SEGMENT-DELTA: EXACT AND SCALABLE REPLACEMENT REFINEMENT FOR TRAJECTORY SIMPLIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Fixed-budget trajectory simplifiers produce compact anchor subsequences, but reconstruction quality depends jointly on all anchors, so their outputs may still admit beneficial same-budget replacements. Exhaustively finding the best such replacement requires evaluating $(k-2)(n-k)$ delete–insert pairs at every refinement round. We introduce Segment-Delta, an exact compiler for this complete neighborhood under segment-local reconstruction objectives. Deleting an anchor merges two adjacent intervals, whereas inserting a point splits one; the effects compose when their supports are disjoint, and only a linear number of pairs overlap. Segment-Delta summarizes the disjoint pairs and evaluates the exceptions, reducing candidate comparison in each scan to $O(n+k)$ values. Direct deltas handle additive costs, while a two-pass value–witness procedure handles bottleneck objectives. In exact arithmetic, both recover the same tie-broken move, refinement path, and terminal subset as exhaustive search. After refinement restricted to overlapping edit supports, a complete-neighborhood audit finds a further improving move in 59.7% of 144 terminal states. Across GeoLife and Porto, complete refinement improves classical and learned simplifications and closes 94.2–99.9% of the aggregate Linear dynamic-programming gap for learned constructors. Segment-Delta gives 1.10–1.54× complete-refinement speedups on published length-1,000 low-retention outputs and 2.11–5.55× on nested-length complete-path tests ($n \leq 96$). With a frozen global Transformer, structurally ranked proposals retain 91.3–98.3% of complete-refinement gain, and every accepted move is verified on the full objective.

1 INTRODUCTION

Classical and learned trajectory simplifiers turn a dense path into an ordered subsequence of k anchors. Their hard decisions jointly determine how well the path can be reconstructed. Our input is the dense trajectory, the constructor's chosen anchors, and a fixed reconstruction evaluator; our output preserves endpoints and budget while decreasing error whenever an improving replacement is accepted. This is an inference-time structured correction layer: the constructor and reconstructor stay frozen while the selected anchors change. It applies after classical or learned construction (Zhang et al., 2018; Wang et al., 2021; Song et al., 2024).

A delete–insert example. Consider ten samples with initial anchors $S = (0, 3, 6, 9)$ ($k = 4$). Replacing anchor 3 by discarded sample 8 yields $(0, 6, 8, 9)$: deletion merges intervals $(0, 3), (3, 6)$ into $(0, 6)$, and insertion splits $(6, 9)$ into $(6, 8), (8, 9)$. The two edits affect disjoint intervals, so their exact costs compose. Inserting 4 instead would split the deletion bridge itself, requiring a joint calculation. For coordinates $(t, 0)$ except $(8, 6)$ and uniform timestamps, the $3 \to 8$ move reduces mean non-anchor Linear error from 1 to 0.5 (six missing points). Figure 1 connects this interface to the complete-neighborhood mechanism; Supplement A plots the numerical example.

The challenge is to find the best such correction among $(k-2)(n-k)$ delete–insert pairs at every round. Segment-Delta uses trajectory order to *compile the complete neighborhood*: each deletion merges adjacent intervals and each insertion splits one; only $O(n-k)$ pairs have overlapping supports. Constant-size summaries recover the best disjoint candidates, while the overlapping pairs are

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

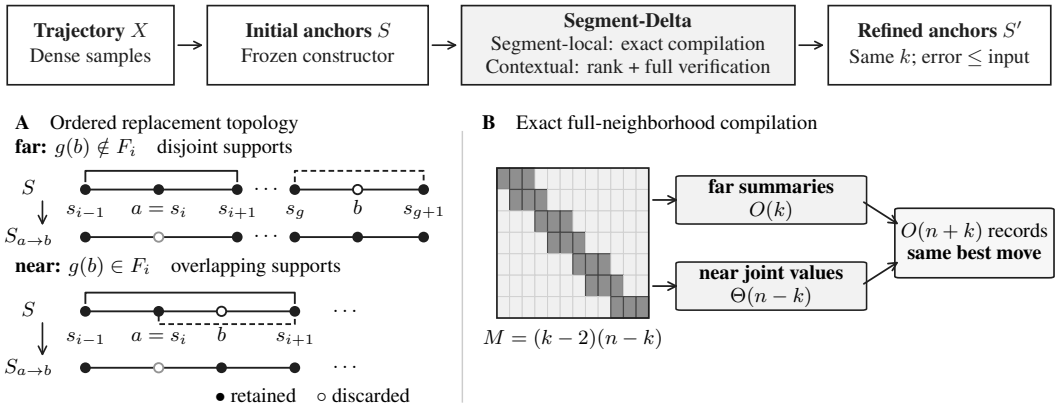


Figure 1: Workflow and exact compilation mechanism. (A) Deletion merges two adjacent gaps; insertion splits one. Their supports are disjoint for far pairs and overlap for near pairs. Positions denote sample order; filled/open circles denote retained/discarded points. (B) The full retained–discarded matrix consists of far pairs represented by $O(k)$ exact summaries and only $\Theta(n-k)$ overlapping exceptions, recovering the same best move. The contextual branch uses approximate ranking followed by full-objective verification.

evaluated jointly. This yields $O(n+k)$ candidate values for both additive and bottleneck objectives, including the same complete-tuple tie rule as exhaustive search. Interval-cost construction is charged separately.

This provides inference-time optimization of interacting anchor choices from a frozen constructor. We evaluate two learned constructors and published MLSimp outputs, with compatible dynamic programming (DP) supplying a global reference. A frozen global Transformer tests the edit structure beyond segment locality.

We contribute a constructor-independent correction interface; exact additive and bottleneck value–witness compilers based on ordered merge–split structure; and a contextual extension with structural ranking and full-objective verification. Experiments measure reconstruction gains and exhaustive-equivalent refinement at reduced cost.

2 Problem Formulation

Let $X = (x_0, \ldots, x_{n-1})$ be a trajectory in sample order, with strictly increasing timestamps when required. A simplification retains $S = (s_0 = 0 < s_1 < \cdots < s_{k-1} = n-1)$, $3 \leq k \leq n-1$. A frozen reconstructor $\mathcal{R}(X, S)$ predicts the path from these anchors; $J_{\mathcal{R}}(X, S)$ is its error. Refinement accepts any classical or learned hard output S and preserves endpoints, budget, reconstructor and objective.

Removable indices are $\mathcal{A}(S) = S \setminus \{0, n-1\}$ and insertable indices are $\mathcal{B}(S) = \{1, \ldots, n-2\} \setminus S$. Thus $|\mathcal{A}| = k-2$, $|\mathcal{B}| = n-k$ and there are

$$M = (k-2)(n-k) \tag{1}$$

pairs. Replacing $a \in \mathcal{A}$ by $b \in \mathcal{B}$ yields the sorted anchor tuple $S_{a\to b} = S \setminus \{a\} \cup \{b\}$.

Definition 1 (Complete one-replacement step)**.** *A complete replacement step returns the neighbor minimizing* $(J_{\mathcal{R}}(X, S_{a\to b}), S_{a\to b})$ *lexicographically over all* $(a, b) \in \mathcal{A} \times \mathcal{B}$*. It is accepted only if its objective is strictly smaller than* $J_{\mathcal{R}}(X, S)$*.*

Comparing complete tuples makes exhaustive equivalence and paths reproducible. A strict decrease makes S *one-replacement improvable*; *complete refinement* repeats the step to a complete one-replacement local optimum. Compatible exact-k dynamic programming (DP) supplies a separate global optimum and *DP optimality gap*; it does not define replacement stopping.

Exact segment-local setting. Consecutive anchors induce $k-1$ intervals with reusable costs $c(i,j) \geq 0$: given endpoints x_i, x_j, the cost is independent of other candidate anchors, and auxiliary context is fixed across candidates and rounds. This covers geometric segments, synchronous interpolation and endpoint-conditioned interval decoders. Selected-set-dependent global reconstructors instead use the contextual extension.

The exact objective $J_{\mathcal{R}}$ is either additive risk Φ (the pth-power form for finite-p error) or bottleneck error E:

$$\Phi(S) = \sum_{t=0}^{k-2} c(s_t, s_{t+1}), \tag{2}$$

$$E(S) = \max_{0 \leq t < k-1} c(s_t, s_{t+1}). \tag{3}$$

Costs account for interior samples $i < r < j$; anchors are copied exactly, so contributions do not overlap. Our canonical bottleneck instances are maximum synchronous Euclidean distance (SED) and maximum perpendicular distance from each sampled point to the infinite supporting line of its anchor pair (PED-line). Hereafter, "PED" in tables denotes this directed sampled PED-line convention.

For sums of pth-power pointwise errors, reported risk is $(\Phi(S)/(n-k))^{1/p}$. Fixed n, k, p make this a monotone transformation preserving objective order and ties. Learned Linear experiments use $p = 1$: mean Euclidean error over the $n-k$ non-anchor samples.

2.1 Ordered Reconstruction Structure

Let $a = s_i$, $1 \leq i \leq k-2$. Deleting a removes intervals (s_{i-1}, s_i) and (s_i, s_{i+1}) and creates bridge (s_{i-1}, s_{i+1}). Every discarded point b lies in a unique current gap $g(b)$ with $s_{g(b)} < b < s_{g(b)+1}$; insertion removes that interval and creates two splits.

