Learning Unified Representation of 3D Gaussian Splatting

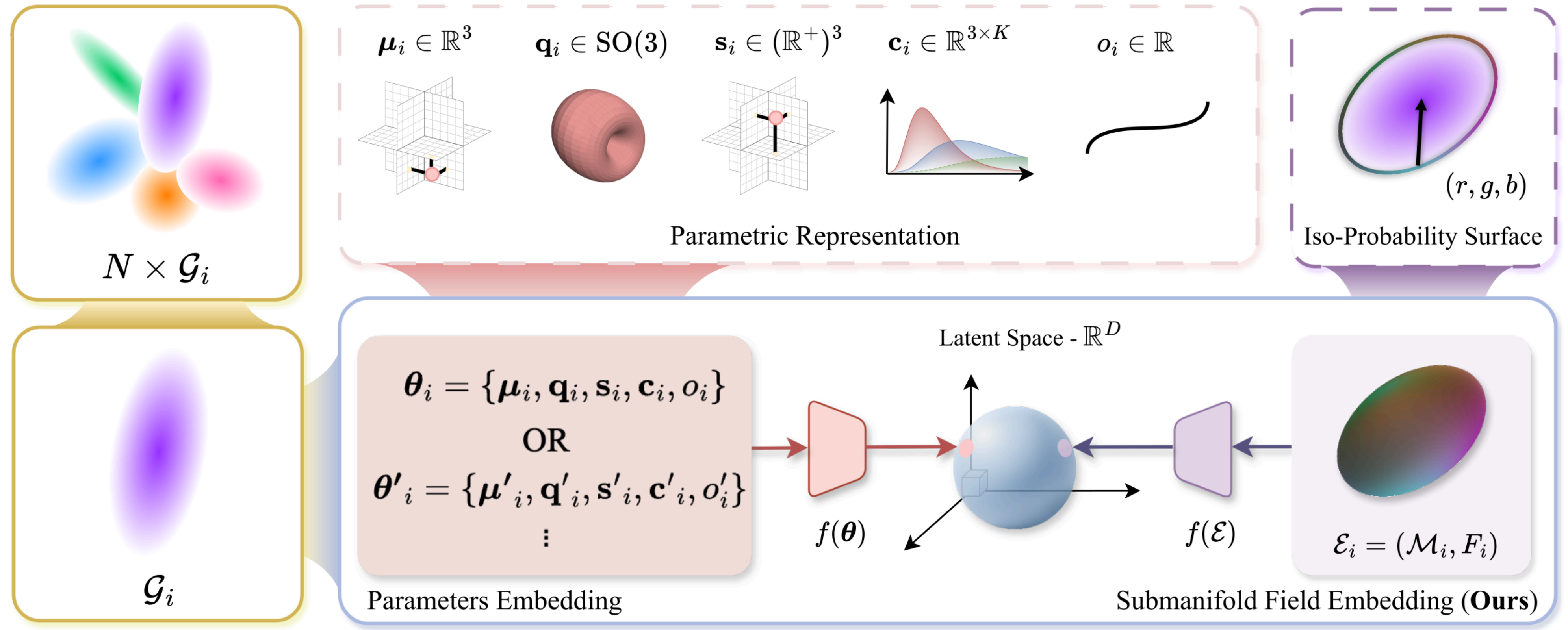
Yuelin Xin[1] Yuheng Liu[1] Xiaohui Xie[1] Xinke Li[2]

[1]UC Irvine [2]City University of Hong Kong

ICLR 2026, Rio de Janeiro. Session 1

Project Website

Motivation



Why parameter space is the wrong learning space

A Gaussian primitive is parameterized as

$$\theta = \{\mu, q, s, c, o\}$$

This space is convenient for rendering, but poorly matched to neural learning:

