Dynamic 3D Gaussian Fields for Urban Areas

Tobias Fischer[1] Jonas Kulhanek[1,3] Samuel Rota Bulò[2] Lorenzo Porzi[2] Marc Pollefeys[1] Peter Kontschieder[2]

tobiasfshr.github.io/pub/4dgf

1 ETH zürich 2 Meta 3

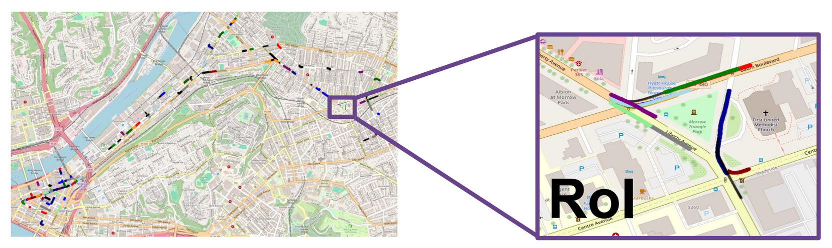
NEURAL INFORMATION PROCESSING SYSTEMS

TL;DR Given a set of *heterogeneous* sequences of a shared geographic area, we optimize a *single dynamic* scene representation that renders arbitrary viewpoints and configurations at interactive speeds.

1. What?

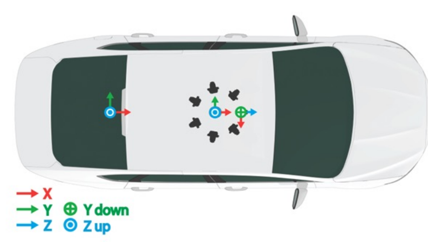
Given: Set of sequences $\mathcal{S}$ of a common geographic region of interest (RoI) with:

- Distinct dynamic objects
- Transient objects (construction sites, …)
- Varying environmental conditions



Each sequence $s \in \mathcal{S}$ has…

Ego-vehicle poses $\mathbf{P}$
Camera calibration $\mathbf{T}, \mathbf{K}$
Dynamic object poses ξ



Goal: Learn function f_θ that, for given viewpoint $(\mathbf{T}, \mathbf{K})$ at sequence s and time t, outputs the image $\mathcal{I} \in [0,1]^{H \times W \times 3}$ with correct appearance & dynamic actors

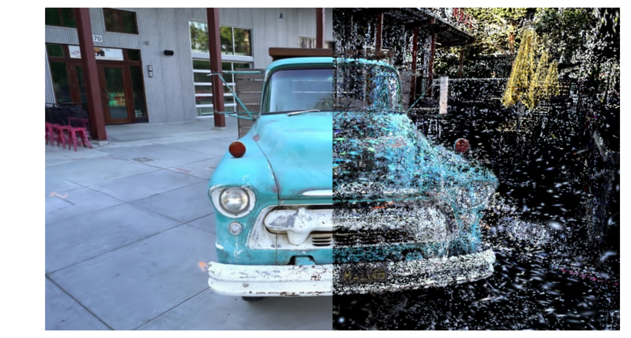
Inputs

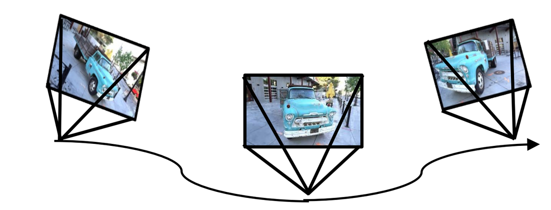
$f_\theta(\mathbf{T}, \mathbf{K}, t, s) \rightarrow \mathcal{I} \in [0,1]^{H \times W \times 3}$

2. Why?

3D Gaussian Splatting achieved high-quality novel view synthesis at high speed, but:

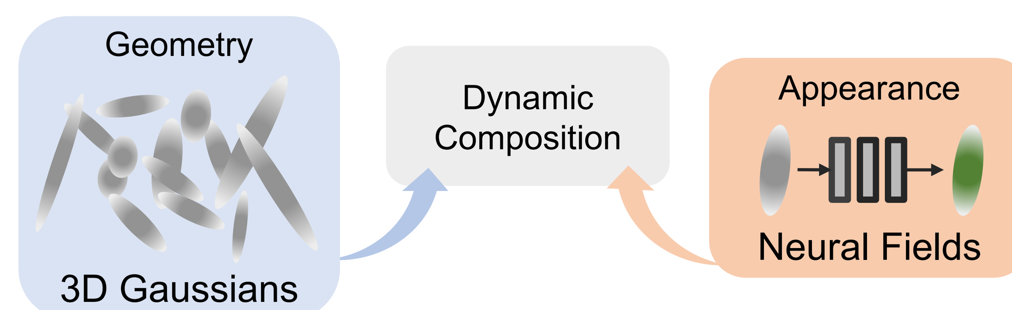
1. Limited **scalability**: Memory footprint increases linearly with number of primitives
2. Requires **homogeneous input data**: Spherical harmonics cannot model large appearance changes
3. Limited to **static** scenes: 3D primitives have a fixed location