Definition 2 (Far and near replacements)**.** *A replacement (s_i, b) is* near *when $g(b) \in \{i-1, i\}$ and* far *otherwise.*

Lemma 1 (Linearly Many Overlap Exceptions)**.** *Deletion and insertion edit disjoint current intervals for every far pair. A discarded point is near to at least one and at most two removable anchors, so the number of near pairs satisfies*

$$n - k \leq N_{\text{near}} \leq 2(n-k). \tag{4}$$

Proof. Deletion touches gaps $i-1, i$; insertion touches $g(b)$. Each discarded point is near to one removable anchor in an endpoint gap and two in an interior gap. Summing proves the bounds. □

Additive merge/split deltas. For deletion i and insertion b in gap $g = g(b)$, define

$$D_i = c(s_{i-1}, s_{i+1}) - c(s_{i-1}, s_i) - c(s_i, s_{i+1}), \tag{5}$$

$$I_{g,b} = c(s_g, b) + c(b, s_{g+1}) - c(s_g, s_{g+1}). \tag{6}$$

By Lemma 1, every far replacement satisfies

$$\Phi(S_{s_i \to b}) = \Phi(S) + D_i + I_{g,b}. \tag{7}$$

For a near replacement, the deletion bridge is immediately split and is not a final interval:

$$\begin{aligned}\Phi(S_{s_i \to b}) = \Phi(S) &- c(s_{i-1}, s_i) - c(s_i, s_{i+1}) \\ &+ c(s_{i-1}, b) + c(b, s_{i+1}).\end{aligned} \tag{8}$$

Bottleneck critical gaps. Let $w_g = c(s_g, s_{g+1})$, $d_i = c(s_{i-1}, s_{i+1})$, and

$$\beta_{g,b} = \max\{c(s_g, b), c(b, s_{g+1})\}. \tag{9}$$

For a far replacement,

$$E(S_{s_i \to b}) = \max\left\{d_i, \beta_{g,b}, \max_{\ell \notin \{i-1, i, g\}} w_\ell\right\}. \tag{10}$$

For a near replacement,

$$E(S_{s_i \to b}) = \max\left\{c(s_{i-1}, b), c(b, s_{i+1}), \max_{\ell \notin \{i-1, i\}} w_\ell\right\}. \tag{11}$$

Equation (11) again excludes d_i because insertion splits the bridge.

3 SEGMENT-DELTA

3.1 ADDITIVE COMPILER

For every nonempty gap g, retain its best split record $r_g = \arg\min_{b \in g}(I_{g,b}, b)$. In one streaming pass, maintain the first three gap records under the order (I_{g,r_g}, r_g, g). For deletion position i, at most two gaps are forbidden; the first allowed of three retained records is therefore the exact best far insertion. We combine it with D_i using (7), and scan every near insertion in gaps $i-1, i$ using (8).

Theorem 1 (Exact additive compilation)**.** *The additive compiler returns the same objective value and best replacement under complete-tuple tie-breaking as complete enumeration. Given the requested interval costs, it uses $O(n+k)$ time and memory and scores at most $N_{\text{near}} + (k-2)$ candidate objectives.*

Proof. For fixed deletion i, D_i is constant over far insertions and every gap contributes only its minimum $I_{g,b}$. Excluding two adjacent gaps leaves the first allowed record among the global top three. Equation (8) evaluates every exception, and Lemma 1 bounds them linearly. The final comparison uses the required complete-tuple order. □

3.2 EXACT BOTTLENECK VALUE AND WITNESS COMPILER

A bottleneck value may be set by an untouched gap, and the point with minimum split value need not be the lexicographically first witness on a max plateau. The compiler therefore separates optimum-value recovery from canonical-witness recovery.

For every nonempty gap define $\bar{\beta}_g = \min_{b \in g} \beta_{g,b}$. A streaming scan stores the four largest (w_g, g) records over all current gaps, retaining multiplicities, and the four smallest $(\bar{\beta}_g, g)$ records over nonempty gaps.

Pass 1: recover the optimum value. For deletion i, exclude $F_i = \{i-1, i\}$. If the largest surviving w_g is repeated, it is an unavoidable floor, so the best far class uses the first $\bar{\beta}_g$ outside F_i. If it is unique at gap h, an optimum either splits h, using the second-largest surviving w_g and $\bar{\beta}_h$, or leaves h intact, using w_h and the first $\bar{\beta}_g$ outside $F_i \cup \{h\}$. The maximum over no untouched gap is zero, and a class with no eligible insertion is omitted. Taking the maximum with bridge cost d_i gives the exact far optimum value. Four records suffice because two gaps are excluded before the two surviving w records are compared, while the split summary may exclude those two gaps and the unique critical gap.

Taking the minimum over these values for all deletions and over every near pair gives the complete-neighborhood optimum $\tau^\star = \min_{(a,b) \in \mathcal{A}(S) \times \mathcal{B}(S)} E(S_{a \to b})$.

Pass 2: recover the canonical witness. Using this global threshold, define

$$H^\star = \{g : w_g > \tau^\star\}, \qquad q_g = \min\{b \in g : \beta_{g,b} \le \tau^\star\}, \tag{12}$$

where the minimum is the smallest eligible original sample index and $q_g = \perp$ when the set is empty. Writing $F_i = \{i-1, i\}$, a far pair in gap $g \notin F_i$ attains $\tau^\star$ exactly when

$$d_i \le \tau^\star, \qquad \beta_{g,b} \le \tau^\star, \qquad H^\star \setminus F_i \subseteq \{g\}. \tag{13}$$

When no above-threshold gap remains, the first three defined q_g records in increasing (q_g, g) order recover the smallest allowed insertion. When exactly one remains, insertion is forced there and requires its q_g to be defined. Every near pair is rescanned with (11).

Theorem 2 (Exact bottleneck compilation)**.** *The two-pass bottleneck compiler returns the same optimum value and lexicographically smallest complete anchor tuple as complete enumeration. Given the requested interval costs, it uses $O(n+k)$ time and memory and forms at most $N_{\text{near}} + 3(k-2)$ distinct candidate-pair records.*

Proof sketch. After deleting two adjacent gaps, a far insertion either removes the unique remaining critical gap or leaves the maximum untouched. The top-four summaries exhaust both cases. At the resulting optimum $\tau^\star$, (13) exactly characterizes feasible far witnesses. If no high gap survives, the first eligible of three q_g records gives the smallest allowed insertion; one high gap forces insertion

Algorithm 1 One exact Segment-Delta step

```
Build/cache requested interval costs; assign discarded points to current gaps.
if the objective is additive then
    Compile deletion deltas and the top three gap insertion records.
    Score one allowed far representative per deletion and every near pair.
else
    Compile the four largest gap costs and four smallest gap split records.
    Score the far critical-gap classes and all near pairs to obtain τ⋆.
    Recover threshold-eligible far witnesses; compare them with all near witnesses.
end if
Return the minimum objective, breaking ties by the complete resulting anchor tuple.
```

there, and two cannot be repaired by one insertion. For each deletion, the smallest eligible insertion gives the smallest resulting tuple, even on a max plateau. Comparing these witnesses and all near pairs recovers the global canonical tuple. Supplement E gives the complete cases. □

The bound counts distinct pairs. Rescanning near pairs in the witness pass gives at most $2N_{\mathrm{near}} + 3(k-2)$ candidate-objective evaluations; both bounds remain linear. Reported candidate savings count distinct pairs.

Complete-tuple ties and refinement. Algorithm 1 summarizes one exact step; strict-improvement iteration includes the terminal scan.

For moves (a, b) and (c, d), the replacement tuples differ only on $\{a, b, c, d\}$. The equal-cardinality set containing the smallest element of their symmetric difference is lexicographically smaller. This compares complete tuples in $O(1)$ without constructing each $O(k)$ tuple.

Corollary 1 (Complete one-replacement condition)**.** *If Segment-Delta returns no strict improvement, no feasible one-replacement improves the current objective. Repeatedly accepting strict improvements terminates at a complete one-replacement local optimum.*

Proof. Theorems 1–2 cover the complete neighborhood. Every accepted step strictly decreases the objective, so a fixed-cardinality anchor tuple cannot repeat; only finitely many tuples exist. □

3.3 Exact search and interval-cost complexity

Let $\mathcal{Q}_{\mathrm{SD}}(S)$ and $\mathcal{Q}_{\mathrm{EX}}(S)$ be distinct interval requests at state S, and $C(\mathcal{Q})$ their actual construction cost, including any interior scans. Both request $O(n+k)$ pairs. With the same lazy cost interface and local formulas, complete-scan costs are

$$T_{\mathrm{SD}}(S) = O\big(C(\mathcal{Q}_{\mathrm{SD}}(S)) + n + k\big), \tag{14}$$

$$T_{\mathrm{EX}}(S) = O\big(C(\mathcal{Q}_{\mathrm{EX}}(S)) + M\big). \tag{15}$$

The exhaustive baseline scores all M pairs. Since $N_{\mathrm{near}} = \Theta(n-k)$, the $\Theta(k)$ reduction in candidate comparisons is tight when $n - k = \Theta(n)$; the compilation layer remains $O(n+k)$.

Compatible exact-k layered DP uses $O(kn^2)$ transitions after $O(n^2)$ edge costs are available. Independent interior scans construct those edges in $\Theta(n^3)$; criterion-specific geometric algorithms can be faster (Related Work). Segment-Delta corrects a supplied output using $O(n+k)$ interval requests per scan.

A complete path with R accepted replacements includes $R+1$ scans, for $O((R+1)(\bar{C}+n+k))$ cost under a uniform construction bound $\bar{C}$. Strict descent gives $R < \binom{n-2}{k-2}$ in general. For bottleneck costs, every objective is one of the fixed interval values $\{c(i,j)\}$, so $R+1 \le \binom{n}{2}$, giving $O(n^2(\bar{C}+n+k))$ total cost. This level-counting bound relies on the bottleneck objective.

Prefix sums give $O(1)$ interval costs for squared synchronous error, making a full scan $O(n+k)$. The measured $p = 1$ Euclidean sums and chord-dependent SED/PED-line maxima charge raw interval construction instead. Theorems 1–2 compare a shared interval-cost table exactly. The

implementation orders computed double-precision objectives without approximate tie equality and applies any tolerance only to winner-versus-current acceptance. Classical exact-core benchmarks use zero; learned Linear runs use a roundoff allowance of $64\,\mathrm{ulp}(\max(1, |\Phi|))$. Their observed terminal equality and two intermediate floating-point witness differences are reported separately (Supplement G).

3.4 Verified contextual extension

For a replacement (a, b), let $\mathcal{I}(a, b)$ be the union of the deletion merge interval and the insertion split interval after removing a. Anchor order determines this set. Adjacent-anchor reconstruction is unchanged outside $\mathcal{I}(a, b)$, giving the exact locality used above. For a global Transformer, the anchor mask, linear bridge, and gap features still change only there, although attention can propagate their effects. This motivates using edit-interval recomputation to rank proposals, with the full reconstruction objective deciding acceptance.