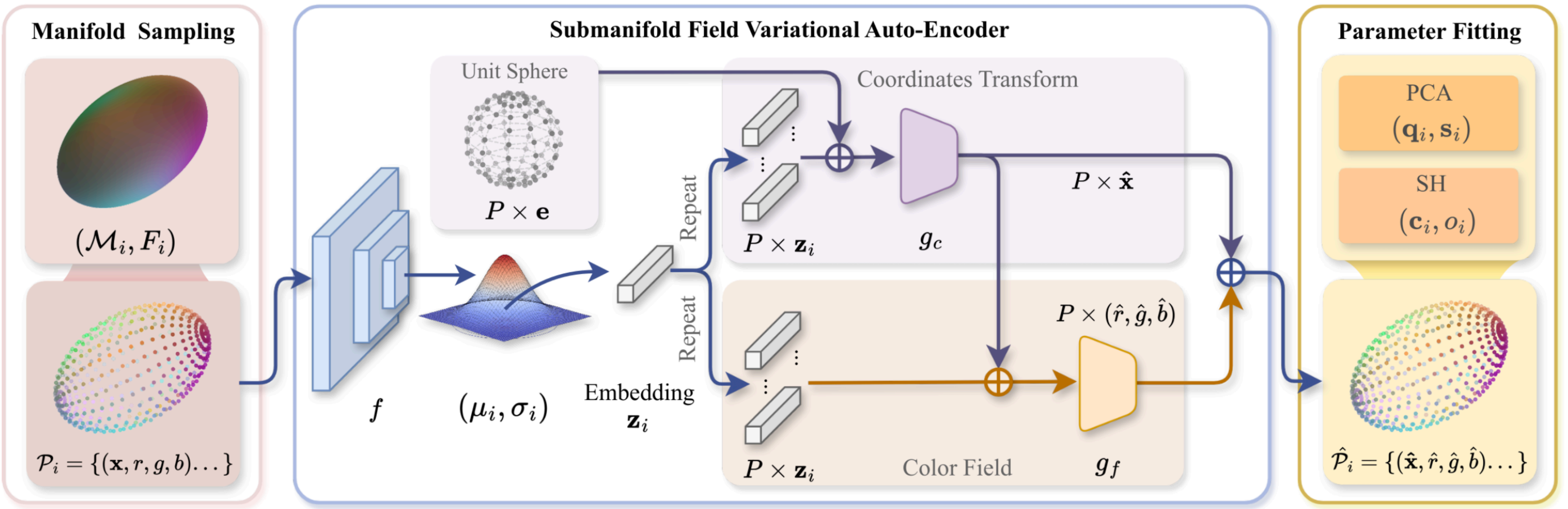
- **non-unique:** distinct θ can induce the same rendered field;
- **heterogeneous:** $\mu \in \mathbb{R}^3, q \in SO(3), s \in (\mathbb{R}_+)^3, c \in SH space$;
- **unstable:** small perturbations in parameter space can produce disproportionate errors.

We formalize the rendered effect of one Gaussian as a single-Gaussian radiance field

$$\phi_G(x, d) = \rho_G(x) \cdot c_G(d)$$

Key point: the map $\theta \mapsto \phi_G$ is not injective.
So parameter regression is fundamentally ambiguous and not suitable for neural learning.

Method



Our representation

We represent each Gaussian by a field on its iso-probability surface:

$$\mathcal{M}_i = \{\mathbf{x} \in \mathbb{R}^3 \mid (\mathbf{x} - \mu_i)^\top \Sigma_i^{-1} (\mathbf{x} - \mu_i) = r^2\}$$

$$F_i(\mathbf{x}) = \sigma(o_i) \cdot \mathrm{Color}_i(\mathbf{d}_\mathbf{x})$$

The unified representation is $E = (\mathcal{M}, F)$
This replaces parameter tuples with a **geometric-photometric object**. Unlike θ, E is constructed to be **unique** with respect to the underlying Gaussian field.

Encoding & decoding

We discretize $(\mathcal{M}, F)$ as a colored point cloud and encode it using a **Submanifold Field VAE (SF-VAE)**:

- Encoder: PointNet maps sampled points to a 32-D latent code.
- Decoder: reconstructs the field and recovers Gaussian parameters via PCA + SH fitting.
- Loss: Wasserstein-2 based **Manifold Distance (M-Dist)**, aligning spatial and color similarity, discretized as:

$$\hat{W}_2^2(\mathcal{P}, \hat{\mathcal{P}}) = \min_{\boldsymbol{\Gamma} \in \Gamma(\hat{\sigma}, \hat{\sigma}')} \sum_{(\mathbf{x}_i, c_{x_i}) \in \mathcal{P}} \sum_{(\mathbf{y}_j, c_{y_j}) \in \hat{\mathcal{P}}} \boldsymbol{\Gamma}_{ij} \left(d^2\left((\mathbf{x}_i, c_{x_i}), (\mathbf{y}_j, c_{y_j})\right)\right)$$

Results

We evaluate on **ShapeSplat** and **Mip-NeRF 360**, comparing submanifold fields with parametric baselines under matched capacity. Key results:

- **Higher fidelity**: SF-VAE achieves PSNR↑63.4 vs 44.7/37.5 for parametric baselines (ShapeSplat).
- **Generalization**: superior cross-domain transfer (object → scene PSNR↑19.2 vs 9.8 for parametric).
- **Robustness**: embeddings resilient to noise; interpolations smooth and semantically consistent.
- **Better latent structure**: unsupervised clustering on SF embeddings shows clearer semantic separation.

Input Representation	Encoder	Decoder	PSNR ↑	SSIM ↑	LPIPS↓	M-Dist	L_1-Dist
ShapeSplat							
Parametric	MLP	MLP	37.512	0.888	0.152	0.184	<u>0.040</u>
Parametric	MLP	SF-VAE	44.725	0.896	0.136	0.051	0.097
Submanifold Field	SF-VAE	SF-VAE	**63.408**	**0.990**	**0.010**	<u>0.041</u>	0.098
Mip-NeRF 360							
Parametric	MLP	MLP	18.818	0.564	0.452	0.510	<u>0.034</u>
Parametric	MLP	SF-VAE	20.923	0.730	0.359	0.055	0.173
Submanifold Field	SF-VAE	SF-VAE	**29.833**	**0.953**	**0.079**	<u>0.048</u>	0.179