Self-supervised Geometry Reasoning for LiDAR Simultaneous Localization and Mapping

Jiwoo Kim[1,*] Jinwoo Lee[1,*] Woojae Shin[1] Giseop Kim[2] Hyondong Oh[1,†]

[1]Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea

[2]Daegu Gyeongbuk Institute of Science and Technology (DGIST), Daegu, Republic of Korea

[*]Equal contribution. [†]Corresponding author.

Emails: {tars0523, jinwoolee, oj7987, h.oh}@kaist.ac.kr, gsk@dgist.ac.kr

***Abstract*—LiDAR simultaneous localization and mapping (SLAM) relies on local geometric quantities such as covariances, correspondences, and surface structures. However, most existing pipelines rely on hand-crafted estimates of local geometry and use them as fixed inputs to LiDAR SLAM, which can make the estimated local geometry noisy and unstable in sparse regions of a point cloud or when using low-resolution LiDAR. To address this issue, this paper introduces a self-supervised framework that learns an explicit symbolic representation of local geometry and uses it to improve LiDAR SLAM recursively. Specifically, each point is represented as a Gaussian distribution, allowing local geometry to be described by a covariance. Without dense geometry labels or ground-truth poses, the framework learns by maximizing the likelihood of local geometry, with self-supervision derived from consistency relations over symbolic geometric representations, including predicted covariances, correspondences, and trajectory from SLAM. The learned geometry is then fed back into LiDAR SLAM, forming a reciprocal loop in which improved geometry enhances localization and mapping, and improved localization provides cleaner supervision for subsequent geometry reasoning. This framework is backend-agnostic and can be plugged into existing LiDAR SLAM pipelines without architectural changes. Experiments on KITTI under varying LiDAR resolutions show that the proposed method improves both odometry and global registration.**

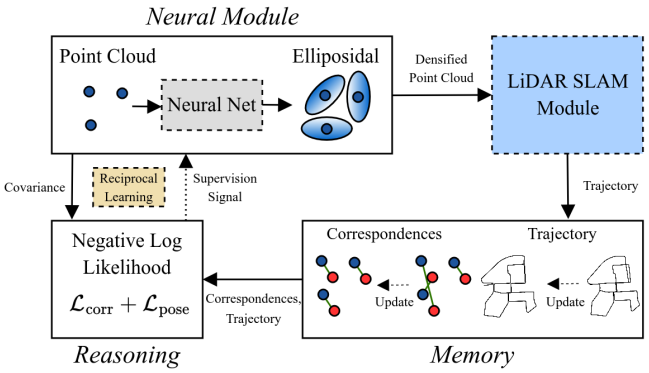


Fig. 1: Overview of the proposed self-supervised geometry reasoning framework. The neural module infers point-wise covariance as local geometry and samples additional points with this covariance to produce a densified point cloud for LiDAR SLAM. The reasoning module provides likelihood-based self-supervision from correspondences and trajectory in memory, while the updated SLAM trajectory is written back to memory, forming a reciprocal loop between geometry reasoning and SLAM.

I. Introduction

LiDAR simultaneous localization and mapping (SLAM) is a fundamental technology that enables robots to explore, localize, and interact with previously unknown environments [1]. A common principle behind many LiDAR SLAM modules is the use of local geometric information extracted from point clouds. For example, LiDAR odometry often relies on local covariance structures to achieve accurate scan matching [18], while global registration methods exploit surface normals and local geometric consistency to establish reliable correspondences across distant scans [8].

However, most existing pipelines estimate such local geometry using hand-crafted neighborhood statistics and then use it as a fixed input to subsequent optimization. These estimates can become noisy and unstable in sparse regions of a point cloud or under low-resolution LiDAR observations, making the overall SLAM performance highly dependent on the environment and sensor characteristics.

In this paper, we move beyond this perspective and propose a self-supervised geometric reasoning framework that recursively infers scene geometry from raw LiDAR observations and leverages it for localization and mapping, without requiring dense geometry labels or ground-truth trajectories. Our formulation is inspired by imperative learning [14], which introduces a general self-supervised neuro-symbolic framework where neural module, symbolic reasoning, and memory are reciprocally optimized through bilevel optimization. In our formulation, the neural module predicts local geometry from raw LiDAR observations, while LiDAR SLAM uses this information to estimate the trajectory. The resulting trajectory, in turn, serves as an evolving memory, providing more reliable poses and correspondences for the next round of geometry reasoning. This reciprocal interaction between the neural module, reasoning module, memory, and SLAM backend is summarized in Fig. 1. The proposed framework is backend-agnostic, enabling learned geometric priors to improve existing LiDAR SLAM pipelines in a self-supervised manner.

II. Related Works

A. Underlying Geometry Estimation from Point Clouds

Estimating the underlying geometry of point clouds is essential for geometric perception [4]. Classical methods typically use hand-crafted neighborhood statistics, such as PCA-based covariance analysis [11] and surface normal estimation [10]. Although efficient, these methods heavily depend on local sampling quality and neighborhood selection, making them fragile under sparse, noisy, and non-uniform LiDAR measurements [9].

Learning-based methods have been proposed to infer richer geometric structures, often by densifying sparse point clouds or reconstructing complete surfaces [13, 15]. However, these approaches usually require dense ground-truth geometry for supervision and may hallucinate unobserved structures that are not supported by the actual measurements [15]. Unlike full-scene completion, our work aims to infer local underlying geometry around observed points, providing a compact, uncertainty-aware, and physically grounded representation for geometric perception.

B. Self-supervised Neuro-Symbolic Learning

Neuro-symbolic learning combines neural perception with symbolic reasoning to improve interpretability and generalization [14]. In robotics, symbolic reasoning can include not only logical rules [14] but also physical principles and geometric optimization, such as bundle adjustment [17] and pose graph optimization [2]. Recent imperative learning frameworks [14] use such reasoning engines as sources of self-supervision, allowing neural modules to learn without explicit labels.

Our work instantiates imperative learning [14] for point cloud geometry reasoning in LiDAR SLAM. We specialize the reasoning component as a maximum likelihood estimation (MLE) framework for local geometry, where point-wise covariances are learned to explain two types of SLAM consistency: the residuals between corresponding points and the relative poses estimated by the SLAM backend. In this formulation, the neural module predicts Gaussian local geometry, the memory stores SLAM-estimated poses and correspondences, and the likelihoods provide self-supervision by enforcing consistency between them. The resulting local geometry is then fed back into the SLAM pipeline to improve downstream tasks such as odometry and global registration, and the refined SLAM outputs in turn provide cleaner memory for subsequent geometry reasoning. This forms a reciprocal structure in which local geometry and SLAM consistency progressively reinforce each other.