One base forward pass caches hidden states, attention keys/values, and errors. We approximate each delta by recomputing states in $\mathcal{I}(a, b)$ with cached global context and candidate-specific keys/values, rank all replacements, and verify rank 1 with a full frozen-reconstructor pass. We accept a true $J_{\mathcal{R}}$ improvement and stop at the first rejection. Full verification ensures descent at every accepted step. Approximate ranking determines which proposals are tested, so termination need not be a Transformer one-replacement local optimum.

4 Experiments

We examine reconstruction gains, the cost of exhaustive-equivalent refinement, and the value of edit structure under global reconstruction.

Data, constructors and evaluators. We use GeoLife motorized trajectories (Zheng et al., 2011) and Porto taxis (Moreira-Matias et al., 2015), projected to planar meters. The learned evaluation uses all eligible entity-disjoint length-56 test windows: 1,782 from 19 GeoLife users and 1,285 from 87 Porto taxis, with $k = 14, 28$. Pointwise-TopK is a bidirectional-LSTM point scorer; Anchor Pointer uses a bidirectional encoder and autoregressive GRU pointer. Both learn from train-only, objective-matched deletion-greedy subsets and select checkpoints by validation error; three selector seeds are frozen before testing. Linear uses both selectors. The global Transformer uses TopK trained against its own objective and one validation-selected, frozen reconstructor per dataset, shared across budgets. Supplement A/H gives architectures, splits and training settings.

Linear is adjacent-anchor time-synchronous interpolation; the Transformer predicts residuals over that interpolation and copies anchors exactly. Error is mean Euclidean distance over the $n - k$ non-anchor samples, in meters. Matching n, k matches retention ratio, not duration: GeoLife/Porto windows span 55/825 seconds. GeoLife is interpolated to 1 s from a median raw cadence of 5 s, producing a small high-budget Linear error floor. Classical controls separately use 99 trajectories per dataset, $k = 5, 11$, maximum SED/PED-line, and Action-0 deletion, fixed-budget Douglas–Peucker or exact-objective forward greedy (Supplement B).

Comparisons and statistical units. Whole-objective exhaustive recomputation independently checks correctness. Localized exhaustive uses the same interval formulas and cache policy but scores all M pairs, isolating the cost of candidate comparison. Near-only best improvement tests the restriction to overlapping edits. Compatible exact-k DP diagnoses the global gap. For the Transformer, complete refinement fully evaluates all M neighbors; structural, Linear-delta and ten-seed random rankers receive one full proposal verification per round. These roles separate compiler cost, neighborhood quality and global optimality.

Quality averages three selector seeds within each window and weights windows equally. The 95% intervals use 5,000 cluster-bootstrap replicates over physical users/taxis. GeoLife's largest user supplies 64.5% of windows. The near-only audit has 144 states from 12 pilot-unseen trajectories; the nested-length timing matrix has 192 states from 16 trajectories. Repeated states and seeds do not increase the number of independent trajectories.

4.1 RECONSTRUCTION QUALITY AND THE ROLE OF COMPLETE REFINEMENT

Data/error	Constructor	k	Initial	Final	Imp. (%)	Gain (%)	DP (%)
GeoLife/SED	Action-0	5	345.24	271.81	87.9	18.6	82.2
GeoLife/SED	Action-0	11	91.76	79.59	75.8	12.3	63.9
Porto/PED	Douglas–P.	5	109.92	83.82	89.9	22.6	85.3
Porto/PED	Douglas–P.	11	30.34	25.58	69.7	14.6	76.3
Porto/PED	Forward	5	144.87	107.37	62.6	20.9	52.8
Porto/PED	Forward	11	101.41	68.87	60.6	27.6	40.2
Porto/SED	Douglas–P.	5	254.08	154.97	96.0	35.0	89.9
Porto/SED	Douglas–P.	11	160.98	66.11	98.0	50.8	83.9
Porto/SED	Forward	5	257.00	195.76	70.7	21.1	55.2
Porto/SED	Forward	11	191.26	132.57	69.7	32.1	51.6

Table 1: Classical controls on outcome-blind 99-trajectory cohorts: mean maximum SED/PED-line errors (m). Imp. is the initially improvable percentage; Gain averages per-trajectory relative reductions; DP averages closure over positive initial DP gaps. Dataset/constructor coverage is asymmetric.

Across learned cells, 95.51–100% of seed-averaged windows have positive initial DP gaps; complete Linear refinement closes 94.23–99.88% of the aggregate initial-to-DP gap (Table 2); the least favorable 95% interval is [93.69%,94.68%]. Exact DP attainment ranges from 13.52 to 89.84%, distinguishing strong correction from global optimality. Porto mean errors fall by 3.317–11.488 m across selectors and budgets, whereas GeoLife gains are 0.0034–0.0553 m in its interpolated regime. Classical maximum SED/PED-line errors also improve: 60.6–98.0% of outputs admit a replacement and mean relative gains are 12.3–50.8% (Table 1). These are reconstruction effects; downstream task benefits were not measured.

Gain capture and terminal residuals. From the same initial states, near-only captures 90.6% of the summed complete-path gain. Yet a complete scan at its own terminals finds an improving move in 86/144 states (59.7%), with positive residual medians of 0.806 m and 4.70%. Many small residuals can coexist with high total gain capture. Moreover, the two policies can follow different paths: near-only finishes better/equal/worse than the same-initial complete path in 9/48/87 states. The gain ratio measures cumulative improvement, while the residual scan detects remaining one-replacement opportunities. Defects occur on all 12 trajectories and every marginal factor level. Near-only is cheaper: median path time is 1.62 ms versus 4.29 ms for complete refinement. Compiling the complete domain adds at most $k-2$ additive or $3(k-2)$ bottleneck candidate values beyond the near pairs, given interval summaries.

4.2 EXACTNESS, MEASURED RUNTIME AND PUBLISHED-OUTPUT TRANSFER

All three solvers follow the same complete paths on 192 nested-length states and 1,314 independent whole-objective scans. The learned Linear matrix adds 36,804 selector-seed states with identical terminal objective and subset; published MLSimp adds 4,000 matching paths at $n = 1{,}000$. These checks include terminal no-improvement scans and canonical ties.

Across the 24 nested-length settings, median complete-refinement speedups are 2.11–5.55× including setup, interval construction, search and updates; search-only speedups are 3.04–8.07×, and candidate formation falls by 79.7–91.7% (Figure 2). Each treatment uses one warmup and five measured paths. The median path accepts four moves (95th percentile 18; maximum 35). Both solvers charge actual interval requests under the same scan-local cache policy. A four-trajectory, high-retention $n = 500$ mechanism test matches all 16 winners with identical interval requests and 9.19–26.96× single-scan total speedups (Supplement D).

On separate matched 128-window timing cohorts, learned Linear refinement gives 1.21× ratio-of-mean speedups at $k = 14$ and 1.44–1.46× at $k = 28$, excluding selector inference. Adding separately measured batch-128-amortized inference gives component-sum estimates of 4.634–13.753 ms for Segment-Delta and 1.207–1.450× speedup (Supplement D). CPU timing uses Python 3.11.14 on an i7-13700K; the supplement records the full environment.

Data	Selector	k	Initial	Final	DP	Closure (%)	Speedup
GeoLife	Pointer	14	0.08392	0.03872	0.03776	97.92	1.21×
GeoLife	Pointer	28	0.05803	0.00277	0.00270	99.88	1.45×
GeoLife	TopK	14	0.07019	0.03873	0.03776	97.01	1.21×
GeoLife	TopK	28	0.00615	0.00274	0.00270	98.81	1.45×
Porto	Pointer	14	22.986	16.612	16.221	94.23	1.22×
Porto	Pointer	28	9.967	6.650	6.522	96.28	1.45×
Porto	TopK	14	28.130	16.642	16.221	96.46	1.21×
Porto	TopK	28	11.063	6.656	6.522	97.05	1.44×

Table 2: Exact Linear refinement: mean non-anchor errors (m) on all test windows and three selector seeds. DP is global; closure is $100\sum(J_0 - J_f)/\sum(J_0 - J_{\mathrm{DP}})$. Speedup compares mean complete-path times against localized exhaustive on separate 128-window cohorts, excluding selector inference.

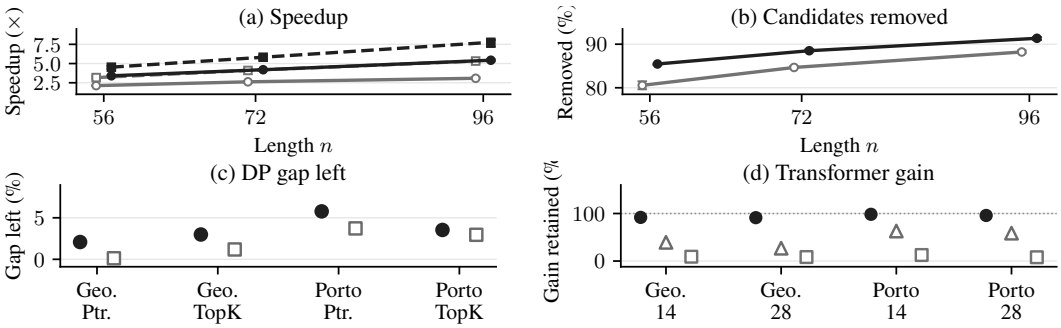


Figure 2: Exact scaling and learned-output value. (a–b) Medians and min–max over four dataset–objective cells: solid/dashed curves show total/search speedup; open/filled markers denote $k/n = 1/4, 1/2$. All 192 states match complete paths (eight trajectories per dataset, nested $n = 56, 72, 96$). (c) Residual Linear-DP gap: circles/squares denote $k = 14/28$; Geo.=GeoLife, Ptr.=Pointer. (d) Complete-refinement gain retained by Structural/Linear/Random (circles/triangles/squares), with 95% entity-cluster intervals for Structural.

Published MLSimp outputs (Song et al., 2024) test transfer beyond the controlled selectors. We preserve the realized k of each of 2,000 length-1,000 outputs at each nominal database retention rate. Table 3 reports absolute error and work-weighted candidate savings. On separate 128-path cohorts with symmetric rolling caches, marginal median complete-path times are 0.230/0.322 s for Segment-Delta and 0.264/0.500 s for localized exhaustive; paired-median speedups are 1.10/1.54×. Far moves occur on 85.4/99.2% of paths and contribute 65.4/75.1% of accepted-step gain along those realized paths, not a causal decomposition. Timestamp ties require sample-index-synchronous Linear reconstruction. This cross-objective transfer does not measure the query-trained selector's native range-query F1.

