Received 6 May 2026; revised 20 May 2026; accepted 29 May 2026. Date of publication 3 June 2026; date of current version 19 June 2026. The review of this article was arranged by Associate Editor X. Guo.

Digital Object Identifier 10.1109/LSMC.2026.3699834

Adaptive Downsampling and Refinement for Fast Iterative Closest Point

NEHA BHATT, ZHAODONG ZHOU, KRUNAL MEHTA, AND JUN CHEN (Senior Member, IEEE)

Department of Electrical and Computer Engineering, Oakland University, Rochester, MI 48309 USA

CORRESPONDING AUTHOR: JUN CHEN (e-mail: junchen@oakland.edu).

This work was supported in part by the National Institute of Standards and Technology under Award 60NANB24D138 and in part by Michigan Economic Development Corporation through Advance Grant Proof-of-Concept Fund.

ABSTRACT Iterative closest point (ICP) and its variants represent a fundamental technique for rigid registration between two point sets, with wide applications ranging from robotics to 3D reconstruction. The primary limitations of ICP include slow convergence due to its linear convergence rate, along with the per-iteration nearest-neighbor search, which scales linearly with source cloud size and dominates runtime on dense datasets. To address these issues, this paper proposes a modified ICP method that achieves fast convergence while maintaining robust alignment. Specifically, we introduce an adaptive threshold-based downsampling technique that selectively reduces the number of source points processed during early iterations, decreasing the computational cost for each iteration. This is followed by a full-cloud refinement stage designed to recover alignment accuracy and mitigate any degradation introduced by the downsampling phase. Simulation results demonstrate that the proposed method achieves notable reductions in computational time across the majority of benchmark datasets, including Stanford mesh models and LiDAR point clouds, while maintaining comparable registration accuracy relative to the baseline method.

INDEX TERMS Rigid registration, iterative closest point, adaptive downsampling, LiDAR point clouds.

I. INTRODUCTION

Rigid registration of 3D point clouds [1], [2], [3] requires estimating an optimal transformation to align the source point cloud to the target point cloud. Iterative closest point (ICP) algorithm has long served as the cornerstone method for addressing this problem, alternating between correspondence establishment through closest point queries using KD-tree [4] and transformation optimization that minimizes the sum of squared distances between corresponding points. While ICP helps in aligning the two point clouds, it has been constrained by certain limitations such as slow convergence due to its linear convergence rate [5], and its per-iteration cost is dominated by KD-tree nearest-neighbor queries on every source point [6], [7], which scales with the source cloud size and becomes a runtime bottleneck on dense point clouds.

Numerous ICP variants have been proposed to improve its accuracy, robustness, and convergence speed [7], including symmetric objective [8], point-to-plane [9], robust cost functions [10], robust estimation with outliers [11], geometric features for robust superpoint matching [12], multi-resolution schemes [7], and graph-based correspondence formulations [13]. Acceleration techniques have also received considerable attention. Anderson-accelerated ICP applies fixed-point iteration theory to speed convergence [14]. The voxelized generalized iterative closest point (VGICP) algorithm extends the generalized ICP approach with voxelization to avoid costly nearest neighbor search while retaining its accuracy [15]. NDT-based methods [16] represent a complementary direction that replaces explicit correspondences with probabilistic voxel representations. Learning-based approaches such as Deep Closest Point [17] have also emerged for point cloud registration, though they require training data and are complementary to classical ICP-based methods. In the LiDAR SLAM domain, classical LiDAR odometry methods such as [18] and its ground-optimized variant [19] establish foundational real-time LiDAR odometry, while more recent methods such as [20] and tightly-coupled LiDAR-inertial systems including [21] have advanced state-of-the-art robustness. KISS-ICP [22] couples point-to-point ICP with adaptive thresholding and voxel-based downsampling for LiDAR

odometry. Our contribution is complementary to these systems; rather than addressing full SLAM pipelines, we focus on improving the ICP registration primitive itself, which serves as a building block in many of these methods. On the other hand, uniform downsampling, voxel grid filtering [2], and random sampling [7] approaches decrease point density prior to correspondence estimation [2]. However, fixed-resolution filtering may either discard informative points or fail to sufficiently reduce early iteration cost. Despite the significant advances in ICP-based registration introduced by [23], each iteration of their method still computes correspondences over the full source; practical convergence behavior remains sensitive to the initial correspondence quality and the aggressiveness of the acceleration strategy.