3. How?

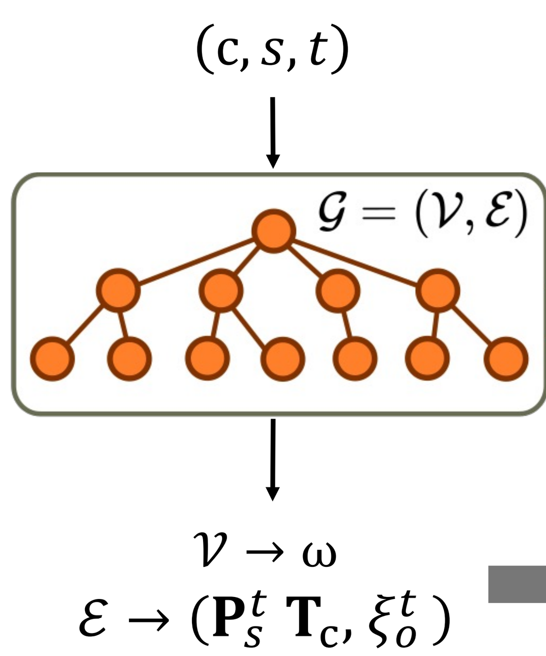
1. **Gaussians** as efficient **geometry scaffold**
2. **Neural fields** as a compact and flexible **appearance model**
3. **Scene dynamics** via scene **graph** at **global** level and **deformations** at **local** level.



Scene Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
Stores scene **configurations**

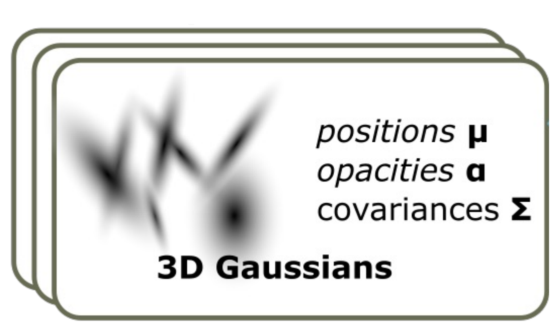
Nodes $\mathcal{V}$: Latent codes ω
Sequence code ω_s^t
Object ID ω_o

Edges $\mathcal{E}$: Rigid transformations
Ego-vehicle poses $\mathbf{P}$
Dynamic object poses ξ
Camera to ego-vehicle $\mathbf{T}$



3D Gaussians as Geometry Scaffold
Multiple sets of 3D Gaussians

- **Static** $\mathrm{G_r}$
  World space
- **Dynamic** $\{\mathrm{G_o} \in \mathcal{O}_s\}$
  Canonical space