4.3 Verified contextual refinement

Matched structural, Linear and random rankers test whether edit structure remains informative with global attention. Structural ranking retains 91.25–98.32% of aggregate complete Transformer-refinement gain (Table 4), versus 26.96–63.11% for the Linear-delta proxy and 7.98–12.67% for matched random. Porto mean error falls by 12.213/5.050 m at $k = 14/28$. Supplement B reports per-seed quality; Supplement F gives evaluation counts.

We measure remaining improvement by fully scanning structural terminals on the frozen 128-window cohort per dataset: 1,536 seed-states from 256 windows. Batch-1-confirmed improving witnesses above the predeclared replay scale occur in 53.4/86.2% of GeoLife and 28.4/50.5% of Porto states at $k = 14/28$. Porto's conditional median residuals are 0.175/0.110 m (1.43/1.52%). Thus high aggregate gain retention coexists with small remaining corrections on many terminals. Supplement F gives the post-hoc diagnostic's protocol and numerical thresholds.

Rate	Realized k	Initial → final (m)	Imp. (%)	EX → SD	Removed	Speedup
1%	7 [4, 31]	488.82 → 135.99	99.6	1.334B → 62.40M	95.32%	1.10×
2%	14 [8, 58]	255.77 → 45.54	100.0	3.194B → 115.98M	96.37%	1.54×

Table 3: MLSimp transfer: 2,000 length-1,000 outputs per rate; realized k is median [P5,P95]. Error is Linear mean non-anchor error; Imp. is the improvable percentage. EX/SD are cumulative localized-exhaustive/compiled candidates (B=10^9, M=10^6); reduction is work-weighted. Speedup is the paired median on separate 128-path cohorts with matched rolling caches.

		Mean error (m)			Complete gain retained (%)		
Data	k	Initial	Complete	Struct.	Structural [95% CI]	Linear	Random
GeoLife	14	0.26824	0.13817	0.14877	91.85 [90.75, 94.95]	39.34	9.13
GeoLife	28	0.04403	0.01155	0.01440	91.25 [87.96, 94.70]	26.96	8.41
Porto	14	25.893	13.471	13.680	98.32 [98.05, 98.58]	63.11	12.67
Porto	28	11.312	6.046	6.262	95.90 [95.47, 96.34]	58.62	7.98

Table 4: Frozen global Transformer, objective-matched TopK. Each ranker fully verifies one proposal per round. Retained gain is $100 \sum(J_0 - J_{\text{ranker}}) / \sum(J_0 - J_{\text{complete}})$; structural 95% CIs use 5,000 physical-user/taxi cluster-bootstrap replicates after averaging selector seeds within windows.

The ranker scores all $M = 504/728$ moves at $k = 14/28$ through cached edit-interval recomputation, then fully evaluates rank 1. In four eight-trajectory validation cells, this implementation is slower than batched complete evaluation: complete/structural time ratios are 0.364–0.559× (Supplement F). Here the reduced number of full evaluations is outweighed by proposal construction cost.

5 Related Work

Trajectory construction. Classical and learned methods construct hard outputs (Zhang et al., 2018; Wang et al., 2021; Fang et al., 2023; Song et al., 2024); nonlocal recovery motivates our contextual test (Xia et al., 2021). Graph formulations solve compatible criteria globally (Imai & Iri, 1988). For PED-line's parallel-strip criterion, exact minimum-error construction takes $O(n^2 \log n)$ time and $O(n^2)$ space under at-most cardinality (Eu & Toussaint, 1994; Chan & Chin, 1996); our interface corrects a supplied exact-k output.

Refinement and swap acceleration. Curve refinement uses restricted DP, split/merge and local vertex relocation (Chen et al., 2012; Marteau & Ménier, 2009; Parvez, 2015); Parvez's relocation is near-type rather than complete replacement. Generic swap accelerations reuse assignment- or factor-local terms (Kaufman & Rousseeuw, 1990; Arya et al., 2004; Resende & Werneck, 2007; Chicano et al., 2014). FastPAM1 and p-center interchange summarize client-to-center assignments for additive and bottleneck moves, respectively (Schubert & Rousseeuw, 2021; Mladenović et al., 2003). Segment-Delta instead exploits ordered gap incidence: only $\Theta(n-k)$ pairs overlap; constant-size gap summaries and a threshold pass recover far optima and canonical bottleneck witnesses.

Alternative search policies. Multiple-anchor exchange and annealing explore broader exact-k searches; adapting tour 2-opt/3-opt requires preserving sample order. Their quality–cost tradeoff is a separate question from compiling a fixed one-replacement search and remains untested here.

6 Conclusion

Segment-Delta uses ordered merge–split structure to correct classical and learned outputs through complete one-replacement refinement. Its additive and bottleneck compilers preserve the exhaustive winner with $O(n + k)$ candidate values, yielding measured reconstruction gains and lower refinement cost. For global reconstruction, edit structure supplies informative proposals and full-objective verification ensures improvement at every accepted step.

AI-USE STATEMENT

Generative AI tools assisted manuscript restructuring and language editing, critical review of theoretical arguments and experimental methodology, interpretation and source checking of existing results, design and implementation of the additional terminal-residual experiment, code for evidence aggregation and artifact preparation, and construction of an illustrative example and workflow figure. Checks included comparison with the existing proof and implementation contracts, hash-traced result aggregation, exact-compiler differential tests, and inspection of the compiled documents. AI-assisted text and artifacts remain the responsibility of the authors.

REPRODUCIBILITY STATEMENT

The supplement provides complete proof cases, data and training protocols, numerical contracts, per-seed results, timing details and source provenance. The anonymous code/data supplement contains the exact compilers, independent references, differential tests, a synthetic demo and compact aggregate evidence with checksums. Raw trajectory datasets, neural checkpoints and the full training pipeline are not included; the package documents this reproduction boundary and the public data sources.

REFERENCES

Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004. doi: 10.1137/S0097539702416402. Conference version in STOC 2001.

W. S. Chan and F. Y. L. Chin. Approximation of polygonal curves with minimum number of line segments or minimum error. *International Journal of Computational Geometry & Applications*, 6(1):59–77, 1996. doi: 10.1142/S0218195996000058.

Minjie Chen, Mantao Xu, and Pasi Fränti. A fast $O(N)$ multiresolution polygonal approximation algorithm for GPS trajectory simplification. *IEEE Transactions on Image Processing*, 21(5): 2770–2785, 2012. doi: 10.1109/TIP.2012.2186146.

Francisco Chicano, Darrell Whitley, and Andrew M. Sutton. Efficient identification of improving moves in a ball for pseudo-boolean problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 437–444, 2014. doi: 10.1145/2576768.2598304.

David Eu and Godfried T. Toussaint. On approximating polygonal curves in two and three dimensions. *CVGIP: Graphical Models and Image Processing*, 56(3):231–246, 1994. doi: 10.1006/cgip.1994.1021.

Ziquan Fang, Changhao He, Lu Chen, Danlei Hu, Qichen Sun, Linsen Li, and Yunjun Gao. A lightweight framework for fast trajectory simplification. In *Proceedings of the 39th IEEE International Conference on Data Engineering*, pp. 2386–2399, 2023. doi: 10.1109/ICDE55515.2023.00184.

Hiroshi Imai and Masao Iri. Polygonal approximations of a curve—formulations and algorithms. In Godfried T. Toussaint (ed.), *Computational Morphology*, volume 6 of *Machine Intelligence and Pattern Recognition*, pp. 71–86. North-Holland, 1988. doi: 10.1016/B978-0-444-70467-2.50011-4.

Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York, 1990. doi: 10.1002/9780470316801.

Pierre-François Marteau and Gildas Ménier. Tabu split and merge for the simplification of polygonal curves. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 1322–1327, 2009. doi: 10.1109/ICSMC.2009.5346240.

Nenad Mladenović, Martine Labbé, and Pierre Hansen. Solving the p-center problem with tabu search and variable neighborhood search. *Networks*, 42(1):48–64, 2003. doi: 10.1002/net.10081.

Luís Moreira-Matias, Michel Ferreira, and João Mendes-Moreira. Taxi service trajectory—prediction challenge, ECML PKDD 2015. UCI Machine Learning Repository, 2015.

Mohammad Tanvir Parvez. Optimized polygonal approximations through vertex relocations in contour neighborhoods. *Image and Vision Computing*, 34:1–10, 2015. doi: 10.1016/j.imavis.2014.10.012.

Mauricio G. C. Resende and Renato F. Werneck. A fast swap-based local search procedure for location problems. *Annals of Operations Research*, 150(1):205–230, 2007. doi: 10.1007/s10479-006-0154-0.

Erich Schubert and Peter J. Rousseeuw. Fast and eager k-medoids clustering: $O(k)$ runtime improvement of the PAM, CLARA, and CLARANS algorithms. *Information Systems*, 101:101804, 2021. doi: 10.1016/j.is.2021.101804.

Yumeng Song, Yu Gu, Tianyi Li, Yushuai Li, Christian S. Jensen, and Ge Yu. Quantifying point contributions: A lightweight framework for efficient and effective query-driven trajectory simplification. *Proceedings of the VLDB Endowment*, 18(2):453–465, 2024. doi: 10.14778/3705829.3705858.

Zheng Wang, Cheng Long, and Gao Cong. Trajectory simplification with reinforcement learning. In *Proceedings of the 37th IEEE International Conference on Data Engineering*, pp. 684–695, 2021. doi: 10.1109/ICDE51399.2021.00065.

Tong Xia, Yunhan Qi, Jie Feng, Fengli Xu, Funing Sun, Diansheng Guo, and Yong Li. AttnMove: History enhanced trajectory recovery via attentional network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4494–4502, 2021. doi: 10.1609/aaai.v35i5.16577.

Dongxiang Zhang, Mengting Ding, Dingyu Yang, Yi Liu, Ju Fan, and Heng Tao Shen. Trajectory simplification: An experimental study and quality analysis. *Proceedings of the VLDB Endowment*, 11(9):934–946, 2018. doi: 10.14778/3213880.3213885.