In this work, we observe that not all source points need to be queried at every iteration, especially during early iterations when the transformation is far from optimal. Based on this observation, we propose an adaptive threshold-based downsampling strategy that operates on a reduced subset of the source cloud during early iterations and progressively increases the working set as alignment improves. Unlike fixed-resolution downsampling, the threshold is halved at each iteration, enabling a coarse-to-fine progression that reduces the dominant correspondence search cost while maintaining well-distributed sampling across the source cloud. Although the transformation is estimated on the filtered subset, the registration energy is consistently evaluated over the full source cloud, ensuring that the original optimization objective is preserved. Anderson Acceleration [24], [25] is incorporated within a dual stabilization mechanism: the accelerated update is accepted only if it reduces the full-cloud energy; otherwise, the algorithm reverts to the backup SVD-based [26] transformation and re-evaluates. If neither yields improvement, the coarse phase terminates. A subsequent full-cloud refinement phase recovers fine-grained alignment accuracy using the complete source cloud.

To test the efficacy of the proposed method, we benchmark it against the point-to-point Fast ICP as implemented in [23]. We use both static and dynamic datasets to assess its computational efficiency, and registration accuracy. Static evaluation is conducted using widely adopted benchmarks from the Stanford 3D repository [27] and EPFL statue dataset [28]. Dynamic datasets are derived from real-world LiDAR sequences containing sensor noise and motion distortion. Our method achieves speedups of 13–58% on four of five static datasets and around 44–64% on all three LiDAR datasets, while maintaining comparable RMSE.

The contributions of this paper are summarized as follows.

- We propose an adaptive threshold-based downsampling strategy that progressively reduces the correspondence computation burden. The threshold is initialized proportional to the minimum inter-point spacing $d_{\min}$ and halved at each iteration, producing a coarse-to-fine alignment progression that enforces minimum inter-point separation in the filtered subset while reducing KD-tree queries.
- We design a two-phase optimization framework combining adaptive downsampling with a dual stabilization mechanism. The coarse phase estimates transformations on filtered subsets while validating energy on the full source cloud, ensuring monotonic energy decrease in each accepted iteration. A subsequent full-cloud refinement phase recovers fine-grained alignment accuracy.
- We conduct comprehensive experiments on both static and dynamic datasets and demonstrate that our method can achieve significant speedups while maintaining comparable accuracy.

The remainder of this paper is organized as follows. Section II describes preliminaries on ICP, while Section III presents the main algorithm. Section IV describes experimental evaluation on Stanford mesh models and EPFL statue datasets and Section V shows experiments on real-world LiDAR data. Section VI concludes the paper with future work directions.

II. PRELIMINARIES

Given two point clouds $P = \{\mathbf{p}_i\}_{i=1}^{M}$ and $Q = \{\mathbf{q}_j\}_{j=1}^{N}$, the goal of ICP is to estimate a rigid transformation $(\mathbf{R}, \mathbf{t})$ that minimizes:

$$E(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{M} \left\| \mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_{c(i)} \right\|^2, \tag{1}$$

where $\mathbf{R} \in SO(3)$ is a rotation matrix, $\mathbf{t} \in \mathbb{R}^3$ is a translation vector, and

$$c(i) = \arg \min_{j \in \{1,\ldots,N\}} \left\| \mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_j \right\| \tag{2}$$

assigns each source point to its nearest neighbor in Q [1]. Correspondences are established via nearest-neighbor queries using a KD-tree [4] built over the target cloud Q, which constitutes the dominant computational cost per ICP iteration. Note that (1) can be solved via Singular Value Decomposition (SVD) of the cross-covariance matrix [23], [26]. For notational brevity, we write $\mathbf{T} = (\mathbf{R}, \mathbf{t})$ to denote the rigid transformation, and $\mathbf{T}P$ to denote the transformed point cloud $\{\mathbf{R}\mathbf{p}_i + \mathbf{t}\}_{i=1}^{M}$.

To accelerate convergence, we incorporate Anderson Acceleration [24], [25], a quasi-Newton technique that uses information from previous iterations to produce a faster-converging update. Anderson Acceleration was first applied to ICP by [14], and [23] subsequently developed an Anderson-accelerated formulation over the Lie algebra of $SE(3)$ with an energy-based stabilization scheme that ensures monotonic decrease of the target energy.

III. FAST ICP WITH ADAPTIVE DOWNSAMPLING

The proposed method operates in three sequential stages. *First*, the minimum inter-point spacing $d_{\min}$ is computed and used to initialize the adaptive threshold $\tau_0 = \kappa d_{\min}$. *Second*, a coarse alignment phase iteratively filters the source cloud $P = \{p_i\}_{i=1}^{M}$ and estimates transformations on the reduced subset X, with each candidate validated using the full-cloud

Algorithm 1: Threshold-Based Downsampling.