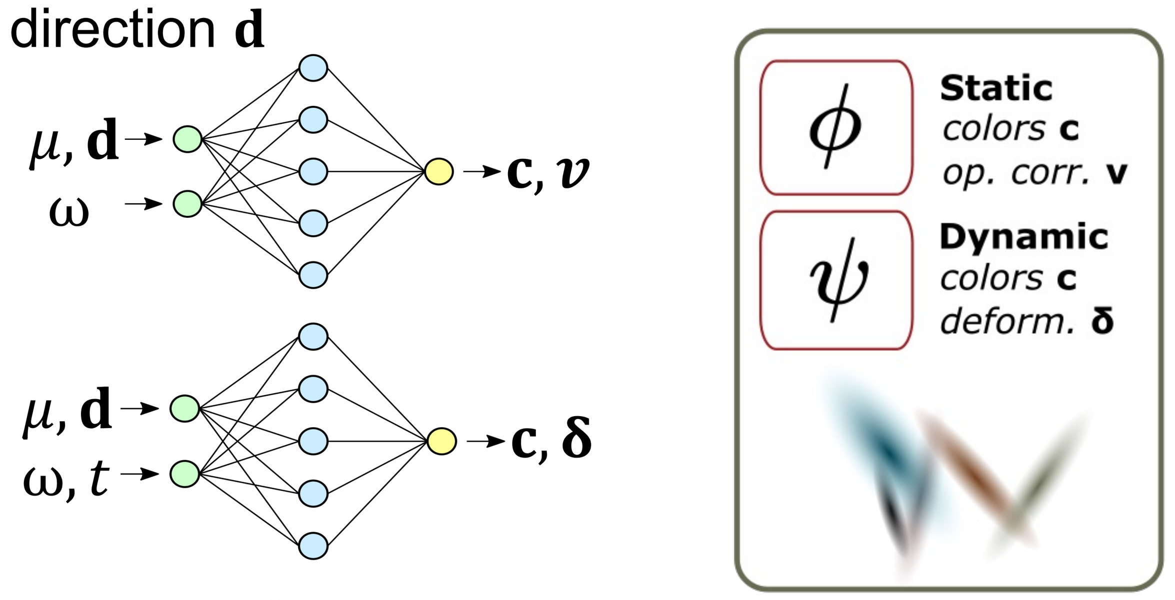


Remove SH-Coefficients $\mathbf{c}_{\text{sh}} \in \mathbb{R}^{48}$
→ Reduced **memory footprint**

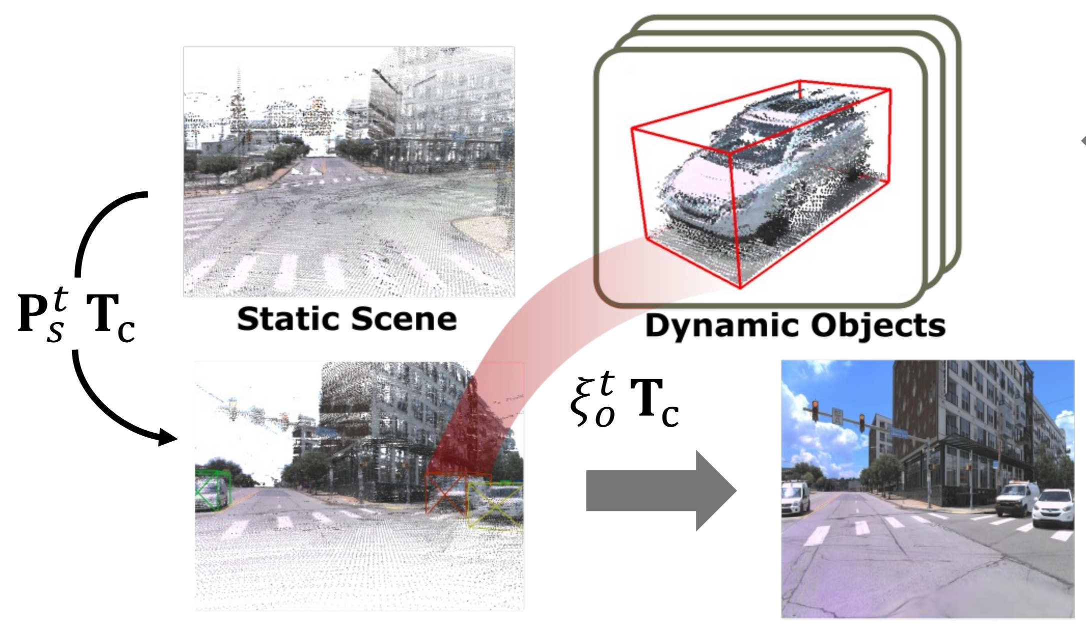
- $N \times 59 \times 4$ bytes = **2.25 GiB**
- $N \times 11 \times 4$ bytes = **0.42 GiB**

… for 10M Gaussians

Neural Fields as Appearance Model
Infer 3D Gaussian color $\mathbf{c}$, opacity correction $\boldsymbol{\nu}$ and deformation $\boldsymbol{\delta}$ from code ω, position μ, and direction $\mathbf{d}$