Yu Zheng, Hao Fu, Xing Xie, Wei-Ying Ma, and Quannan Li. GeoLife GPS trajectory dataset: User guide. Microsoft Research, 2011. URL https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/.

SUPPLEMENTARY MATERIAL

This supplement reports the frozen protocol and disaggregated evidence behind the main paper. It introduces no additional method or post-test model choice. Numerical tables are generated from frozen result files; the source manifests and generators record provenance. Timing component analyses reuse existing measurements; the new terminal-residual audit in Section F makes additional full-model evaluations on frozen states.

A DATA AND PROTOCOL DETAILS

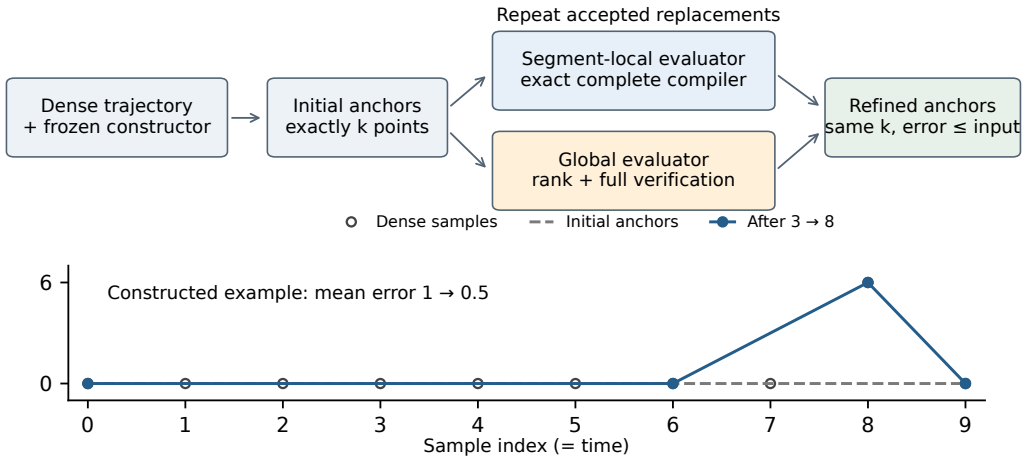


Figure 3: The constructed ten-sample example used in the introduction. With uniform timestamps, all points have coordinates $(t, 0)$ except $(8, 6)$. Replacing anchor 3 by 8 changes $(0, 3, 6, 9)$ to $(0, 6, 8, 9)$ and total Linear error from 6 to 3; six non-anchor points give mean error 1 to 0.5. This is an analytical illustration, not an empirical result.

The controlled learned matrix retains both endpoints and uses $k \in \{14, 28\}$. GeoLife uses a frozen Beijing region of interest (ROI) and Porto uses a frozen city-center ROI. All eligible systematic windows are retained; no result-dependent trajectory sampling is performed. The primary objective is mean Euclidean reconstruction error over the $n - k$ missing points in meters. Coordinate normalization is used only for model inputs and training; every reported objective is evaluated after inverse transformation in local planar meters.

Paper-facing point estimates average the three selector seeds within each trajectory UID and then weight trajectory UIDs equally. Population uncertainty uses 5,000 physical-entity cluster-bootstrap replicates: 19 GeoLife users or 87 Porto taxis are sampled with replacement, and every test window of a sampled entity is retained. For ratio statistics, the complete entity population is sampled before the frozen positive-denominator UID mask is applied. This retains the frozen point estimates. One GeoLife user contributes 1,150/1,782 windows (64.53%); the largest Porto taxi contributes 72/1,285 (5.60%). Cluster intervals account for within-entity dependence but the estimand still weights windows, not users, equally.

Reconstructors. Linear denotes adjacent-anchor time-synchronous interpolation. The global Transformer has three layers, four heads, 96-dimensional states, a 192-dimensional feed-forward block, and dropout 0.1. Its inputs are the linear bridge, normalized time, anchor mask, linear and log time gaps to the neighboring anchors, budget ratio, and normalized ROI origin. It predicts a coordinate residual and copies anchors exactly.

Each dataset has its own Transformer, shared across both budgets. Models use AdamW, batch size 128, gradient clipping at 1.0, and three seeds 20260725–20260727. Training uses fresh endpoint-preserving uniform exact-k masks; checkpoint selection uses four frozen validation masks per tra-

jectory and budget. The predeclared validation-median seed is frozen before selector training. Structured anchor masks are out-of-distribution diagnostics and do not select or veto a checkpoint.

Learned simplifiers. Pointwise-TopK uses a two-layer bidirectional LSTM with hidden size 96 and selects the $k-2$ largest interior logits. Anchor Pointer uses the same-size encoder and an autoregressive GRU pointer constrained to increasing indices. Models are shared across both budgets within a dataset–reconstructor pair. Supervision consists only of train-set exact-objective deletion-greedy subsets, obtained by starting from all samples and repeatedly deleting the point that minimizes the fixed reconstructor's exact current objective until k anchors remain. Checkpoints minimize validation reconstruction error, and all three predeclared seeds are retained.

Data	Cadence	Duration	Train	Validation	Test
GeoLife	1 s	55 s	3,967	438	1,782
Porto	15 s	825 s	3,665	1,228	1,285

Table 5: Entity-disjoint length-56 cohorts. Matching n and k matches retention ratio, not physical duration or spatial scale.

B Classical and Learned-Output Quality

The classical controls use 99 trajectories per dataset and the asymmetric combinations shown in Table 1 of the main paper; the learned matrix below uses all eligible test windows.

The Linear rows verify exhaustive equivalence separately for every retained selector seed. The Transformer rows show that gain capture is not produced by seed averaging: the structural proposal retains most complete-refinement gain for every dataset, budget, and seed, while matched random is weaker in every primary aggregate reported in the main paper.

Recon.	Selector	k	Seed	Initial	Seg.-Δ gain	Comp. gain	Improvable (%)	Capture (%)
Linear	Pointer	14	25	0.09634	0.05766	0.05766	82.94	100.00
Linear	Pointer	28	25	0.04063	0.03788	0.03788	98.99	100.00
Linear	Pointer	14	26	0.07208	0.03353	0.03353	83.39	100.00
Linear	Pointer	28	26	0.01959	0.01684	0.01684	98.82	100.00
Linear	Pointer	14	27	0.08333	0.04440	0.04440	81.87	100.00
Linear	Pointer	28	27	0.11386	0.11108	0.11108	98.99	100.00
Linear	TopK	14	25	0.08573	0.04672	0.04672	79.80	100.00
Linear	TopK	28	25	0.00840	0.00566	0.00566	98.65	100.00
Linear	TopK	14	26	0.06122	0.02266	0.02266	76.71	100.00
Linear	TopK	28	26	0.00476	0.00203	0.00203	96.97	100.00
Linear	TopK	14	27	0.06362	0.02501	0.02501	77.33	100.00
Linear	TopK	28	27	0.00529	0.00255	0.00255	97.14	100.00
Trans.	TopK	14	25	0.25876	0.11137	0.12071	97.59	92.26
Trans.	TopK	28	25	0.04465	0.03048	0.03308	99.38	92.13
Trans.	TopK	14	26	0.27206	0.12232	0.13360	96.91	91.56
Trans.	TopK	28	26	0.04015	0.02574	0.02862	99.21	89.96
Trans.	TopK	14	27	0.27391	0.12472	0.13592	97.14	91.76
Trans.	TopK	28	27	0.04730	0.03269	0.03573	99.21	91.48

Table 6: GeoLife per-seed quality; seeds 25–27 denote 20260725–20260727. Initial errors and mean reductions are non-anchor meters. Improvable is the fraction with positive complete-refinement gain; Capture is the ratio of cohort-mean gains. Linear rows are exhaustive-equivalent.

Recon.	Selector	k	Seed	Initial	Seg.-Δ gain	Comp. gain	Improvable (%)	Capture (%)
Linear	Pointer	14	25	22.391	5.761	5.761	99.69	100.00
Linear	Pointer	28	25	9.532	2.886	2.886	99.30	100.00
Linear	Pointer	14	26	22.300	5.707	5.707	99.61	100.00
Linear	Pointer	28	26	9.477	2.826	2.826	99.38	100.00
Linear	Pointer	14	27	24.266	7.653	7.653	99.92	100.00
Linear	Pointer	28	27	10.893	4.240	4.240	99.84	100.00
Linear	TopK	14	25	28.273	11.630	11.630	100.00	100.00
Linear	TopK	28	25	11.123	4.466	4.466	100.00	100.00
Linear	TopK	14	26	28.448	11.815	11.815	99.92	100.00
Linear	TopK	28	26	11.115	4.460	4.460	100.00	100.00
Linear	TopK	14	27	27.671	11.019	11.019	99.92	100.00
Linear	TopK	28	27	10.950	4.295	4.295	100.00	100.00
Trans.	TopK	14	25	25.991	12.315	12.497	99.92	98.54
Trans.	TopK	28	25	11.392	5.130	5.344	100.00	95.99
Trans.	TopK	14	26	25.826	12.115	12.348	100.00	98.12
Trans.	TopK	28	26	11.221	4.979	5.175	100.00	96.21
Trans.	TopK	14	27	25.863	12.210	12.422	99.92	98.29
Trans.	TopK	28	27	11.322	5.040	5.277	100.00	95.51

Table 7: Porto per-seed quality; seeds 25–27 denote 20260725–20260727. Initial errors and mean reductions are non-anchor meters. Improvable is the fraction with positive complete-refinement gain; Capture is the ratio of cohort-mean gains. Linear rows are exhaustive-equivalent.

C Global-Optimum Context for Linear Reconstruction

Dynamic programming supplies the global fixed-k reference. Complete refinement removes most of the measurable initial-to-DP gap; exact DP attainment varies by dataset, budget, selector, and seed. The tables report gap closure and exact attainment separately to locate the terminal solutions relative to that reference.

