MCGI: Manifold-Consistent Graph Indexing for Billion-Scale Disk-Resident Vector Search

Anonymous Author(s)

Abstract

Graph-based Approximate Nearest Neighbor (ANN) search often suffers from performance degradation in high-dimensional spaces due to the “Euclidean-Geodesic mismatch,” where greedy routing diverges from the underlying data manifold. To address this challenge, we propose Manifold-Consistent Graph Indexing (MCGI), a geometry-aware and disk-resident indexing method that leverages Local Intrinsic Dimensionality (LID) to dynamically adapt search strategies to the data’s intrinsic geometry. Unlike standard algorithms that treat dimensions uniformly, MCGI modulates its beam search budget based on in situ geometric analysis, eliminating the dependency on static hyperparameters. Theoretical analysis confirms that MCGI enables improved approximation guarantees by preserving manifold-consistent topological connectivity. We conducted extensive evaluations against three industry-standard baselines across five datasets ranging from million to billion scales. Empirically, MCGI achieves 5.8× higher throughput at 95% recall on the high-dimensional GIST1M compared to the state-of-the-art DiskANN. On the billion-scale SIFT1B and T2I-1B datasets, MCGI further validates its scalability by reducing high-recall query latency by 3× while maintaining performance parity on standard lower-dimensional benchmarks. [Dongfang: Probably need to add SPANN results]

CCS Concepts

• **Information systems → Top-k retrieval in databases**.

Keywords

Approximate Nearest Neighbor Search, Disk-resident Indexing, Local Intrinsic Dimensionality

1 Introduction

The advent of Large Language Models (LLMs) [4, 35] has fundamentally transformed the landscape of information retrieval and knowledge management. To address the inherent limitations of LLMs, such as hallucinations [15] and knowledge cutoff dates, Retrieval-Augmented Generation (RAG) [23] has emerged as a critical architectural paradigm. RAG relies heavily on the ability to retrieve semantically relevant context from massive corpora in real-time, typically via dense vector representations [20]. This dependency has placed Approximate Nearest Neighbor Search (ANNS) at the core of modern data infrastructure, demanding vector indices that can scale to billion-point datasets [31] while maintaining low latency and high recall under strict production constraints.

State-of-the-art ANNS solutions have noticeably converged on graph-based indices, with DiskANN (Vamana) [32] being a representative example for large-scale, SSD-resident workloads. These algorithms typically employ greedy routing on a proximity graph to navigate from an entry point to the query target. While such methods exhibit exceptional performance on standard benchmarks like SIFT1M [17] (128 dimensions), their efficiency degrades significantly in high-dimensional spaces, such as GIST1M (960 dimensions). This degradation is often attributed to the curse of dimensionality [2], where the distance contrast diminishes, and the Euclidean shortest path on the graph diverges from the geodesic path on the underlying data manifold. We refer to this phenomenon as the *Euclidean-Geodesic mismatch*. When the routing algorithm ignores the intrinsic geometry of the data, it performs excessive backtracking and disk I/O, rendering the search inefficient, particularly problematic in high-recall regimes where production systems usually expect Recall@10 ≥ 95%.

Our key insight is that high-dimensional real-world data is rarely uniformly distributed. Instead, it typically adheres to the Manifold Hypothesis [29, 33], residing on lower-dimensional structures embedded within the ambient space. Consequently, the search difficulty is not uniform across the dataset but is modulated by the Local Intrinsic Dimensionality (LID) [22]. In regions where the data manifold is flat (low LID), greedy routing is effective; however, in regions with high curvature or complex topology (high LID), standard greedy strategies fail to identify the correct descent direction. We argue that an optimal indexing strategy must be manifold-aware, dynamically allocating computational resources based on the local geometric complexity. To address these challenges, we introduce Manifold-Consistent Graph Indexing (MCGI), a geometry-aware disk-based indexing architecture designed to align Euclidean search with the underlying data manifold. By integrating LID estimation directly into the routing logic, MCGI adapts its traversal strategy to the local topology of the data.

Our contributions are as follows:

- We establish a theoretical method linking local intrinsic dimensionality to graph navigability, offering a justification for adaptive beam search on non-Euclidean manifolds.
- We develop a lightweight adaptive routing algorithm that dynamically modulates the search budget based on real-time geometric analysis. This design eliminates the dependency on static and manually tuned hyperparameters that often limit the adaptability of existing methods.
- We conduct comprehensive empirical evaluations against three industry-standard baselines across five datasets ranging from million to billion scales. Results demonstrate that MCGI achieves up to 5.8× higher throughput on high-dimensional vectors (e.g., 960-dimensional GIST1M) and reduces query latency by 3× on the billion-scale SIFT1B compared to DiskANN, confirming its robustness in mitigating both I/O and computational bottlenecks across diverse production environments.

2 Related Work

[Dongfang: TODO: Expand the references a bit]

2.1 Vector Indexing Paradigms

While traditional sparse retrieval methods like BM25 [28] rely on lexical matching, the surge of neural networks has shifted the focus to dense vector retrieval. In the memory-resident regime, graph-based indices, particularly Hierarchical Navigable Small World (HNSW) [25], have established state-of-the-art performance by enabling logarithmic complexity scaling. However, the high memory consumption of HNSW poses challenges for billion-scale datasets. To mitigate this, disk-based approaches have emerged. DiskANN (Vamana) [32] adapts the graph topology for SSDs by relaxing sparsity constraints to maximize neighborhood coverage. Concurrently, methods like SPANN [5] argue against pure graph traversal on disk due to random I/O latency, advocating instead for an inverted index (IVF) structure combined with centroid-based routing. Despite their differences, both DiskANN and SPANN, as well as their predecessors like NSG [8], are predominantly evaluated on standard benchmarks such as SIFT and DEEP with moderate dimensionality (96 to 128).

The above works largely rely on static routing parameters or centroid layouts that do not explicitly account for the local intrinsic dimensionality. In contrast, MCGI distinguishes itself by abandoning static routing configurations in favor of a geometry-aware strategy. By dynamically modulating the search budget based on estimated LID, our method aligns the graph traversal with the underlying manifold structure, thereby overcoming the efficiency bottlenecks that hinder rigid indexing schemes in high-dimensional spaces.

2.2 High-Dimensional Indexing

Early approaches relied on space-partitioning trees. The KD-tree [3] divides the space using axis-aligned hyperplanes, while the R-tree [11] utilizes hierarchical bounding rectangles. However, these strict partitioning schemes suffer from the curse of dimensionality, typically degrading to linear scan performance when the dimension exceeds 20. To address this, approximate methods like Locality-Sensitive Hashing (LSH) [7, 14] were introduced, offering sub-linear search time guarantees. Yet, achieving high recall with LSH often requires maintaining significant redundancy via multiple hash tables, resulting in excessive storage overhead. Another line of work involves subspace quantization, exemplified by Product Quantization (PQ) [17] and Optimized PQ (OPQ) [9], which decompose high-dimensional vectors into lower-dimensional subspaces for compression. Additionally, randomized structures like RP-Trees [6] attempt to adapt to the data geometry via random projections.

[Dongfang: Expand the following paragraph] MCGI differs from these approaches by retaining the connectivity benefits of graph traversal while avoiding the aggressive quantization loss or the storage redundancy of hashing methods.

2.3 Intrinsic Dimensionality

Theoretical analysis of nearest neighbor search often relies on characterizing the intrinsic difficulty of the dataset. Fundamental concepts such as the doubling dimension [10] and expansion dimension [19] provide asymptotic bounds on search complexity in growth-restricted metrics. Bridging theory and practice, Levina and Bickel [22] introduced maximum likelihood estimators for Local Intrinsic Dimensionality (LID), enabling robust estimation on real-world data. While subsequent works have utilized LID for tasks such as query hardness prediction [12] or detecting adversarial examples [24], these applications are typically passive, utilizing LID primarily for post-hoc analysis or pre-query estimation without altering the underlying index structure.