Input: Source cloud P, threshold τ
Output: Filtered subset X

```
X ← {p_0}, p_ref ← p_0;
for i = 1 to N do
    if ‖p_i − p_ref‖ ≥ τ then
        X ← X ∪ {p_i}, p_ref ← p_i;
    end
end
return X
```

registration energy in (1). *Third*, a refinement phase further improves the alignment accuracy using the complete source cloud P. Each stage is described in the following subsections.

A. ADAPTIVE THRESHOLD-BASED DOWNSAMPLING

Instead of computing correspondences for all M source points at every ICP iteration, the proposed method operates on a filtered subset $X \subseteq P$ obtained through adaptive spatial downsampling. The initial threshold is defined as $\tau_0 = \kappa d_{\min}$, where $d_{\min}$ denotes the minimum inter-point distance in the source cloud P and κ is selected such that τ initially covers the spatial extent of the source cloud, ensuring that the coarse phase begins with a sparse but well-distributed subset.

The threshold is progressively halved at each iteration, i.e., $\tau_{k+1} = \frac{\tau_k}{2}$. At each threshold level τ_k, the filtered subset X is constructed from P using Algorithm 1, which enforces minimum inter-point separation. This strategy maintains well-distributed sampling across the source cloud while preventing point clustering. As τ_k decreases, the filtered subset X increases, enabling progressively finer registration refinement. This coarse-to-fine strategy ensures that early iterations operate on a sparsified point set for computational efficiency and faster convergence, while later iterations gradually increase point density.

B. CORRESPONDENCE SEARCH AND ENERGY-BASED ACCEPTANCE

Given source cloud P and target cloud Q, the objective is to estimate a rigid transformation $(\mathbf{R}, \mathbf{t})$ that minimizes (1). The adaptively downsampled source subset $X \subset P$ is used for correspondence search and transformation generation. Although the transformation is estimated on the filtered subset X, its quality is validated by computing correspondences over the entire source cloud and evaluating the full registration energy. This ensures that the downsampling does not compromise the original optimization objective (1).

Following the framework of [23], we adopt SVD-based transformation estimation, Anderson Acceleration for fixed-point convergence, and energy-based acceptance with fallback to the SVD solution. Before applying Anderson Acceleration, the SVD-derived transformation is stored as $\mathbf{T}_{\text{backup}}$. Anderson Acceleration is applied only if it reduces full-cloud energy; otherwise, the method reverts to the standard SVD solution. If neither the accelerated nor the backup transformation yields an improvement, the coarse phase terminates. Once a transformation is accepted, it is applied to the source cloud $P \leftarrow \mathbf{T}P$, so that subsequent iterations work on the updated geometry rather than the original. Combined with the halving threshold, this produces a coarse-to-fine progression where each filtered subset is denser and better-aligned than the previous one. This dual-tier acceptance strategy allows the algorithm to benefit from Anderson Acceleration when it improves convergence, while guaranteeing stable monotonic energy decrease through fallback to the classical SVD estimate.

C. COMPLETE ALGORITHM

Algorithm 2 presents the proposed ICP framework with adaptive thresholding, energy-based reversion, and full-cloud refinement. The method consists of two main phases: a coarse alignment phase driven by adaptive sampling, followed by a refinement stage using the complete source cloud. The algorithm begins by computing the minimum inter-point spacing $d_{\min}$ of the source cloud and initializing the threshold to $\kappa d_{\min}$. A KD-tree is constructed on the target cloud Q to accelerate nearest neighbor queries. The transformation is initialized as an identity matrix and the initial full-cloud registration energy E_{prev} is computed.

At each iteration, P is filtered using Algorithm 1 with the current threshold τ, producing a reduced subset $X \subseteq P$. A candidate transformation is estimated on X by minimizing (1), refined through Anderson Acceleration, and validated by recomputing correspondences and energy over the full source cloud, following the dual-tier acceptance scheme described in Section III-B. The phase terminates either when $\tau < d_{\min}$, or when neither the accelerated nor the SVD candidate reduces the full-cloud energy. This energy-based acceptance criterion preserves the monotonic energy-decrease across all accepted iterations, property of classical ICP, while the halving schedule enables a coarse-to-fine progression from structural alignment to fine geometric matching.

Phase 2 ensures the accuracy of the final alignment, with its behavior adapted based on the termination condition of Phase 1. If Phase 1 terminates because the threshold has decayed to $\tau < d_{\min}$, the alignment is already near-optimal, and only a few full-cloud iterations are needed before the energy-based stopping criterion is satisfied. If Phase 1 terminates early due to energy-based fallback (before τ reaches $d_{\min}$), the source cloud may still require additional correction, and Phase 2 allows up to $k_{\max}$ full-cloud refinement iterations, each guarded by the same energy-based stopping criterion. In both cases, transformations are estimated on the complete source cloud P, without downsampling or Anderson Acceleration. Upon termination, the final accumulated transformation $\mathbf{T}$ is returned.

IV. EXPERIMENTS ON STATIC DATASET

We evaluate Algorithm 2 on several benchmark datasets and compare against the Fast ICP [23]. All experiments are conducted on a PC with a 4-cores CPU at 1.60 GHz, 8 GB

Algorithm 2: Fast ICP with Adaptive Downsampling and Energy-Based Reversion.

Input: Source cloud P, Target cloud Q, κ, $k_{\max}$
Output: Transformation $\mathbf{T}$

Initialization:
Compute $d_{\min}$; $\tau \leftarrow \kappa\, d_{\min}$; Build KD-tree on Q;
Compute $\mathbf{T} \leftarrow I$; Initial energy E_{prev} on P using Eq 1;
% Phase 1: Adaptive threshold-based downsampling
while $\tau \geq d_{\min}$ **do**
  $X \leftarrow \text{FILTER}(\text{P}, \tau)$ via Algorithm 1;
  Compute nearest neighbors for X in Q via KD-tree;
  $\mathbf{T} \leftarrow$ minimizing (1) with X as source and Q as target;
  $\mathbf{T}_{\text{backup}} \leftarrow \mathbf{T}$;
  $\mathbf{T} \leftarrow AndersonAccel(\mathbf{T})$;
  Recompute nearest neighbors for P using $\mathbf{T}$;
  Compute full-cloud registration energy E_{new} on P using (1);
  if $E_{new} < E_{prev}$ **then**
    $P \leftarrow \mathbf{T}\,P$, $E_{\text{prev}} \leftarrow E_{\text{new}}$;
  else
    $\mathbf{T} \leftarrow \mathbf{T}_{\text{backup}}$;
    Recompute nearest neighbors for P using $\mathbf{T}_{\text{backup}}$;
    Compute E_{revert} on P using (1);
    if $E_{revert} < E_{prev}$ **then**
      $P \leftarrow \mathbf{T}\,P$, $E_{\text{prev}} \leftarrow E_{\text{revert}}$;
    else
      break;
    end
  end
  $\tau \leftarrow \tau/2$;
end
% Phase 2: Refinement
$k \leftarrow 0$;
while $\tau < d_{\min}$ *or* $k < k_{\max}$ **do**
  $\mathbf{T}_{\text{backup}} \leftarrow \mathbf{T}$;
  Recompute nearest neighbors for P using $\mathbf{T}$;
  $\mathbf{T} \leftarrow$ minimizing (1);
  Compute full-cloud energy E on P using (1);
  if $E > E_{prev}$ **then**
    break;
  end
  $E_{\text{prev}} \leftarrow E$, $k \leftarrow k+1$;
end
return $\mathbf{T}$

RAM. We use datasets from the Stanford 3D scanning repository [27] and EPFL statues [28]. We set $\kappa = 16$ and $k_{\max} = 8$ for static dataset.

A. NUMERICAL RESULTS

The coarse-to-fine behavior of Algorithm 2 is illustrated in Fig. 1 using the Stanford Bunny, where Fig. 1(a) illustrates the original source (14,806 points) and target (15,446 points) clouds. Fig. 1(b)–1(f) demonstrate the adaptive downsampling phase of Algorithm 2 while Fig. 1(g) corresponds to the refinement phase. As can be seen, during the adaptive downsampling phase as the threshold decreases, the point cloud becomes denser indicating an increase in the number of points in the source point cloud. Particularly, the source cloud X increases from only 1,338 points at Iteration 0 ($\tau = 16\,d_{\min}$) to 14,805 points by Iteration 5 ($\tau = d_{\min}$). At each iteration, the transformation estimated on the filtered subset is applied to the entire source cloud before the next round of downsampling, so the source progressively reorients toward the target while the working subset becomes denser. As shown in Fig. 1, the source cloud aligns aggressively with the target within only five coarse-phase iterations, leaving the refinement phase to resolve fine-scale residuals.

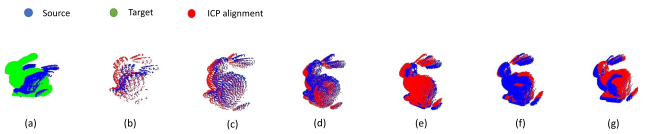

FIGURE 1. Illustration of the coarse-to-fine behavior of Algorithm 2 using the Stanford Bunny. (a) Original source and target clouds; (b)–(f) Adaptive downsampling phase; (g) Refinement phase.

TABLE 1. Comparison of Algorithm 2 Versus Baseline Fast ICP and Robust ICP [23] on Static Datasets

Dataset	Method	Ave. Time ± STD (s)	RMSE (m)
Bunny	Fast ICP	0.417 ± 0.002	2.277×10^{-2}
	Robust ICP	0.569 ± 0.0158	2.277×10^{-2}
	Ours	$\mathbf{0.275 \pm 0.002}$	$\mathbf{2.273 \times 10^{-2}}$
Buddha	Fast ICP	8.923 ± 0.069	$\mathbf{1.233 \times 10^{-2}}$
	Robust ICP	10.974 ± 0.0476	1.233×10^{-2}
	Ours	$\mathbf{3.777 \pm 0.028}$	1.236×10^{-2}
Dragon	Fast ICP	6.271 ± 0.316	$\mathbf{5.75 \times 10^{-9}}$
	Robust ICP	6.732 ± 0.629	5.67×10^{-9}
	Ours	$\mathbf{5.467 \pm 0.170}$	5.67×10^{-6}
Monkey	Fast ICP	$\mathbf{3.577 \pm 0.081}$	$\mathbf{1.48 \times 10^{-8}}$
	Robust ICP	8.765 ± 2.336	1.44×10^{-8}
	Ours	6.814 ± 0.049	6.59×10^{-5}
William Tell	Fast ICP	2.895 ± 0.272	$\mathbf{2.29 \times 10^{-7}}$
	Robust ICP	5.747 ± 1.706	2.29×10^{-7}
	Ours	$\mathbf{1.916 \pm 0.039}$	2.204×10^{-3}

Table 1 presents a comparison between Algorithm 2 against the baseline Fast ICP and Robust ICP [23] across several datasets, evaluated on runtime (seconds) and registration error (RMSE in meters). On the Bunny dataset, the proposed Algorithm 2 achieves a 34% speedup with a virtually identical RMSE. On the Buddha dataset, a 58% speedup is observed with a negligible RMSE difference. On the Dragon and William Tell datasets, Algorithm 2 is 13% and 34% faster, respectively, with a slight degradation in accuracy. This trade-off arises because the adaptive downsampling phase operates on spatially filtered subsets that prioritize computational efficiency over geometric completeness during early iterations.

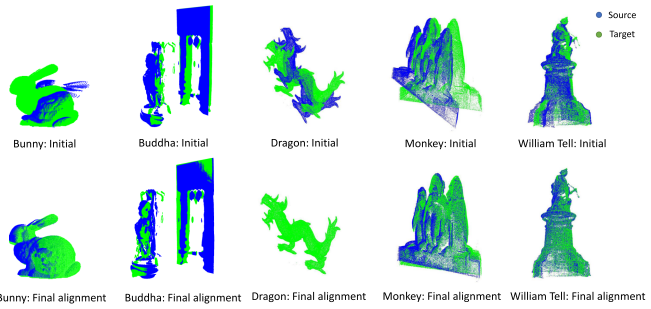

FIGURE 2. **Registration results on static datasets. *Top:* Initial overlap of target (green) and source (blue). *Bottom:* Final alignment by Algorithm 2.**

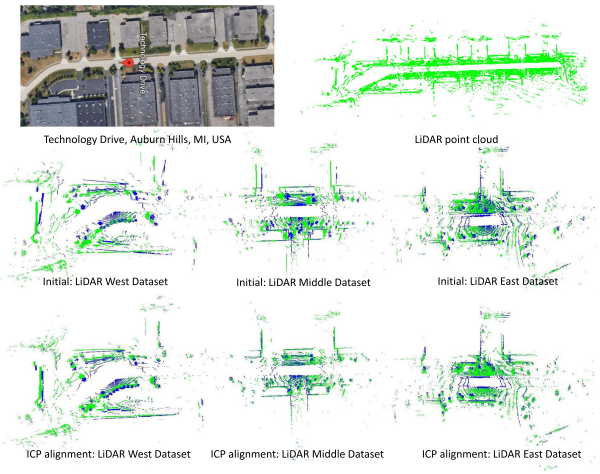

FIGURE 3. **LiDAR experiments on Technology Drive, Auburn Hills, MI, USA. *Top:* Drive route map and aggregated LiDAR point cloud. *Middle:* Initial overlap of target (green) and source (blue) for West, Middle, and East segments. *Bottom:* Final alignment achieved by Algorithm 2.**

Importantly, despite the numerical increase in RMSE, the final alignments produced by the proposed method are visually indistinguishable from those of Fast ICP, as illustrated in Fig. 2, suggesting that the observed RMSE differences correspond to sub-millimeter residuals that do not affect practical registration quality. Monkey dataset is the only case where Algorithm 2 performs poorer in both speed and accuracy. Specifically, our method is slower than Fast ICP; however, the alignment quality remains accurate in practice. We attribute this to the complex geometry of the Monkey mesh, where the spatially filtered subset produces transformations that fail the energy check, causing early termination of the adaptive downsampling phase before τ reaches $d_{\min}$. The refinement phase must then compensate from a less favorable initialization, requiring additional iterations.

Remark 1: The Monkey dataset consists of a larger point cloud compared to the other benchmark models, comprising three separate figures in distinct poses, each with concentrated high-curvature features (concave eye socket, hands covering eyes, mouth and ears). These structures increase nearest-neighbor ambiguity after adaptive downsampling and reduce correspondence stability during Phase 1, which terminates early through the energy-plateau criterion before reaching the finest threshold level. As a result, the downsampling stage introduces computational overhead without proportionally reducing the refinement burden, leading to higher total runtime than the benchmarks. However, despite its inability to establish a strong coarse initialization on this geometry, Phase 1 still advances the alignment enough to provide a measurable accuracy benefit when combined with Phase 2, the full Algorithm 2 achieves the lowest RMSE of 6.59×10^{-5} m. Future work will investigate curvature-aware and geometry-preserving downsampling strategies to better handle complex non-uniform point cloud structures.

Finally, the results of Algorithm 2 over all tested datasets are illustrated in Fig. 2, where the good alignment between source and target can be observed.

TABLE 2. **Sensitivity With Respect to κ, on the Bunny Dataset With $k_{\max} = 8$**

κ	Time (s)	RMSE (m)
8	0.268 ± 0.008	2.27×10^{-2}
16	0.267 ± 0.003	2.27×10^{-2}
32	0.258 ± 0.001	2.28×10^{-2}
64	0.292 ± 0.001	2.27×10^{-2}
128	0.267 ± 0.001	4.68×10^{-2}

TABLE 3. **Sensitivity With Respect to $k_{\max}$, on the Bunny Dataset, With $\kappa = 16$**

$k_{\max}$	Time (s)	RMSE (m)
4	0.268 ± 0.003	2.27×10^{-2}
16	0.267 ± 0.002	2.27×10^{-2}
32	0.268 ± 0.004	2.27×10^{-2}
64	0.289 ± 0.012	2.27×10^{-2}

TABLE 4. **Sensitivity With Respect to κ on the Buddha Dataset, With $k_{\max} = 8$**

κ	Time (s)	RMSE (m)
4	2.90 ± 0.01	1.23×10^{-2}
16	3.82 ± 0.01	1.24×10^{-2}
64	3.95 ± 0.25	1.24×10^{-2}
80	4.01 ± 0.32	1.24×10^{-2}

TABLE 5. **Sensitivity With Respect to $k_{\max}$ on the Buddha Dataset, With $\kappa = 16$**

$k_{\max}$	Time (s)	RMSE (m)
4	2.93 ± 0.01	1.24×10^{-2}
8	3.82 ± 0.01	1.23×10^{-2}
17	5.89 ± 0.03	1.23×10^{-2}
32	6.83 ± 0.04	1.23×10^{-2}

B. SENSITIVITY ANALYSIS

We characterize the sensitivity of Algorithm 2 to its two tunable parameters, κ and $k_{\max}$, using the Bunny and Buddha datasets. The results are shown in Tables 2, 3, 4, and 5.

For Bunny, RMSE is robust to κ in the range [8,64], varying by less than 0.3%. At $\kappa = 128$, however, RMSE degrades sharply to 4.68×10^{-2} m, approximately twice the baseline error. Such a degradation originates from excessive sparsification during the initial stage of Phase 1, where the adaptive filter retains only 19 source points. This sparse and non-representative subset weakens the geometric constraints available for rigid registration, producing unstable correspondence structure and a poor intermediate transformation estimate. On the other hand, both runtime and RMSE are invariant to $k_{\max}$ across the tested range, with all values agreeing around 2.27×10^{-2} m. This invariance is structural rather than empirical. On Bunny, Phase 1 exhausts its full threshold cascade and exits with $\tau < d_{\min}$, and so Phase 2 always terminates via its energy-plateau break regardless of $k_{\max}$. Note that the small variation of runtime is attributable to CPU variation rather than algorithmic work.

For Buddha dataset, runtime varies significantly with respect to κ. Larger κ requires more Phase 1 iterations before the cascade either exhausts or triggers the energy-plateau break, hence increasing total runtime without improving RMSE. For $k_{\max}$, the behavior reveals a pattern of diminishing-returns. At $k_{\max} = 4$, the algorithm yields an RMSE of 1.24×10^{-2} m in 2.93 ± 0.01 s. At $k_{\max} = 17$, the algorithm reaches near-converged RMSE of 1.23×10^{-2} m in 5.89 ± 0.03 s. At $k_{\max} \geq 32$, Phase 2 actually exits via the energy-plateau break before reaching $k_{\max}$ and converges to 1.23×10^{-2} m in approximately 6.8 s. Beyond $k_{\max} \approx 8$, additional iterations continue to decrease the energy by progressively smaller amounts, but the alignment is already near-perfect and the extra iterations consume runtime without producing visible accuracy improvement. The 1% RMSE reduction from $k_{\max} = 4$ to $k_{\max} \geq 32$ comes at a 2.3× runtime cost, illustrating that an excessive refinement budget on already-converged datasets adds runtime without practical benefit.

V. EXPERIMENTS ON LIDAR DATASET

Real-world experiments were conducted using a Ford Mustang Mach-E electric vehicle equipped with a roof-mounted LiDAR sensor and a tightly integrated GPS/IMU navigation unit. The vehicle platform is instrumented to collect time-synchronized LiDAR point clouds together with dual GPS measurements during on-road driving. LiDAR measurements are acquired using an Ouster OS1-64 sensor [29], a 64-beam spinning LiDAR widely adopted in robotics and autonomous systems research. The sensor provides a full 360° horizontal field of view and operates at 10–20 Hz, generating dense 3D point clouds with meter-to-centimeter scale resolution.

Data collection is performed across urban environments, including paved road surfaces, building facades, curbs, roadside vegetation (trees and shrubs), and static infrastructure elements such as poles and signage. These scenes contain both structured geometry (e.g., buildings and curbs) and unstructured elements (e.g., vegetation and uneven terrain), providing realistic conditions representative of autonomous driving scenarios.

TABLE 6. Target Sub-Map and Source Scan Timestamps for Each LiDAR Segment

Segment	Target Sub-map (Global Frames)	Source Scan (t_l)
East	0 s + 5 s	3 s
Middle	12 s + 16 s	15 s
West	36 s + 40 s	38 s

TABLE 7. Comparison of Algorithm 2 Versus Baseline Fast ICP and Robust ICP [23] on LiDAR Datasets

LiDAR set	Method	Ave. Time ± STD (s)	RMSE (m)
East	Fast ICP	0.371 ± 0.002	$\mathbf{2.92 \times 10^{-3}}$
	Robust ICP	0.634 ± 0.006	$\mathbf{2.92 \times 10^{-3}}$
	Ours	$\mathbf{0.157 \pm 0.002}$	2.92×10^{-3}
Middle	Fast ICP	0.730 ± 0.003	2.66×10^{-3}
	Robust ICP	0.925 ± 0.011	$\mathbf{2.66 \times 10^{-3}}$
	Ours	$\mathbf{0.266 \pm 0.002}$	$\mathbf{2.65 \times 10^{-3}}$
West	Fast ICP	0.543 ± 0.002	5.71×10^{-3}
	Robust ICP	0.698 ± 0.001	$\mathbf{5.71 \times 10^{-3}}$
	Ours	$\mathbf{0.304 \pm 0.002}$	$\mathbf{5.60 \times 10^{-3}}$

To reduce computational complexity and improve convergence stability, the proposed approach registers each scan against a temporally localized target sub-map rather than the complete global map. Three sub-maps — corresponding to the East, Middle, and West segments of the drive — are constructed by aggregating global frames acquired at the timestamps shown in Table 6. The source point cloud is a local scan acquired at time t_l, while its initial pose (GPS and heading) is taken from the preceding timestamp $t_l - 1$ to simulate misalignment present in real-time SLAM scenarios. The evaluation is done on the same computing hardware as in the static dataset. We set $\kappa = 1,000$ and $k_{\max} = 8$ for LiDAR experiment.

Table 7 presents a comparison between Algorithm 2 against the baseline Fast ICP and Robust ICP [23] across the three LiDAR sequences, evaluated on runtime (seconds) and registration error (RMSE in meters). During the East segment, the proposed method is approximately 50% faster with virtually identical RMSE, confirming that speedup is achieved without compromising accuracy. The largest improvement is observed during the Middle segment, with a 64% reduction in runtime, while producing a nearly identical RMSE. During the West segment, the method is 44% faster with a marginally lower RMSE, indicating a slight accuracy improvement alongside the runtime gain.

Across all three LiDAR benchmarks, the proposed approach consistently achieves faster runtime with speedups ranging from 44–64 %, while maintaining registration accuracy comparable to Fast-ICP.

VI. CONCLUSION

This paper presents an adaptive downsampling strategy combined with the refinement framework for iterative closest point (ICP) algorithm. The key insight is that coarse alignment

does not require all source points, and a progressively refined subset suffices, while full-cloud energy validation preserves the original optimization objective. By coupling adaptive downsampling with Anderson Acceleration and a dual-tier acceptance criterion, the proposed two-phase pipeline reduces per-iteration computational cost. Comprehensive experiments on Stanford mesh models, EPFL statues, and real-world LiDAR scans demonstrate 13–64% runtime improvements, while maintaining comparable or improved RMSE relative to baseline Fast ICP. The proposed method is simple to implement, introduces only two tunable parameters, and integrates naturally with existing ICP frameworks. Future work will explore the extensions to point-to-plane formulations and robust error metrics for improved accuracy on noisy data, geometry-aware downsampling strategies that incorporate surface normals or curvature information, learned adaptive strategies for threshold selection and integration within a full SLAM pipeline to evaluate long-term trajectory accuracy and map consistency.

REFERENCES

[1] P. J. Besl and N. D. McKay, "A method for registration of 3D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[2] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 1–4.

[3] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Foundations Trends Robot.*, vol. 4, no. 1, pp. 1–104, 2013.

[4] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[5] H. Pottmann, Q.-X. Huang, Y.-L. Yang, and S.-M. Hu, "Geometry and convergence analysis of algorithms for registration of 3D shapes," *Int. J. Comput. Vis.*, vol. 67, no. 3, pp. 277–296, 2006.

[6] M. Greenspan and M. Yurick, "Approximate KD-tree search for efficient ICP," in *Proc. 4th Int. Conf. 3D Digit. Imag. Model.*, 2003, pp. 442–448.

[7] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Proc. Int. Conf. 3D Digit. Imag. Model.*, 2001, pp. 145–152.

[8] S. Rusinkiewicz, "A symmetric objective function for ICP," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–7, 2019.

[9] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vis. Comput.*, vol. 10, no. 3, pp. 145–155, 1992.

[10] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse iterative closest point," *Comput. Graph. Forum*, vol. 32, no. 5, pp. 113–123, 2013.

[11] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 314–333, Apr. 2021.

[12] Z. Qin et al., "GeoTransformer: Fast and robust point cloud registration with geometric transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 9806–9821, Aug. 2023.

[13] L. Li, M. Yang, and C. Wang, "Graph correspondence-based point set registration," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 54, no. 7, pp. 4101–4112, Jul. 2024.

[14] A. L. Pavlov, G. W. Ovchinnikov, D. Y. Derbyshev, D. Tsetserukou, and I. V. Oseledets, "AA-ICP: Iterative closest point with anderson acceleration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 3407–3412.

[15] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Voxelized GICP for fast and accurate 3D point cloud registration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11054–11059.

[16] Z. Zhou et al., "NDT-transformer: Large-scale 3D point cloud localisation using the normal distribution transform representation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5654–5660.

[17] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3523–3532.

[18] J. Zhang et al., "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst.*, Berkeley, CA, USA, 2014, vol. 2, no. 9, pp. 1–9.

[19] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.

[20] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "CT-ICP: Real-time elastic LiDAR odometry with loop closure," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 5580–5586.

[21] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang, "FAST-LIVO: Fast and tightly-coupled sparse-direct LiDAR-inertial-visual odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 4003–4009.

[22] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "KISS-ICP: In defense of point-to-point ICP–simple, accurate, and robust registration if done the right way," *IEEE Robot. Automat. Lett.*, vol. 8, no. 2, pp. 1029–1036, Feb. 2023.

[23] J. Zhang, Y. Yao, and B. Deng, "Fast and robust iterative closest point," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3450–3466, Jul. 2022.

[24] D. G. Anderson, "Iterative procedures for nonlinear integral equations," *J. ACM*, vol. 12, no. 4, pp. 547–560, 1965.

[25] H. F. Walker and P. Ni, "Anderson acceleration for fixed-point iterations," *SIAM J. Numer. Anal.*, vol. 49, no. 4, pp. 1715–1735, 2011.

[26] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.

[27] Stanford Computer Graphics Laboratory, "The Stanford 3D scanning repository," 1996. Accessed: Feb. 19, 2026. [Online]. Available: http://graphics.stanford.edu/data/3Dscanrep/

[28] EPFL Computer Graphics and Geometry Laboratory, "EPFL statue model repository," 2012. [Online]. Available: https://lgg.epfl.ch/statues_dataset.php

[29] Inc Ouster, "OS1-64 product specification," 2021. Accessed: Feb. 19, 2026. [Online]. Available: https://ouster.com/products/os1-sensor/