Data	Selector	k	Seed	Gap>0 (%)	Initial	Final	DP	Gap closed (%)	Reach DP (%)
GeoLife	Pointer	14	25	94.73	0.09634	0.03868	0.03776	98.43	92.59
GeoLife	Pointer	28	25	99.38	0.04063	0.00276	0.00270	99.84	26.09
GeoLife	Pointer	14	26	94.56	0.07208	0.03855	0.03776	97.69	93.04
GeoLife	Pointer	28	26	99.27	0.01959	0.00275	0.00270	99.67	27.55
GeoLife	Pointer	14	27	93.60	0.08333	0.03893	0.03776	97.43	91.92
GeoLife	Pointer	28	27	99.38	0.11386	0.00279	0.00270	99.92	25.48
GeoLife	TopK	14	25	92.37	0.08573	0.03901	0.03776	97.39	91.53
GeoLife	TopK	28	25	99.10	0.00840	0.00274	0.00270	99.19	35.02
GeoLife	TopK	14	26	90.24	0.06122	0.03857	0.03776	96.55	92.93
GeoLife	TopK	28	26	97.70	0.00476	0.00273	0.00270	98.32	31.54
GeoLife	TopK	14	27	90.63	0.06362	0.03861	0.03776	96.72	92.54
GeoLife	TopK	28	27	97.70	0.00529	0.00274	0.00270	98.38	33.84

Table 8: GeoLife per-seed fixed-k Linear dynamic-programming audit. Gap closed is the aggregate initial-to-DP gap removed by complete refinement. Gap>0 is the prevalence of a positive initial-to-DP gap. Initial, Final, and DP are mean missing-point Euclidean errors in meters after direct evaluation on the stored metric coordinates. Reach DP reports exact global-optimum attainment and is not used as a theoretical guarantee.

Data	Selector	k	Seed	Gap>0 (%)	Initial	Final	DP	Gap closed (%)	Reach DP (%)
Porto	Pointer	14	25	99.92	22.391	16.629	16.221	93.38	39.77
Porto	Pointer	28	25	99.61	9.532	6.646	6.522	95.87	46.69
Porto	Pointer	14	26	99.84	22.300	16.593	16.221	93.88	42.33
Porto	Pointer	28	26	99.77	9.477	6.651	6.522	95.63	46.93
Porto	Pointer	14	27	99.92	24.266	16.613	16.221	95.13	42.88
Porto	Pointer	28	27	99.92	10.893	6.653	6.522	97.00	46.07
Porto	TopK	14	25	100.00	28.273	16.642	16.221	96.51	40.62
Porto	TopK	28	25	100.00	11.123	6.657	6.522	97.06	45.60
Porto	TopK	14	26	99.92	28.448	16.633	16.221	96.63	40.54
Porto	TopK	28	26	100.00	11.115	6.656	6.522	97.09	44.36
Porto	TopK	14	27	99.92	27.671	16.652	16.221	96.24	40.16
Porto	TopK	28	27	100.00	10.950	6.655	6.522	96.99	45.68

Table 9: Porto per-seed fixed-k Linear dynamic-programming audit. Gap closed is the aggregate initial-to-DP gap removed by complete refinement. Gap>0 is the prevalence of a positive initial-to-DP gap. Initial, Final, and DP are mean missing-point Euclidean errors in meters after direct evaluation on the stored metric coordinates. Reach DP reports exact global-optimum attainment and is not used as a theoretical guarantee.

D Exact Runtime and Scaling Details

Data	Constructor/k	Error (m) initial$\rightarrow$final	Selector (ms)	EX sum (ms)	SD sum (ms)	Ratio
GeoLife	Pointer/14	$0.08392 \rightarrow 0.03872$	0.049	5.820	4.810	1.210$\times$
GeoLife	Pointer/28	$0.05803 \rightarrow 0.00277$	0.299	18.804	13.069	1.439$\times$
GeoLife	TopK/14	$0.07019 \rightarrow 0.03873$	0.024	5.594	4.634	1.207$\times$
GeoLife	TopK/28	$0.00615 \rightarrow 0.00274$	0.150	19.940	13.753	1.450$\times$
Porto	Pointer/14	$22.986 \rightarrow 16.612$	0.168	11.125	9.180	1.212$\times$
Porto	Pointer/28	$9.967 \rightarrow 6.650$	0.205	10.935	7.602	1.438$\times$
Porto	TopK/14	$28.130 \rightarrow 16.642$	0.132	12.813	10.593	1.210$\times$
Porto	TopK/28	$11.063 \rightarrow 6.656$	0.111	14.345	9.984	1.437$\times$

Table 10: Existing-measurement estimate for learned Linear inference plus complete refinement. Error uses all 1,782 GeoLife or 1,285 Porto test windows; EX and SD have identical terminal quality. Timing uses separate matched 128-window cohorts. Selector time is batch-128 inference divided by 128. EX/SD sums add this amortized selector cost to the respective measured refinement mean; Ratio is EX sum divided by SD sum. Five measured repeats are reduced by their median within selector seed and trajectory, then three seeds are averaged within trajectory and trajectories equally weighted. These are sums of separately measured components, not measured single-query latency; loading, training, and data ingest are excluded.

The component-sum table combines batch-amortized selector inference with per-window refinement. It reaggregates 36,864 existing timing rows and checks 18,432 paired terminal-quality signatures. The sum estimates the cost of these measured components; single-query latency and deployment throughput would additionally depend on batching and system overhead.

The controlled timing comparison changes only complete-neighborhood evaluation. Selector inference is outside both refinement times. Localized exact exhaustive and Segment-Delta use the same current subset, interval-cost interface, acceptance rule, and complete-tuple tie breaking.

The scaling matrix repeats complete best-improvement refinement through the terminal no-improvement scan. "Total speed" includes setup, requested interval-cost construction, search, and state updates. "Search speed" isolates candidate comparison after interval costs are available.

High-n fixed-ratio mechanism matrix. A separately locked single-scan preflight uses centered $n = 500$ windows from two outcome-blind official MLSimp GeoLife test paths and two outcome-blind, distinct-entity Porto paths. It crosses SED/PED-line with $k/n \in \{1/4, 1/2\}$ from the same

Data	Selector	k	Localized exact (ms)	Seg.-Δ (ms)	Mean-time speedup	95% CI
GeoLife	Pointer	14	5.772	4.762	1.212	[1.209,1.215]
GeoLife	Pointer	28	18.505	12.770	1.449	[1.447,1.451]
GeoLife	TopK	14	5.570	4.610	1.208	[1.205,1.212]
GeoLife	TopK	28	19.790	13.603	1.455	[1.450,1.460]
Porto	Pointer	14	10.957	9.012	1.216	[1.213,1.219]
Porto	Pointer	28	10.731	7.398	1.451	[1.448,1.454]
Porto	TopK	14	12.680	10.461	1.212	[1.209,1.216]
Porto	TopK	28	14.234	9.873	1.442	[1.415,1.469]

Table 11: Controlled learned Linear complete-path timing on each fixed 128-path cohort. Localized exact exhaustive and Segment-Delta return the same terminal subset and objective; Speedup is the ratio of cohort-mean refinement times, and the paired bootstrap interval is over the frozen timing cohort.

fixed-budget Douglas–Peucker initializer, using one warmup and three measured repeats. Segment-Delta matches all 16 localized-exhaustive winners. The two solvers request exactly the same interval costs in every state; compiling 46,125/62,000 pairs into 988–1,101/991–996 candidate values yields 9.19–26.96× total and 25.31–41.53× search speedup. This four-trajectory result isolates the predicted high-retention comparison mechanism; it is not a complete-refinement or population estimate.

Per-scan and complete-refinement cost. The per-scan complexity equation in the main paper is the cost of one complete-neighborhood scan at one current anchor tuple. If a path accepts R replacements, it contains $R+1$ scans including the terminal no-improvement scan. Its exact cost is the sum of the per-state construction and search costs; using a uniform per-state construction bound gives $O((R+1)(\bar{C}+n+k))$ for Segment-Delta. In general, strict descent prevents a tuple from recurring, so $R < \binom{n-2}{k-2}$. For the bottleneck objective, let $\mathcal{V}_X = \{c(i,j) : 0 \le i < j < n\}$ be the distinct values in the fixed instance-level interval table. Since every state satisfies $E(S) \in \mathcal{V}_X$, $R+1 \le |\mathcal{V}_X| \le \binom{n}{2}$; hence complete bottleneck refinement costs $O(n^2(\bar{C}+n+k))$ over at most $\binom{n}{2}$ scans. This level bound need not hold for additive or selected-set-dependent objectives. Across the 192 frozen nested-length scaling paths, the number of accepted moves has mean 5.84, median 4, 95th percentile 18, and maximum 35; the corresponding scan counts are one larger and sum to 1,314.

The reported $O(n+k)$ search memory is also per scan. The implementation creates a fresh lazy interval cache for each current state and discards it after an accepted replacement in the main scaling matrix. The separate rolling-cache experiment in Section I retains only the current and immediately preceding scan generations. Since either generation contains $O(n+k)$ requests, this policy still stores $O(n+k)$ interval records; its peak entries are reported rather than treated as free. Prefix sums give constant-time costs for squared synchronous error, but not for the reported $p=1$ Euclidean sum or chord-dependent SED/PED-line maxima. Their raw oracle scans represented samples and can make construction quadratic per scan; a specialized constant-time oracle would leave the complete Segment-Delta scan linear.

Data	Metric	k/n	n	k	Moves	Candidate red. (%)	Total speedup (×)	Search speedup (×)
GeoLife	PED	1/2	56	28	1.0	85.53	3.436	4.582
GeoLife	PED	1/2	72	36	1.5	88.81	4.290	5.935
GeoLife	PED	1/2	96	48	1.5	91.50	5.507	7.929
GeoLife	PED	1/4	56	14	1.5	79.81	2.109	3.077
GeoLife	PED	1/4	72	18	1.5	85.00	2.632	4.154
GeoLife	PED	1/4	96	24	2.5	88.35	3.076	5.379
GeoLife	SED	1/2	56	28	8.5	85.20	3.380	4.338
GeoLife	SED	1/2	72	36	7.5	88.31	4.168	5.544
GeoLife	SED	1/2	96	48	6.5	90.98	5.297	7.157
GeoLife	SED	1/4	56	14	7.5	79.65	2.152	3.037
GeoLife	SED	1/4	72	18	11.0	84.60	2.707	4.051
GeoLife	SED	1/4	96	24	15.0	87.94	3.226	5.127
Porto	PED	1/2	56	28	1.5	85.46	3.445	4.697
Porto	PED	1/2	72	36	2.0	88.64	4.227	6.046
Porto	PED	1/2	96	48	2.0	91.65	5.547	8.066
Porto	PED	1/4	56	14	1.0	81.48	2.119	3.361
Porto	PED	1/4	72	18	2.5	84.72	2.492	4.169
Porto	PED	1/4	96	24	2.0	88.56	3.090	5.401
Porto	SED	1/2	56	28	8.5	85.45	3.362	4.472
Porto	SED	1/2	72	36	5.0	88.36	4.139	5.707
Porto	SED	1/2	96	48	8.0	91.21	5.356	7.595
Porto	SED	1/4	56	14	6.5	81.30	2.134	3.268
Porto	SED	1/4	72	18	4.0	84.58	2.614	4.029
Porto	SED	1/4	96	24	8.0	88.07	3.052	5.269

Table 12: Full nested-length complete-refinement matrix. Values are cell medians over eight trajectories and five measured repeats for each local solver. Moves is the number of accepted replacements; candidate reduction is the cumulative reduction relative to localized exact exhaustive. Every one of the 192 complete-refinement paths matches the whole-objective exhaustive reference. PED denotes PED-line.