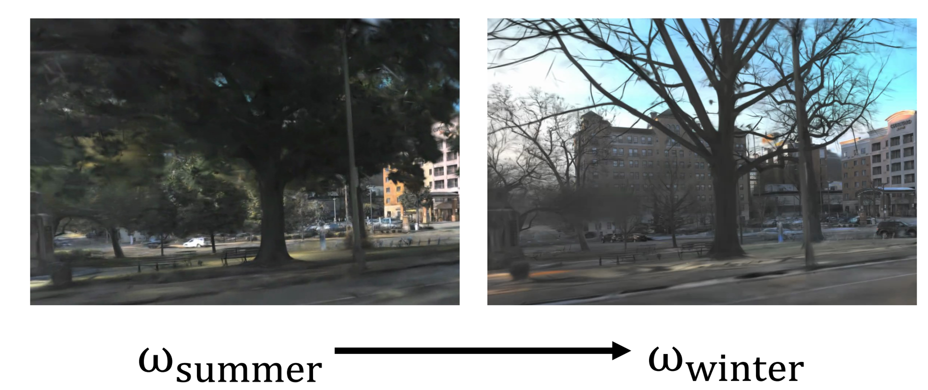
Dynamic Composition
Compose retrieved information into scene at (c, s, t) and render the image:

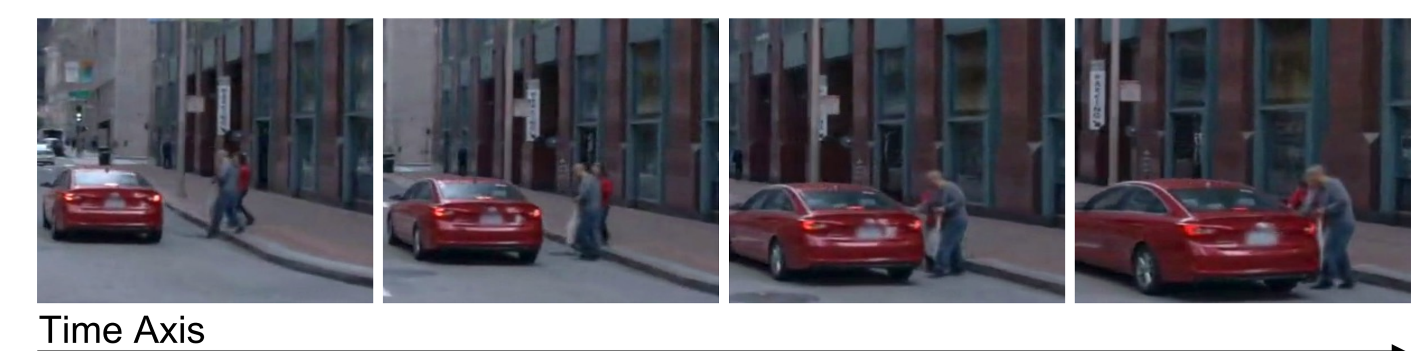


4. And?

We can now **model multiple complex traffic scenarios**

- Change scene appearance *and* geometry by exchanging latent code ω
- Model articulated motion: walking, holding a shopping bag, opening car door, …





Key insights

Neural fields

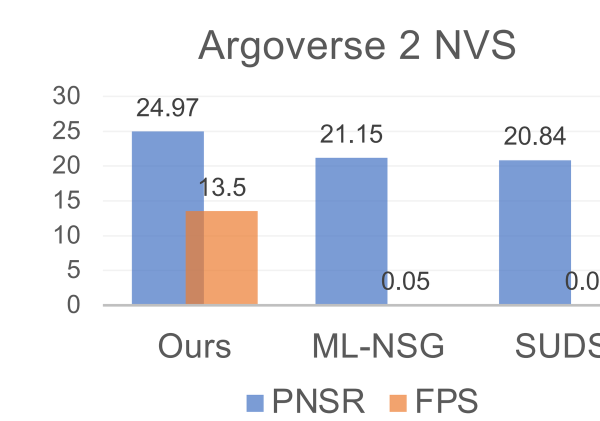
- Achieve **comparable accuracy** to spherical harmonics
- Scale better
- Are *much* more **flexible**
- Do **not hurt speed** critically

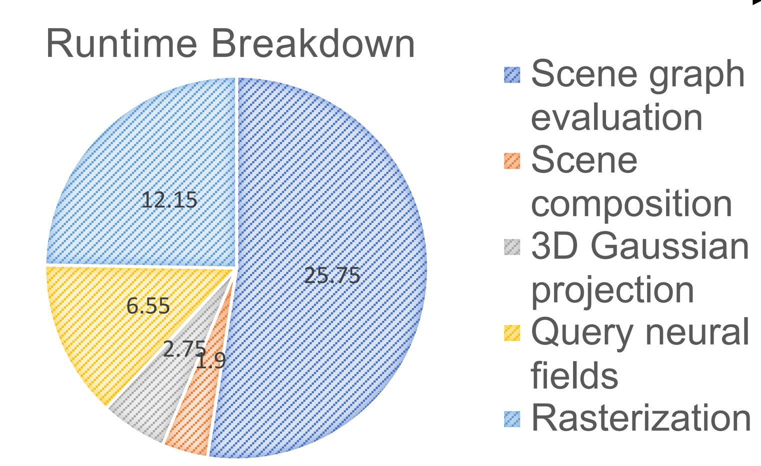
Opacity correction term $\boldsymbol{\nu}$

- Essential for *realistic* rendering of heterogeneous captures (Geom. + App.)