[Dongfang: Expand the following paragraph] MCGI diverges from this paradigm by employing LID as an active control signal. By dynamically modulating the graph traversal parameters based on local geometry, our method transforms LID from a descriptive metric into a prescriptive mechanism for efficient routing.

3 Methodology

3.1 Definitions and Preliminaries

We start by briefly introducing the notions of Local Intrinsic Dimensionality (LID) in the words of analysis. A full treatment can be found, for example, in Houle [13].

Definition 3.1 (Local Intrinsic Dimensionality)**.** Let $\mathcal{X}$ be a domain equipped with a distance measure $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$. For a reference point $x \in \mathcal{X}$, let $F_x(r) = \mathbb{P}(d(x, Y) \leq r)$ denote the cumulative distribution function (CDF) of the distance between x and a random variable Y drawn from the underlying data distribution. The Local Intrinsic Dimensionality (LID) of x, denoted as $\mathrm{LID}(x)$, is defined as the intrinsic growth rate of the probability measure within the neighborhood of x:

$$\mathrm{LID}(x) \triangleq \lim_{r \to 0} \frac{r \cdot F'_x(r)}{F_x(r)} = \lim_{r \to 0} \frac{d \ln F_x(r)}{d \ln r}, \tag{1}$$

provided the limit exists and $F_x(r)$ is continuously differentiable for $r > 0$.

Remark 3.2 (Institution of LID). The definition of LID can be understood as a measure of the multiplicative growth rate of the volume of a ball centered at x with radius r as r approaches 0. Let D denote the dimensionality of the ambient space. If the data lies on a local D-dimensional manifold, then the CDF around an infinitely small neighborhood of x satisfies:

$$F_x(r) \approx C \cdot r^D, \tag{2}$$

where C is a constant. Thus, the following holds:

$$F'_x(r) \approx C \cdot D \cdot r^{D-1}. \tag{3}$$

Combining equations (2) and (3), we get:

$$D \approx \frac{F'_x(r)}{F_x(r)} \cdot r, \tag{4}$$

thus Eq. (1).

While Eq. 3.1 provides an intuitive closed-form formula for intrinsic dimensionality, in practice we usually do not have access to the true CDF $F_x(r)$. Luckily, we can estimate LID from a finite sample of distances from x to its neighbors using Maximum Likelihood Estimation (MLE) as proposed by [22]. According to [1], LID can be estimated as follows.

Definition 3.3 (LID Maximum Likelihood Estimator)**.** Given a reference point x and its k-nearest neighbors determined by the distance measure d, let $r_i = d(x, v_i)$ denote the distance to the i-th nearest neighbor, sorted such that $r_1 \leq \cdots \leq r_k$. Following the formulation in [1], which adapts the Hill estimator for intrinsic dimensionality, the LID at x is estimated as:

$$\widehat{\mathrm{LID}}(x) = -\left(\frac{1}{k}\sum_{i=1}^{k} \ln \frac{r_i}{r_k}\right)^{-1}. \tag{5}$$

3.2 Mapping Function

The primary goal of Manifold-Consistent Graph Indexing is that the graph topology should adapt to the local geometric complexity. In regions where the Local Intrinsic Dimensionality (LID) is low, the data manifold approximates a flat Euclidean subspace. In such isotropic regions, the Euclidean metric is a reliable proxy for geodesic distance, allowing for aggressive edge pruning (larger α in [32]) to permit long-range direct connections without risking semantic shortcuts. Conversely, regions with high LID typically exhibit significant curvature, noise, or singularity. In this case, the Euclidean distance often violates the manifold geodesic structure. To preserve topological fidelity, the indexing algorithm must adopt a conservative pruning strategy (smaller α in [32]), thereby forcing the search to take smaller, safer steps along the manifold surface.

Let $u \in V$ be a node in the graph, and $\widehat{\mathrm{LID}}(u)$ be its estimated LID. We define the pruning parameter $\alpha(u)$ as:

$$\alpha(u) \triangleq \Phi(\widehat{\mathrm{LID}}(u)). \tag{6}$$

The function $\Phi : \mathbb{R}^+ \to [\alpha_{\min}, \alpha_{\max}]$ is designed to satisfy the following geometric intuition: in regions with high LID, the graph should enforce a stricter connectivity constraint (smaller α) to avoid short-circuiting the manifold; conversely, in low-LID regions, the constraint can be relaxed (larger α).

To ensure the mapping is robust across datasets with varying complexity scales, we employ Z-score normalization based on the empirical distribution of the LID estimates. We first compute the normalized score $z(u)$:

$$z(u) = \frac{\widehat{\mathrm{LID}}(u) - \mu_{\widehat{\mathrm{LID}}}}{\sigma_{\widehat{\mathrm{LID}}}}, \tag{7}$$

where $\mu_{\widehat{\mathrm{LID}}}$ and $\sigma_{\widehat{\mathrm{LID}}}$ denote the mean and standard deviation of the set of estimated LID values $\{\widehat{\mathrm{LID}}(v) \mid v \in V\}$ computed across the entire graph.

We then formulate Φ using a logistic function to smoothly map the Z-score to the operational range $[\alpha_{\min}, \alpha_{\max}]$:

$$\Phi(\widehat{\mathrm{LID}}(u)) = \alpha_{\min} + \frac{\alpha_{\max} - \alpha_{\min}}{1 + \exp(z(u))}. \tag{8}$$

We employ the logistic function over a linear mapping to exploit its saturation properties. LID estimates often exhibit heavy-tailed distributions with extreme outliers. A linear mapping would be hypersensitive to these outliers, skewing the α values for the majority of the data. The logistic function acts as a robust soft-thresholding mechanism: it reduces the variance in the high-LID and low-LID tails (saturating towards $\alpha_{\min}$ and $\alpha_{\max}$, respectively) while maintaining sensitivity in the transition region around the population mean. We set $\alpha_{\min} = 1.0$ and $\alpha_{\max} = 1.5$ following standard practices in graph indexing [32], unless otherwise stated. This formulation ensures that nodes with average complexity ($z(u) \approx 0$) are assigned $\alpha \approx 1.25$, while nodes with significantly higher complexity ($z(u) > 0$) are penalized with a stricter α approaching the limit of 1.0.

The mapping function Φ satisfies the following geometric properties that are essential for stable graph construction: Monotonicity and Boundedness.

Proposition 3.4 (Monotonicity). *The mapping function Φ is strictly decreasing with respect to the estimated local intrinsic dimensionality. Formally, given that the standard deviation of the LID estimates $\sigma_{\widehat{LID}} > 0$ and the pruning range $\alpha_{\max} > \alpha_{\min}$, the derivative satisfies:*

$$\frac{d\Phi}{d\widehat{LID}(u)} < 0. \tag{9}$$

[Dongfang: Move full proof to appendix]

Proof. Let $L = \widehat{\mathrm{LID}}(u)$ be the independent variable. We define the normalized Z-score z as a function of L:

$$z(L) = \frac{L - \mu_{\widehat{\mathrm{LID}}}}{\sigma_{\widehat{\mathrm{LID}}}}. \tag{10}$$

The mapping function is defined as:

$$\Phi(L) = \alpha_{\min} + \frac{C}{1 + \exp(z(L))}, \tag{11}$$

where $C = \alpha_{\max} - \alpha_{\min}$. Since we strictly require $\alpha_{\max} > \alpha_{\min}$, it follows that $C > 0$. To determine the sign of the gradient, we apply the chain rule:

$$\frac{d\Phi}{dL} = \frac{d\Phi}{dz} \cdot \frac{dz}{dL}. \tag{12}$$

First, we differentiate the Z-score term with respect to L:

$$\frac{dz}{dL} = \frac{1}{\sigma_{\widehat{\mathrm{LID}}}}. \tag{13}$$

Next, we differentiate the logistic component Φ with respect to z:

$$\frac{d\Phi}{dz} = \frac{d}{dz}\left(\alpha_{\min} + C(1 + e^z)^{-1}\right) \tag{14}$$

$$= C \cdot (-1) \cdot (1 + e^z)^{-2} \cdot \frac{d}{dz}(1 + e^z) \tag{15}$$

$$= -C \cdot \frac{e^z}{(1 + e^z)^2}. \tag{16}$$

Combining these terms yields the full derivative:

$$\frac{d\Phi}{dL} = -\frac{C}{\sigma_{\widehat{\mathrm{LID}}}} \cdot \frac{e^z}{(1 + e^z)^2}. \tag{17}$$

We analyze the sign of each component:

- The operational range constant $C > 0$.
- The standard deviation $\sigma_{\widehat{\mathrm{LID}}} > 0$, assuming the dataset exhibits non-zero geometric variance.
- The exponential function $e^z > 0$ for all $z \in \mathbb{R}$.
- The denominator $(1 + e^z)^2 > 0$.