E Complete Bottleneck Proof

We give the omitted cases behind Theorem 2. We use the main paper's notation $w_g, d_i, \beta_{g,b}, \tau^\star, H^\star$, and q_g. Let $F_i = \{i-1, i\}$ be the gaps removed by deleting anchor s_i, and let $\bar{\beta}_g = \min_{b \in g} \beta_{g,b}$ for each nonempty gap.

Value pass. Fix a deletion i and consider only far insertions. If the largest w_g among gaps outside F_i has multiplicity at least two, removing one insertion gap cannot remove every copy. The surviving bottleneck floor is therefore fixed, and the best far value is obtained by the smallest $\bar{\beta}_g$ outside F_i. If the largest surviving gap is unique, say h, every far insertion belongs to exactly one of two classes: it splits h, leaving the second-largest surviving w_g, or it splits another gap and leaves w_h. The first class is absent when h is empty; the second is absent when no other nonempty gap exists.

The four largest current-gap records suffice because excluding the two gaps in F_i leaves the first two survivors among those four. Likewise, the four smallest per-gap $\bar{\beta}_g$ records suffice: the repeated-maximum case excludes two gaps, while the unique-maximum outside class excludes $F_i \cup \{h\}$, at most three gaps. Consequently the value pass evaluates the exact best far value for every deletion. Direct evaluation of all near pairs by Equation (11) completes the exact optimum value $\tau^\star$ over the full neighborhood. When fewer than four records exist, the scan retains all of them; a class whose required record or eligible insertion is absent is simply omitted.

Threshold witness pass. For a far pair in gap $g \notin F_i$, Equation (10) is at most $\tau^\star$ if and only if

$$d_i \le \tau^\star, \qquad \beta_{g,b} \le \tau^\star, \qquad H^\star \setminus F_i \subseteq \{g\}.$$

Necessity follows because the bridge, both split costs, and every untouched above-threshold gap remain in the maximum. Sufficiency follows because these are all terms in Equation (10). Since $\tau^\star$ is the minimum value over the complete neighborhood, every such candidate attains it exactly.

Define q_g only when gap g contains a point with $\beta_{g,b} \leq \tau^\star$, choosing the smallest such index. If $H^\star \setminus F_i$ is empty, the lexicographically smallest far witness for this deletion is the smallest defined q_g outside F_i; the first three global records suffice after excluding two gaps. If the remaining set is the singleton $\{h\}$, insertion is forced into h and exists exactly when q_h is defined. With two or more remaining high gaps, no single insertion can remove them all. Empty gaps create no q_g record.

For a fixed deletion, the smaller inserted index gives the smaller complete resulting tuple. Across different deletions, the constant-time complete-tuple comparator inspects the at most four indices in the symmetric difference. All near pairs are explicitly rescanned, so the recovered minimum tuple is exactly the exhaustive tie-broken witness. The argument also covers $k = 3$ (where every legal replacement may be near), consecutive anchors and empty gaps, repeated maxima, and states with no feasible far witness.

F Global-Transformer Refinement and Stopping Behavior

Data	k	Validation trajectories	Complete/Structural time
GeoLife	14	8	0.382×
GeoLife	28	8	0.559×
Porto	14	8	0.380×
Porto	28	8	0.364×

Table 13: Validation-only contextual timing audit for Pointwise-TopK selector seed 20260725. Entries are median paired complete-refinement time divided by structural-refinement time; values below one mean structural recomputation is slower. These four eight-trajectory validation cells motivated omitting a formal test-runtime claim. They are separate from the full held-out quality evaluation and are not test-population runtime estimates.

The frozen validation timing audit found that all-pair structural recomputation was slower than batched complete evaluation. The table reports these four validation cells; formal test-set contextual timing was not run. Proposal quality is evaluated separately on the complete test population.

For a contextual reconstructor, Segment-Delta scores the same retained– discarded candidate domain through localized recomputation and fully evaluates the rank-1 proposal. A move is accepted only when that full evaluation improves the frozen true objective, so every accepted move decreases the evaluated reconstruction error. Across the complete test matrix, 75,942 proposals pass this check. Structural ranking first scores all M candidates through cached edit-interval recomputation and then fully evaluates rank 1; complete refinement fully evaluates all M candidates. Each of the 18,402 paths ends with one rejected proposal, giving an aggregate proposal acceptance rate of 80.49%; this is path-length-dependent workload accounting. Complete and proposal paths may diverge after their first different move. We measure remaining improvement directly at the proposal terminals through the complete scans below.

F.1 Complete-neighborhood terminal audit

Data	k	Verified residual states/384	Median positive gain (m)	Median relative gain (%)
GeoLife	14	205/384	0.00419	3.44
GeoLife	28	331/384	0.00042	5.83
Porto	14	109/384	0.175	1.43
Porto	28	194/384	0.110	1.52

Table 14: Complete-neighborhood audit at structural Transformer terminal states on the frozen 128-window cohort per dataset, both budgets, and three selector seeds. A counted residual is confirmed by separate full batch-1 forwards and exceeds the predeclared replay scale $5 \times 10^{-5} + 10^{-5} \max(|J_{\text{base}}|, |J_{\text{witness}}|)$ meters. Medians are conditional on these positive states. This directly diagnoses proposal stopping; the 1,536 states represent 256 windows, not independent trajectories.

Data	k/seed	Nominal /128	Robust /128	Median positive gain (m)	P95 positive gain (m)
GeoLife	14/25	68	67	0.00419	0.03761
GeoLife	14/26	70	69	0.00475	0.02452
GeoLife	14/27	70	69	0.00411	0.03303
GeoLife	28/25	119	111	0.00046	0.00612
GeoLife	28/26	118	113	0.00038	0.00896
GeoLife	28/27	118	107	0.00045	0.00708
Porto	14/25	38	38	0.187	0.807
Porto	14/26	31	31	0.225	1.116
Porto	14/27	40	40	0.149	0.819
Porto	28/25	65	65	0.122	0.670
Porto	28/26	63	63	0.101	0.554
Porto	28/27	66	66	0.107	0.590

Table 15: Per-selector-seed terminal residual audit. Seed suffixes 25–27 mean 20260725–20260727. Nominal counts use the original 10^{-6} m acceptance threshold after batch-1 confirmation; robust counts use the stricter predeclared replay-scale threshold. Residual magnitudes condition on robust positives. All nonrobust signed deltas, including nominal improvements below the robust threshold, are retained in the result supplement.

We therefore audit the actual structural terminals directly. This is a post-hoc diagnostic after the original quality results were opened, with its cohort and numerical rules fixed before any new neighborhood evaluations. It uses all 128 windows per dataset in the original outcome-blind timing cohorts, both budgets and all three selector seeds: 1,536 states from 256 windows and 70 physical entities (11 GeoLife users, 59 Porto taxis). Every frozen checkpoint, initial subset and previously accepted path remains unchanged. We do not continue refinement or retune the proposal.

At each terminal we enumerate the full $M = 504/728$ candidates under the frozen Transformer, then separately confirm the current state and batch-ranked best candidate with ordinary full batch-1 forwards. Nominal positive gain uses the original 10^{-6} m threshold. A robust witness must additionally exceed

$$5 \times 10^{-5} + 10^{-5} \max(|J_{\text{base}}|, |J_{\text{witness}}|) \quad \text{meters.}$$

This is the existing empirical replay scale, not a proven floating-point error bound. The audit evaluates 946,176 candidates. It finds 866 nominal and 839 robust positive witnesses; all 27 nominal-but-nonrobust cases are GeoLife states and remain in the reporting artifacts. All original terminal objectives replay exactly; the largest batch-versus-singleton winner difference is 4.01×10^{-5} m. Independent complete enumeration on 24 states selected before execution reproduces all objective scores with zero discrepancy. Independent aggregate checks cover every saved state and classification.

Tables 14–15 report cells and seeds rather than treating repeated states as independent trajectories. Porto's robust-positive conditional median residuals are 0.175/0.110 m (1.43/1.52%) at $k = 14/28$. These remaining corrections are small relative to its initial refinement gains, but positively identify nonoptimal proposal stopping. The absence of a robust witness is not a mathematical local-optimality certificate under approximate neural arithmetic.

A. Full-objective operation accounting

Data	k	Seg.-Δ full evals	Complete full evals	Reduction (%)	Localized candidate scores
GeoLife	14	24,227	11,360,970	99.787	9,516,024
GeoLife	28	31,697	31,420,730	99.899	19,183,528
Porto	14	27,892	12,989,415	99.785	12,114,648
Porto	28	28,930	20,108,303	99.856	18,254,600

B. Matched proposal-ranking ablation

Data	k	Structural	Linear	Random
		Complete-refinement gain retained (%)		
GeoLife	14	91.85	39.34	9.13
GeoLife	28	91.25	26.96	8.41
Porto	14	98.32	63.11	12.67
Porto	28	95.90	58.62	7.98

Table 16: Global-Transformer accounting and matched ranking ablation over all three selector seeds. In A, localized candidate scores rank the complete domain; one initial base pass per path, accepted proposals and terminal checks invoke full frozen-reconstructor evaluation. These counts support evaluation reduction, not a contextual wall-clock claim. In B, all methods start from the same frozen Pointwise-TopK states and fully verify one proposal per round. Linear uses the exact additive time-synchronous Linear objective; Random is verification-matched over ten seeds.

