IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Density-Aware Point Cloud Upsampling via Relational Graph Flow Matching

Yuzhong Deng, Dongzhen Liu, Di Wu, Junlin Song, Jianxiao Zou, *Member, IEEE*, and Shicai Fan

***Abstract*—Real-world point clouds exhibit non-uniform density distributions, varying across distance and scale. Conventional upsampling methods typically treat points homogeneously, which over-smooths sparse regions while over-processing dense regions. We propose PURF, a density-aware point cloud upsampling framework based on relational graph flow matching. PURF leverages a heterogeneous graph representation to capture density variations through relational graph construction, and employs transformer-based flow matching to predict timestep-dependent velocity fields. This design enables a density-aware and efficient mapping from sparse inputs to dense point clouds, reducing computational overhead compared to recent approaches. Extensive experiments on synthetic, KITTI, and a proprietary campus dataset collected by our team demonstrate that PURF achieves advanced performance in upsampling point clouds qualitatively and quantitatively.**

***Index Terms*—Deep learning for visual perception, computer vision for automation, representation learning.**

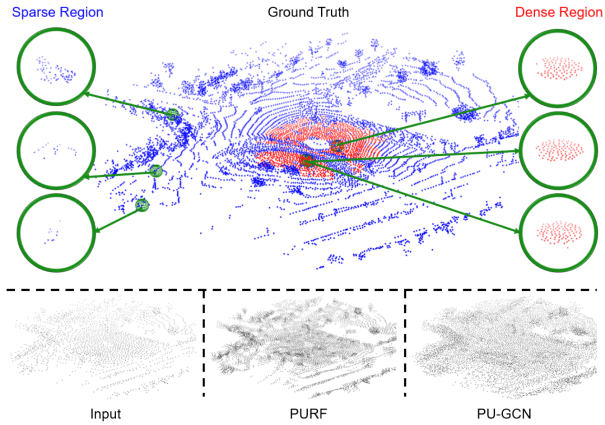

Fig. 1. PURF distinguishes dense and sparse regions in point cloud upsampling, resulting in more precise geometric structures compared to over-smoothed details of homogeneous methods.

I. Introduction

POINT clouds have become a fundamental 3D data representation in a wide range of applications, including autonomous driving, robotics, large-scale mapping, augmented and virtual reality, and industrial inspection [1], [2], [3]. Their ability to capture fine-grained geometric details and complex spatial structures makes them indispensable for downstream tasks, such as high-level perception and scene understanding.

However, unlike structured 2D data, point clouds are inherently irregular and often exhibit highly variable spatial density. This non-uniformity arises from diverse sources, such as sensor characteristics, occlusions, surface reflectance, scanning perspectives, and environmental factors. As a result, regions close to the sensor or with favorable granularity are often densely sampled, while distant, occluded, or low-reflectance areas are sparsely populated. Such density variation, coupled with the presence of outliers and incomplete coverage, poses significant challenges for downstream tasks including mapping, object detection, semantic segmentation, and surface reconstruction [4], [5], [6]. Existing upsampling methods often treat all regions homogeneously, failing to distinguish patterns between sparse and dense regions as in Fig. 1, which would result in over-smoothing geometric details in sparse areas and redundant computation in dense ones.

Traditional upsampling methods, while effective on synthetic or uniformly sampled data, often struggle to generalize to the complex, heterogeneous density patterns characteristic of real-world point clouds [7], [8], [9], [10], [11], [12]. Recent generative models, particularly diffusion-based approaches, have demonstrated impressive fidelity by modelling the gradient of data distribution from dense point clouds [13], [14], [15]. However, these methods typically require hundreds of iterative refinement steps to achieve satisfying results, resulting in substantial computational overhead and slow inference, which limits their scalability and real-time applicability.

To address these challenges, we propose PURF, a density-aware point cloud upsampling framework based on relational graph flow matching. PURF (i) discretizes local density into dense/sparse relations via a learnable threshold in relational graph construction, enabling relation-specific message passing and preserving density-aware structures, and (ii) predicts velocity fields with a transformer-based flow matching paradigm, which are integrated with few ODE steps at inference to reduce

Received 7 June 2025; accepted 6 December 2025. Date of publication 12 January 2026; date of current version 22 January 2026. This article was recommended for publication by Associate Editor O. Mees and Editor A. Faust upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62473078 and in part by Shenzhen Science and Technology Program under Grant JCYJ20210324140407021. *(Corresponding author: Shicai Fan.)*

Yuzhong Deng, Dongzhen Liu, and Di Wu are with the School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China.

Junlin Song is with the School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with Dongfang Electric Digital Technology Company, Ltd., Chengdu 610218, China.

Jianxiao Zou and Shicai Fan are with the School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China, and also with the Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China (e-mail: shicaifan@uestc.edu.cn).

Digital Object Identifier 10.1109/LRA.2026.3653291

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

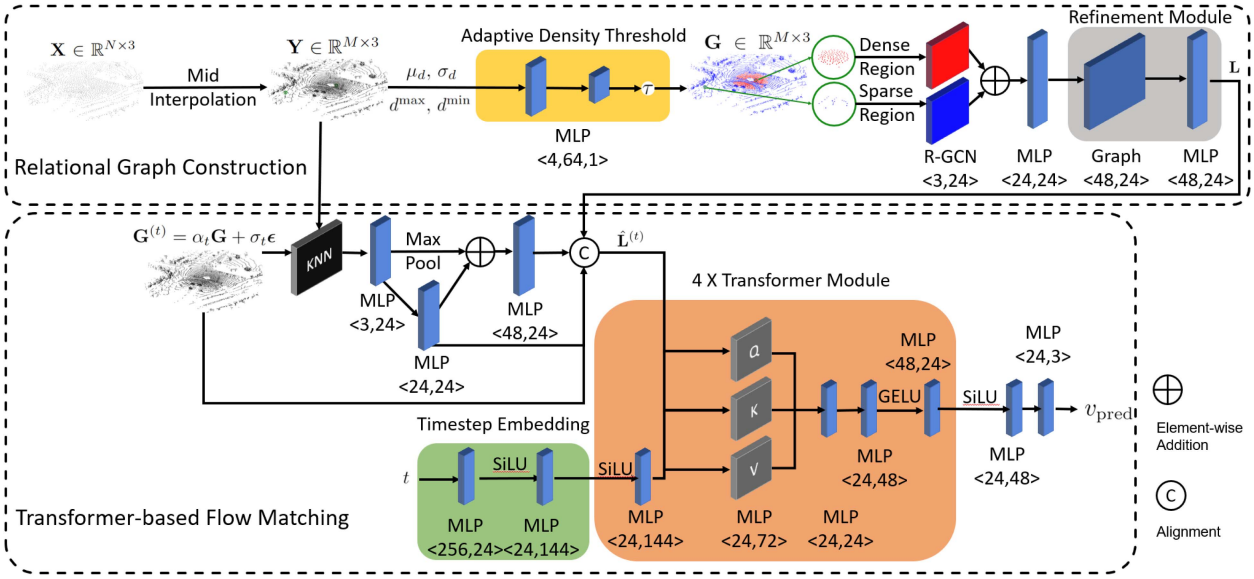

Fig. 2. Overview of the PURF framework for density-aware point cloud upsampling. The upper part (**Relational Graph Construction**) constructs a heterogeneous graph from the mid-interpolated input by estimating adaptive local density thresholds and extracting region-aware features using R-GCNs. The lower part (**Transformer-based Flow Matching**) aligns noisy ground truth with relational features, and iteratively predicts denoising velocity fields via a multi-layer transformer module, conditioned on timestep embeddings.

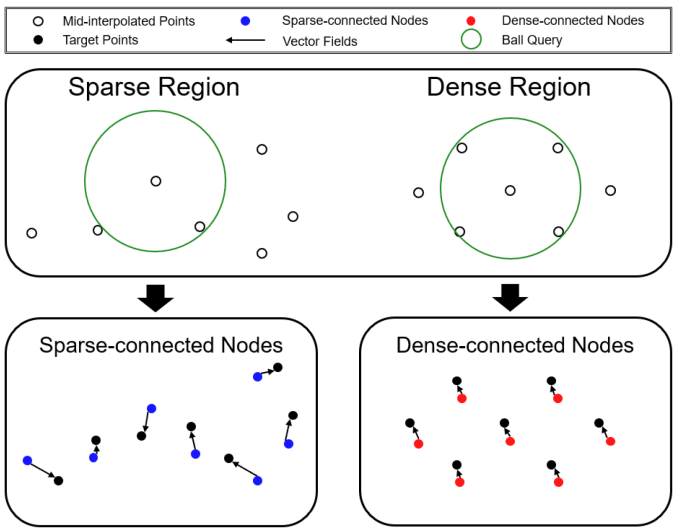

Fig. 3. The graph is made heterogeneous by binarizing per-point density values estimated via ball-query spatial analysis, partitioning nodes into sparse-connected and dense-connected types. The threshold for binarization is learned via MLP based on density statistics over the patch.

computation cost. The accuracy and efficiency of PURF make it well-suited for deployment in resource-constrained robots.

II. Related Work

A. Learnable Point Cloud Upsampling

Learnable point cloud upsampling methods aim to generate dense point clouds from sparse inputs using neural networks. Early approaches such as PU-Net [7] introduced the use of multi-layer perceptrons (MLPs) based on PointNet++ [16] for upsampling, a concept later extended by patch-based progressive

TABLE I
4× Upsampling on PUGAN and PU1K Datasets. Metrics are CD ($\times 10^{-3}$), HD ($\times 10^{-3}$), and P2F ($\times 10^{-3}$). Bold Text and Underline Indicate the Best and Second-Best Performance.

Datasets	PUGAN			PU1K		
Methods	CD↓	HD↓	P2F↓	CD↓	HD↓	P2F↓
PU-Net [7]	0.529	6.805	4.460	0.175	3.684	4.822
MPU [8]	0.292	6.672	2.822	0.146	3.337	3.537
PU-GAN [9]	0.282	5.577	2.016	0.108	2.584	2.311
PU-GCN [10]	0.268	3.201	2.489	0.115	1.748	2.487
Grad-PU [11]	0.245	1.479	**1.893**	**0.096**	0.695	**1.673**
PUDM [13]	0.242	1.366	1.912	0.114	0.692	2.882
PURF	**0.234**	**1.357**	2.215	**0.096**	**0.676**	1.736

models (e.g., MPU [8]) and generative adversarial networks such as PU-GAN [9]. However, these methods generally treat all points in the point cloud as homogeneous entities, thereby failing to distinguish regions with differing density levels. This homogeneous treatment leads to several undesirable effects, including noise, outliers, and local shrinkage, especially in areas where the density contrast is pronounced. Even recent methods like Grad-PU [11] that decouple the upsampling process into coarse interpolation and iterative refinement have not fully addressed the challenges associated with non-uniform density distributions. Consequently, the inability to differentiate between sparse and dense regions often results in suboptimal upsampling performance.