Therefore, the term $\frac{C}{\sigma_{\widehat{\mathrm{LID}}}} \frac{e^z}{(1+e^z)^2}$ is strictly positive. The leading negative sign guarantees that $\frac{d\Phi}{dL} < 0$. This confirms that the pruning parameter α strictly decreases as the local geometric complexity increases, thereby enforcing a more conservative graph topology in high-LID regions to prevent topology distortion in complex regions. □

Proposition 3.5 (Boundedness). *The pruning parameter $\alpha(u)$ derived from the mapping function is strictly bounded within the prescribed operational interval. For any node u with a finite LID estimate:*

$$\alpha_{\min} < \alpha(u) < \alpha_{\max}. \tag{18}$$

[Dongfang: Move proofs to appendix]

Proof. Let $S(u)$ denote the logistic component of the mapping function:

$$S(u) = \frac{1}{1 + \exp(z(u))}. \tag{19}$$

For any finite input $\widehat{\text{LID}}(u)$, the Z-score $z(u)$ is finite. The exponential function maps the real line to the positive real line, i.e., $\exp(z(u)) \in (0, \infty)$. Consequently, the denominator lies in the interval $(1, \infty)$. Taking the reciprocal yields the bounds for the logistic component:

$$0 < S(u) < 1. \tag{20}$$

Substituting $S(u)$ back into the definition of Φ:

$$\alpha(u) = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot S(u). \tag{21}$$

Since $(\alpha_{\max} - \alpha_{\min}) > 0$, we can apply the inequality boundaries:

$$\alpha(u) > \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot 0 = \alpha_{\min}, \tag{22}$$

$$\alpha(u) < \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot 1 = \alpha_{\max}. \tag{23}$$

This proves that the topology is strictly confined. The pruning behavior never exceeds the relaxation upper limit ($\alpha_{\max}$) and never becomes stricter than the lower limit ($\alpha_{\min}$), ensuring graph connectivity and preventing degree explosion. □

3.3 Manifold-Consistent Graph Indexing

The MCGI algorithm (Algorithm 1) introduces a geometric calibration phase to a graph indexing procedure. Unlike static methods that apply a uniform connectivity rule, MCGI executes in two distinct stages to ensure the topology respects the manifold structure.

Phase 1: Geometric Calibration. Before modifying the graph topology, the system first performs a global analysis of the dataset geometry. We estimate the LID for every point and aggregate the population statistics (μ, σ) defined in Section 3.2. This phase "freezes" the geometric profile of the dataset. By pre-computing these statistics, we decouple the complexity estimation from the graph update loop, ensuring that the mapping function Φ remains stable and computationally efficient during the intensive edge-selection process.

Phase 2: Manifold-Consistent Refinement. The index construction follows an iterative refinement strategy. Let $\mathcal{N}(u)$ denote the set of neighbors for node u in the graph G. In each iteration, the algorithm dynamically updates $\mathcal{N}(u)$ by:

1. Queries the pre-computed geometric profile to determine the node-specific constraint $\alpha(u)$.
2. Explores the graph to identify a candidate pool C.
3. Filters connections using the dynamic occlusion criterion.

3.3.1 Computational Complexity. We analyze the computational complexity as follows. Let N be the total number of vectors in dataset $\mathcal{X}$, R the maximum degree, L the construction beam width, and T the number of refinement iterations.

Algorithm 1 Manifold-Consistent Graph Indexing (MCGI)

```
Input: Dataset $X$, Max Degree $R$, Beam Width $L$
Output: Optimized Graph $G$

// Phase 1: Geometric Calibration
$\mathcal{L} \leftarrow$ ParallelEstimateLID($X$)
$\mu \leftarrow$ Mean($\mathcal{L}$)
$\sigma \leftarrow$ StdDev($\mathcal{L}$)
for each node $u \in V$ in parallel do
    $z_u \leftarrow (\mathcal{L}[u] - \mu)/\sigma$
    $\alpha_u \leftarrow \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})/(1 + \exp(z_u))$
end for

// Phase 2: Topology Refinement
$G \leftarrow$ RandomGraph($X, R$)
for $iter \leftarrow 1$ to $MaxIter$ do
    for each node $u \in G$ in parallel do
        $C \leftarrow$ GreedySearch($u, G, L$)
        $\mathcal{N}_{new} \leftarrow \emptyset$
        for $v \in$ SortByDistance($C \cup \mathcal{N}(u)$) do
            $pruned \leftarrow$ False
            for $n \in \mathcal{N}_{new}$ do
                if $\alpha_u \cdot d(n, v) \leq d(u, v)$ then
                    $pruned \leftarrow$ True; break
                end if
            end for
            if not $pruned$ and $|\mathcal{N}_{new}| < R$ then
                $\mathcal{N}_{new}$.add($v$)
            end if
        end for
        $\mathcal{N}(u) \leftarrow \mathcal{N}_{new}$
    end for
end for
```

- *Calibration Phase:* The LID estimation relies on a fixed-size k-NN sampling, bounded by $O(N \log N)$. The subsequent parameter mapping is a linear scan $O(N)$.
- *Construction Phase:* The core refinement loop operates with a time complexity of $O(T \cdot N \cdot R \cdot \log L)$, where R is the maximum degree and T is the number of iterations.

Since the calibration is a one-pass pre-processing step (as opposed to the multi-pass iterative refinedment in Phase 2), the total time complexity remains dominated by the graph refinement, ensuring that MCGI scales linearly with N, consistent with state-of-the-art methods like DiskANN [32].

3.3.2 Online MCGI. Calculating the exact Local Intrinsic Dimensionality (LID) for the entire dataset serves as a computational bottleneck on billion-scale data, typically requiring $O(N^2)$ complexity or a pre-built index. To address this, we propose *Online-MCGI* (Algorithm 2), which integrates LID estimation directly into the graph construction process. First, we bootstrap global statistics (μ, σ) using a small random subset of the data, reducing the preprocessing cost to negligible levels. Then, during the iterative graph refinement, we estimate the local LID $\hat{\ell}_u$ for each node u on-the-fly using its current candidate set C obtained from the greedy search. While

Algorithm 2 Online Manifold-Consistent Graph Indexing

```
Input: Dataset $X$, Max Degree $R$, Beam Width $L$, Sample Rate $S$
Output: Optimized Graph $G$

// Phase 1: Bootstrap Statistics (Fast Approximation)
$X_{sample} \leftarrow$ RandomSample($X, S$)
$\mathcal{L}_{sample} \leftarrow$ EstimateLID($X_{sample}$) {Compute LID on subset}
$\mu \leftarrow$ Mean($\mathcal{L}_{sample}$), $\sigma \leftarrow$ StdDev($\mathcal{L}_{sample}$)

// Phase 2: Online Topology Refinement
$G \leftarrow$ RandomGraph($X, R$)
for $iter \leftarrow 1$ to $MaxIter$ do
  for each node $u \in G$ in parallel do
    $C \leftarrow$ GreedySearch($u, G, L$)
    // Online LID Estimation & Parameter Adaptation
    $\hat{\ell}_u \leftarrow$ ComputeLID($u, C$) {Estimate from neighbors}
    $z_u \leftarrow (\hat{\ell}_u - \mu)/\sigma$
    $\alpha_u \leftarrow \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})/(1 + \exp(z_u))$
    $\mathcal{N}_{new} \leftarrow \emptyset$
    for $v \in$ SortByDistance($C \cup \mathcal{N}(u)$) do
      $pruned \leftarrow$ False
      for $n \in \mathcal{N}_{new}$ do
        // Pruning with dynamic $\alpha_u$
        if $\alpha_u \cdot d(n, v) \leq d(u, v)$ then
          $pruned \leftarrow$ True; break
        end if
      end for
      if not $pruned$ and $|\mathcal{N}_{new}| < R$ then
        $\mathcal{N}_{new}$.add($v$)
      end if
    end for
    $\mathcal{N}(u) \leftarrow \mathcal{N}_{new}$
  end for
end for
```

the initial estimates may be noisy, they progressively converge to the true manifold dimensionality as the neighbor quality improves with each iteration. This approach allows the pruning parameter α_u to dynamically adapt to local geometric complexity without incurring the prohibitive cost of offline global analysis.

4 Theoretical Analysis

4.1 Geometric Complexity and Adaptive Routing

The efficiency of greedy routing on a proximity graph is intrinsically tied to the local geometric properties of the underlying data manifold $\mathcal{M}$. Under the Manifold Hypothesis, the search space is locally characterized by the Local Intrinsic Dimensionality (LID) at query q. We formalize the relationship between this local geometry and the computational cost required to identify the nearest neighbor.

Lemma 4.1 (Local Complexity Lower Bound). *Consistent with the complexity bounds established for growth-restricted metrics [19], for a query q on a manifold $\mathcal{M}$, the expected number of distance evaluations N_{dist} required for successful greedy routing scales exponentially with the local intrinsic dimensionality $d = LID(q)$:*

$$\mathbb{E}[N_{dist}] \geq \Omega\left(\frac{1}{\sqrt{d}} \cdot \exp(\lambda \cdot d)\right), \tag{24}$$

where $\lambda > 0$ is a geometric constant derived from the required convergence rate.

Proof. Let the local dataset around a query q be modeled as a uniform distribution on a manifold of intrinsic dimension d. Consider a greedy routing step where the algorithm is currently at a node u with distance $r = \|u - q\|$.

1. Geometric Condition for Improvement. To make progress, the algorithm must identify a neighbor v such that the distance to the query is reduced by a factor ϵ (where $0 < \epsilon < 1$), i.e., $\|v - q\| \leq (1 - \epsilon)r$. Let $s = \|u - v\|$ be the distance between the current node and the candidate neighbor (the step size). Applying the Law of Cosines to triangle $\triangle uvq$, we have:

$$\|v - q\|^2 = r^2 + s^2 - 2rs\cos\theta, \tag{25}$$

where θ is the angle between the vector $\vec{uq}$ and $\vec{uv}$. The improvement condition $\|v - q\|^2 \leq (1 - \epsilon)^2 r^2$ implies:

$$\begin{aligned} r^2 + s^2 - 2rs\cos\theta &\leq (1 - \epsilon)^2 r^2, \\ 2rs\cos\theta &\geq r^2 + s^2 - r^2(1 - \epsilon)^2, \\ \cos\theta &\geq \frac{r^2 + s^2 - r^2(1 - \epsilon)^2}{2rs}. \end{aligned} \tag{26}$$

Assuming the step size is small relative to the distance (i.e., $s \ll r$), the right-hand side is dominated by $1 - (1 - \epsilon)^2 \approx 2\epsilon$, yielding a critical angle threshold θ_{max} such that any successful neighbor must fall within the cone defined by $0 \leq \theta \leq \theta_{max}$.

2. Volume Concentration of the Improving Region. In the tangent space at u, the directions of neighbors are uniformly distributed on the unit hypersphere $\mathbb{S}^{d-1}$. The probability $P_{success}$ of finding a neighbor in the improving cone corresponds to the ratio of the surface area of the spherical cap $C(\theta_{max})$ to the total surface area of $\mathbb{S}^{d-1}$. This ratio is given by the regularized incomplete beta function, which can be expressed in terms of the integral over the polar angle ϕ:

$$P_{success} = \frac{\text{Area}(C(\theta_{max}))}{\text{Area}(\mathbb{S}^{d-1})} = \frac{\int_0^{\theta_{max}} \sin^{d-2}\phi \, d\phi}{\int_0^{\pi} \sin^{d-2}\phi \, d\phi}. \tag{27}$$

3. Asymptotic Analysis. We analyze the asymptotic behavior of this ratio as $d \to \infty$. The denominator relates to the surface area of the hypersphere and can be evaluated using the standard integral identity for powers of sine. Exploiting the symmetry of $\sin\phi$ around $\pi/2$, we have:

$$\begin{aligned} \int_0^{\pi} \sin^{d-2}\phi \, d\phi &= 2\int_0^{\frac{\pi}{2}} \sin^{d-2}\phi \, d\phi \\ &= 2 \cdot \frac{\sqrt{\pi}}{2} \frac{\Gamma(\frac{d-1}{2})}{\Gamma(\frac{d}{2})} = \sqrt{\pi}\frac{\Gamma(\frac{d-1}{2})}{\Gamma(\frac{d}{2})}. \end{aligned} \tag{28}$$

For high-dimensional spaces ($d \gg 1$), we apply the asymptotic property of the Gamma function ratio, $\frac{\Gamma(x+a)}{\Gamma(x)} \approx x^a$. Setting $x = d/2$

and $a = -1/2$, we obtain the approximation:

$$\sqrt{\pi}\frac{\Gamma(\frac{d}{2}-\frac{1}{2})}{\Gamma(\frac{d}{2})} \approx \sqrt{\pi}\left(\frac{d}{2}\right)^{-\frac{1}{2}} = \sqrt{\frac{2\pi}{d}}. \tag{29}$$

For the numerator, we exploit the concentration of measure near the integration boundary θ_{max}. Since the integrand $\sin^{d-2}\phi$ decays exponentially fast away from θ_{max}, the integral is dominated by the contribution near the upper limit. We apply integration by parts to extract the leading order term. Recall that $\frac{d}{d\phi}(\sin^{d-1}\phi) = (d-1)\sin^{d-2}\phi\cos\phi$. We rewrite the integrand as:

$$\int_0^{\theta_{max}} \sin^{d-2}\phi\, d\phi = \int_0^{\theta_{max}} \frac{1}{(d-1)\cos\phi}\frac{d}{d\phi}(\sin^{d-1}\phi)\, d\phi. \tag{30}$$

Neglecting lower-order terms generated by the derivative of $\frac{1}{\cos\phi}$, the integral approximates to the boundary term:

$$\left[\frac{\sin^{d-1}\phi}{(d-1)\cos\phi}\right]_0^{\theta_{max}} = \frac{\sin^{d-1}\theta_{max}}{(d-1)\cos\theta_{max}}. \tag{31}$$

Substituting these approximations back into the probability ratio, we obtain:

$$P_{success} \approx \frac{1}{\sqrt{2\pi}\cos\theta_{max}} \cdot \frac{\sqrt{d}}{d-1} \cdot (\sin\theta_{max})^{d-1}. \tag{32}$$

For large d, the term $\frac{\sqrt{d}}{d-1}$ is asymptotically equivalent to $d^{-1/2}$. Since θ_{max} is fixed by the geometric constraints, the trigonometric factors are constant relative to d. Let $\gamma = \sin\theta_{max} < 1$. The probability scales as:

$$P_{success} \propto d^{-1/2} \cdot \gamma^d = d^{-1/2}\exp(d\ln\gamma). \tag{33}$$

Defining the decay rate $\lambda = -\ln\gamma > 0$ (since $\gamma < 1$), we strictly obtain:

$$P_{success} \propto d^{-1/2}\exp(-\lambda \cdot d). \tag{34}$$

4. Complexity Bound. The expected number of distance evaluations follows $\mathbb{E}[N_{dist}] = 1/P_{success}$. Thus:

$$\mathbb{E}[N_{dist}] \geq \Omega\left(\sqrt{d}\cdot\exp(\lambda\cdot d)\right). \tag{35}$$

This confirms that the routing cost is dominated by the exponential term $\exp(\lambda \cdot \mathrm{LID}(q))$, necessitating the adaptive beam width design in MCGI. □

Lemma 4.1 reveals a fundamental geometric barrier: for a fixed budget L, the probability of routing failure grows exponentially in high-LID regions. To address this, MCGI adopts an *Iso-Recall* strategy, aiming to homogenize the search reliability across the heterogeneous manifold.

Proposition 4.2 (Optimal Budget Allocation). *To maintain a uniform bound on the routing failure probability δ across the manifold (i.e., $\mathbb{P}(fail|q) \leq \delta$ for all $q \in \mathcal{M}$), the search beam width $L(q)$ must scale exponentially with the local intrinsic dimensionality:*

$$L(q) \propto \exp\left(\lambda \cdot LID(q)\right). \tag{36}$$

Proof. Consider a beam search with width L. The probability of failing to find a descent step in one round is approximately $(1-P_{success})^L$. To guarantee a recall target, we enforce a constant upper bound δ on the failure probability:

$$(1-P_{success})^L \leq \delta. \tag{37}$$

Taking the natural logarithm and using the approximation $\ln(1-x) \approx -x$ for small $P_{success}$, we require:

$$-L\cdot P_{success} \leq \ln\delta \implies L \geq \frac{-\ln\delta}{P_{success}}. \tag{38}$$

Substituting the bound from Lemma 4.1 ($P_{success} \propto \exp(-\lambda \cdot \mathrm{LID}(q))$), the necessary budget becomes:

$$L(q) \geq C_\delta \cdot \exp(\lambda\cdot\mathrm{LID}(q)), \tag{39}$$

where $C_\delta = -\ln\delta$. □

Proposition 4.2 demonstrates that the exponential mapping in MCGI is not merely a heuristic, but the *necessary condition* for decoupling search recall from geometric complexity. By enforcing this relationship, MCGI achieves a *topological isomorphism* between the search budget and the manifold structure: computational resources are dynamically allocated to strictly *counteract* the exponential decay of routing probability in high-LID regions.

4.2 Topological Fidelity and Connectivity

A primary theoretical concern with aggressive edge pruning is the potential fracture of the connectivity backbone [36]. We prove that MCGI guarantees global reachability by strictly preserving the underlying manifold skeleton.

The edge selection in MCGI is governed by the adaptive pruning parameter $\alpha(u)$. Specifically, an edge (u,v) is pruned if there exists a witness node n such that $\alpha(u)\cdot d(n,v) \leq d(u,v)$. This condition defines an *exclusion region* around the midpoint of u and v. As $\alpha(u)$ increases, this region shrinks, making pruning more conservative.

Proposition 4.3 (Connectivity Preservation). *Let G_{EMST} and G_{RNG} denote the Euclidean Minimum Spanning Tree and the Relative Neighborhood Graph, respectively. For any configuration of points in general position, provided that $\alpha(u) \geq 1.0$ for all $u \in V$, the graph G_{MCGI} satisfies the strict inclusion hierarchy:*

$$E_{EMST} \subseteq E_{RNG} \subseteq E_{MCGI}. \tag{40}$$

Consequently, G_{MCGI} is connected.

Proof. The proof proceeds in two steps. First, we invoke the classical result established by Toussaint [34], which proves that the Relative Neighborhood Graph (RNG) is a supergraph of the EMST for any finite set of points in a metric space:

$$E_{EMST} \subseteq E_{RNG}. \tag{41}$$

Since the EMST connects all vertices in a single component, G_{RNG} is necessarily connected.

Second, we show that $E_{RNG} \subseteq E_{MCGI}$. The RNG is defined by the strict condition that an edge (u,v) exists iff no witness n satisfies $d(n,v) \leq d(u,v)$ (and $d(n,u) \leq d(u,v)$). This corresponds to our pruning condition with $\alpha = 1.0$. In MCGI, we enforce $\alpha(u) \geq 1.0$. The condition for pruning becomes stricter: a witness n must satisfy $d(n,v) \leq \frac{1}{\alpha(u)}d(u,v)$. Since $\frac{1}{\alpha(u)} \leq 1$, any witness that prunes an edge in MCGI would also prune it in the RNG, but the converse does not hold. Geometrically, the exclusion region of MCGI is strictly contained within the RNG lune:

$$\mathcal{R}_{MCGI} \subset \mathcal{R}_{RNG}. \tag{42}$$

Thus, MCGI retains all edges present in the RNG (plus additional shortcuts), inheriting the global connectivity of the EMST derived in [34]. □

Proposition 4.3 ensures that there are no structural dead ends. While Lemma 4.1 dictates the *cost* of routing, Proposition 4.3 ensures the *possibility* of routing. Even in the worst-case scenario where heuristics fail, a path exists from the entry point to any target via the edges of the underlying EMST, provided the graph traversal algorithm (beam search) has sufficient width to discover these backbone links.

5 Experimental Evaluation

We implemented MCGI on top of the official C++ codebase of DiskANN [32]. MCGI implementation is open-sourced as a subproject of the *AdaDisk* project, available at: https://github.com/hpdic/AdaDisk. To ensure fair comparison and production-level efficiency, we integrated the Maximum Likelihood Estimator (MLE) for LID directly into the graph building process. The core distance computations were optimized using AVX SIMD instructions to maximize hardware utilization. All latency-critical components, including the dynamic candidate list management and the LID-aware routing logic, were implemented in C++ to avoid the overhead of high-level language interpreters. We evaluate the performance of MCGI against state-of-the-art disk-based and memory-mapped approximate nearest neighbor (ANN) search algorithms. We focus on answering the following research questions:

- RQ1 (High-Dimensional Effectiveness): How does MCGI perform compared to baselines on high-dimensional data (e.g., GIST1M) where the curse of dimensionality typically degrades the efficiency of graph-based indices?
- RQ2 (High-Recall Efficiency): Can MCGI maintain high query throughput (QPS) under strict recall requirements (e.g., Recall@10 $\geq$ 95%), making it suitable for latency-critical production environments?
- RQ3 (Billion-Scale Scalability): Does the proposed method scale effectively to billion-scale datasets, including SIFT1B and T2I-1B, and demonstrate superiority over industry-standard baselines across both I/O-bound and compute-bound workloads?
- RQ4 (Resource Efficiency): Does the manifold-aware routing strategy translate to lower I/O overhead and reduced query latency on modern hardware (e.g., NVMe SSDs and AVX2 enabled CPUs)?

5.1 Experimental Setup

Platform and Environment. All experiments were conducted on the Chameleon Cloud [21] platform using a compute node equipped with dual-socket Intel Xeon Platinum 8380 CPUs (Ice Lake architecture, 2.30GHz, 80 cores total) and 256 GiB of RAM. To simulate a cost-effective large-scale retrieval scenario, the indices are stored on a single 480 GB Micron 5300 PRO Enterprise SSD. The operating system is Ubuntu 24.04 LTS. All algorithms were compiled using GCC 11.4 with -O3 and AVX2 optimizations enabled to fully utilize the hardware instruction set.

Datasets. We evaluate our method on four standard benchmarks, ranging from million-scale to billion-scale, to comprehensively test robustness across varying intrinsic dimensionalities and data volumes. In the following descriptions, N denotes the number of base vectors and D represents the vector dimensionality:

- SIFT1M [17] ($N = 10^6, D = 128$): A standard computer vision dataset using Euclidean distance (L_2).
- GloVe-100 [26] ($N = 1.2 \times 10^6, D = 100$): Word embedding vectors measuring semantic similarity. Following standard practice, we normalize the vectors to unit length and use Euclidean distance as a proxy for Cosine similarity.
- GIST1M [17] ($N = 10^6, D = 960$): A high-dimensional dataset representing global image features. This dataset is particularly challenging for index structures due to the sparsity of the space and the high intrinsic dimensionality.
- SIFT1B [18] ($N = 10^9, D = 128$): A billion-scale dataset used to evaluate the scalability and I/O efficiency of our method. It shares the same feature distribution as SIFT1M but scales the volume by three orders of magnitude, representing a realistic industrial scenario.
- T2I-1B [30] ($N = 10^9, D = 200$): A billion-scale dataset consisting of CLIP [27] embeddings derived from the LAION-5B dataset. Unlike visual descriptors like SIFT, T2I-1B represents a cross-modal retrieval scenario with a more complex manifold structure and higher intrinsic dimensionality, posing greater challenges for graph-based indexing algorithms.

Baselines. We compare MCGI against two baselines, each serving a distinct role:

- DiskANN (Vamana) [32]: The state-of-the-art disk-based graph index. Since MCGI builds upon the Vamana architecture, this comparison isolates the specific gains derived from our manifold-aware routing strategy.
- IVF-Flat (Faiss) [16]: An industry-standard inverted index serving as a performance roofline. Although running in memory-mapped mode, its sequential scanning pattern allows aggressive OS caching, effectively simulating in-memory performance. We include it to benchmark how closely our disk-resident solution approaches the throughput limits of memory-resident systems.
- HNSW [25]: The state-of-the-art in-memory graph index known for its superior recall-latency trade-off. Unlike disk-resident approaches, HNSW maintains the entire hierarchical graph topology in DRAM, completely eliminating I/O latency. We include it to establish a performance ceiling for graph-based search, allowing us to quantify the gap between our disk-resident solution and the theoretical limit of unconstrained memory-resident traversal.

Evaluation Metrics. We adhere to the standard evaluation protocol for ANN search. We measure Recall@10 against QPS (Queries Per Second). Additionally, we report the query latency at critical high-recall operating points (e.g., 95% Recall) to assess the tail latency characteristics.

Table 1: Peak QPS at high recalls.

Algorithm	Data Locality	Recall@10 ≥ 95%
DiskANN [32]	disk-resident	64.7
MCGI (this work)	disk-resident	375.1
IVF-Flat [16]	in-memory	590.5
HNSW [25]	in-memory	2,165

5.2 High-Dimensional Effectiveness (RQ1)

The primary advantage of MCGI lies in its robust handling of high-dimensional spaces where traditional disk-based graph indices typically degrade. Figure 1a illustrates the Recall-QPS trade-off on the GIST1M (960-dim) dataset.

As expected, the in-memory baseline, Faiss (IVF), exhibits exceptional throughput. This result serves as a performance roofline, representing the theoretical upper bound achievable when the entire index resides in DRAM, thereby completely eliminating the overhead of disk I/O. In contrast, disk-based indices like DiskANN and MCGI operate under the strict constraints of SSD latency, where performance is bounded by random I/O operations (IOPS) rather than memory bandwidth. Under these harsh conditions, the baseline DiskANN struggles, achieving approximately 64 QPS at 95% recall. However, MCGI significantly mitigates this I/O bottleneck. By leveraging manifold-aware routing to minimize strictly necessary disk reads, MCGI achieves 375 QPS, a 5.8× speedup over DiskANN. This result demonstrates that MCGI can drive disk-resident graph search to performance levels that begin to rival in-memory systems, effectively closing the gap between disk-based scalability and in-memory efficiency.

To ensure that our optimizations for high-dimensional manifolds do not negatively impact performance on standard tasks, we also evaluate MCGI on the lower-dimensional SIFT1M and GloVe-100 datasets. Figures 1b and 1c present the results. On both datasets, MCGI achieves performance parity with DiskANN. For instance, on SIFT1M, the curves for MCGI and DiskANN are nearly identical, converging to approximately 720 QPS at 98% recall. This indicates that our adaptive routing mechanism is robust. It identifies the simpler geometry of low-dimensional data and reduces to standard greedy search, incurring no overhead. This makes MCGI a general-purpose solution that matches state-of-the-art performance on simple tasks while unlocking speedups on challenging workloads.

5.3 High-Recall Efficiency (RQ2)

Real-world applications typically demand strict accuracy guarantees, such as Recall@10 being greater than or equal to 95%. We analyze the performance stability of all methods under these constraints, as summarized in Table 1.

HNSW establishes the performance ceiling for high-dimensional search, achieving a staggering 2,165 QPS by leveraging efficient graph traversal within the DRAM subsystem. While IVF-Flat provides a solid in-memory baseline (590.5 QPS) through contiguous scanning, the disparity between memory-resident and disk-resident methods remains vast. DiskANN, representing the state-of-the-art for disk-based indexing, faces a severe bottleneck. At 95% recall, it plateaus at 64.7 QPS, as its static routing strategy incurs excessive random I/O when navigating the sparse high-dimensional space.

MCGI bridges this divide effectively. At 95% recall, it delivers 375.1 QPS, outperforming the direct competitor DiskANN by a factor of 5.8×. Notably, MCGI's throughput begins to rival in-memory solutions, reaching approximately 64% of the performance of the memory-resident IVF-Flat index. While a natural latency gap remains compared to the HNSW roofline, MCGI demonstrates that optimized I/O management can push disk-based indexing speeds into a range previously reserved for in-memory structures like inverted files.

5.4 Billion-Scale Scalability (RQ3)

To validate the scalability of MCGI to billion-scale datasets, we conducted comprehensive evaluations on two benchmarks: SIFT1B and T2I-1B.

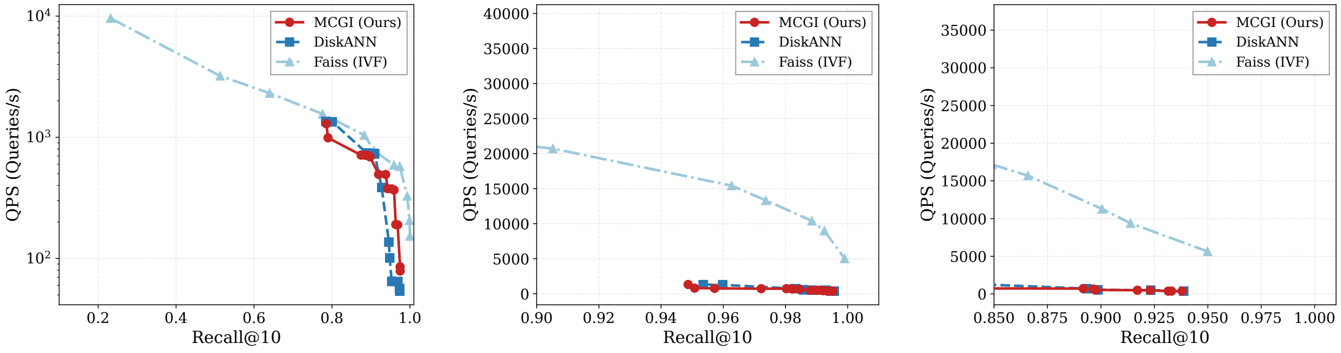
Figure 2 illustrates the recall-QPS trade-off and latency comparisons on the SIFT1B dataset. While DiskANN maintains competitive performance at moderate recall levels, its throughput degrades sharply as retrieval requirements tighten. Specifically, at a strict recall target of approximately 90%, DiskANN drops to 2,597 QPS. In contrast, MCGI maintains a robust throughput of 3,436 QPS, representing a 1.32× speedup. The advantage of MCGI is even more pronounced in terms of query latency, as shown in Figure 2 (b). At 90% recall, DiskANN incurs a mean latency of 49.06 ms, largely due to excessive backtracking and redundant I/O operations. Conversely, MCGI leverages its geometry-aware routing to reduce the mean latency to just 16.20 ms, a 3× reduction. This confirms that MCGI's manifold-consistent pruning strategy effectively identifies "high-value" neighbors, enabling the search agent to achieve high recall with significantly fewer disk accesses.

[Dongfang: Move T2I-1B experiment to appendix]

As shown in Figure 3, we further evaluated MCGI on the T2I-1B dataset. Unlike SIFT1B, T2I-1B introduces a distinct challenge characterized by higher dimensionality ($D = 200$) and 32-bit floating-point precision. These factors make distance computations significantly more expensive, shifting the system bottleneck from pure I/O latency towards CPU computation. In the critical high-recall regime (Recall@10 $> 90\%$), MCGI demonstrates a clear advantage over DiskANN, delivering higher QPS at reduced latency. Notably, this performance gain persists even though the I/O reduction is less pronounced than on SIFT1B. The advantage stems from MCGI's geometric pruning strategy: by identifying simpler local manifolds, our algorithm aggressively prunes redundant edges, reducing the average node degree without compromising navigability. This optimization translates directly to fewer expensive floating-point operations per hop, highlighting MCGI's versatility in handling both I/O-bound (SIFT1B) and compute-bound (T2I-1B) billion-scale workloads.

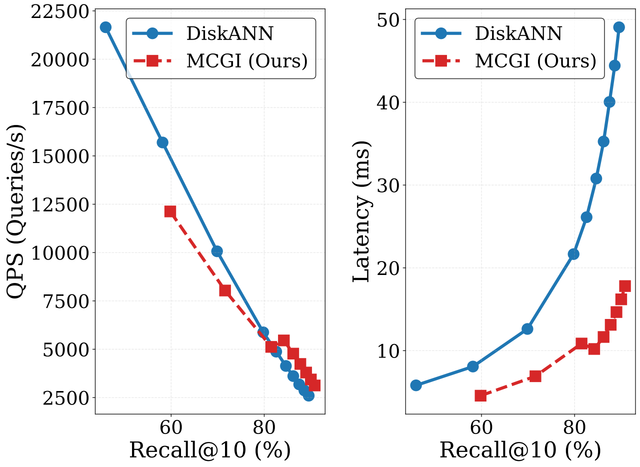
5.5 Resource Efficiency (RQ4)

Parameter Sensitivity. The search list size parameter, L, governs the trade-off between search quality and computational cost. Figure 4a illustrates the recall performance as a function of L. As observed, MCGI exhibits a recall trajectory that closely mirrors that of DiskANN, maintaining performance parity across the tested range.



(a) GIST1M (960-dim) **(b) SIFT1M (128-dim)** **(c) GloVe-100 (100-dim)**

Figure 1: Recall-QPS Trade-off. Comparison of MCGI against DiskANN and Faiss (mmap) on three datasets.

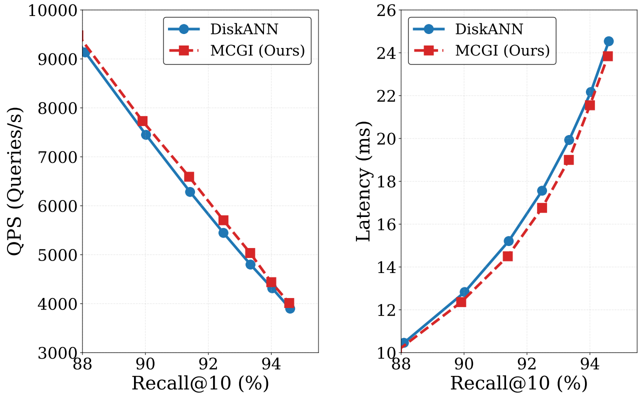


(a) Throughput (QPS) (b) Latency

Figure 2: Billion-Scale Performance on SIFT1B.

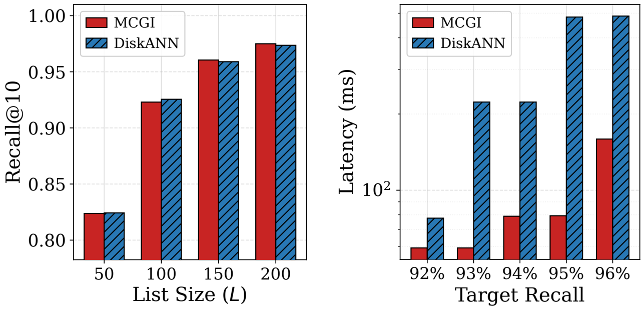
This parity is a critical validation of our approach. It demonstrates that our geometry-aware routing is robust: despite dynamically pruning the search space based on LID, MCGI retains the high recall capabilities of the baseline graph without degrading search quality. Consequently, MCGI achieves target accuracy levels using standard L configurations, validating that our efficiency gains (shown in latency analysis) do not come at the cost of retrieval accuracy.

Latency Analysis. Figure 4b details the query latency across different recall levels on GIST1M. While QPS reflects throughput, latency is critical for online services. The results show that MCGI significantly reduces latency in the high-recall regime compared to DiskANN. By minimizing the number of random I/O operations required to escape local minima, MCGI ensures that tail latency remains bounded even for difficult queries in high-dimensional spaces. This demonstrates that the manifold-aware routing strategy translates to lower I/O overhead on modern hardware, validating the benefits of our approach for resource-constrained deployments.



(a) Throughput (High Recall) (b) Latency (High Recall)

Figure 3: Performance on T2I-1B.



(a) Sensitivity **(b) Latency**

Figure 4: Resource Efficiency (RQ4).

6 Conclusion

We presented Manifold-Consistent Graph Indexing (MCGI), a framework that fundamentally rethinks vector search by aligning graph routing with the intrinsic geometry of high-dimensional data. By resolving the discrepancy between Euclidean and Geodesic distances, MCGI provides a robust theoretical guarantee on connectivity and effectively mitigates the curse of dimensionality. Comprehensive evaluations on billion-scale benchmarks, including SIFT1B and T2I-1B, confirm the versatility of MCGI across both I/O-intensive and compute-intensive scenarios. Our results demonstrate that MCGI reduces query latency by up to 3× compared to the state-of-the-art DiskANN and delivers throughput that approaches the performance of in-memory inverted indices.

References

[1] Amsaleg, L., Chelly, O., Furon, T., Girard, S., Houle, M. E., Kawarabayashi, K.-i., and Nett, M. Estimating local intrinsic dimensionality. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2015), KDD '15, Association for Computing Machinery, p. 29–38.

[2] BELLMAN, R., and Dreyfus, S. *Dynamic Programming*, vol. 33. Princeton University Press, 2010.

[3] Bentley, J. L. Multidimensional binary search trees used for associative searching. *Commun. ACM 18*, 9 (Sept. 1975), 509–517.

[4] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2020), NIPS '20, Curran Associates Inc.

[5] Chen, Q., Zhao, B., Wang, H., Li, M., Liu, C., Li, Z., Yang, M., and Wang, J. Spann: highly-efficient billion-scale approximate nearest neighbor search. In *Proceedings of the 35th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2021), NIPS '21, Curran Associates Inc.

[6] Dasgupta, S., and Freund, Y. Random projection trees and low dimensional manifolds. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2008), STOC '08, Association for Computing Machinery, p. 537–546.

[7] Datar, M., Immorlica, N., Indyk, P., and Mirrokni, V. S. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry* (New York, NY, USA, 2004), SCG '04, Association for Computing Machinery, p. 253–262.

[8] Fu, C., Xiang, C., Wang, C., and Cai, D. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proc. VLDB Endow. 12*, 5 (Jan. 2019), 461–474.

[9] Ge, T., He, K., Ke, Q., and Sun, J. Optimized product quantization. vol. 36, IEEE Computer Society, p. 744–755.

[10] Gupta, A., Krauthgamer, R., and Lee, J. Bounded geometries, fractals, and low-distortion embeddings. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.* (2003), pp. 534–543.

[11] Guttman, A. R-trees: a dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data* (New York, NY, USA, 1984), SIGMOD '84, Association for Computing Machinery, p. 47–57.

[12] He, J., Kumar, S., and Chang, S.-F. On the difficulty of nearest neighbor search. In *Proceedings of the 29th International Coference on International Conference on Machine Learning* (Madison, WI, USA, 2012), ICML'12, Omnipress, p. 41–48.

[13] Houle, M. E. Local intrinsic dimensionality I: an extreme-value-theoretic foundation for similarity applications. In *Similarity Search and Applications - 10th International Conference, SISAP 2017, Munich, Germany, October 4-6, 2017, Proceedings* (2017), C. Beecks, F. Borutta, P. Kröger, and T. Seidl, Eds., vol. 10609 of *Lecture Notes in Computer Science*, Springer, pp. 64–79.

[14] Indyk, P., and Motwani, R. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1998), STOC '98, Association for Computing Machinery, p. 604–613.

[15] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. Survey of hallucination in natural language generation. *ACM Comput. Surv. 55*, 12 (Mar. 2023).

[16] Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data 7*, 3 (2021), 535–547.

[17] Jégou, H., Douze, M., and Schmid, C. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, 1 (2011), 117–128.

[18] Jégou, H., Tavenard, R., Douze, M., and Amsaleg, L. Searching in one billion vectors: Re-rank with source coding. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2011), pp. 861–864.

[19] Karger, D. R., and Ruhl, M. Finding nearest neighbors in growth-restricted metrics. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2002), STOC '02, Association for Computing Machinery, p. 741–750.

[20] Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (Online, Nov. 2020), B. Webber, T. Cohn, Y. He, and Y. Liu, Eds., Association for Computational Linguistics, pp. 6769–6781.

[21] Keahey, K., Anderson, J., Zhen, Z., Riteau, P., Ruth, P., Stanzione, D., Cevik, M., Colleran, J., Gunawi, H. S., Hammock, C., Mambretti, J., Barnes, A., Halbach, F., Rocha, A., and Stubbs, J. Lessons learned from the chameleon testbed. In *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*. USENIX Association, July 2020.

[22] Levina, E., and Bickel, P. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems* (2004), L. Saul, Y. Weiss, and L. Bottou, Eds., vol. 17, MIT Press.

[23] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Red Hook, NY, USA, 2020), NIPS '20, Curran Associates Inc.

[24] Ma, X., Li, B., Wang, Y., Erfani, S. M., Wijewickrema, S. N. R., Houle, M. E., Schoenebeck, G., Song, D., and Bailey, J. Characterizing adversarial subspaces using local intrinsic dimensionality. In *International Conference on Learning Representations* (2018).

[25] Malkov, Y. A., and Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. vol. 42, IEEE Computer Society, p. 824–836.

[26] Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL* (2014), A. Moschitti, B. Pang, and W. Daelemans, Eds., ACL, pp. 1532–1543.

[27] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event* (2021), M. Meila and T. Zhang, Eds., vol. 139 of *Proceedings of Machine Learning Research*, PMLR, pp. 8748–8763.

[28] Robertson, S., and Zaragoza, H. The probabilistic relevance framework: Bm25 and beyond. In *Foundations and Trends in Information Retrieval*, vol. 3. Now Publishers, Inc., 2009, pp. 333–389.

[29] Roweis, S. T., and Saul, L. K. Nonlinear dimensionality reduction by locally linear embedding. *Science 290*, 5500 (2000), 2323–2326.

[30] simhadri, H. V., Aumüller, M., Douze, M., Baranchuk, D., Ingber, A., Liberty, E., Williams, G., Landrum, B., Manohar, M. D., Karjikar, M., Dhulipala, L., Chen, M., Chen, Y., Ma, R., Zhang, K., Cai, Y., Shi, J., Zheng, W., Chen, Y., Yin, J., and Huang, B. Results of the big ANN: NeurIPS'23 competition. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track* (2025).

[31] Simhadri, H. V., Williams, G., Aumüller, M., Douze, M., Babenko, A., Baranchuk, D., Chen, Q., Hosseini, L., Krishnaswamy, R., Srinivasa, G., Subramanya, S. J., and Wang, J. Results of the neurips'21 challenge on billion-scale approximate nearest neighbor search. In *NeurIPS 2021 Competitions and Demonstrations Track* (2022), PMLR, pp. 177–189.

[32] Subramanya, S. J., Devvrit, Kadekodi, R., Krishaswamy, R., and Simhadri, H. V. Diskann: fast accurate billion-point nearest neighbor search on a single node. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (2019).

[33] Tenenbaum, J. B., de Silva, V., and Langford, J. C. A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 5500 (2000), 2319–2323.

[34] Toussaint, G. T. The relative neighbourhood graph of a finite planar set. *Pattern Recognition 12*, 4 (1980), 261–268.

[35] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models.

[36] Zhao, D., and Yang, L. Incremental isometric embedding of high-dimensional data using connected neighborhood graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence 31*, 1 (2009), 86–98.

A Complete Proofs

[Dongfang: TODO]

B More Experimental Details

Build Configuration. Table 2 summarizes the dataset specifications and indexing parameters used during the construction of MCGI and baseline DiskANN indices. R denotes the maximum out-degree of each node in the graph. L_{build} is the size of the candidate list during index construction. α Range indicates the range of pruning aggressiveness explored during the LID-based Sigmoid mapping. m_{PQ} specifies the number of bytes allocated for Product Quantization (PQ) codes when applicable. Data Type indicates whether the dataset vectors are stored as 32-bit floating-point numbers or 8-bit unsigned integers.

Table 2: Dataset Specifications and Indexing Parameters.

Dataset	R	L_{build}	α Range	m_{PQ} (Byte)	Data Type
SIFT1M	64	100	[1.0, 1.5]	N/A	float32
GloVe-100	64	100	[1.0, 1.5]	N/A	float32
GIST1M	96	150	[1.0, 1.5]	N/A	float32
SIFT1B	32	50	[1.0, 1.5]	16	uint8
T2I-1B	32	50	[1.0, 1.5]	16	float32

Search Configuration. Table 3 lists the statistical parameters derived from the dataset's LID distribution used in the Sigmoid mapping function $\Phi(\cdot)$, alongside the runtime search sweep parameters. μ_{LID} and σ_{LID} denote the mean and standard deviation of the Local Intrinsic Dimensionality (LID) estimates across the dataset. Beam (L_{search}) indicates the range of candidate list sizes evaluated during query execution to capture the Recall-QPS trade-off. Threads specifies the number of parallel threads utilized during search to maximize hardware concurrency.

Table 3: Runtime Search Parameters and LID Statistics.

Dataset	μ_{LID} (Mean)	σ_{LID} (Std)	Beam (L_{search})	Threads
SIFT1M	14.2	3.1	$10 \longrightarrow 100$	1
GloVe-100	18.5	4.2	$10 \longrightarrow 120$	1
GIST1M	22.1	5.8	$20 \longrightarrow 200$	1
SIFT1B	19.5	7.9	$10 \longrightarrow 400$	80
T2I-1B	18.3	7.0	$10 \longrightarrow 200$	80

Additional Experimental Results. [Dongfang: TODO]