III. Probabilistic Modeling of Underlying Geometry as a Statistical Manifold

To formalize this geometry reasoning, we first introduce a probabilistic representation of the underlying geometry [7].

Definition 1 (Statistical Manifold of World Geometry)**.** *The statistical manifold $\mathcal{M}_{\boldsymbol{g}}$ represents the underlying world surface manifold through a probabilistic approximation in a global reference frame. The manifold $\mathcal{M}_{\boldsymbol{g}}$ is defined as a family of 3D Gaussian distributions, each parameterized by a mean $\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} \in \mathbb{R}^3$ and a covariance matrix $\boldsymbol{C}_{\boldsymbol{g}_i} \in \mathbb{S}^3_+$. Each Gaussian distribution defines the probability density that an observed point $\boldsymbol{x}_i$ is measured with respect to a local tangent plane at $\bar{\boldsymbol{x}}_{\boldsymbol{g}_i}$.*

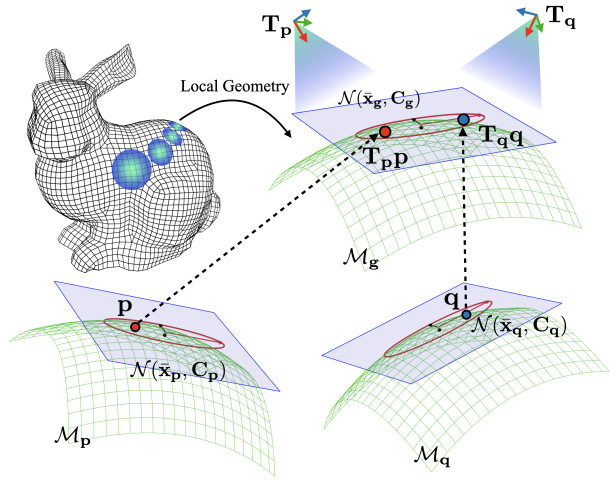


Fig. 2: A visualization illustrating how the underlying geometry can be represented as a statistical manifold. It explains that two points observed in different coordinate frames can be regarded as samples drawn from the same underlying distribution.

When $\boldsymbol{x}_i$ is acquired from a high-precision LiDAR sensor, the resulting Gaussian distribution approximates the local planar geometry. In this formulation, the covariance matrix $\boldsymbol{C}_{\boldsymbol{g}_i}$ provides a quantitative representation of the geometry in the neighborhood of $\bar{\boldsymbol{x}}_{\boldsymbol{g}_i}$, as shown in Fig. 2.

Proposition 1 (Statistical Manifold Representation from Different Sensor Frames)**.** *Given the sensor pose $\boldsymbol{T_p} = (\boldsymbol{R_p}, \boldsymbol{t_p}) \in SE(3)$ relative to the global reference frame, the parameters $\bar{\boldsymbol{x}}_{\boldsymbol{p}_i}$ and $\boldsymbol{C}_{\boldsymbol{p}_i}$ of the local statistical manifold $\mathcal{M}_{\boldsymbol{p}}$ are defined by transforming the global model $\mathcal{M}_{\boldsymbol{g}}$ into the sensor frame via $\boldsymbol{T}_{\boldsymbol{p}}^{-1}$:*

$$\bar{\boldsymbol{x}}_{\boldsymbol{p}_i} \triangleq \boldsymbol{R}_{\boldsymbol{p}}^\top(\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} - \boldsymbol{t_p}), \qquad \boldsymbol{C}_{\boldsymbol{p}_i} \triangleq \boldsymbol{R}_{\boldsymbol{p}}^\top \boldsymbol{C}_{\boldsymbol{g}_i} \boldsymbol{R_p}.$$

Analogously, for a second observation frame with pose $\boldsymbol{T_q} = (\boldsymbol{R_q}, \boldsymbol{t_q}) \in SE(3)$, the parameters $\bar{\boldsymbol{x}}_{\boldsymbol{q}_i}$ and $\boldsymbol{C}_{\boldsymbol{q}_i}$ of the statistical manifold $\mathcal{M}_{\boldsymbol{q}}$ are defined via $\boldsymbol{T}_{\boldsymbol{q}}^{-1}$:

$$\bar{\boldsymbol{x}}_{\boldsymbol{q}_i} \triangleq \boldsymbol{R}_{\boldsymbol{q}}^\top(\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} - \boldsymbol{t_q}), \qquad \boldsymbol{C}_{\boldsymbol{q}_i} \triangleq \boldsymbol{R}_{\boldsymbol{q}}^\top \boldsymbol{C}_{\boldsymbol{g}_i} \boldsymbol{R_q}.$$

Theorem 1 (Residual Distribution)**.** *Let $\mathcal{P} \triangleq (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N)$ and $\mathcal{Q} \triangleq (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_N)$ denote two scans acquired from frames $\boldsymbol{T_p}$ and $\boldsymbol{T_q}$, respectively, with known correspondences $(\boldsymbol{p}_i, \boldsymbol{q}_i)$. Each point is independently sampled as a noisy observation from its local Gaussian distribution, i.e., $\boldsymbol{p}_i \sim \mathcal{N}(\bar{\boldsymbol{x}}_{\boldsymbol{p}_i}, \boldsymbol{C}_{\boldsymbol{p}_i})$ and $\boldsymbol{q}_i \sim \mathcal{N}(\bar{\boldsymbol{x}}_{\boldsymbol{q}_i}, \boldsymbol{C}_{\boldsymbol{q}_i})$, with per-point covariances $\boldsymbol{C_p} \triangleq (\boldsymbol{C}_{\boldsymbol{p}_1}, \ldots, \boldsymbol{C}_{\boldsymbol{p}_N})$ and $\boldsymbol{C_q} \triangleq (\boldsymbol{C}_{\boldsymbol{q}_1}, \ldots, \boldsymbol{C}_{\boldsymbol{q}_N})$. Since $\mathcal{P}$ and $\mathcal{Q}$ originate from manifolds related by an isometric rigid-body transformation of the same underlying world surface,*

they share identical local geometry, i.e., $\boldsymbol{C}_{\boldsymbol{g}_i}$. Under this local geometric model, the displacement $\boldsymbol{d}_i$ for the i-th correspondence, defined by the relative transformation $\boldsymbol{T} \triangleq \boldsymbol{T}_{\boldsymbol{q}}^{-1}\boldsymbol{T}_{\boldsymbol{p}}$, satisfies

$$\boldsymbol{d}_i \triangleq \boldsymbol{q}_i - \boldsymbol{T}\boldsymbol{p}_i, \qquad \boldsymbol{d}_i \sim \mathcal{N}(\boldsymbol{0},\, 2\boldsymbol{C}_{\boldsymbol{q}_i})\,. \tag{1}$$

Proof. Let $\boldsymbol{T} = (\boldsymbol{R}, \boldsymbol{t}) \triangleq \boldsymbol{T}_{\boldsymbol{q}}^{-1}\boldsymbol{T}_{\boldsymbol{p}}$, where $\boldsymbol{R} = \boldsymbol{R}_{\boldsymbol{q}}^\top \boldsymbol{R}_{\boldsymbol{p}}$ and $\boldsymbol{t} = \boldsymbol{R}_{\boldsymbol{q}}^\top(\boldsymbol{t}_{\boldsymbol{p}} - \boldsymbol{t}_{\boldsymbol{q}})$. Using the Gaussian perturbation form, we write

$$\boldsymbol{p}_i = \boldsymbol{R}_{\boldsymbol{p}}^\top(\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} - \boldsymbol{t}_{\boldsymbol{p}}) + \boldsymbol{\epsilon}_{\boldsymbol{p}_i}, \qquad \boldsymbol{q}_i = \boldsymbol{R}_{\boldsymbol{q}}^\top(\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} - \boldsymbol{t}_{\boldsymbol{q}}) + \boldsymbol{\epsilon}_{\boldsymbol{q}_i},$$

where

$$\boldsymbol{\epsilon}_{\boldsymbol{p}_i} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_{\boldsymbol{p}_i}), \qquad \boldsymbol{\epsilon}_{\boldsymbol{q}_i} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_{\boldsymbol{q}_i}),$$

and the two perturbations are independent. Then,

$$\begin{aligned}\boldsymbol{d}_i &\triangleq \boldsymbol{q}_i - \boldsymbol{T}\boldsymbol{p}_i \\ &= \boldsymbol{R}_{\boldsymbol{q}}^\top(\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} - \boldsymbol{t}_{\boldsymbol{q}}) - \big(\boldsymbol{R}\boldsymbol{R}_{\boldsymbol{p}}^\top(\bar{\boldsymbol{x}}_{\boldsymbol{g}_i} - \boldsymbol{t}_{\boldsymbol{p}}) + \boldsymbol{t}\big) + \boldsymbol{\epsilon}_{\boldsymbol{q}_i} - \boldsymbol{R}\boldsymbol{\epsilon}_{\boldsymbol{p}_i}.\end{aligned}$$

Since $\boldsymbol{R} = \boldsymbol{R}_{\boldsymbol{q}}^\top \boldsymbol{R}_{\boldsymbol{p}}$ and $\boldsymbol{t} = \boldsymbol{R}_{\boldsymbol{q}}^\top(\boldsymbol{t}_{\boldsymbol{p}} - \boldsymbol{t}_{\boldsymbol{q}})$, the deterministic terms cancel, yielding

$$\boldsymbol{d}_i = \boldsymbol{\epsilon}_{\boldsymbol{q}_i} - \boldsymbol{R}\boldsymbol{\epsilon}_{\boldsymbol{p}_i}.$$

Therefore,

$$\mathbb{E}[\boldsymbol{d}_i] = \boldsymbol{0},$$

and, by independence (i.e., sampling independently),

$$\begin{aligned}\mathrm{Cov}[\boldsymbol{d}_i] &= \boldsymbol{C}_{\boldsymbol{q}_i} + \boldsymbol{R}\boldsymbol{C}_{\boldsymbol{p}_i}\boldsymbol{R}^\top \\ &= \boldsymbol{R}_{\boldsymbol{q}}^\top \boldsymbol{C}_{\boldsymbol{g}_i}\boldsymbol{R}_{\boldsymbol{q}} + \boldsymbol{R}_{\boldsymbol{q}}^\top \boldsymbol{R}_{\boldsymbol{p}}\left(\boldsymbol{R}_{\boldsymbol{p}}^\top \boldsymbol{C}_{\boldsymbol{g}_i}\boldsymbol{R}_{\boldsymbol{p}}\right)\boldsymbol{R}_{\boldsymbol{p}}^\top \boldsymbol{R}_{\boldsymbol{q}} \\ &= 2\boldsymbol{R}_{\boldsymbol{q}}^\top \boldsymbol{C}_{\boldsymbol{g}_i}\boldsymbol{R}_{\boldsymbol{q}} = 2\boldsymbol{C}_{\boldsymbol{q}_i}.\end{aligned}$$

Hence,

$$\boldsymbol{d}_i \sim \mathcal{N}(\boldsymbol{0},\, 2\boldsymbol{C}_{\boldsymbol{q}_i})\,,$$

which proves Eq. (1). □

Corollary 1 (Distribution of a Point)**.** *Given the relative transformation $\boldsymbol{T}$ and source point $\boldsymbol{p}_i$, the target point $\boldsymbol{q}_i$ can be represented as*

$$\boldsymbol{q}_i = \boldsymbol{T}\boldsymbol{p}_i + \boldsymbol{d}_i.$$

From Theorem 1, the residual follows

$$\boldsymbol{d}_i \sim \mathcal{N}(\boldsymbol{0}, 2\boldsymbol{C}_{\boldsymbol{q}_i}).$$

Therefore, conditioned on $\boldsymbol{p}_i$ and $\boldsymbol{T}$, the target point follows

$$\boldsymbol{q}_i \mid \boldsymbol{p}_i, \boldsymbol{T}, \boldsymbol{C}_{\boldsymbol{q}_i} \sim \mathcal{N}(\boldsymbol{T}\boldsymbol{p}_i, 2\boldsymbol{C}_{\boldsymbol{q}_i}).$$

Theorem 2 (Distribution of Point Cloud)**.** *Assuming the different correspondences are conditionally independent, the conditional likelihood of the target set $\mathcal{Q}$ given the fixed source set $\mathcal{P}$ factorizes as*

$$\begin{aligned}p(\mathcal{Q} \mid \mathcal{P}, \boldsymbol{T}, \boldsymbol{C}_{\boldsymbol{q}}) &= \prod_{i=1}^{N} p(\boldsymbol{q}_i \mid \boldsymbol{p}_i, \boldsymbol{T}, \boldsymbol{C}_{\boldsymbol{q}_i}) \\ &= \prod_{i=1}^{N} \frac{1}{\sqrt{(2\pi)^3\,|2\boldsymbol{C}_{\boldsymbol{q}_i}|}} \exp\!\left(-\frac{1}{2}\,\boldsymbol{d}_i^\top (2\boldsymbol{C}_{\boldsymbol{q}_i})^{-1}\boldsymbol{d}_i\right),\end{aligned} \tag{2}$$

where the last equality follows from Corollary 1.

IV. Maximum Likelihood Estimation of Geometry

In this section, we introduce two MLE methods for the local geometry (i.e., $\boldsymbol{C}_{\boldsymbol{q}_i}$).

A. Correspondence Likelihood

The first estimator follows directly from the likelihood in Theorem 2.

Proposition 2 (Maximize Correspondence Likelihood)**.** *The maximum likelihood estimator of $\boldsymbol{C}_{\boldsymbol{q}}$ from Eq.* (2) *is*

$$\hat{\boldsymbol{C}}_{\boldsymbol{q}} = \arg\max_{\boldsymbol{C}_{\boldsymbol{q}}}\; p(\mathcal{Q} \mid \mathcal{P}, \boldsymbol{T} = \boldsymbol{T}^\star, \boldsymbol{C}_{\boldsymbol{q}}). \tag{3}$$

We refer to this formulation as the maximization of correspondence likelihood, as it is defined over given pairs of corresponding points and characterizes the residual between them.

Taking the negative log-likelihood of Eq. (3) and discarding constants reduces the MLE to

$$\hat{\boldsymbol{C}}_{\boldsymbol{q}} = \arg\min_{\boldsymbol{C}_{\boldsymbol{q}}}\; \tfrac{1}{2}\sum_{i=1}^{N}\left[\log|2\boldsymbol{C}_{\boldsymbol{q}_i}| + \boldsymbol{d}_i^{\star\top}(2\boldsymbol{C}_{\boldsymbol{q}_i})^{-1}\boldsymbol{d}_i^\star\right], \tag{4}$$

where $\boldsymbol{d}_i^\star = \boldsymbol{q}_i - \boldsymbol{T}^\star\boldsymbol{p}_i$. In practice, the ground-truth pose $\boldsymbol{T}^\star$ is rarely available; instead, it can be obtained from LiDAR SLAM.

B. Pose Likelihood

The second estimator takes a different view. We ask: *given a candidate $\boldsymbol{C}_{\boldsymbol{q}}$, does the pose implied by the correspondence likelihood agree with the observed pose $\tilde{\boldsymbol{T}}$ (e.g., from LiDAR SLAM)?* To formalize this, we first define the pose implied by the correspondence likelihood for a given covariance set:

$$\boldsymbol{T}(\boldsymbol{C}_{\boldsymbol{q}}) \triangleq \arg\max_{\boldsymbol{T}}\; p(\mathcal{Q} \mid \mathcal{P}, \boldsymbol{T}, \boldsymbol{C}_{\boldsymbol{q}}), \tag{5}$$

which is equivalent to the weighted least-squares problem

$$\boldsymbol{T}(\boldsymbol{C}_{\boldsymbol{q}}) = \arg\min_{\boldsymbol{T}}\; \sum_{i=1}^{N} \boldsymbol{d}_i(\boldsymbol{T})^\top (2\boldsymbol{C}_{\boldsymbol{q}_i})^{-1}\boldsymbol{d}_i(\boldsymbol{T}).$$

We now characterize the sampling distribution of $\boldsymbol{T}(\boldsymbol{C}_{\boldsymbol{q}})$.

Proposition 3 (Distribution of the Correspondence-Implied Pose)**.** *The pose estimator in Eq.* (5) *is the weighted least-squares MLE of $\boldsymbol{T}^\star$. By the standard asymptotic theory of MLE,*

$$\boldsymbol{T}(\boldsymbol{C}_{\boldsymbol{q}}) = \boldsymbol{T}^\star \mathrm{Exp}(\boldsymbol{\eta}), \quad \boldsymbol{\eta} \sim \mathcal{N}\big(\boldsymbol{0},\, \boldsymbol{H}^{-1}(\boldsymbol{C}_{\boldsymbol{q}})\big)\,,$$

where $\mathrm{Exp}(\cdot)$ denotes the exponential map of $SE(3)$, which lifts a $\mathbb{R}^6$ Lie algebra perturbation to a relative transformation, and $\boldsymbol{H}(\boldsymbol{C}_{\boldsymbol{q}})$ is the Hessian of the weighted least-squares objective at $\boldsymbol{T}(\boldsymbol{C}_{\boldsymbol{q}})$.

We additionally adopt a Gaussian pose observation model for $\tilde{\boldsymbol{T}}$, which is typically obtained from a measurement (e.g., pose from LiDAR SLAM):

$$\tilde{\boldsymbol{T}} = \boldsymbol{T}^\star \mathrm{Exp}(\boldsymbol{\xi}), \qquad \boldsymbol{\xi} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Gamma}), \tag{6}$$

where $\boldsymbol{\xi} \in \mathbb{R}^6$ is the pose noise in the Lie algebra and $\boldsymbol{\Gamma}$ is its covariance. Combining Eq. (6) and Proposition 3 yields the distribution of the discrepancy between the correspondence-implied pose and the observed pose.

Proposition 4 (Distribution of the Pose Discrepancy)**.** *Under the independence of the pose measurement noise $\boldsymbol{\xi}$ and the estimation noise $\boldsymbol{\eta}$, the discrepancy between $\boldsymbol{T}(\boldsymbol{C_q})$ and $\tilde{\boldsymbol{T}}$ in the Lie algebra satisfies*

$$\mathrm{Log}\Big(\boldsymbol{T}(\boldsymbol{C_q})^{-1}\tilde{\boldsymbol{T}}\Big) \approx \boldsymbol{\xi} - \boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{0},\, \boldsymbol{W}(\boldsymbol{C_q}))\,,$$

where the approximation follows from the first-order Baker–Campbell–Hausdorff (BCH) expansion, and

$$\boldsymbol{W}(\boldsymbol{C_q}) \triangleq \boldsymbol{\Gamma} + \boldsymbol{H}^{-1}(\boldsymbol{C_q}).$$

Here, $\mathrm{Log}(\cdot)$ denotes the logarithm map of $SE(3)$, which maps a relative transformation to its $\mathbb{R}^6$ Lie algebra representation.

Corollary 2 (Distribution of the Observed Pose)**.** *Given the correspondence-implied pose $\boldsymbol{T}(\boldsymbol{C_q})$ and the discrepancy covariance $\boldsymbol{W}(\boldsymbol{C_q})$, the observed pose $\tilde{\boldsymbol{T}}$ can be represented as*

$$\tilde{\boldsymbol{T}} = \boldsymbol{T}(\boldsymbol{C_q})\,\mathrm{Exp}(\boldsymbol{\mu}),$$

where, from Proposition 4, the discrepancy follows

$$\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{W}(\boldsymbol{C_q})).$$

Therefore, conditioned on $\boldsymbol{T}(\boldsymbol{C_q})$ and $\boldsymbol{W}(\boldsymbol{C_q})$, the observed pose follows

$$\tilde{\boldsymbol{T}} \mid \boldsymbol{T}(\boldsymbol{C_q}), \boldsymbol{W}(\boldsymbol{C_q}) \sim \mathcal{N}(\boldsymbol{T}(\boldsymbol{C_q}),\, \boldsymbol{W}(\boldsymbol{C_q}))\,,$$

in the sense that its Lie algebra perturbation about $\boldsymbol{T}(\boldsymbol{C_q})$ is Gaussian.

Proposition 5 (Maximize Pose Likelihood)**.** *The maximum likelihood estimator of $\boldsymbol{C_q}$ from the distribution in Corollary 2 is*

$$\hat{\boldsymbol{C}}_{\boldsymbol{q}} = \arg\max_{\boldsymbol{C_q}}\; p\Big(\tilde{\boldsymbol{T}} \mid \boldsymbol{T}(\boldsymbol{C_q}), \boldsymbol{W}(\boldsymbol{C_q})\Big)\,. \tag{7}$$

We refer to this as the maximization of pose likelihood, as it is defined over the discrepancy between the correspondence-implied pose $\boldsymbol{T}(\boldsymbol{C_q})$ and the observed pose $\tilde{\boldsymbol{T}}$.

Taking the negative log-likelihood of Eq. (7) and discarding constants reduces the MLE to

$$\hat{\boldsymbol{C}}_{\boldsymbol{q}} = \arg\min_{\boldsymbol{C_q}}\; \tfrac{1}{2}\,\boldsymbol{\mu}(\boldsymbol{C_q})^\top \boldsymbol{W}(\boldsymbol{C_q})^{-1}\boldsymbol{\mu}(\boldsymbol{C_q}) + \tfrac{1}{2}\log|\boldsymbol{W}(\boldsymbol{C_q})|, \tag{8}$$

where $\boldsymbol{\mu}(\boldsymbol{C_q}) \triangleq \mathrm{Log}\Big(\boldsymbol{T}(\boldsymbol{C_q})^{-1}\tilde{\boldsymbol{T}}\Big)$.

V. Geometry Reasoning for LiDAR SLAM

Building on the above likelihood-based formulation, we now instantiate geometry reasoning within a LiDAR SLAM pipeline. The proposed framework consists of three components: a *neural module* that predicts local geometry from raw point clouds, a *reasoning* module that evaluates geometric likelihoods, and a shared *memory* that stores the data exchanged between them. These components are coupled through three procedures operating at different scales: Algorithm 1 trains the neural module from a fixed memory snapshot; Algorithm 2 uses the trained module to refine the memory through SLAM; and Algorithm 3 alternates the two until convergence. We describe each in turn.

Algorithm 1 Self-supervised Training of Geometry Reasoning

Require: Neural module $f_{\boldsymbol{\theta}}$, memory $\mathbf{M} = (\{\mathcal{S}_k\}_{k=1}^K, \hat{\mathcal{T}}, \mathcal{C})$, hyperparams $\Omega = \{\eta, I, \lambda_{\mathrm{corr}}, \lambda_{\mathrm{pose}}, \epsilon\}$
Ensure: Trained neural net $f_{\boldsymbol{\theta}^\star}$

1: **for** $i = 1$ to I **do**
2: $\quad\mathcal{L}(\boldsymbol{\theta}) \leftarrow 0$
3: $\quad$**for** each pair $(k,l) \in \mathcal{E}$ **do**
4: $\quad\quad\mathcal{P} \leftarrow \mathcal{S}_k,\ \mathcal{Q} \leftarrow \mathcal{S}_l$
5: $\quad\quad\hat{\boldsymbol{T}}_{k,l} \leftarrow \hat{\mathcal{T}}_{k,l},\quad \mathcal{C}_{k,l} \leftarrow \mathcal{C}(k,l)$
6: $\quad\quad$/* $\mathcal{Q} \to \boldsymbol{C_q}$ via Cholesky parameterization */
7: $\quad\quad\boldsymbol{L_q} \leftarrow f_{\boldsymbol{\theta}}(\mathcal{Q}),\ \boldsymbol{C_q} \leftarrow \boldsymbol{L_q}\boldsymbol{L_q}^\top + \epsilon\boldsymbol{I}$
8: $\quad\quad$/* Correspondence likelihood, Eq. (4) */
9: $\quad\quad\mathcal{L}_{\mathrm{corr}} \leftarrow \sum_i \Big[\log|2\boldsymbol{C}_{\boldsymbol{q}_i}| + \tilde{\boldsymbol{d}}_i^\top(2\boldsymbol{C}_{\boldsymbol{q}_i})^{-1}\tilde{\boldsymbol{d}}_i\Big]$
10: $\quad\quad$/* Pose likelihood, Eq. (8) */
11: $\quad\quad\boldsymbol{T}(\boldsymbol{C_q}) \leftarrow \mathtt{SDPRLayer}(\boldsymbol{C_q}, \mathcal{C}_{k,l}, \mathcal{P}, \mathcal{Q})$
12: $\quad\quad\mathcal{L}_{\mathrm{pose}} \leftarrow \boldsymbol{\mu}^\top\boldsymbol{W}^{-1}\boldsymbol{\mu} + \log|\boldsymbol{W}|$
13: $\quad\quad\mathcal{L}(\boldsymbol{\theta}) \mathrel{+}= \lambda_{\mathrm{corr}}\mathcal{L}_{\mathrm{corr}} + \lambda_{\mathrm{pose}}\mathcal{L}_{\mathrm{pose}}$
14: $\quad\quad\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\,\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta})$
15: $\quad$**end for**
16: **end for**
17: **return** $f_{\boldsymbol{\theta}^\star}$

Memory. We define a memory module $\mathbf{M}$ as

$$\mathbf{M} \triangleq \big\{\{\mathcal{S}_k\}_{k=1}^K,\ \hat{\mathcal{T}},\ \mathcal{C}\big\},$$

where $\mathcal{S}_k$ is the k-th LiDAR scan. Given a scan-pair graph $\mathcal{E}$, the pose and correspondence memories are defined as

$$\hat{\mathcal{T}} \triangleq \{\hat{\boldsymbol{T}}_{k,l}\}_{(k,l)\in\mathcal{E}}, \qquad \mathcal{C} \triangleq \{\mathcal{C}_{k,l}\}_{(k,l)\in\mathcal{E}},$$

where $\hat{\boldsymbol{T}}_{k,l} \in SE(3)$ is the estimated relative pose from $\mathcal{S}_k$ to $\mathcal{S}_l$, and $\mathcal{C}_{k,l} = \{(\boldsymbol{p}_i, \boldsymbol{q}_i) \mid \boldsymbol{p}_i \in \mathcal{S}_k, \boldsymbol{q}_i \in \mathcal{S}_l\}$ is the corresponding point-pair set. The memory provides scans, poses, and correspondences to the neural and spatial reasoning modules, and is updated by the SLAM module $\mathcal{F}_{\mathrm{SLAM}}$ during reciprocal learning.

Self-supervised training of the neural module. Given the current memory $\mathbf{M}$, Algorithm 1 trains $f_{\boldsymbol{\theta}}$ to predict point-wise covariances that are consistent with the trajectory and correspondences stored in memory. The training signal is composed of two likelihoods. The (*i*) correspondence likelihood (line 8) asks whether the predicted covariance statistically explains the residual between corresponding points, following the Gaussian distribution established in Proposition 2. The (*ii*) pose likelihood (line 11) asks whether the trajectory implied by the predicted covariances agrees with the trajectory already in memory, following Proposition 5. Evaluating the pose

Algorithm 2 SLAM with Reasoned Geometry

Require: Memory $\mathbf{M} = (\{\mathcal{S}_k\}, \hat{\mathcal{T}}, \mathcal{C})$, trained net $f_{\boldsymbol{\theta}^\star}$, SLAM module $\mathcal{F}_{\text{SLAM}}$, hyperparams $\Omega = \{\tau_{\text{ani}}, M\}$
Ensure: Updated memory $\mathbf{M}^+ = (\{\mathcal{S}_k^+\}, \hat{\mathcal{T}}^+, \mathcal{C}^+)$

1: **for** $k = 1$ to K **do**
2: /* Construct positive-definite covariance */
3: $\boldsymbol{L}_k \leftarrow f_{\boldsymbol{\theta}^\star}(\mathcal{S}_k)$, $\boldsymbol{C}_k \leftarrow \boldsymbol{L}_k \boldsymbol{L}_k^\top + \epsilon \boldsymbol{I}$
4: $\mathcal{S}_k^+ \leftarrow \emptyset$
5: **for** each point $\boldsymbol{s}_{k,i} \in \mathcal{S}_k$ **do**
6: Compute anisotropy $a_{k,i}$ from $\boldsymbol{C}_{k,i}$
7: /* Symbolic-grounded outlier rejection */
8: **if** $a_{k,i} > \tau_{\text{ani}}$ **then**
9: Sample $\{\boldsymbol{s}_{k,i}^{(m)}\}_{m=1}^M \sim \mathcal{N}(\boldsymbol{s}_{k,i}, \boldsymbol{C}_{k,i})$
10: $\mathcal{S}_k^+ \leftarrow \mathcal{S}_k^+ \cup \{\boldsymbol{s}_{k,i}^{(m)}\}_{m=1}^M$
11: **end if**
12: **end for**
13: **end for**
14: /* Memory update */
15: $\hat{\mathcal{T}}^+ \leftarrow \mathcal{F}_{\text{SLAM}}(\{\mathcal{S}_k^+\})$
16: $\mathcal{C}^+ \leftarrow \texttt{UpdateCorrespondence}(\hat{\mathcal{T}}^+, \mathcal{C})$
17: **return** $\mathbf{M}^+ = (\mathcal{S}_k, \hat{\mathcal{T}}^+, \mathcal{C}^+)$

Algorithm 3 Reciprocal Learning

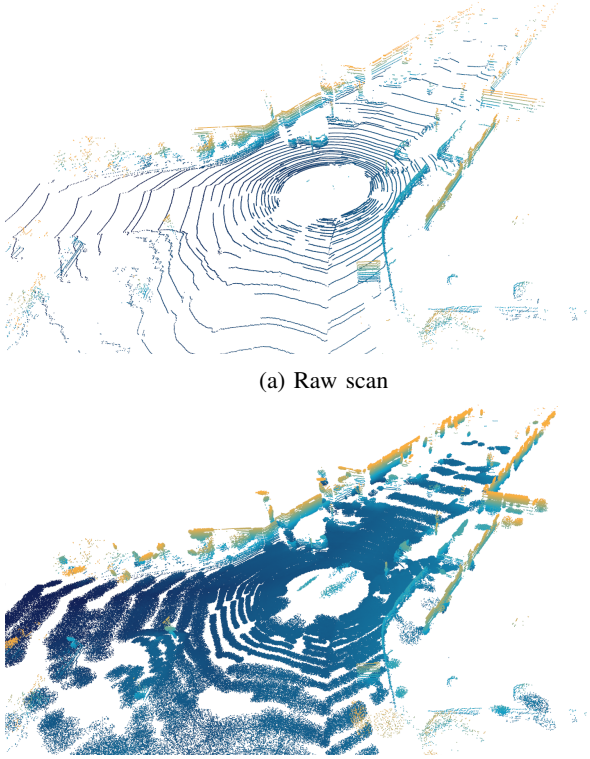
Require: Raw scans $\{\mathcal{S}_k\}_{k=1}^K$, initial net $f_{\boldsymbol{\theta}^{(0)}}$, SLAM module $\mathcal{F}_{\text{SLAM}}$, hyperparams Ω, threshold δ
Ensure: Refined memory $\mathbf{M}^\star$, trained net $f_{\boldsymbol{\theta}^\star}$

1: /* Initialize memory via SLAM */
2: $(\hat{\mathcal{T}}^{(0)}, \mathcal{C}^{(0)}) \leftarrow \mathcal{F}_{\text{SLAM}}(\{\mathcal{S}_k\})$
3: $\mathbf{M}^{(0)} \leftarrow (\{\mathcal{S}_k\}, \hat{\mathcal{T}}^{(0)}, \mathcal{C}^{(0)})$
4: $r \leftarrow 0$
5: **while** not converged **do**
6: /* Update neural module from memory */
7: $f_{\boldsymbol{\theta}^{(r+1)}} \leftarrow$ **Alg. 1**$(f_{\boldsymbol{\theta}^{(r)}}, \mathbf{M}^{(r)}, \Omega)$
8: /* Update memory via reasoning */
9: $\mathbf{M}^{(r+1)} \leftarrow$ **Alg. 2**$(\mathbf{M}^{(r)}, f_{\boldsymbol{\theta}^{(r+1)}}, \mathcal{F}_{\text{SLAM}}, \Omega)$
10: **if** $\|\hat{\mathcal{T}}^{(r+1)} - \hat{\mathcal{T}}^{(r)}\| < \delta$ **then** converged
11: $r \leftarrow r + 1$
12: **end while**
13: **return** $\mathbf{M}^\star = \mathbf{M}^{(r)}$, $f_{\boldsymbol{\theta}^\star} = f_{\boldsymbol{\theta}^{(r)}}$

likelihood requires the global optimum of a weighted least-squares problem, as well as gradients through that optimum. We obtain both using the SDPR layer [5], which leverages the implicit function theorem (IFT) to differentiate through the optimization problem (line 10).

Remark 1 (Self-supervision from Symbolic Consistency)**.** *No ground-truth covariance label is required: the supervision arises entirely from the consistency between the predicted geometry, correspondences, and trajectory.*

SLAM with reasoned geometry. Once $f_{\boldsymbol{\theta}^\star}$ is trained, Algorithm 2 applies it to every scan and refines the memory accordingly. For each scan, the network predicts per-point covariances (line 3), and each point is filtered by its anisotropy (line 8). As established in Section III, points sampled from local surfaces under high-precision LiDAR induce anisotropic covariances; near-isotropic predictions therefore indicate that the Gaussian model is not faithful to the underlying geometry. Retained points are resampled from the predicted distributions (line 9) to form geometry-enhanced scans, qualitatively shown in Fig. 3. These scans are then passed to $\mathcal{F}_{\text{SLAM}}$ to update the trajectory and correspondences (lines 15–16). Note that the SLAM module is treated as a black box, so any LiDAR SLAM pipeline can be plugged in.



(a) Raw scan

(b) Densified scan

Fig. 3: Raw and densified point clouds. (a) Raw 32-channel scan. (b) Densified scan from learned covariance.

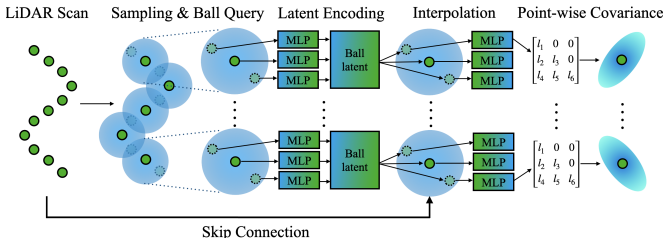
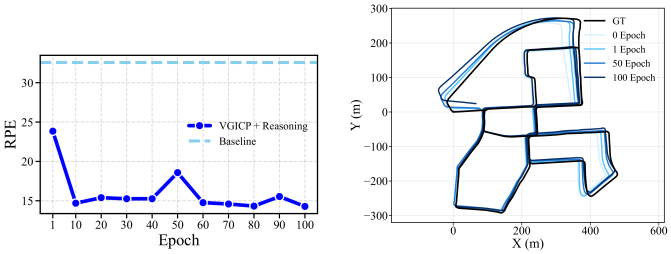


Fig. 4: Network architecture of covariance estimator.

Reciprocal learning. Algorithms 1 and 2 depend on each other: the neural module is trained from the memory, and the memory is refined using the neural module. Algorithm 3 resolves this circularity by alternating between the two, starting



(a) RPE over epochs (b) Trajectory comparison

Fig. 5: Odometry performance over different training epochs. (a) Translational RPE over 300-frame intervals. (b) Estimated trajectories at the corresponding epochs.

TABLE I: Odometry performance under different LiDAR resolutions. We report translational RPE over 300-frame intervals, with relative error reductions from Raw in parentheses.

Input	16 ch.	32 ch.	64 ch.
Raw	15.55	7.88	3.68
1-step	10.02 (↓35.6%)	4.80 (↓39.1%)	3.50 (↓4.9%)
2-step	**7.85** (↓49.5%)	**4.11** (↓47.8%)	**3.46** (↓6.0%)

from an initial memory obtained by running SLAM on the raw scans (lines 2–3). Each iteration updates the neural module (line 7) and then the memory (line 9), and the loop terminates when the trajectory change falls below δ (line 10).

VI. Experiments

A. Experimental Setup

We validate the proposed framework on KITTI odometry Sequence 00 [3] under 64-, 32-, and 16-channel LiDAR settings. The 32- and 16-channel scans are generated by downsampling the original 64 LiDAR channels with strides of 2 and 4, respectively. We use a lightweight VGICP-based odometry pipeline [6] as $\mathcal{F}_{\text{SLAM}}$ and a PointNet++ [12]-inspired architecture as the neural module $f_{\boldsymbol{\theta}}$ (Fig. 4). Starting from the raw memory information obtained from VGICP, we run two reciprocal rounds as a minimal experimental setting to examine the reciprocal effect between geometry reasoning and SLAM. Raw scans are replaced with densified scans generated from the reasoned underlying geometry, denoted as + *Reasoning* in the following experiments, as shown in Algorithm 2 lines 3–9, and Fig. 3.

B. Evaluation Protocol

We assess both odometry and global registration. For odometry, we compare the initial VGICP trajectory (*Raw*) against the trajectories after one (*1-step*) and two (*2-step*) reciprocal updates, using the translational relative pose error (RPE) over 300-frame intervals. For global registration, we randomly sample 100 scan pairs whose ground-truth translation distance is within 10m and align them with TEASER [16], counting a pair as successful when the rotation error is below 10° and the translation error is below 2m.

(a) TEASER (b) TEASER + Reasoning

Fig. 6: Qualitative comparison of global registration results. (a) TEASER fails under sparse LiDAR input. (b) TEASER with the proposed reasoning succeeds.

TABLE II: Global registration performance under different LiDAR resolutions. We report success rates, with relative improvements from Raw in parentheses.

Input	16 ch.	32 ch.	64 ch.
Raw	79.00%	90.00%	97.00%
1-step	**83.00%** (↑5.1%)	**96.00%** (↑6.7%)	97.00% (−0.0%)
2-step	80.00% (↑1.3%)	94.00% (↑4.4%)	**98.00%** (↑1.0%)

C. Results and Analysis

Odometry. Table I shows that geometry reasoning reduces translational drift at all resolutions, with larger gains for sparser inputs. At 16 and 32 channels, the 2-step model cuts drift by roughly half (49.5% and 47.8%), while the gain at 64 channels is marginal (6.0%) because the dense scans already provide reliable geometry. The second reciprocal round further improves odometry in all settings, indicating progressive trajectory refinement (Fig. 5).

Global registration. Table II shows consistent success-rate improvements, with the largest gain at 32 channels (+6.7% at 1-step). Most gains are obtained in the first reasoning round, while later updates show diminishing returns except at 64 channels, which reaches 98%. Figure 6 shows a case where TEASER fails on raw scans but succeeds with reasoning.

VII. Limitations

Although the proposed framework shows promising results, it has several limitations that require further investigation.

- The current densification strategy samples only from highly anisotropic points. More principled covariance-based densification remains future work;
- The training process is time-consuming, taking about 15 minutes per epoch in our setup. Pretrained geometry reasoning models could reduce this cost; and
- Reciprocal updates do not always improve all metrics, as shown in Table II. A reliable stopping criterion remains an open issue.

VIII. Conclusion

We presented a self-supervised geometry reasoning framework that learns underlying geometry for LiDAR SLAM without ground-truth poses or dense geometric labels. While the results demonstrate the potential of learned local geometry for LiDAR SLAM, addressing the above limitations is necessary to make the framework broadly applicable.

References

[1] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jos'e Neira, Ian Reid, and John J Leonard. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics (T-RO)*, 2017.

[2] Taimeng Fu, Shaoshu Su, Yiren Lu, and Chen Wang. iSLAM: Imperative SLAM. *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[3] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[4] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep Learning for 3D Point Clouds: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[5] Connor Holmes, Frederike D"umbgen, and Timothy D Barfoot. SDPRLayers: Certifiable Backpropagation Through Polynomial Optimization Problems in Robotics. *IEEE Transactions on Robotics (T-RO)*, 2025.

[6] Kenji Koide, Masashi Yokozuka, Shuji Oishi, and Atsuhiko Banno. Voxelized GICP for Fast and Accurate 3D Point Cloud Registration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[7] Jinwoo Lee, Jiwoo Kim, Woojae Shin, Giseop Kim, and Hyondong Oh. Learning Point Cloud Geometry as a Statistical Manifold: Theory and Practice. *arXiv preprint arXiv:2605.10456*, 2026.

[8] Hyungtae Lim, Beomsoo Kim, Daebeom Kim, Eungchang Mason Lee, and Hyun Myung. Quatro++: Robust Global Registration Exploiting Ground Segmentation for Loop Closing in LiDAR SLAM. *The International Journal of Robotics Research (IJRR)*, 2024.

[9] Chao-Hung Lin, Jyun-Yuan Chen, Po-Lin Su, and Chung-Hao Chen. Eigen-Feature Analysis of Weighted Covariance Matrices for LiDAR Point Cloud Classification. *ISPRS Journal of Photogrammetry and Remote Sensing (ISPRS JPRS)*, 2014.

[10] Niloy J Mitra and An Nguyen. Estimating Surface Normals in Noisy Point Cloud Data. In *Annual Symposium on Computational Geometry (SoCG)*, 2003.

[11] Mark Pauly, Richard Keiser, and Markus Gross. Multi-Scale Feature Extraction on Point-Sampled Surfaces. In *Computer Graphics Forum (CGF)*, 2003.

[12] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[13] Ignacio Vizzo, Benedikt Mersch, Rodrigo Marcuzzi, Louis Wiesmann, Jens Behley, and Cyrill Stachniss. Make It Dense: Self-Supervised Geometric Scan Completion of Sparse 3D LiDAR Scans in Large Outdoor Environments. *IEEE Robotics and Automation Letters (RA-L)*, 2022.

[14] Chen Wang, Kaiyi Ji, Junyi Geng, Zhongqiang Ren, Taimeng Fu, Fan Yang, Yifan Guo, Haonan He, Xiangyu Chen, Zitong Zhan, et al. Imperative Learning: A Self-Supervised Neuro-Symbolic Learning Framework for Robot Autonomy. *The International Journal of Robotics Research (IJRR)*, 2024.

[15] Bin Yang, Patrick Pfreundschuh, Roland Siegwart, Marco Hutter, Peyman Moghadam, and Vaishakh Patil. Tulip: Transformer for Upsampling of LiDAR Point Clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[16] Heng Yang, Jingnan Shi, and Luca Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Transactions on Robotics (T-RO)*, 2020.

[17] Zitong Zhan, Dasong Gao, Yun-Jou Lin, Youjie Xia, and Chen Wang. iMatching: Imperative Correspondence Learning. In *European Conference on Computer Vision (ECCV)*, 2024.

[18] Ji Zhang, Sanjiv Singh, et al. LOAM: Lidar Odometry and Mapping in Real-Time. In *Robotics: Science and Systems (RSS)*, 2014.