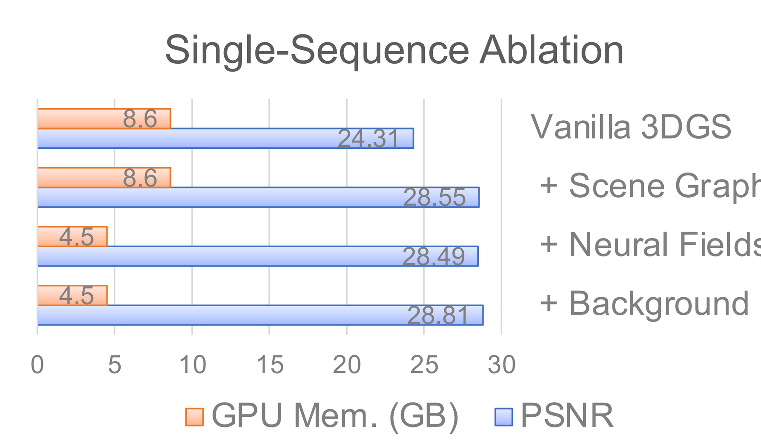
Helpful tricks

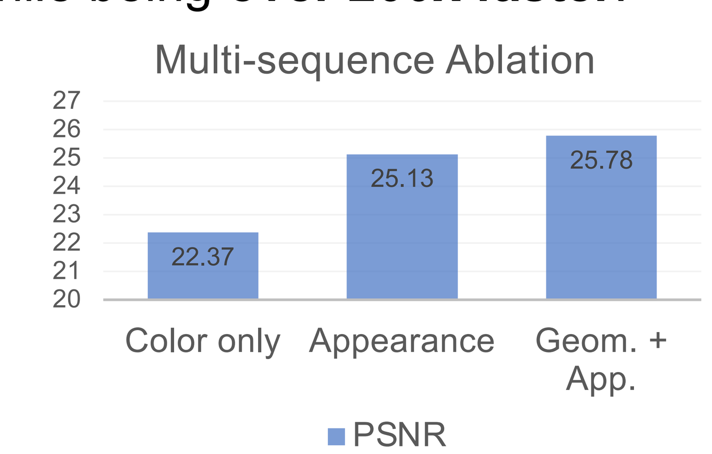
- Multi-GPU training & ADC
- Camera optimization
- Background modeling
- 4D Hash grids

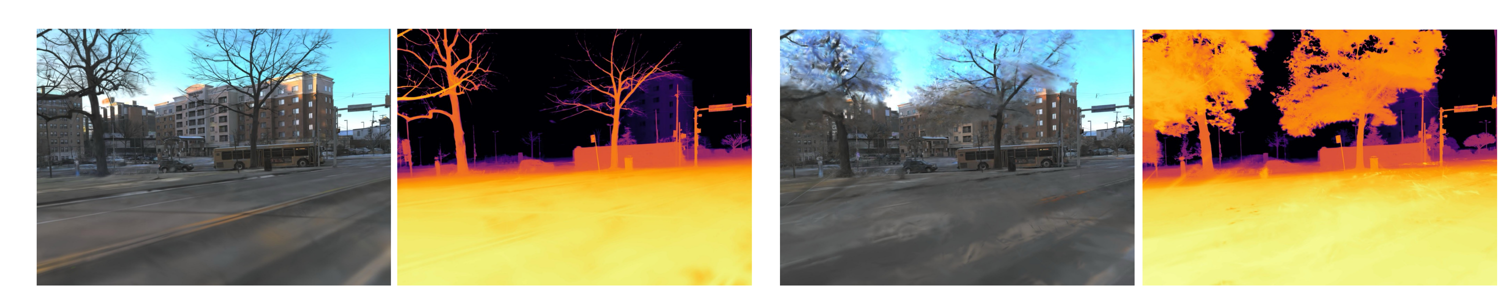




We **beat SOTA by over 3 dB** while being **over 200x faster**!







Qualitative example: Geom. + App. *vs* Appearance