B. Graph Representation for Point Processing

Graph-based representations have shown great promise in processing irregular point clouds by exploiting their inherent non-Euclidean structures. Early works, such as DGCNN [17] and Point-GNN [18], demonstrated the effectiveness of local neighborhood aggregation using learned features. Most existing point cloud upsampling methods treat point clouds as homogeneous graphs [8], [10], which limits their ability to capture

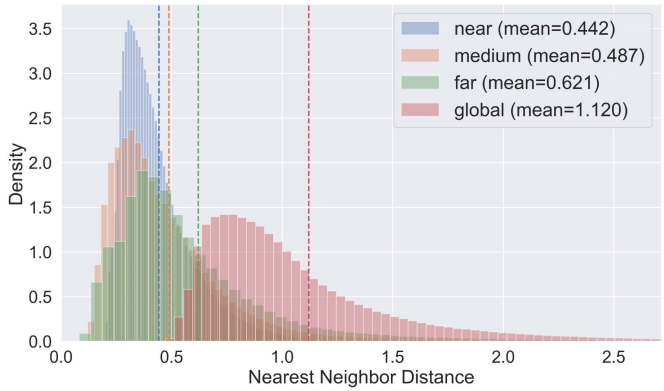

Fig. 4. Distribution of nearest neighbor distances of KITTI dataset in different regions.

and exploit the spatial density variations naturally present in real-world data.

In contrast, representing point clouds as *heterogeneous graphs*—in which nodes and edges can encode diverse attributes such as local density information—presents an exciting research direction for point cloud upsampling [19]. Such an approach, as relational graph convolutional networks (R-GCN) [20], holds the potential to better handle the challenges posed by inconsistent sparsity due to sharp edges, object boundaries, and distant surfaces. By differentiating between regions of varying density, a heterogeneous graph representation could facilitate adaptive feature aggregation and refined sampling strategies that lead to more accurate and robust upsampling outcomes.

C. Diffusion and Flow Models for Point Processing

Diffusion models have recently shown impressive performance for 2D and 3D generative tasks [21], [22], [23], including point cloud upsampling. Methods such as PUDM [13] model the gradient of the data distribution from noise to clean samples. However, they typically require hundreds of refinement steps and may struggle to recover fine-grained structures in high-resolution or anisotropic regions. Alternatively, normalizing flow approaches (e.g., PU-Flow [24]) learn invertible mappings between noisy and clean point clouds but incur significant computational overhead due to the need for Jacobian determinant evaluations. In contrast, our method leverages a flow matching framework [25] that directly learns a time-dependent velocity field without the constraints of invertibility or explicit likelihood estimation. This allows our approach to efficiently bridge the gap from noisy inputs to true geometry while maintaining flexibility and computational efficiency.

III. Method

Fig. 2 presents the overall architecture of PURF. The framework is divided into two key components: (1) a relational graph construction module that adaptively encodes density variations and spatial relationships through heterogeneous graph representation, and (2) a transformer-based flow matching module that leverages both graph features and time-dependent embeddings to predict velocity fields for iterative point cloud denoising.

Given a sparse input point cloud $\mathbf{X} \in \mathbb{R}^{N\times 3}$ and a dense ground truth $\mathbf{G} \in \mathbb{R}^{M\times 3}$, our goal is to produce a clean and dense point cloud via a learned flow matching process. Our framework consists of a heterogeneous graph feature constructor that captures density-aware local structures; and a velocity-based denoising module that, following the flow matching paradigm, predicts a time-dependent velocity field $v_\theta(\cdot, t)$ conditioned on both the current noisy state and the heterogeneous graph features.

A. Relational Graph Construction and Alignment

First, a mid-interpolation operation is applied to the sparse input:

$$\mathbf{Y} = f_{\text{interp}}(\mathbf{X}) \in \mathbb{R}^{M\times 3}. \tag{1}$$

The patterns between sparse and dense regions are different, which leads mid-interpolated points to different velocity fields in approaching target points (see Fig. 3). To capture the density-aware context, we construct a relational graph from $\mathbf{Y}$:

$$\mathcal{H}(\mathbf{Y}) = \{(\mathbf{v}_i, \mathbf{e}_{ij}) \mid \mathbf{v}_i \in \mathbf{Y},\ \mathbf{e}_{ij} \in E\}, \tag{2}$$

where edges E are generated via a ball-query based density estimation and subsequent binarization.

To adaptively distinguish between dense and sparse regions in the constructed graph, we first estimate the local density for each point using a ball-query mechanism. Specifically, for each point $\mathbf{y}_i \in \mathbf{Y}$, we count the number of neighbors n_i within a fixed radius R, and normalize this count to obtain a density score $d_i \in \mathbf{d}$ (density scores over the input point cloud patch):

$$n_i = |\{j \mid \|\mathbf{y}_i - \mathbf{y}_j\|_2 < R\}| \tag{3}$$

$$d_i = \frac{n_i}{\max_j n_j + \eta} \tag{4}$$

where $\mathbf{y}_j$ denotes a candidate neighbor in the same patch and η is a small constant to prevent division by zero.

To achieve adaptive thresholding, we employ a learnable multi-layer perceptron (MLP), denoted as $\mathcal{M}_\theta$, which predicts a threshold τ ensured by a sigmoid activation $\sigma(\cdot)$:

$$\tau = \mathcal{M}_\theta(\mathbf{f}_d) = \sigma\left(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1\mathbf{f}_d + \mathbf{b}_1) + \mathbf{b}_2\right). \tag{5}$$

For predicting the threshold τ, patch density statistics (mean μ_d, deviation σ_d, max $d^{\max}$, min $d^{\min}$) of density scores $\mathbf{d}$ are stacked into the density feature vector $\mathbf{f}_d$ as input to the MLP. $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2$ are learnable parameters.

Given normalized density d_i and the threshold τ, we compute:

$$r_i^{\text{soft}} = \sigma((d_i - \tau)/temp), r_i^{\text{hard}} = \mathbf{1}[\,d_i \geq \tau\,], \tag{6}$$

$$r_i = r_i^{\text{hard}} + \left(r_i^{\text{soft}} - \text{sg}(r_i^{\text{soft}})\right), \tag{7}$$

where $temp$ is the temperature and $\text{sg}(\cdot)$ stops gradients of STE (Straight-Through Estimator) [26]. Here $r_i^{\text{hard}} = 1$ denotes a dense relation and $r_i^{\text{hard}} = 0$ a sparse one. During training, edges are typed by the hard gate ($r_i = r_i^{\text{hard}}$) in the forward pass while gradients update via r_i^{soft} to τ.

This adaptive binarization enables the model to partition the graph into dense and sparse regions, thereby facilitating effective density-aware feature aggregation and message passing throughout the network.

Let the initial latent feature of node i be $\mathbf{h}_i$ (initialized from its 3D coordinate $\mathbf{y}_i \in \mathbf{Y}$ and relation type r_i). An R-GCN aggregates neighborhood features using the message passing

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

rule [20]:

$$\mathbf{F}_i = \sigma\left(\sum_{r_i\in\{0,1\}}\sum_{j\in\mathcal{N}_\mathbf{r}(i)}\frac{1}{c_{i,r}}\mathbf{W}_r\,\mathbf{h}_j + \mathbf{W}_0\,\mathbf{h}_i\right), \quad (8)$$

where $\mathcal{N}_r(i)$ denotes the set of node i's neighbors under relation r, $c_{i,r}$ is a normalization constant (e.g., $|\mathcal{N}_r(i)|$), $\mathbf{W}_r$ and $\mathbf{W}_0$ are learnable weight matrices, and $\sigma(\cdot)$ is an activation function (e.g., ReLU).

A DGCNN-based [17] refinement module $\psi(\cdot)$ (with dilation factor d) further processes the aggregated features:

$$\mathbf{L} = \psi\big(\mathbf{F}^{(d)}\big) \in \mathbb{R}^{M\times C}, \quad (9)$$

where $\mathbf{L}$ represents the dynamic relational graph features and C is the feature dimension.

To incorporate time-dependent alignment, we sample a noise vector:

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad (10)$$

and define the noisy ground truth state at time t as

$$\mathbf{G}^{(t)} = (1-t)\mathbf{G} + t\boldsymbol{\epsilon}, \quad t \in [0, 0.02]. \quad (11)$$

An MLP-based alignment module $\mathcal{A}$ fuses the noisy state $\mathbf{G}^{(t)}$ with the heterogeneous features $\mathbf{L}$ to produce the aligned latent state:

$$\hat{\mathbf{L}}^{(t)} = \mathcal{A}\big(\mathbf{G}^{(t)}, \mathbf{L}; \Theta_\mathcal{A}\big) \in \mathbb{R}^{M\times C}. \quad (12)$$

B. Transformer-Based Flow Matching With Velocity Loss

Following the flow matching paradigm [25], we aim to learn a time-dependent velocity field $v_\theta(\hat{\mathbf{L}}^{(t)}, t)$ that guides the state toward clean geometry. The target velocity field is defined as

$$v_{\text{true}} = \frac{d\mathbf{G}^{(t)}}{dt} = \frac{d}{dt}\Big((1-t)\mathbf{G} + t\boldsymbol{\epsilon}\Big). \quad (13)$$

At each time step t, the current aligned state $\hat{\mathbf{L}}^{(t)}$ is passed through a transformer module $\mathcal{T}$ to capture long-range dependencies:

$$\tilde{\mathbf{L}}^{(t)} = \mathcal{T}\big(\hat{\mathbf{L}}^{(t)}\big), \quad (14)$$

and then fed to a displacement decoder $\mathcal{D}'$ that predicts the 3D velocity:

$$v_{\text{pred}} = v_\theta\big(\hat{\mathbf{L}}^{(t)}, t\big) = \mathcal{D}'\Big(\tilde{\mathbf{L}}^{(t)}, t\Big). \quad (15)$$

We update the state via Euler's method:

$$\hat{\mathbf{G}}^{(t_{\text{next}})} = \hat{\mathbf{G}}^{(t)} + (t_{\text{next}} - t)\, v_{\text{pred}} \quad (16)$$

The training objective is to minimize the mean squared error (MSE) between the predicted velocity and the target velocity:

$$\mathcal{L}_{\text{velocity}} = \mathbb{E}_{t\sim\mathcal{U}[0,1]}\Big[\|v_\theta(\hat{\mathbf{L}}^{(t)}, t) - v_{\text{true}}\|_2^2\Big]. \quad (17)$$

As for inference (Algorithm 1), the predicted velocity can be applied in one (or fewer) steps to generate the final upsampled point cloud.

Overall, our method integrates heterogeneous graph feature construction with a flow matching denoising framework. The velocity predictor v_θ, leveraging transformer-based encoding and a displacement decoder, produces incremental displacements that gradually transform the aligned noisy state toward the clean geometry.

In summary, we model flow matching as learning a time-dependent velocity field $v_\theta(\hat{\mathbf{L}}^{(t)}, t)$ that transforms the aligned noisy state toward clean geometry. Minimizing the MSE between the predicted and target velocities ensures accurate recovery of the true density and structure of $\mathbf{G}$.

Algorithm 1: Inference.

Input: $\mathbf{X} \in \mathbb{R}^{N\times 3}$, max time $t^{\max}$, number of steps T
$\mathbf{Y} \leftarrow f_{\text{interp}}(\mathbf{X})$ *mid interpolation*
$\mu_d, \sigma_d, d^{\max}, d^{\min} \leftarrow \mathbf{Y}$ *density statistics*
$\tau \leftarrow \mathcal{M}_\theta(\mu_d, \sigma_d, d^{\max}, d^{\min})$ *dense / sparse threshold*
$\mathbf{r} \leftarrow \mathcal{B}(\tau, \mathbf{d})$ *relation type*
$\mathbf{L} \leftarrow \psi\big(\mathcal{F}(\mathbf{Y}, \mathbf{r})\big)$ *refinement module*
$\hat{\mathbf{L}}^{(0)} \leftarrow \mathcal{A}(\mathbf{Y}, \mathbf{L})$ *alignment module*
$\hat{\mathbf{G}}^{(0)} \leftarrow \mathbf{Y}$ *initialize state*
for $t = 0$ **to** $t^{\max}(1 - 1/T)$ **do**
  $t_{\text{next}} = t + t^{\max}/T$
  $\tilde{\mathbf{L}}^{(t)} \leftarrow \mathcal{T}\big(\hat{\mathbf{L}}^{(t)}\big)$ *transformer module*
  $v_{\text{pred}} \leftarrow \mathcal{D}'\big(\tilde{\mathbf{L}}^{(t)}, t\big)$ *predicted velocity*
  $\hat{\mathbf{G}}^{(t_{\text{next}})} \leftarrow \hat{\mathbf{G}}^{(t)} + (t_{\text{next}} - t)\, v_{\text{pred}}$ *Euler step*
end for
Output: Final state $\hat{\mathbf{G}}^{(t^{\max})}$

IV. Experiments

A. Experimental Settings

Datasets: We evaluate our method on four benchmark datasets. PUGAN dataset [9], PU1K dataset [7], KITTI dataset [1], and a private data-CAMPUS. The first two are two well-established synthetic datasets with various geometric structures, which contain 24,000 and 69,000 patches for training respectively. To evaluate the robustness of methods on real-scan lidar data across varying ranges or scales, we conduct experiments on the KITTI dataset [1] with a focus on different distant regions for upsampling real-time LiDAR scans, and the CAMPUS dataset with a focus on different scale grids for consolidating mapping results.

For the KITTI dataset, we partitioned the raw KITTI LiDAR point clouds into four distinct datasets corresponding to near (<25 m), medium (25–50 m), far (>50 m), and global (all points) regions. In each dataset, input samples were generated via farthest point sampling while a dense set of ground truth points were directly captured by the sensor. To further analyze the density characteristics of the KITTI dataset, we examine the distribution of the 5-nearest neighbor (kNN) distances across four distant regions: near, medium, far, and global. As illustrated in Fig. 4, the mean and standard deviation of the kNN distances increase progressively from the near region (mean = 0.442, std = 0.204) to the medium (mean = 0.487, std = 0.381) and far regions (mean = 0.621, std = 0.585), with the global region exhibiting the highest values (mean = 1.120, std = 0.673). This trend reflects the inherent sparsity and non-uniformity of real-world LiDAR point clouds, where points become increasingly dispersed with distance from the sensor. The broader spread and higher mean in the global region underscore the challenge of upsampling under highly variable density conditions. These observations highlight

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

(a) (b) (c)

Fig. 5. (a) Real aerial view of the main building and its environment. (b) The autonomous vehicle used for data collection, equipped with a Livox Mid-360 LiDAR sensor. (c) The 3D point cloud reconstruction result, including a main building and surrounded paths.

TABLE II
COMPARISON ACROSS DIFFERENT REGIONS ON KITTI DATASET. METRICS ARE CD AND HD. BOLD TEXT AND UNDERLINE INDICATE THE BEST AND SECOND-BEST PERFORMANCE.

Ranges	Near		Medium		Far		Global	
Methods	CD↓	HD↓	CD↓	HD↓	CD↓	HD↓	CD↓	HD↓
PU-Net [7]	0.417	17.640	1.154	67.006	2.271	107.432	4.264	107.286
MPU [8]	0.326	14.650	1.092	78.110	1.972	110.399	3.897	107.369
PU-GAN [9]	0.248	9.611	0.780	46.228	1.534	96.333	2.862	99.353
PU-GCN [10]	0.269	10.353	1.059	58.111	2.132	99.098	3.739	94.298
Grad-PU [11]	0.153	9.369	0.577	**46.233**	1.131	95.254	1.516	91.364
PUDM [13]	0.150	9.297	0.561	46.656	1.126	94.972	1.507	90.827
PURF	**0.149**	**9.260**	**0.556**	46.288	**1.063**	**93.283**	**1.480**	**90.746**

TABLE III
COMPARISON ACROSS DIFFERENT SCALES ON CAMPUS DATASET. METRICS ARE CD AND HD. BOLD TEXT AND UNDERLINE INDICATE THE BEST AND SECOND-BEST PERFORMANCE.

Ranges	5x5		10x10		15x15		20x20		25x25	
Metrics	CD↓	HD↓	CD↓	HD↓	CD↓	HD↓	CD↓	HD↓	CD↓	HD↓
PU-Net [7]	1.807	63.921	0.513	34.778	0.264	22.265	0.167	17.666	0.121	15.258
MPU [8]	1.883	73.848	0.537	35.493	0.275	24.290	0.173	19.689	0.126	16.337
PU-GAN [9]	1.448	52.103	0.425	26.610	0.224	19.585	0.141	15.421	0.104	13.484
PU-GCN [10]	1.543	51.068	0.440	28.532	0.227	19.154	0.140	16.074	0.103	13.857
Grad-PU [11]	0.772	50.055	0.242	26.850	0.128	13.874	0.087	11.876	0.063	9.391
PUDM [13]	0.816	**48.388**	0.329	26.261	0.202	**13.338**	0.144	11.818	0.117	9.306
PURF	**0.752**	50.431	**0.231**	**25.958**	**0.121**	13.660	**0.081**	**11.780**	**0.056**	**9.246**

TABLE IV
COMPARISON OF TRAINING AND INFERENCE COSTS ON PUGAN DATASET. BOLD TEXT AND UNDERLINE INDICATE THE BEST AND SECOND-BEST PERFORMANCE.

Stages		Inference		Training
Metrics	Params.(kb)↓	Memory(MB)↓	Time(s)↓	Time(hrs)↓
Grad-PU [11]	67.1	**1959**	0.407	6
PUDM [13]	16414.7	13137	1.551	96
PURF	**46.0**	2075	**0.363**	**3**

the necessity for density-aware upsampling strategies capable of robustly handling such diverse spatial distributions.

For the CAMPUS dataset, we sampled input and ground truth point clouds by partitioning a raw point cloud (52,772,410 points) into $n_{grids} \times n_{grids}$ grids ($n_{\text{grids}} \in \{5, 10, 15, 20, 25\}$). As demonstrated in Fig. 5, the raw point cloud was collected by a Livox Mid-360 LiDAR sensor, which is reconstructed based on FAST-LIO2 [27]. We constructed this dataset by sampling 8192 points from each grid as ground truth and 2048 points as input by furthest point sampling, to obtain different scales of point clouds with varied densities.

Baselines: We retrained PU-Net [7], PU-GAN [9], PU-GCN [10], MPU [8], Grad-PU [11], and PUDM [13] on the PUGAN and PU1K dataset. The settings followed the original papers. We used the same metrics for all the methods.

Metrics: Chamfer Distance (CD), Hausdorff Distance (HD), and Point-to-Surface Distance (P2F) are used to assess the performance of the methods following [10].

Implementation: We implemented methods based on a single NVIDIA RTX 3090ti GPU with 24 GB memory. Datasets and detailed hyperparameters are available on https://github.com/yuzhong-deng/PURF/.

V. RESULTS

The proposed PURF framework is evaluated across synthetic and real-world benchmarks, demonstrating its effectiveness in addressing density heterogeneity and achieving high computational efficiency. Below, we analyze results through four critical dimensions: synthetic benchmark performance, real-scan generalization, computational efficiency, and ablation study.

A. Synthetic Datasets

Table I summarizes the quantitative performance of several state-of-the-art upsampling methods on synthetic datasets. Notably, our proposed PURF framework achieves the best or competitive performance across both datasets. For instance, on the PUGAN dataset, PURF attains the best metrics among the compared methods in terms of CD and HD distances. Similarly,

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

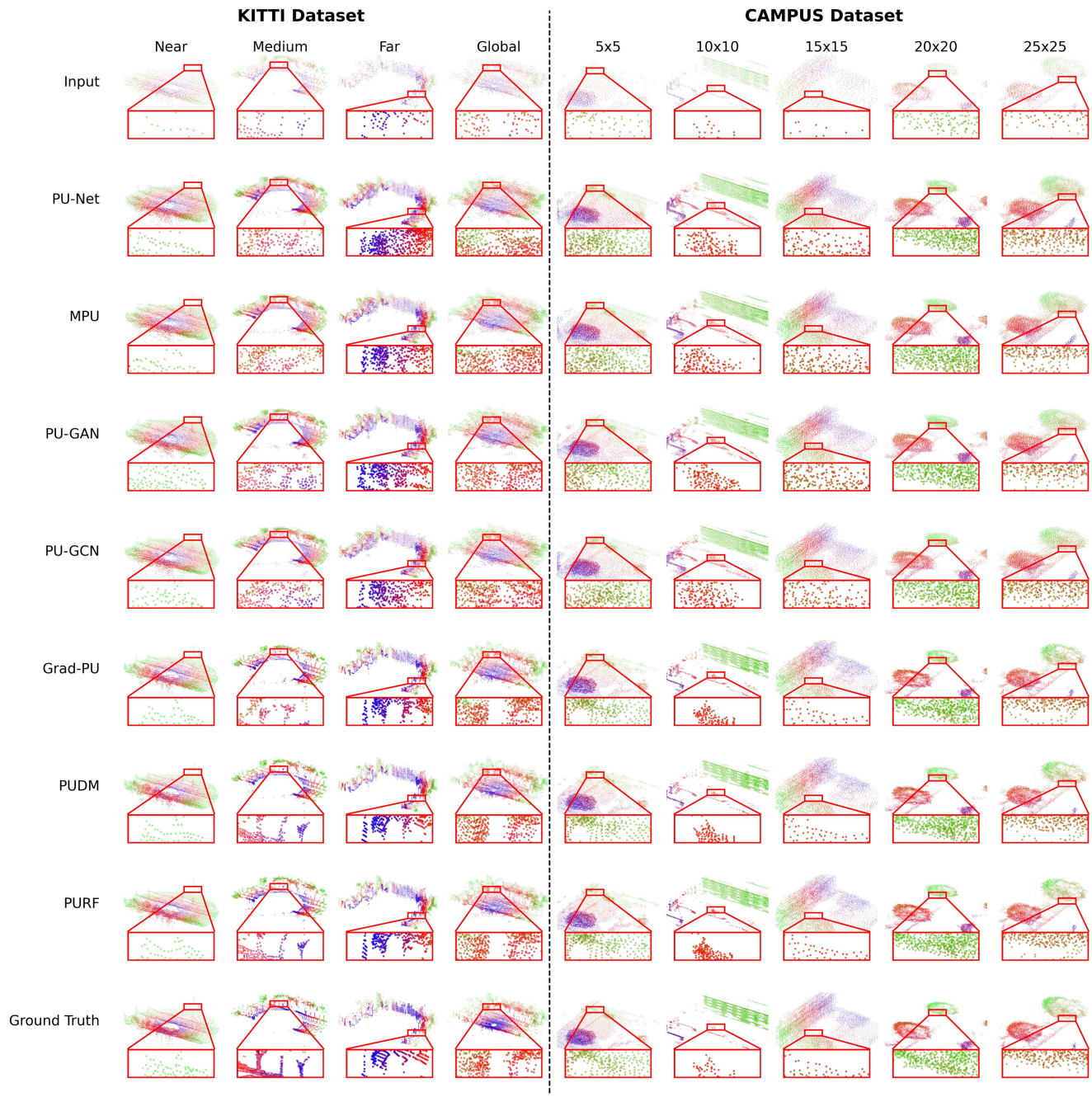

Fig. 6. Qualitative comparison of upsampling on KITTI regions and CAMPUS scales. PURF exhibits fewer outliers and smoother surfaces, preserving fine geometric details even in challenging areas.

on the PU1K dataset, PURF exhibits surpassing or matching the performance of existing approaches. Note that the P2F metric is inherently asymmetric—measuring distances only from the upsampled points to the ground-truth surface and not in the reverse direction. Consequently, the symmetric CD and HD provide a more balanced and informative evaluation [11].

These results indicate that PURF's heterogeneous graph-driven design and flow matching strategy effectively capture fine-grained geometric structures. The lower error metrics suggest that our method not only reconstructs dense point clouds more accurately but also preserves geometric detail better than conventional homogeneous methods. Overall, these findings validate the efficiency and robustness of PURF for point cloud upsampling across diverse spatial scales in real-world scenarios.

B. *Real-Scanned Results*

Table II further details the performance of various upsampling methods across four distant regions of the KITTI dataset. Notably, the density distribution in the point clouds varies significantly with distance: points in the near region are relatively dense, whereas those in the far and global regions become increasingly sparse. This inherent non-uniformity poses an additional challenge for out-of-distribution (OOD) evaluation, as

the model parameters are trained exclusively on the PU1K dataset. Despite these distributional shifts, our proposed PURF framework achieves superior performance, achieving lower CD and HD values across all regions. It is worth noting that the performance gap between PURF and other methods widens in the far and global regions, where the density of points is lower. This not only demonstrates the robustness of our heterogeneous graph-driven design and flow matching strategy but also highlights its impressive out-of-distribution generalization capabilities, when applied to real-scan data with varying density profiles. Because there are no ground truth meshes for the dataset, we did not include the P2F metric in this case.

To further evaluate the robustness of the proposed method under varying densities of different scales, we conduct an analysis by dividing the input point cloud into block grids of increasing granularity: 5×5, 10×10, 15×15, 20×20, and 25×25. As the division becomes finer, each predicted patch covers a smaller field of view and should infer more intricate local structures, which poses a denser distribution for upsampling models. As shown in Table III, PURF outperforms state-of-the-art methods (Grad-PU and PUDM) in terms of CD across all spatial ranges, demonstrating superior point-wise accuracy. While Grad-PU shows competitive HD in the coarsest grid (5×5), it degrades as the granularity increases. PUDM exhibits strong HD in medium-scale ranges, but its CD performance lags significantly, particularly at finer scales. In contrast, PURF not only maintains the lowest CD across all scales—indicative of faithful point reconstruction—but also achieves highly competitive HD values, especially in high-resolution grids like 25×25. These results highlight PURF's capacity to generalize effectively under limited receptive fields, and its ability to preserve global structure while refining local geometry.

Fig. 6 presents qualitative upsampling results for multiple methods across various distance and scale. Conventional upsampling methods, including PU-Net, MPU, PU-GAN, and PU-GCN, tend to generate noticeable outliers and less continuous surface geometry, with these issues becoming more pronounced in sparser or more finely partitioned regions. Although Grad-PU and PUDM display relatively enhanced structure and fewer artifacts, they are not entirely free from inconsistencies, especially in the Far and Global views. PURF, on the other hand, demonstrates a trend toward preserving surface continuity and minimizing noise, maintaining visually coherent reconstructions even in challenging distant and low-scale scenarios. This suggests that PURF's heterogeneous, density-aware graph design would offer benefits for upsampling under varying spatial distributions.

C. Computational Efficiency

Table IV provides a comparison of the computational costs associated with recent upsampling methods on the PUGAN dataset. Notably, our proposed PURF framework achieves substantial efficiency gains across all measured dimensions. Specifically, PURF maintains the lowest parameter count at 46.0 kb—significantly below Grad-PU's 67.1 kb and dramatically less than PUDM's 16,414.7 kb. In terms of inference, PURF achieves the fastest runtime (0.363 s) and competitive memory usage (2075 MB), while also reducing training time to just 3 hours, compared to 6 hours for Grad-PU and a prohibitive 96 hours for PUDM. These results underscore PURF's suitability for real-time or resource-constrained deployment scenarios.

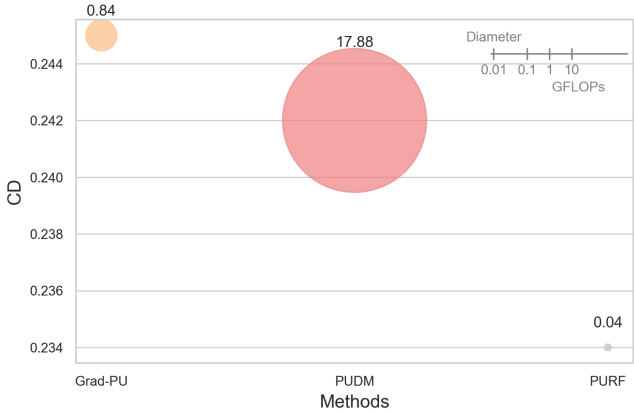

Fig. 7. GFLOPs vs. CD for different upsampling methods on PUGAN dataset. Bubble diameter is proportional to GFLOPs. PURF achieves the best trade-off, combining the lowest computational cost with the highest accuracy.

TABLE V
ABLATION ON PUGAN. METRICS ARE CD ($\times 10^{-3}$), HD ($\times 10^{-3}$), AND P2F ($\times 10^{-3}$). BOLD TEXT AND UNDERLINE INDICATE THE BEST AND SECOND-BEST PERFORMANCE.

Ablation Settings	CD↓	HD↓	P2F↓	Infer. time (s)↓
Graph+Diffusion	0.284	1.571	2.510	0.647
Relational Graph+Diffusion	0.256	1.466	<u>2.341</u>	0.675
Graph+Flow	0.237	<u>1.426</u>	2.412	**0.253**
Concat Graph+Flow	<u>0.236</u>	1.457	2.397	<u>0.276</u>
Relational Graph+Flow	**0.234**	**1.357**	**2.215**	0.277

To further illustrate the relationship between computational complexity and upsampling accuracy, we visualize the trade-off between GFLOPs (floating-point operations) and CD on PUGAN dataset (Fig. 7). In this bubble plot, each method is represented by a bubble whose diameter is proportional to its GFLOPs, with the vertical axis indicating CD and the horizontal axis denoting the method. PURF is positioned at the lower right, achieving both the lowest computational cost and the best accuracy, while PUDM incurs orders of magnitude higher computational burden with no corresponding gain in performance. This visualization highlights the remarkable efficiency of PURF, which delivers state-of-the-art results with minimal computational overhead.

D. Ablation Study

Table V disentangles the effects of the graph representation (homogeneous vs. relational) and the sampling paradigm (diffusion vs. flow matching), within the same PURF framework. Relational graphs help regardless of the sampler. Replacing a homogeneous graph with a density-aware relational graph improves CD/HD/P2F under both diffusion and flow matching, indicating that edge-level modeling of dense/sparse interactions is beneficial. Edge-level relations outperform node concatenation. At the same sampling steps and similar inference time, Relational Graph+Flow outperforms Concat Graph+Flow, showing that simply appending density to node features (homogeneous message passing) is weaker than relation-specific message passing with R-GCN. Flow matching reduces latency at matched

steps. Under the same number of steps (20) following a DDPM-style variance-preserving process with a linear-β schedule from 0 to 0.02, flow matching (Euler) attains lower inference latency and better HD/P2F, consistent with the fact that flow matching effectively predicts a velocity field that can be integrated in a few ODE steps, avoiding the long denoising chain required by diffusion.

VI. Conclusion

In this letter, we introduced PURF, a density-aware point cloud upsampling framework that leverages relational graph flow matching to tackle non-uniform density and sparsity in real-world 3D data. By combining relational graph construction with transformer-based flow matching, PURF captures local geometric and density variations and enables accurate upsampling on both synthetic and real-scanned datasets. Experiments show that PURF achieves state-of-the-art fidelity and efficiency, especially on highly sparse or out-of-distribution cases. Ablation studies further validate the effectiveness of each component and the synergy between relational graphs and flow matching. Overall, PURF offers an efficient solution for point cloud upsampling and a viable basis for scalable 3D geometric learning.

References

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, pp. 1231–1237, 2013.

[2] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[3] B. Fei et al., "Comprehensive review of deep learning-based 3D point cloud completion processing and analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 22862–22883, Dec. 2022.

[4] Z. Zhao, W. Zhang, J. Gu, J. Yang, and K. Huang, "LiDAR mapping optimization based on lightweight semantic segmentation," *IEEE Trans. Intell. Veh.*, vol. 4, no. 3, pp. 353–362, Sep. 2019.

[5] H. Li et al., "DAUP: Enhancing point cloud homogeneity for 3D industrial anomaly detection via density-aware point cloud upsampling," *Adv. Eng. Inform.*, vol. 62, Oct. 2024, Art. no. 102823.

[6] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, and M. Wimmer, "Continuous projection for fast L1 reconstruction," *ACM Trans. Graph.*, vol. 33, no. 4, Jul. 2014, Art. no. 47.

[7] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-Net: Point cloud upsampling network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2790–2799.

[8] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3D point set upsampling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 5951–5960.

[9] R. Li, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-GAN: A point cloud upsampling adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7202–7211.

[10] G. Qian, A. Abualshour, G. Li, A. Thabet, and B. Ghanem, "PU-GCN: Point cloud upsampling using graph convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 11683–11692.

[11] Y. He, D. Tang, Y. Zhang, X. Xue, and Y. Fu, "GradPU: Arbitrary-scale point cloud upsampling via gradient descent with learned distance functions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Vancouver, BC, Canada, 2023, pp. 5354–5363.

[12] Y. Bai, X. Wang, M. H.A. Jr, and D. Rus, "BIMS-PU: Bi-directional and multi-scale point cloud upsampling," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7447–7454, Jul. 2022.

[13] W. Qu, Y. Shao, L. Meng, X. Huang, and L. Xiao, "A conditional denoising diffusion probabilistic model for point cloud upsampling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 20786–20795.

[14] Z. Lyu, Z. Kong, X. Xu, L. Pan, and D. Lin, "A conditional point diffusion-refinement paradigm for 3D point cloud completion," in *Proc. Int. Conf. Learn. Representations*, 2022, pp. 6254–6277.

[15] S. Mo et al., "DiT-3D: Exploring plain diffusion transformers for 3D shape generation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2023, pp. 67960–67971.

[16] C. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.

[17] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, pp. 1–12, 2018.

[18] W. Shi and R. R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 1708–1716.

[19] X. Wang, D. Bo, C. Shi, S. Fan, Y. Ye, and P. S. Yu, "A survey on heterogeneous graph embedding: Methods, techniques, applications and sources," *IEEE Trans. Big Data*, vol. 9, no. 2, pp. 415–436, Apr. 2023.

[20] M. Schlichtkrull, T. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. Extended Semantic Web Conf.*, 2017, pp. 593–607.

[21] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst.*, Red Hook, NY, USA, Curran Associates Inc., 2020, pp. 6840–6851.

[22] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 240–275.

[23] R. Zhang, D. Xue, Y. Wang, R. Geng, and F. Gao, "Towards dense and accurate radar perception via efficient cross-modal diffusion model," *IEEE Robot. Automat. Lett.*, vol. 9, no. 9, pp. 7429–7436, Sep. 2024.

[24] A. Mao, Z. Du, J. Hou, Y. Duan, Y.-J. Liu, and Y. He, "PU-Flow: A point cloud upsampling network with normalizing flows," *IEEE Trans. Vis. Comput. Graph.*, vol. 29, no. 12, pp. 4964–4977, Dec. 2023.

[25] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *Proc. Int. Conf. Learn. Representations*, 2023, pp. 1–28.

[26] Y. Bengio, N. Léonard, and A. C. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," 2013, *arXiv:1308.3432*.

[27] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.