf