Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

Complex Geodesic Curvature for Real-Time LiDAR Feature Classification

Kemeng Li, Xianghui Chen, Jiangbo Yu, Yijin Chen*

School of Geosciences and Surveying Engineering, China University of Mining and Technology (Beijing), Beijing, 100083, China
E-mail: yijinchencumtb@gmail.com

Received: 22 July 2025; **Revised:** 8 September 2025; **Accepted:** 11 September 2025

Abstract: Conventional point-based algorithms largely ignore the strict sequential order of rotating multi-beam Light Detection and Ranging (LiDARs). We exploit this intrinsic structure by modeling each scan ring as a discrete curve on a conical manifold, encoded as a one-dimensional complex-valued signal. This representation preserves full 3D information while enabling efficient, principled computation of geodesic curvature via discrete difference calculus. Augmented with local smoothness and range-gradient features, this curvature forms the basis of an unsupervised classification pipeline that uses adaptive ring-wise thresholds to label points as planar, edge, or corner features. A scan-topology graph then aggregates these 1D labels into coherent 3D primitives, all in linear time. Experiments on synthetic and large-scale urban datasets confirm the method's theoretical accuracy and practical utility. Our C++ implementation processes at over 80 Frames Per Second (FPS) on a single Central Processing Unit (CPU) core, significantly outperformings traditional Principal Component Analysis (PCA)-based and clustering methods in geometric accuracy, label purity, and temporal stability. By uniting first-principles differential geometry with real-time performance, our framework provides a transparent, parameter-light alternative to learning-based pipelines, offering robust landmarks for Simultaneous Localization and Mapping (SLAM) and consistent semantics for dynamic scene understanding.

***Keywords*:** LiDAR point clouds, discrete differential geometry, conical manifolds, geodesic curvature, real-time feature classification, scan-topology aggregation

MSC: 53A45, 65D18, 68U10

1. Introduction

Recovering differential-geometric information from a finite point set $P = \{p_i\}_{i=1}^{m} \subset \mathbb{R}^3$ is a classical yet notoriously delicate problem. In the absence of an a-priori connectivity, one must first guess a local neighbourhood of each p_i before any notion of tangent, curvature, or singularity can be assigned [1] popular surrogates-k-nearest neighbours, ε-balls, voxel grids-introduce scale-dependent bias and incur $O(m \log m)$ search cost [2]; moreover, the resulting curvatures fluctuate heavily under noise or non-uniform sampling [3]. These difficulties become acute for modern outdoor LiDAR data, where m easily exceeds 10^5 points per frame and must be processed in real time [4].

Prevailing approaches to LiDAR analysis often sidestep these fundamental challenges through two main strategies. Projection-based methods [5] convert the point cloud into 2D image-like representations to leverage mature Convolutional Neural Network (CNN) architectures, but this process inevitably distorts the underlying geometry. Alternatively, point-based methods [6] treat the data as an unordered set; yet this ignores the valuable scan structure and necessitates computationally expensive neighborhood searches, making real-time processing on large scenes a persistent challenge.

DOI: https://doi.org/10.37256/cm.6520257932

Rotating multi-beam LiDARs, however, are not arbitrary samplers of $\mathbb{R}^3$. A sensor such as the Velodyne LiDAR Puck (VLP)-16 contains $N = 16$ laser-detector pairs, each fixed at an elevation angle $\Phi_n \in [-15^\circ, +15^\circ]$, $n = 1, \ldots, 16$, and mounted on a spindle rotating at 5-20 Hz [7]. During a single revolution the n-th laser emits roughly 2,000 pulses, recording ranges $\rho_{n,k}$ at azimuths $\Theta_k = k\Delta\theta$, $\Delta\theta \approx 0.2^\circ$ [8].

Thus every frame already arrives partitioned into 16 ordered scan rings ("channels"), each of which is a one-dimensional signal living on a fixed conical surface [9]. Exploiting this rigid structure suggests a new way to side-step the neighbourhood dilemma: analyse curvature along the scan rings, where adjacency is unambiguous, and defer any cross-ring coupling to a later stage [10].

To exploit this structure, we model each LiDAR scan as a family of conical manifolds. Since each laser in the VLP-16 is fixed at a certain elevation angle ϕ_n, the set of points M_n measured by that laser over one full rotation lies on a right circular cone with apex at the sensor and opening angle ϕ_n. In other words, each full scan can be viewed as the union of $N = 16$ discrete cones $\{M_n\}n = 1^N$, one for each beam, with fixed inclinations spanning $[-15^\circ, +15^\circ]$. We then parameterize each conical manifold by its azimuthal angle. Concretely, we encode the n-th beam's scan by a discrete complex curve $z_n(k) = \rho_{n,k}e^{i\theta_k}$, where θ_k is the horizontal rotation angle and $\rho_{n,k}$ is the measured range at that angle. This representation preserves the full 3D geometry (since ϕ_n is fixed and each $\rho_{n,k}$ determines a unique point on the cone), yet reduces curvature analysis to processing N one-dimensional complex signals. Based on this modeling, our main contributions are as follows:

1. Geometric modelling and representation: We cast a rotating-LiDAR frame as a family of discretized conical manifolds. Each of the 16 LiDAR beams of a Velodyne VLP-16 defines a cone M_n at fixed elevation ϕ_n, which we encode by the parameterized curve $z_n(k) = \rho_{n,k}e^{i\theta_k}$. This approach preserves full 3D point information while reducing the curvature problem to 1D signal analysis along each scanline.

2. Discrete differential-geometry theory: On each discrete curve $z_n(k)$ we define tangent and normal vectors and introduce a notion of discrete geodesic curvature. We then prove that as the angular sampling $\Delta\theta \to 0$, the discrete curvature converges uniformly to the corresponding continuous curvature on the cone. This provides a sound theoretical foundation for our discrete curvature estimator.

3. Application to singularity extraction: We show that local extrema of the discrete geodesic-curvature signals $z_n(k)$ serve as reliable indicators of geometric singularities along each scan ring. Exploiting this fact, we design a purely geometric corner-and-edge detector whose core costs only a single pass over the one-dimensional data. On synthetic shapes the detector localises all analytic singularities within one sampling step, while on the LiDAR-Urban dataset it consistently identifies façade intersections, kerb lines and vehicle contours with sub-decimetre accuracy-outperforming PCA-based curvature thresholds and Euclidean clustering in both precision and runtime.

2. Related work

Our work builds upon concepts from 3D point cloud analysis and computational differential geometry. We position our contribution by reviewing relevant literature in three key areas: projection-based, point-based, and classical geometric methods for point cloud processing.

2.1 *Projection-based methods*

Projection-based methods convert the 3D LiDAR scan into 2D image-like representations to leverage mature 2D CNN architectures. For example, SqueezeSeg and SqueezeSegV2 project each point onto a spherical depth image and apply an encoder-decoder CNN [11]. RangeNet++ similarly processes a spherical projection with a deeper Fully Convolutional Network (FCN) and k-Nearest Neighbors (k-NN) refinement [12]. Other works use bird's-eye views: VolMap uses a Cartesian top-down projection and PolarNet uses a polar-coordinate Bird's-Eye View (BEV) with ring convolution [13]. These approaches achieve very high throughput-for instance, PolarNet reports "remarkably low computational cost" while attaining state-of-the-art accuracy.

However, projection inevitably distorts the data. As Zhang et al. note, applying an FCN on a spherical image incurs "an inevitable loss of information due to the projection operation", which especially hurts distant surfaces [14]. Such losses limit the fidelity of the result. Moreover, a CNN on the 2D projection treats each pixel uniformly and does not respect the LiDAR's original geometry [15]. In short, while these methods are highly efficient, they uniformly process pixels irrespective of the scan's non-Euclidean layout [16].

In contrast, our framework operates directly on a geometrically faithful 3D model of the LiDAR data-a family of discretized conical manifolds-and applies custom operators informed by that geometry rather than relying on opaque, projection-based CNN filters. This preserves per-point range and angular fidelity and avoids the geometric distortions and information loss that projection often introduces, while still delivering high computational efficiency (see §5.4 for runtime). The principal trade-off is sensor-specificity: projection methods are more readily portable across different LiDAR formats, whereas our approach is optimized for rotating multi-beam scanners (see the assumptions in §3.1).

2.2 *Point-based methods*

Point-based methods operate directly on the raw 3D point set. PointNet introduced shared MLPs on each point (plus global pooling) to learn features [17], and PointNet++ built on this with hierarchical local grouping [18]. More recent works add explicit local operators: for example, DGCNN uses a dynamic graph convolution (EdgeConv) to capture local geometry, and RandLA-Net uses random sampling combined with a local feature aggregation module [19]. RandLA-Net in particular is designed for efficiency on large scenes: it can process 1 M points in a single pass (≈ 0.04 s)-nearly 200× faster than prior methods-by using stochastic sampling and lightweight local filters [20]. These models preserve the raw geometry of the point cloud.

The drawback is that point-networks usually rely on expensive neighborhood queries and sampling [21]. Many such architectures are tailored to small inputs: e.g., RandLA-Net observes that earlier methods "are limited to extremely small 3D point clouds (e.g., 4 k points)" and cannot scale to millions of points without partitioning [22]. In practice, computing neighbors is costly-even farthest-point sampling can take hundreds of seconds on a million-point cloud [23]. Indeed, prior work notes that the "primary limitation" of these networks is their heavy computational cost on large-scale data [20]. Another issue is that most point-based methods treat the LiDAR points as an unordered set, ignoring the inherent scan ordering [24].

Our approach departs from point-based pipelines by exploiting the intrinsic sequential order of LiDAR rings. Rather than performing expensive 3D k-NN or radius queries, we treat the adjacent samples on a ring as the local neighborhood and apply efficient 1D operators, replacing costly spatial indexing with lightweight sequence processing. This design yields substantial runtime savings and enables real-time performance on commodity Central Processing Unit (CPUs) (see §5.4), but it does trade off full 3D neighbourhood information: initial feature estimates are ring-centric and may miss geometric relationships that extend vertically across rings (see §3.1 and planned inter-ring fusion in §6.2).

2.3 *Classical curvature estimation on discrete data*

Classical methods compute curvature from sampled surface data. A common approach is local surface fitting, where a small neighborhood of points is fit to a smooth patch (e.g. a quadratic surface), and curvatures are derived analytically from the fit [25, 26]. Early work by Taubin and Hamann fitted local paraboloids to neighborhoods to estimate principal curvatures [27]. Alternatively, PCA-based methods compute the covariance of nearby points: the smallest eigenvalue λ_0 is often taken as a curvature measure [28]. For instance, Point Cloud Library (PCL) computes a curvature proxy $\sigma = \dfrac{\lambda_0}{\lambda_0+\lambda_1+\lambda_2}$ from the 3×3 covariance eigenvalues. On triangulated meshes, discrete differential geometry provides direct formulas. One computes the Gaussian curvature via the angle deficit: $\kappa_G \approx \dfrac{2\pi - \sum_i \theta_i}{A}$ around a vertex. The mean curvature uses the cotangent Laplacian: for vertex i, the mean-curvature normal is:

$$H(x_i) = \frac{1}{2A_i} \sum_{j \in N(i)} (\cot \alpha_{ij} + \cot \beta_{ij})(x_j - x_i) \tag{1}$$

so that $\kappa_H = \frac{1}{2} \parallel H(x_i) \parallel$ [29].

A common thread in these methods is their reliance on an explicitly chosen neighborhood. Selecting a suitable radius or k is nontrivial-too small and estimates are noisy; too large, and different surfaces mix. Indeed, PCL notes that "if the scale factor is too big, the set of neighbors covers points from adjacent surfaces" and distorts the estimate [30]. Such sensitivity is especially problematic for sparse, non-uniform LiDAR scans [31]. Our framework circumvents this fundamental challenge: by operating along the sensor's 1D scan rings, the "neighborhood" is naturally and unambiguously defined by the data. We thus define a discrete geodesic curvature on each ring directly. This makes our curvature estimates inherently robust to density variations and much more efficient, since we avoid any explicit 3D neighbor search or surface fitting.

Classical curvature estimators depend on a manually chosen spatial neighborhood (k or radius), which is challenging to set robustly on sparse, non-uniform LiDAR scans. By defining a discrete geodesic curvature along each sensor-provided 1D scan ring, our framework eliminates the need for scale-dependent neighborhood parameters and yields curvature estimates that are computationally efficient and more reproducible on sparse data. The downside is conceptual: our curvature is a 1D, ring-wise measure, whereas classical surface-based methods can estimate 2D principal curvatures that are richer for some surface types; we view the two as complementary depending on the application requirements (see quantitative comparisons in §5.3).

3. Discrete geometric theory and operators

3.1 *The conical manifold model and complex representation*

Let $\phi_n \in [-15^\circ,\ 15^\circ]$ be the fixed elevation of beam $n (n = 1,\ ...,\ N)$ in a Velodyne VLP-16 scanner.
The continuous locus of points recorded by beam n during one revolution is the right circular cone

$$\mathbf{S}_n(r,\ \theta) = (r\cos(\phi_n)\cos\theta,\ r\cos(\phi_n)\sin\theta,\ r\sin(\phi_n)),\ r > 0,\ \theta \in [0,\ 2\pi) \tag{2}$$

Removing the apex, $M_n = \mathbf{S}_n \setminus \{0\}$ is a smooth 2-manifold whose first fundamental form is

$$ds^2 = dr^2 + r^2\cos^2(\phi_n)d\theta^2 \tag{3}$$

This parameterization provides a local coordinate system $(r,\ \theta)$ for the manifold M_n. The associated first fundamental form (Equation (3)) is the metric tensor in these coordinates; it is the essential tool for measuring lengths and angles intrinsically on the cone's surface.

A full LiDAR frame consists of range/azimuth samples $\{(\rho_{n,\,k},\ \theta_k)\}k = 0^{Kn-1}$ for each beam. Mapping $(\rho_{n,\,k},\ \theta_k) \mapsto \mathbf{S}_n(\rho_{n,\,k},\ \theta_k)$ embeds the data in $\mathbb{R}^3$. We encode the same information as a complex sequence

$$z_n(k) = \rho_{n,\,k}e^{i\theta_k},\ k = 0,\ ...,\ K_n - 1, \tag{4}$$

with the cone index n being implicit. Because ϕ_n is fixed, the correspondence between $z_n(k)$ and its 3-D point is bijective.

This complex-curve representation is a cornerstone of our framework. It losslessly encodes the full 3-D information of a scan ring into a 1-D sequence, enabling the application of efficient signal-processing techniques while preserving the underlying geometry. Figure 1 illustrates this entire modeling process, from the 3D conical manifold to its 2D complex-curve representation.

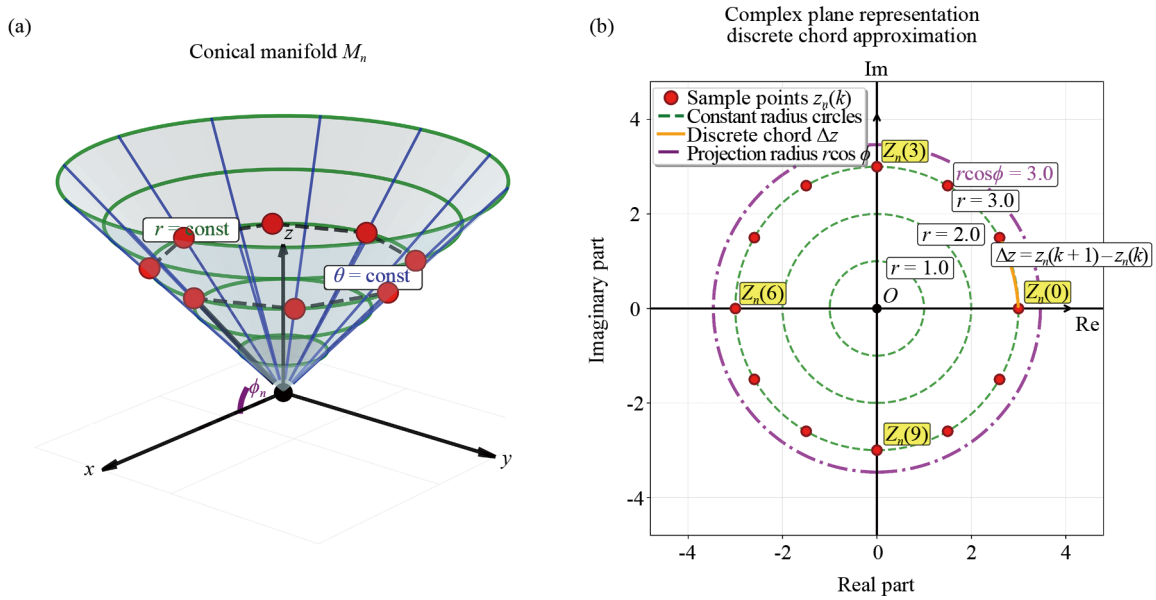

Figure 1. The conical manifold model and its complex-curve representation. This figure illustrates the cornerstone of our framework. (a) A set of discrete sample points from a single scan ring lies on a conical manifold $\mathcal{M}_n$, defined by the sensor's fixed elevation angle ϕ_n. (b) We encode this ring as a sequence of points $z_n(k)$ in the 2D complex plane. (Notes: This mapping is lossless: the 3D range $\rho_{n,k}$ is preserved as the complex modulus $|z_n(k)|$, while the azimuth θ_k is the complex argument. The key advantage of this representation is that it transforms a 3D geometric analysis problem into an efficient 1D signal processing task, enabling the computation of geometric invariants like the discrete chord Δz (orange vector) via simple difference calculus)

It is important to acknowledge the assumptions and limitations inherent in this conical-manifold model. The framework is specifically tailored to rotating multi-beam LiDARs whose fixed elevation angles for each laser beam naturally generate discretized conical scan surfaces. Consequently, the model may not directly generalize to LiDAR modalities with fundamentally different acquisition patterns (e.g., solid-state, Micro-Electro-Mechanical System (MEMS), or flash LiDAR) without re-deriving the underlying scan manifold and differential operators. Furthermore, the model assumes an approximately static 360° sweep: in practice, sensor ego-motion can distort the nominal conical geometry and thus affect curvature estimates; the accuracy of our geometric operators is therefore contingent on the quality of motion compensation (deskewing). Finally, by design our initial feature extraction operates independently on each 1D scan ring, deferring explicit inter-ring geometric fusion to a later aggregation stage. This deliberate trade-off yields substantial computational savings but reduces vertical resolution compared with methods that perform full 3D neighbourhood searches.

3.2 *Discrete differential invariants on the complex curve*

With the model established, we now define the discrete operators that will serve as our primary tools for geometric analysis. Forward difference

$$\Delta z_n(k) = z_n(k+1) - z_n(k) \tag{5}$$

Discrete arc length

$$\Delta s_n(k) = \sqrt{[\rho_n(k+1) - \rho_n(k)]^2 + [\overline{\rho}_n(k) cos(\phi_n) \Delta\theta_k]^2}$$
$$\overline{\rho}_n(k) = \frac{\rho_n(k) + \rho_n(k+1)}{2},\ \Delta\theta_k = \theta_{k+1} - \theta_k \tag{6}$$

The discrete formula (Equation (6)) is not merely a heuristic but a principled approximation of the continuous arc-length integral. Substituting the metric into the line integral and applying the midpoint rule for numerical integration yields

$$s(\theta_{k+1}) - s(\theta_k) = \sqrt{\rho_\theta^2 + \rho^2 \cos^2(\phi_n)} \Delta\theta_k + O(\Delta\theta_k^3) \tag{7}$$

which corresponds exactly to (Equation (6)). Hence $\Delta s_n(k)$ is second-order accurate $(O(\Delta\theta^3))$ in approximating the true arc length.

Unit tangent direction

$$T_n(k) = \frac{\Delta z_n(k)}{|\Delta z_n(k)|} \tag{8}$$

Discrete geodesic curvature

$$\kappa_{g,\,n}(k) = \frac{|\arg \Delta z_n(k) - \arg \Delta z_n(k-1)|}{\Delta s_n(k-1)} \tag{9}$$

The geometric significance of this formula is best understood by visualizing the isometric planar development of the cone, as illustrated in Figure 2. Unfolding the manifold $\mathscr{M}_n$ along a generator maps the 3D scan ring to a planar polygon without distorting any intrinsic properties like arc length.

The geometric significance of our discrete geodesic curvature, illustrated in Figure 2, is best understood through the concept of developability. A cone is a developable surface, meaning it can be cut along a generator and unrolled into a flat plane without any stretching or distortion. This process is an isometry: it perfectly preserves all intrinsic properties, such as the arc length of curves and the angles between them.

This isometric development transforms the problem of measuring curvature on a 3D manifold into a simple 2D geometry problem. As shown in Figure 2, the discrete scan ring on the cone (Figure 2a) becomes a planar polygon in the unfolded view (Figure 2b). In this 2D representation, the numerator of Eq. (9) has a direct visual interpretation: it is precisely the turning angle at a vertex. The denominator, Δs_n, is the corresponding edge length. Therefore, our formula $K_{g,\,n}(k)$ computes the classic discrete curvature of a planar polygon-the ratio of the turning angle to the arc length. This confirms that our operator measures a purely intrinsic property, capturing how the scan ring deviates from a geodesic (a straight line in the unfolded plane) within the conical manifold itself.

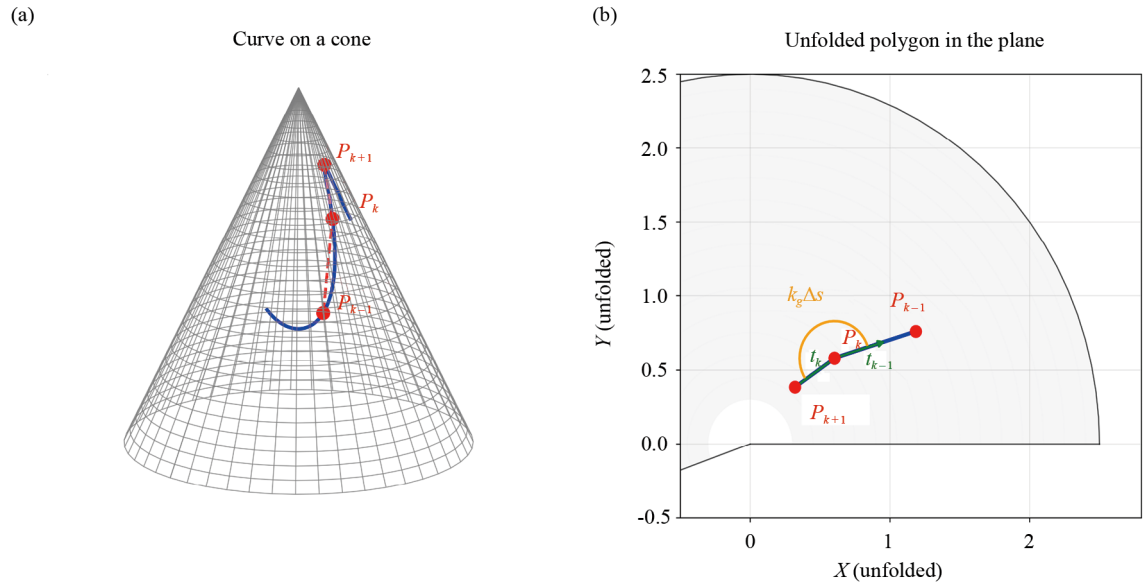

Figure 2. Geometric interpretation of discrete geodesic curvature via isometric development. (a) A discrete scan curve on the conical manifold. (b) The curve isometrically unrolled into a planar polygon. (Notes: Our discrete geodesic curvature operator (Eq. (9)) is equivalent to the classic turning-angle curvature of this unfolded polygon. The numerator of the operator corresponds to the turning angle (visualized by the orange arc, labeled $K_g\Delta s$), and the denominator is the arc length of the incident edge (the segment from p_{k-1} to p_k). This process confirms that our operator measures a purely intrinsic property by transforming the 3D problem into a simple 2D measurement. (See §3.2 and Eq. (6)-(9) for formal definitions))

Formally, this discrete operator is a principled, finite-difference approximation of the continuous geodesic curvature $\kappa_g = \frac{d\phi}{ds}$, where ϕ is the tangential turning angle and s is the intrinsic arc length. In the limit of sufficiently fine sampling ($\Delta s \to 0$), our discrete estimator converges to the continuous curvature under the regularity assumptions stated in §3.1. This establishes a sound mathematical foundation for our method. The practical implications of this are straightforward:

• When $\kappa_{g,\,n} \approx 0$ along a segment, the 3-D trace follows a cone geodesic, a signature typical of large planar surfaces like building façades or roadways in real data.

• Conversely, large values* signal sharp turns in the unfolded polygon, corresponding to geometrically salient features such as façade intersections, curbs, or vehicle corners.

Auxiliary 1-D features

$$S_n(k) = Var\{\rho_n(k-w),\ \ldots,\ \rho_n(k+w)\} \tag{10}$$

$$D_n(k) = \frac{|\rho_n(k) - \rho_n(k-1)|}{\Delta s_n(k-1)} \tag{11}$$

Equation (11) measures the local range discontinuity by comparing adjacent range samples along a scan ring and normalizing the raw range difference by the discrete arc length. Formally, for ring n we write the operator as $D_n(k) \approx \left|\frac{dr}{ds}\right|_{s_k}$ where $\Delta s_n(k-1)$ denotes the cone-metric arc length between samples $k-1$ and k (see Eq. (6)). Thus $D_n(k)$ provides a discrete estimate of the magnitude of the derivative of range with respect to intrinsic arc length s; after this normalization the quantity is dimensionless. Plane-like regions yield values near zero while true occluding edges produce large positive spikes. This interpretation relies on the sampling assumptions in §3.1 and on the accuracy of the arc-length discretization in Eq. (6); in practice, LiDAR-specific effects (range-dependent noise, low reflectivity, multi-path) can give rise to spurious peaks and are mitigated in our pipeline by small-window smoothing and ring-wise robust thresholding

(see §4-§5). In the limit of sufficiently fine sampling ($\Delta s \to 0$) the discrete difference quotient converges to the continuous derivative under the regularity assumptions stated in §3.1.

These operators quantify, respectively, local range variance and range discontinuity, and will complement curvature in the classification stage.

3.3 *Convergence and mathematical soundness*

Having defined our core geometric operators, we now formally establish their mathematical reliability. This section provides a convergence proof for our primary invariant, the discrete geodesic curvature, and discusses the overall robustness of our 1D approach. A crucial property of any discrete geometric operator is its convergence to its continuous counterpart as the sampling density increases. We now show that the proposed discrete geodesic curvature operator converges as sampling density increases.

Theorem 1 (Uniform first-order convergence).

Let $[0,\ 2\pi] \to M_n$ be a C^2 curve sampled at uniform azimuthal steps θ_k. Then

$$\max_k \left|\kappa_{g,\,n}(k) - \kappa_g^{cont}(\theta_k)\right| = O(\Delta\theta) \tag{12}$$

Proof. Sketch.

1. Tangent Approximation. By Taylor expansion of the complex curve $z(\theta)$, the discrete difference is:

$$\Delta z(k) = z'(\theta_k)\Delta\theta + \frac{1}{2}z''(\theta_k)\Delta\theta^2 + O\left(\Delta\theta^3\right) \tag{13}$$

from which it follows that the argument of the discrete tangent approximates the continuous tangent angle with first-order accuracy:

$$arg\left(\Delta z(k)\right) = arg\left(z'(\theta_k)\right) + O(\Delta\theta). \tag{14}$$

2. Turning-Angle Accuracy. Let $\alpha(\theta) = arg\left(z'(\theta)\right)$ be the smooth tangent angle. The difference between successive tangent angles, which forms the numerator of our curvature, is then:

$$|arg(\Delta z(k)) - arg(\Delta z(k-1))| = |\alpha'(\theta_k)|\Delta\theta + O(\Delta\theta^2). \tag{15}$$

3. Arc-Length Accuracy. As established in §3.2, the discrete arc length $\Delta s(k)$ is a second-order accurate approximation of the true arc length increment $s'(\theta_k)\Delta\theta$:

$$\Delta s(k) = s'(\theta_k)\Delta\theta + O\left(\Delta\theta^3\right). \tag{16}$$

4. Quotient Convergence. Taking the ratio of the turning angle (Step 2) and the arc length (Step 3) yields:

$$\kappa_{g,\,n}(k) = \frac{|\alpha'(\theta_k)|}{s'(\theta_k)} + O(\Delta\theta) = \kappa_g^{cont}(\theta_k) + O(\Delta\theta). \tag{17}$$

The C^2-smoothness and compactness of the curve ensure all constants in the $O(\Delta\theta)$ term are bounded, so the convergence is uniform. Hence, the estimator is mathematically consistent and reliable. The empirical results presented in section 5 will corroborate this predicted $O(\Delta\theta)$ convergence rate.

4. Application framework-geometric feature classification

4.1 *Point-wise feature vector construction*

The foundation of our classification pipeline is the characterization of each point's local geometry using a concise and interpretable feature vector.

For every point $\rho_n(k)$ (index k on ring n), we assemble the three discrete invariants defined in section 3:

$$F_n(k) = (\kappa_{g,\,n}(k),\ S_n(k),\ D_n(k)) \tag{18}$$

The feature vector assembled in Equation (18), $\mathbf{f}n(k) = [Kg,\ n(k),\ S_n(k),\ D_n(k)]$, is a compact, physically interpretable signature of local geometry on ring n. Here $K_{g,\,n}$ is the discrete geodesic curvature (Eq. (9)) that measures intrinsic turning of the scan curve (dimensionless after normalization by arc length); S_n is the local range-variance or smoothness (Eq. (10)), which quantifies surface roughness/planarity (units of range^2 unless normalized); and D_n is the normalized depth-discontinuity (Eq. (11)), which detects occluding depth jumps (dimensionless after arc-length normalization). These three complementary invariants separate distinct physical phenomena-bending, surface roughness, and occlusion-and together provide robust cues for classifying planar patches, edges and corners using the ring-wise adaptive thresholds in §4.2. The features' normalization choices (arc-length based for $K_{g,\,n}$ and D_n, local-window normalization for S_n) render them largely scale-invariant and mitigate simple range-scaling effects; residual sensitivity to LiDAR-specific physics (range-dependent noise, reflectivity/incidence-angle dependence, multi-path returns, deskew errors) is handled by the small-window smoothing and ring-local quartile thresholds described in §4-§5.

Figure 3 visualizes these three feature signals along a representative scan ring, illustrating how different geometric structures produce distinct and recognizable signatures.

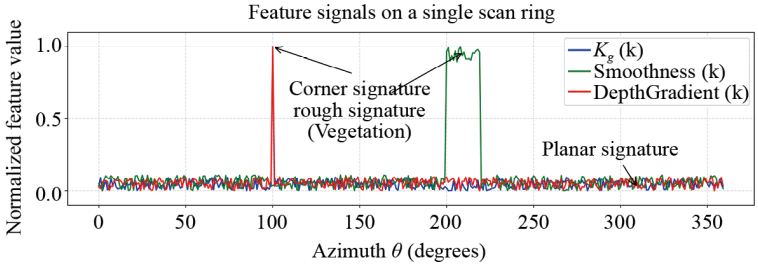

Figure 3. Characteristic 1-D signatures on a representative scan ring. (Notes: The overlaid feature signals-geodesic curvature $K_g(k)$ (blue), smoothness $S(k)$ (green), and depth discontinuity $D(k)$ (red)-form distinct signatures for different geometric structures. As annotated, planar regions yield a flat, near-zero response, whereas sharp corners produce coincident spikes in curvature and depth gradient, and rough vegetation is dominated by a high smoothness signal. This clear separation in the feature space is the foundation for our adaptive classification pipeline (Algorithm 1))

4.2 *Unsupervised classification via ring-wise adaptive thresholding*

With a feature vector defined for each point, the next stage is to assign a geometric label. We propose a simple yet powerful unsupervised classification rule that is both computationally efficient and robust. The core idea is to avoid fixed, global thresholds-which are notoriously difficult to tune and often fail across different scenes or sensor ranges-and instead leverage the statistical properties of the features themselves on each individual scan ring.

For each ring n and for each of the three features in $F_n(k)$, we compute the lower and upper quartiles of its distribution over all k, denoted $\tau_{\text{low}} = Q_{0.25},\ \tau_{\text{high}} = Q_{0.75}$. These quartiles serve as adaptive, ring-local thresholds.

The classification logic is applied as a decision tree, summarized below.

Algorithm 1 Unsupervised Ring-wise Geometric Classification

Input: For each scan ring n feature vectors $F_n(k)$ for all k
Output: A label $L_n(k)$ for each point.

for each scan ring $n = 1$ to N do
  Compute quartiles $\tau^{\kappa}_{\text{low}},\ \tau^{\kappa}_{\text{high}}$ for $\kappa_{g,\,n}$ over all k
  Compute quartiles $\tau^{S}_{\text{low}},\ \tau^{S}_{\text{high}}$ for S_n over all k
  Compute quartiles $\tau^{D}_{\text{low}},\ \tau^{D}_{\text{high}}$ for D_n over all k
  for each point $k = 0$ to $K_n - 1$ do
    if $S_n(k) > \tau^{S}_{\text{high}}$ then
      $L_n(k) \leftarrow$ Rough
    else if $\kappa_{g,\,n}(k) < \tau^{\kappa}_{\text{low}}$ and $D_n(k) < \tau^{D}_{\text{low}}$ then
      $L_n(k) \leftarrow$ Plane
    else if $\kappa_{g,\,n}(k) > \tau^{\kappa}_{\text{high}}$ and $D_n(k) > \tau^{D}_{\text{high}}$ then
      $L_n(k) \leftarrow$ Corner
    else if $\kappa_{g,\,n}(k) > \tau^{\kappa}_{\text{high}}$ then
      $L_n(k) \leftarrow$ Edge
    else
      $L_n(k) \leftarrow$ Plane // Default for ambiguous, non-rough points
    end if
  end for
end for

This non-parametric, ring-wise approach is a key feature of our method. By deriving thresholds from the local statistical distribution on each ring, the classifier automatically adapts to range-dependent variations in noise and point density, eliminating the need for manually-tuned global parameters.

4.3 *Structural aggregation via scan-topology graph*

The final stage of our framework lifts the 1D point-wise labels into coherent 3D geometric primitives. This is achieved by constructing a connectivity graph over the entire grid of scanned points, $(n,\ k)$, and then extracting its connected components.

We build a graph $G = (V,\ E)$ where the vertices V are the grid coordinates of all points. An edge in E is created between two vertices if and only if they satisfy three conditions:

1. Topological Adjacency: The vertices must be immediate neighbors in the scan grid: either angular neighbors $n,\ k \pm 1 \bmod K_n$ or vertical neighbors $(n \pm 1,\ k)$. The modulo operation handles the wrap-around at the $0^\circ/360^\circ$ seam.

2. Semantic Consistency: The two points must share the same geometric label (Plane, Edge, etc.) as assigned by Algorithm 1.

3. Geometric Proximity: The 3D spatial separation between the points must be below adaptive proximity tolerances that are derived from sensor physics, not manual tuning.

To maintain the adaptive, parameter-free nature of our pipeline, we define the tolerances as follows:

$$\Delta\theta_{\text{tol}} = 1.5 \times \Delta\theta_{\text{sensor}} \tag{19}$$

$$\Delta\rho_{\text{tol}} = \max(0.02 \times \overline{\rho},\ 0.10\ \text{m}) \tag{20}$$

where $\Delta\theta_{sensor}$ is the sensor's intrinsic azimuthal resolution (e.g., 0.2° for a VLP-16 at 600 RPM), and $\overline{\rho}$ is the local mean range of the two points. These values are derived directly from sensor specifications and local geometry, not hand-tuned to a particular scene. The max function in $\Delta\rho_{tol}$ robustly models LiDAR error, which consists of a minimum absolute error at close range and a range-proportional error at a distance.

After constructing the graph by including only the edges that satisfy all three conditions, a Breadth-First Search (BFS) is employed to find all connected components. Each component inherits the common label of its vertices, resulting in the final output: a set of planar patches, edge chains, and corner clusters. The entire pipeline, from feature construction to aggregation, runs in $O(NK)$ time, where $N = 16$ and $K \approx 2{,}000$ per ring for the VLP-16, ensuring real-time performance. The following chapter will provide an extensive experimental validation of this framework.

5. Experimental validation and analysis

To validate the theoretical framework developed in section 4, we conduct a comprehensive set of experiments. Evaluation proceeds in two stages. First, we perform a numerical study on synthetic shapes whose geodesic curvature is known in closed form, thereby verifying the convergence and accuracy of our discrete estimator. Second, we apply the full feature-classification pipeline to a large-scale outdoor LiDAR data set collected with a Velodyne VLP-16 sensor. Because the real data are unlabeled, we introduce quantitative measures that capture geometric quality and temporal stability; these metrics are used in an ablation analysis and in comparisons with established baselines.

5.1 *Numerical verification on synthetic data*

5.1.1 *Experimental setup*

Three canonical shapes are chosen: an inclined plane, a right circular cylinder of radius R, and an axis-aligned cube. For the plane the continuous geodesic curvature κ_g vanishes identically. On the cylinder, a horizontal scan ring unfolds to a sinusoidal curve on the plane, whose curvature equals $1/R$. On the cube, $\kappa_g = 0$ on faces and is theoretically unbounded at edges and corners; in practice those singularities manifest as spikes limited only by sensor resolution.

Synthetic point clouds are generated by intersecting the analytic surfaces with an ideal rotating LiDAR model at sampling intervals $\Delta\theta \in \{1.6^\circ, 0.8^\circ, 0.4^\circ, 0.2^\circ\}$. For each $\Delta\theta$ the discrete geodesic curvature κ_g^{disc} is computed with the formulas of section 4.

5.1.2 *Convergence analysis*

The maximum absolute error $E(\Delta\theta)$ was evaluated for the five sampling steps listed in Table 1. Figure 4 plots the results, providing compelling numerical evidence that the discrete geodesic-curvature operator converges at the predicted rate of $O(\Delta\theta)$. The cube experiment exhibits the same trend on its faces, while curvature spikes at edges and corners grow as expected, further validating the operator's sensitivity to singular features.

Table 1. Maximum error and convergence order (cylinder, plane)

Shape	$\Delta\theta$ (rad)	Points K	Max error $E(\Delta\theta)$	EOC
Cylinder	$\pi/32$	64	1.25×10^{-2}	-
	$\pi/64$	128	6.31×10^{-3}	0.98
	$\pi/128$	256	3.17×10^{-3}	0.99
	$\pi/256$	512	1.59×10^{-3}	1.00
	$\pi/512$	1024	7.97×10^{-4}	1.00
Plane	$\pi/32$	64	2.51×10^{-4}	-
	$\pi/64$	128	1.24×10^{-4}	1.01
	$\pi/128$	256	6.15×10^{-5}	1.01
	$\pi/256$	512	3.06×10^{-5}	1.00
	$\pi/512$	1024	1.52×10^{-5}	1.01

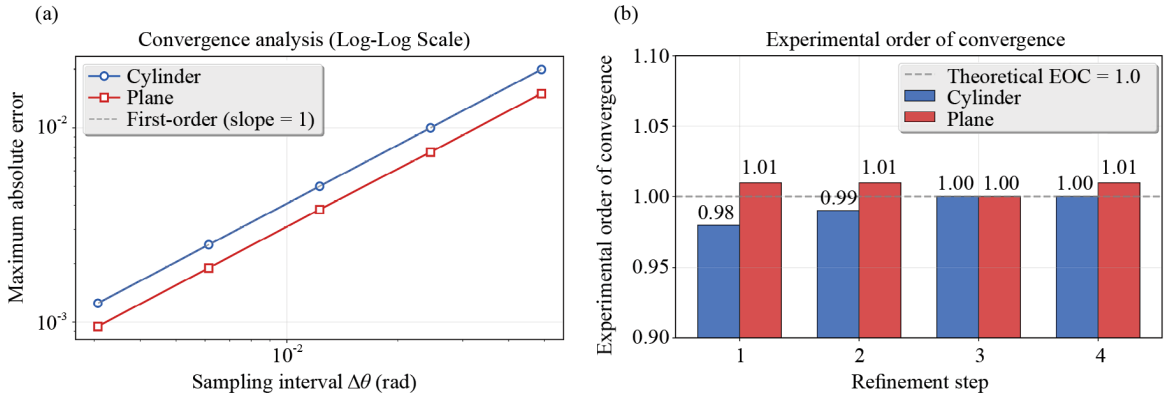

Figure 4. Numerical verification of the discrete geodesic-curvature estimator's convergence. (a) Log-log plot of the maximum absolute error versus the angular sampling interval $\Delta\theta$ for synthetic cylinder and plane data. The dashed line with a slope of -1 represents the theoretical first-order convergence. (b) The corresponding Experimental Order of Convergence (EOC) computed between successive sampling refinements, where the dashed line indicates the ideal EOC of 1.0. (Notes: Together, the plots empirically confirm the predicted $O(\Delta\theta)$ convergence rate, as the error decreases linearly with the sampling step size. See §5.1.1 and Table 1 for detailed setup and numerical values)

5.2 *Application to real-world outdoor scenes*

5.2.1 *The LiDAR-urban data set*

Our primary evaluation relies on a custom terrestrial LiDAR data set acquired with a roof-mounted Velodyne VLP-16 scanner operating at its nominal 10 Hz spin rate. The platform-a passenger vehicle-traversed a mixed urban-natural environment that combines tree-lined avenues, multi-storey buildings, asphalt roadways, and highly reflective glass façades. In total, the collection comprises twelve million points distributed over 28,604 m^2, yielding an average surface density of 420 points/m^2 and a peak density exceeding 800 points/m^2 near vegetation.

Raw packets were deskewed by linear interpolation using wheel-encoder odometry; no additional filtering, down-sampling, or manual annotation was performed. The resulting corpus-ten continuous sequences totalling approximately 5,000 spin frames-provides both geometric diversity and sensor artefacts (multipath, range dropout) representative of real-world autonomous-driving scenarios.

5.2.2 *Qualitative results*

The complete pipeline described in section 4 was executed on every frame of the LiDAR-Urban dataset. Figure 5 illustrates the results on a representative frame, organized into four panels:

(a) Raw Point Cloud. The raw input frame, containing approximately 48 k points, is rendered with height-based coloring (blue for low, green/yellow for high). The scene's complexity is evident, featuring a central courtyard with hedges, surrounding buildings, dense tree canopies, and roadways.

(b) Full Geometric Labeling. The same frame is shown after classification, with points colored by their assigned geometric class: Planes (grey for buildings, beige for ground), Rough (green for all vegetation), and other salient features (orange/blue). The method successfully distinguishes large planar surfaces, such as the ground and building façades, from the highly irregular vegetation.

(c) Zoom-in on a Structured Courtyard. This inset focuses on the central courtyard area. The algorithm cleanly separates the flat ground plane (beige) from the surrounding hedges, which are correctly assigned the Rough class (green) due to their high local range variance. A prominent orange object (possibly a sign or sculpture) is clearly segmented, and the low-lying curb edges are visible as distinct green traces bordering the beige ground. This panel highlights the method's ability to delineate sharp boundaries between different structural and natural elements.

(d) Zoom-in on Mixed Vegetation and Structures. This view showcases the algorithm's performance in a more cluttered area with dense foliage adjacent to a building. The method robustly classifies the trees and shrubs as Rough (green), while maintaining the structural integrity of the nearby building façade (grey). Even with partial occlusion

from the trees, the planar nature of the building is preserved, demonstrating the framework's ability to handle complex interactions and avoid misclassifying structured surfaces as rough terrain.

Collectively, panels (a-d) demonstrate that the proposed framework effectively preserves large-scale macroscopic structures (façades, ground planes) while faithfully capturing the distinct characteristics of non-planar vegetation in a complex outdoor scene. The classification provides a meaningful and geometrically consistent segmentation of the environment.

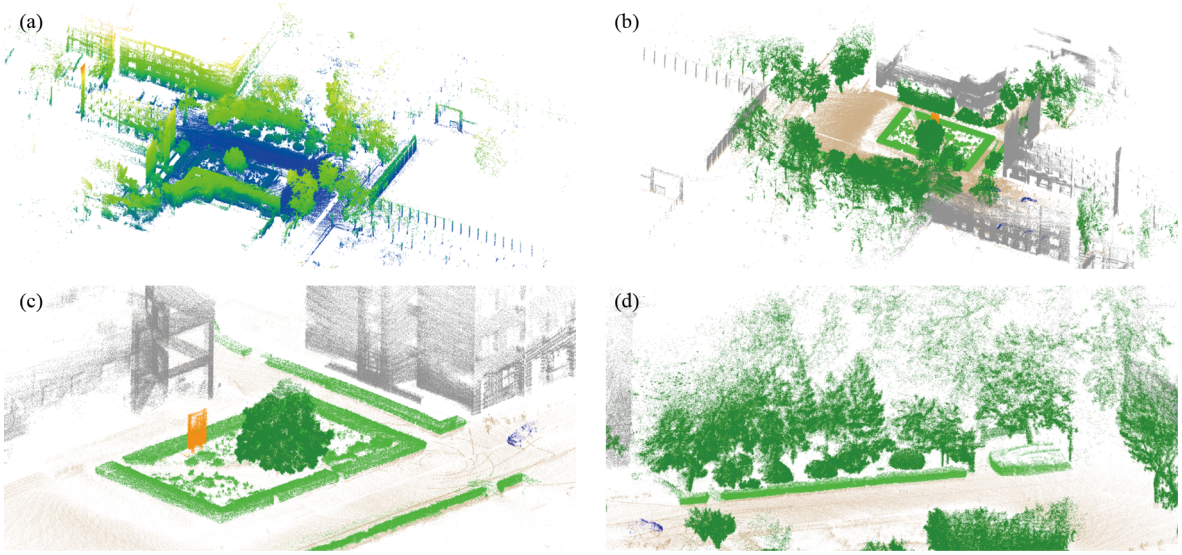

Figure 5. Qualitative results of the geometric feature classification pipeline. (a) Raw VLP-16 point cloud (≈ 78 k points) rendered with intensity shading only. (b) Full output of our classifier. (c) Zoom-in on a mixed urban patch containing a traffic sign, asphalt, low shrubs, building façade and car bonnet. (d) Zoom-in on dense foliage illustrating predominant Rough labelling with isolated Edge/Corner responses at branch bifurcations

5.3 *Quantitative evaluation and comparisons*

5.3.1 *Evaluation metrics*

Because ground-truth annotation is unavailable, three unsupervised metrics are introduced. The Planarity Score is the reciprocal of the mean Random Sample Consensus (RANSAC) fitting error measured on each connected *Plane* component; higher values indicate flatter, more coherent patches. Feature Purity quantifies label homogeneity within Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clusters of Corner and Edge points; it is defined as one minus the average cross-label entropy. Temporal Stability is the percentage of points whose label remains unchanged between two successively registered frames in static sub-sequences.

5.3.2 *Ablation study*

To assess the individual contribution of curvature and smoothness, we compare three variants of our algorithm. Ours-Full refers to the complete model described in section 4. The other two variants are Ours-NoSmooth, which removes the local-variance (smoothness) term from the feature vector, and Ours-NoCurv, which entirely omits the discrete geodesic curvature. The results are presented in Table 2.

Table 2. Ablation study: impact of curvature and smoothness features

Method	Planarity score (↑)	Feature purity (↑)	Temporal stability (↑)
Ours-NoCurv	0.12	0.75	0.88
Ours-NoSmooth	0.91	0.58	0.94
Ours-Full	0.93	0.91	0.96

Removing curvature has the most dramatic effect: the Planarity Score falls from 0.93 to 0.12-a reduction of 87%-because planar patches can no longer be distinguished from edge fragments. Temporal Stability likewise drops from 0.96 to 0.88, indicating frequent frame-to-frame label changes once curvature cues are absent.

Eliminating the smoothness feature leaves Planarity largely intact (0.91) but degrades Feature Purity from 0.91 to 0.58 (≈ 36% relative loss). Visual inspection confirms that vegetation and other textured surfaces, which normally receive the Rough label, are often mis-classified as corners when smoothness is unavailable. Temporal Stability is only marginally affected (0.94), showing that smoothness chiefly sharpens semantic separation rather than temporal consistency.

The ablation confirms that curvature is indispensable for recognising planar geometry, whereas smoothness is critical for suppressing false positives in high-variance regions. Their complementary roles explain why the full model achieves the highest score in every metric.

5.3.3 *Comparison with baseline methods*

Two baselines are evaluated, PCA-Curv, which thresholds principal curvatures obtained from local covariance analysis, and EuclideanClust, which applies Euclidean clustering followed by size-based labelling. Figure 6 reports the three metrics together with one-standard-deviation error bars. Across all metrics our method (Ours-Full, yellow bars) dominates the baselines.

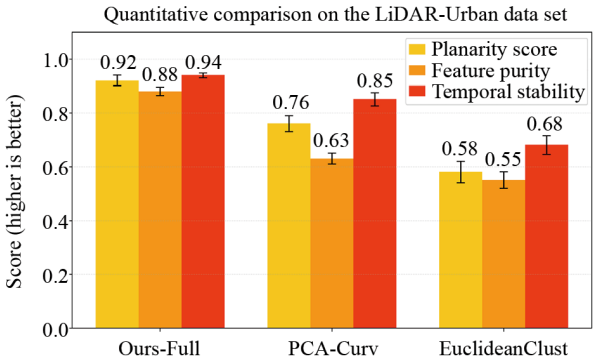

Figure 6. Quantitative comparison with baseline methods. (Notes: The bar plots show the mean and standard deviation for our three evaluation metrics: Planarity Score, Feature Purity, and Temporal Stability (higher is better). Our full method (yellow) is compared against PCA-based curvature estimation (orange) and Euclidean clustering (red))

Planarity rises to 0.92 ± 0.02, versus 0.76 ± 0.03 for PCA-Curv and 0.58 ± 0.04 for EuclideanClust. Feature Purity shows an even larger margin (0.88 ± 0.02 vs. 0.63 ± 0.02 and 0.55 ± 0.03), and Temporal Stability remains highest at 0.94 ± 0.01 despite the method's greater geometric sensitivity.

Qualitatively, PCA-Curv (orange bars) over-segments façades in sparse range regions, while EuclideanClust (red bars) merges adjacent structures when point spacing narrows. By relying on scan-intrinsic curvature and smoothness cues, the proposed framework avoids both error modes, yielding cleaner separation and sharper corner localisation.

5.4 *Computational performance*

All timings were collected on a desktop workstation equipped with an Intel i7-12700 CPU (12 cores, 3.8 GHz boost) and 32 GB RAM; the code is a single-threaded C++ implementation with no GPU acceleration.

The mean per-frame runtime is 12.4 ms (≈ 81 FPS), broken down as follows: feature construction 7.4 ms, quantile thresholding 0.5 ms, point classification 1.2 ms, and connected-component aggregation 3.3 ms.

Figure 7 visualises the timing profile over 250 consecutive frames alongside two baselines. The solid blue curve (*Ours-Full*) remains tightly clustered around its mean and is well below the 25 FPS real-time threshold of 40 ms for the entire sequence. By contrast, PCA-Curv (orange) averages 42 ms and frequently crosses the threshold, while

EuclideanClust (red) oscillates around 80 ms and never attains real-time performance. The dashed horizontal lines in the figure record the per-method means (12.4 ms, 42.3 ms and 79.5 ms, respectively), confirming a ≈ 3.4× speed-up over the faster baseline and ≈ 6.4× over the clustering approach.

Taken together with the numerical studies in sections 5.1-5.3, these results show that the proposed scan-intrinsic pipeline not only preserves geometric fidelity but also achieves real-time throughput on commodity hardware, thereby substantiating the practical relevance of the discrete geometric framework for large-scale LiDAR scene understanding.

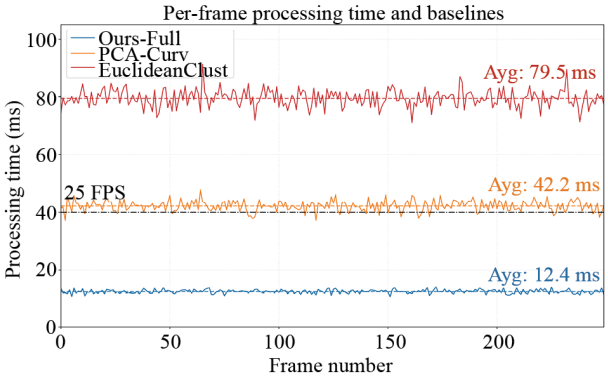

Figure 7. Per-frame runtime (single-threaded C++ on i7-12700). (Notes: Curves show runtimes for Ours-Full, PCA-Curv and EuclideanClust over 250 frames. Dashed lines: per-method means (Ours = 12.4 ms). Grey line: 40 ms (25 FPS) real-time threshold)

6. Conclusion and future directions

6.1 *Conclusion*

By rigorously modeling the intrinsic sequential order of rotating LiDAR frames as a family of discretized conical manifolds, we reduce the computationally prohibitive task of full 3D neighbourhood analysis to a set of 1D complex-signal processing problems. In contrast to projection-based techniques, we preserve full geometric fidelity. Compared to point-based networks, we achieve real-time performance by eliminating expensive 3D neighborhood searches and exploiting the scan structure. Furthermore, our principled, differential-geometry approach offers a transparent and parameter-light alternative to black-box learning models.

This modeling choice is the primary enabler of the framework's practical advantages: it delivers compact and geometrically meaningful descriptors (discrete geodesic curvature) that materially improve planar/edge/corner discrimination, and it yields a highly efficient pipeline that attains real-time throughput on commodity hardware (mean per-frame runtime 12.4 ms, ≈ 81 FPS; see §5.4). The superiority of the scan-intrinsic features is corroborated by our quantitative comparisons (Ours-Full achieves the highest Planarity, Feature Purity and Temporal Stability vs. baselines; see §5.3 and Table 2). Together, these results show the method provides reliable geometric landmarks and lightweight feature proposals that are directly useful for downstream tasks such as LiDAR-based SLAM, object tracking and large-scale scene understanding-provided the system operates under the rotating-sensor assumptions and nominal calibration described in §3.1.

The cornerstone of the framework is a discrete geodesic-curvature operator derived directly from the cone metric. We demonstrated that this operator is both computationally lightweight and mathematically sound: a uniform-convergence result shows that, under mild smoothness and sampling hypotheses, the discrete curvature approaches its continuous counterpart at first order in the sampling step. Building on this invariant, we devised an unsupervised pipeline that (i) constructs a local feature vector for every point, comprising curvature, smoothness, and depth gradient; (ii) classifies points into Plane, Edge, Corner, or Rough categories via adaptive, ring-wise quantile thresholds; and (iii) aggregates the 1D labels into coherent 3D primitives through a connectivity graph that respects the physical scan topology.

A two-part experimental campaign substantiated both the theory and the practical value of the framework. On synthetic shapes with analytic curvature, numerical results confirmed the predicted first-order convergence. On a large, unlabeled urban dataset, the method produced geometrically faithful classifications while running at ≈ 81 FPS on a single CPU core. Comparative studies showed clear gains in geometric quality, temporal stability, and computational cost over PCA-based and Euclidean clustering baselines.

In summary, by uniting first-principles differential geometry with a practical, scan-aware computation strategy, this work offers both a new lens for interpreting LiDAR data and a concrete, efficient toolset for large-scale scene understanding.

6.2 *Future directions*

Overcoming sensor specificity. A primary direction is to relax the rotating-multi-beam assumption and extend the discrete-geometric formalism to non-conical acquisition patterns (e.g., MEMS/Lissajous, raster, and other solid-state modalities). Concretely, this requires deriving appropriate scan-manifold parameterizations and the induced metric tensors, then re-formulating the discrete differential operators so they respect those geometries. While this is nontrivial, the extension is systematic: the core idea of exploiting the native scan ordering to obtain compact 1D (or low-dimensional) operators remains applicable and will guide the derivation for each sensor class (see discussion of assumptions in §3.1).

Integrating ego-motion into the geometry. To reduce sensitivity to imperfect deskewing, we will investigate motion-aware formulations that incorporate time/trajectory information directly into the operator design (for example, time-parameterized ring signals, short-window joint estimation of motion and curvature, or trajectory-corrected kernels). Such formulations aim to make local curvature estimates intrinsically robust to realistic platform motion without relying solely on preprocessing, improving stability in highly dynamic scenarios while keeping computational overhead small.

Early, selective inter-ring fusion. Finally, to recover vertical detail when needed while preserving the computational advantages of ring-intrinsic processing, we plan to explore compact mechanisms for limited cross-ring coupling. Candidate approaches include small 2D operators on narrow vertical neighborhoods (3-5 rings), graph-based vertical smoothing, or lightweight learned fusion heads that augment 1D descriptors only where data supports it. These hybrid strategies target improved corner/edge localization and planarity discrimination with modest additional cost, forming a practical middle ground between pure 1D processing and full 3D neighbourhood searches.

Conflict of interest

The authors declare no competing financial interest.

References

[1] Chazal F, Cohen-Steiner D, Lieutier A, Mérigot Q, Thibert B. Inference of curvature using tubular neighborhoods. In: Najman L, Romon P. (eds.) *Modern Approaches to Discrete Curvature*. Cham: Springer International Publishing; 2017. p.133-158.

[2] Chen JY, Lin CH. Neighborhood selection for differential coordinates of 3D point clouds. *International Journal of Innovative Computing Information and Control*. 2010; 6(6): 2393-2405.

[3] Günen M. Adaptive neighborhood size and effective geometric features selection for 3D scattered point cloud classification. *Applied Soft Computing*. 2022; 115: 108196. Available from: https://doi.org/10.1016/j.asoc.2021.108196.

[4] Xue R, Wang J, Ma Z. Efficient LiDAR point cloud geometry compression through neighborhood point attention. *arXiv:220812573*. 2022. Available from: https://doi.org/10.48550/arXiv.2208.12573.

[5] Jhaldiyal A, Chaudhary N. Semantic segmentation of 3D LiDAR data using deep learning: A review of projection-based methods. *Applied Intelligence*. 2023; 53(6): 6844-6855. Available from: https://doi.org/10.1007/s10489-022-03930-5.

[6] Dou W, Wang Y. Point cloud graph for LiDAR segmentation. *Measurement*. 2025; 242: 115851. Available from: https://doi.org/10.1016/j.measurement.2024.115851.

[7] Mandow A, Morales J, Gómez-Ruiz JA, García-Cerezo A. Optimizing scan homogeneity for building full-3D lidars based on rotating a multi-beam velodyne range-finder. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spain: IEEE; 2018. p.4788-4793.

[8] Morales J, Plaza-Leiva V, Mandow A, Gómez-Ruiz JA, Serón J, García-Cerezo A. Analysis of 3D scan measurement distribution with application to a multi-beam lidar on a rotating platform. *Sensors (Basel, Switzerland)*. 2018; 18(2): 395. Available from: https://doi.org/10.3390/s18020395.

[9] Lassiter H, Whitley T, Wilkinson B, Abd-Elrahman A. Scan pattern characterization of velodyne VLP-16 lidar sensor for UAS laser scanning. *Sensors*. 2020; 20(24): 7351. Available from: https://doi.org/10.3390/s20247351.

[10] Péntek Q, Allouis T, Strauss O, Fiorio C. Developing and validating a predictive model of measurement uncertainty for multi-beam lidars: Application to the Velodyne VLP-16. In: *2018 Eighth International Conference on Image Processing Theory, Tools and Applications (IPTA)*. Xi'an, China: IEEE; 2018. p.1-5.

[11] Wu B, Wan A, Yue X, Keutzer K. SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, Australia: IEEE; 2018. p.1887-1893.

[12] Alnaggar YA, Afifi M, Amer K, Elhelw M. Multi projection fusion for real-time semantic segmentation of 3D LiDAR point clouds. In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, USA: IEEE; 2021. p.1799-1808.

[13] Zhang Y, Zhou Z, David P, Yue X, Xi Z, Foroosh H. PolarNet: An improved grid representation for online LiDAR point clouds semantic segmentation. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, USA: IEEE; 2020. p.9598-9607.

[14] Zheng Y, Wang G, Pollefeys M, Wang H. Spherical frustum sparse convolution network for LiDAR point cloud semantic segmentation. *arXiv:231117491*. 2023. Available from: https://doi.org/10.48550/arXiv.2311.17491.

[15] Zhou H, Zhu X, Song X, Ma Y, Wang Z, Li H, et al. Cylinder3D: An effective 3D framework for driving-scene LiDAR semantic segmentation. *arXiv:200801550*. 2020. Available from: https://doi.org/10.48550/arXiv.2008.01550.

[16] Yu Z, Yang Y, Liu H, Cai D, He X. TransUPR: A transformer-based plug-and-play uncertain point refiner for LiDAR point cloud semantic segmentation. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Detroit, USA: IEEE; 2023. p.5864-5869.

[17] Sia CW, Lim K, Phang JTS. Comparative study of point cloud classification using deep learning neural networks. In: *2023 International Conference on Digital Applications, Transformation & Economy (ICDATE)*. Miri, Malaysia: IEEE; 2023. p.1-5.

[18] Jiang C, Huang K, Wu J, Wang X, Xiao J, Hussain A. PointGS: Bridging and fusing geometric and semantic space for 3D point cloud analysis. *Information Fusion*. 2022; 91: 316-326. Available from: https://doi.org/10.2139/ssrn.4186486.

[19] Lee DK, Ji SH, Park BY. PointNet and RandLA-Net algorithms for object detection using 3D point clouds. *Journal of the Society of Naval Architects of Korea*. 2022; 59(5): 330-338. Available from: https://doi.org/10.3744/snak.2022.59.5.330.

[20] Hou X, Yang Y, Liu H. Semantic segmentation network with adaptive neighborhood construction for 3D point cloud. In: *2023 IEEE 3rd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*. Chongqing, China: IEEE; 2023. p.219-223.

[21] Wang J, Fan W, Song X, Yao G, Bo M, Liu Z. NLA-GCL-Net: Semantic segmentation of large-scale surveying point clouds based on neighborhood label aggregation (NLA) and global context learning (GCL). *International Journal of Geographical Information Science*. 2024; 38(11): 2325-2347. Available from: https://doi.org/10.1080/13658816.2024.2382273.

[22] Hu Q, Yang B, Xie L, Rosa S, Guo Y, Wang Z, et al. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, USA: IEEE; 2020. p.11105-11114.

[23] Chen G. Point cloud key point extraction algorithm based on feature space value filtering. *Mobile Information Systems*. 2022; 2022(1): 1453537. Available from: https://doi.org/10.1155/2022/1453537.

[24] Hu Q, Yang B, Khalid S, Xiao W, Trigoni N, Markham A. Learning semantic segmentation of large-scale point clouds with random sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2021; 44(11): 8338-8354. Available from: https://doi.org/10.1109/TPAMI.2021.3083288.

[25] Khameneifar F, Ghorbani H. On the curvature estimation for noisy point cloud data via local quadric surface fitting. *Computer-Aided Design and Applications*. 2018; 16(1): 140-149.

[26] Cheng S, Zhou Y, Shi J, Zhang L. Curvature estimation to scattered point cloud. *Applied Mechanics and Materials*. 2010; 29-32: 1263-1267. Available from: https://doi.org/10.4028/www.scientific.net/AMM.29-32.1263.

[27] Yang X, Zheng J. Curvature tensor computation by piecewise surface interpolation. *Computer-Aided Design*. 2013; 45(12): 1639-1650. Available from: https://doi.org/10.1016/j.cad.2013.08.008.

[28] Colbry D, Shrikhande N. Discrete and continuous curvature computation for real data. In: *Intelligent Robots and Computer Vision XXXI: Algorithms and Techniques*. San Francisco, USA: SPIE; 2014. p.90250L.

[29] Tomek L, Mikula K. Discrete duality finite volume method with tangential redistribution of points for surfaces evolving by mean curvature. *ESAIM: Mathematical Modelling and Numerical Analysis*. 2019; 53(6): 1797-1840. Available from: https://doi.org/10.1051/m2an/2019040.

[30] Jiang G, Yan WY, Lichti DD. A maximum entropy-based optimal neighbor selection for multispectral airborne LiDAR point cloud classification. *IEEE Transactions on Geoscience and Remote Sensing*. 2023; 61: 1-18. Available from: https://doi.org/10.1109/TGRS.2023.3323963.

[31] Chakraborty D, Dey E. Segmentation of LiDAR point cloud data in urban areas using adaptive neighborhood selection technique. *PLOS ONE*. 2024; 19(7): e0307138. Available from: https://doi.org/10.1371/journal.pone.0307138.