G Numerical Contract and Artifact Provenance

Theorems 1–2 use exact real arithmetic and the canonical complete resulting tuple order. The implementation uses IEEE 754 double precision. Candidate ordering compares the computed objective and then the exact tuple order; it does not use a non-transitive approximate-equality relation. An `improvement_tolerance`, when nonzero, is applied only after the best neighbor has been selected, to the winner-versus-current acceptance test. Classical exact-core complete-refinement experiments use tolerance zero. Controlled learned Linear runs use $64\,\mathrm{ulp}(\max(1, |\Phi|))$, in additive-sum units, only for winner-versus-current acceptance. This does not redefine the candidate tie order.

The nested-length scaling audit contains 192 initial states. Whole-objective exhaustive replay performs 1,314 complete scans. Whole-objective exhaustive, localized exact exhaustive, and Segment-Delta agree on every round, accepted move, objective, subset, and terminal local optimum. Five measured repeats per local solver yield 1,920 measured complete-path records; including one warmup per treatment gives 2,304 records.

The controlled learned Linear matrix contains 36,804 selector-seed states. Segment-Delta and complete refinement have zero terminal-objective and terminal-subset mismatches. Two intermediate paths differ through floating-point-equivalent witnesses before reaching the same terminal subset and objective.

Physical-unit replay. The stored `x_m`/`y_m` arrays are projected local-planar meters. Linear reconstruction consumes them directly; the Transformer divides coordinates by a train-only physical scale and multiplies the missing-point error by the same scale. An independent replay on the stored metric coordinates reproduces all 21,384 GeoLife Linear Segment-Delta state objectives exactly; independent Transformer replay differs by at most 9.54×10^{-6} m. GeoLife's small high-budget values are not kilometers: the frozen source linearly resamples at 1 s from contributing segments with 5 s median raw cadence, leaving many samples on a common interpolation span.

The source-to-table hashes are recorded in the supplement result-source manifest, separately from the main-paper asset manifests.

H Reproducibility Contract

Randomness and inference. Neural reconstructors and simplifiers use seeds $\{20260725, 20260726, 20260727\}$. Each training job resets Python, NumPy, and PyTorch

generators before model initialization; deterministic CUDA evaluation is used for the frozen test matrix. The contextual matched-random control uses seeds 20260800–20260809. Controlled learned quality and DP 95% intervals use 5,000 paired physical-entity cluster-bootstrap replicates with seed 20260726. Controlled Linear timing uses 5,000 paired window-bootstrap replicates; MLSimp cache and far-use analyses use 5,000 and 10,000 trajectory-bootstrap replicates, respectively. Timing order is determined before measurement by the SHA256 hash of dataset, trajectory UID, configuration, budget, repetition, and method.

Final neural settings. Both dataset-specific Transformers use learning rate 10^{-3}, AdamW weight decay 10^{-4}, batch size 128, gradient clipping at 1.0, and ReduceLROnPlateau with factor 0.5, patience 20, and minimum learning rate 10^{-5}; checkpoint selection minimizes the pooled validation error over both budgets and four frozen masks per trajectory. A frozen train/validation learning-rate diagnostic compared $\{3\times10^{-4}, 10^{-3}, 3\times10^{-3}\}$ on GeoLife and $\{10^{-4}, 3\times10^{-4}, 10^{-3}\}$ on Porto. Pointwise-TopK and Anchor Pointer use AdamW with learning rate 10^{-3}, weight decay 10^{-4}, batch size 128, dropout 0.1, and gradient clipping at 1.0. Pointwise-TopK uses 20–80 epochs with early-stopping patience 12; Anchor Pointer uses 20–100 epochs with patience 15. Validation reconstruction error, with earliest-epoch tie breaking, selects every checkpoint. Exact solvers have no learned hyperparameters; their complete-tuple order and acceptance tolerances are specified in the main paper and the numerical contract above.

Execution environment. CPU experiments use 64-bit Windows 11 Pro, Python 3.11.14, an Intel Core i7-13700K (16 cores, 24 logical processors), and 64 GB RAM. Neural training and inference use an NVIDIA GeForce RTX 3060 Ti with 8 GB VRAM, driver 591.86, PyTorch 2.1.1 with CUDA 11.8, and NumPy 1.23.5. Mixed precision is disabled.

I Published Learned-Upstream Length-1000 Transfer

We freeze the official MLSimp GeoLife range/data output at nominal database rates 1% and 2%. A rate ρ draws $\lfloor \rho N n \rfloor$ locations without replacement from the flattened $N \times n$ score matrix and then forces both endpoints in every row. For $N = 2{,}000, n = 1{,}000$, the resulting endpoint-inclusive rates are 1.1983%/2.1963%; realized k has median [P5,P95] $7[4, 31]/14[8, 58]$ and range $[2, 282]/[3, 513]$. Each row keeps this k throughout refinement. This is a published-upstream compatibility and length-1000 stress test, not an objective-matched MLSimp comparison: the released selector is query-trained, whereas refinement uses the separately declared sample-index-synchronous linear reconstruction objective. Native range-query F1 is not evaluated, and we make no claim about its change.

All 4,000 method–budget paths match localized exact exhaustive search through the terminal scan, including complete anchor tuples and objectives. With R_t accepted moves and one terminal scan, cumulative candidate reduction is $1 - C_{\mathrm{SD}}/D$, where $D = \sum_t (R_t + 1)(k_t - 2)(1000 - k_t)$ and C_{SD} sums the unique scored candidate pairs over the same states. It is therefore a work-weighted aggregate, not the reduction for a typical trajectory; it equals 95.32%/96.37%. Across nonempty per-trajectory paths, candidate reduction has P5/median/P95 30.29/69.21/93.41% and 71.76/84.22/96.39%, respectively. The eight 1% rows with $k = 2$ have empty neighborhoods and are retained as valid no-ops. Accepted replacements have mean/median/P95/maximum 16.795/10/58/230 and 30.873/22/92/291, respectively. At these low median budgets, interval construction remains substantial, so the measured median complete-path speedups are 1.032/1.173. Independent replay reconstructs the official masks bit-exactly and reports zero path or objective mismatches.

The complete paths also expose whether the far branch is materially used. At 1%/2%, 85.35% [83.80,86.85]/99.20% [98.80,99.55] of trajectories contain at least one accepted far winner. Far moves comprise 30.87% [29.44,32.18]/33.56% [32.11,34.95] of accepted moves but contribute 65.40% [61.88,68.72]/75.05% [72.14,77.71] of cumulative exact step gain (trajectory bootstrap, 10,000 replicates). A far winner proves that a near-only scan cannot reproduce the exact complete-tuple-tie-broken move at that frozen state. These are descriptive path-use statistics, not candidate-count-normalized enrichment; this trace audit does not estimate the counterfactual terminal solution of a near-only rerun.

Near-only terminal confirmation. A locked confirmation crosses all 12 trajectories not exposed to the pilot with both datasets and objectives, $n \in \{56, 72, 96\}$, and $k/n \in \{1/4, 1/2\}$, giving 144 balanced primary states. Exact near-only best improvement runs through its terminal scan; one complete Segment-Delta scan then tests for a residual move. Eighty-six terminals remain strictly improvable (59.7%), with at least one defect on every trajectory and at every marginal dataset, objective, length, and ratio level. Among positive residuals, the best remaining move has median 0.806 m and median relative magnitude 4.70%. Near-only captures 90.6% of summed same-initial complete-path gain. Under the matched one-warmup/five-repeat protocol, median near-only and complete path times are 1.62 and 4.29 ms, respectively, including raw lazy interval construction.

The same residual scan is positive in 113/192 states in the descriptive full census; the exact 16 pilot states separately replay their original 11/16 result. Independent validation recomputes all near-only paths and complete residual scans with exact objective ordering and complete-tuple tie breaking. In the primary 144 states, near-only terminals are better/equal/worse than the same-initial complete terminals in 9/48/87 cases. The same-initial gain ratio is path-dependent because restricted and complete best-improvement paths may diverge; the residual scan, not their terminal comparison, establishes that near-only termination can remain one-replacement defective.

A. Reconstruction quality and realized budgets

Nominal (%)	Actual (%)	k: median [P5,P95]	Initial (m)	Refined (m)	Improvable (%)
1	1.1983	7 [4,31]	488.819	135.993	99.6
2	2.1963	14 [8,58]	255.771	45.541	100.0

B. Complete-refinement search and timing

Nominal (%)	Candidate reduction median [P5,P95]	Work-weighted reduction (%)	Scan-local speedup median [P10,P90]	Exact paths
1	69.21 [30.29,93.41]	95.32	1.032 [0.996,1.087]	2,000/2,000
2	84.22 [71.76,96.39]	96.37	1.173 [1.085,1.300]	2,000/2,000

Table 17: Published-MLSimp reconstruction transfer on all 2,000 official GeoLife trajectories of length 1,000. Actual is the database-level retained fraction after forcing row endpoints; each realized row budget stays fixed. Improvable is the fraction with a positive complete-refinement gain. The candidate-reduction median and percentiles are over nonempty per-trajectory complete paths; Work-wtd. is $1 - \sum_{t,r} C_{\mathrm{SD,tr}} / \sum_{t,r} M_{tr}$ over all trajectory–round scans. The separately declared objective is sample-index-synchronous linear reconstruction, not MLSimp's native range-query F1. The speedup in Panel B uses scan-local caching and summarizes 640 paired paths per budget; the main paper separately reports the matched rolling-cache regime. Exact-path agreement includes the terminal scan.

I.1 Rolling Cross-Round Interval Reuse

The formal cache matrix freezes 128 trajectories at each nominal rate and crosses the two exact solvers with two symmetric policies. Scan-local caching covers the base objective and candidate solver within one complete scan and is then cleared. Rolling two-generation caching adds only the immediately preceding complete-scan generation. Each treatment has one warmup and five measured repetitions; treatment order is balanced separately in the warmup and measured phases, and the two rates run serially. All treatments reproduce the frozen complete path, terminal tuple, accepted-round count, and objective. Within each solver, source evaluations saved by rolling caching equal its previous-generation hits exactly.

The preregistered validator also contained an unregistered cross-solver check requiring localized exhaustive and Segment-Delta to issue identical intermediate interval-request streams. It stopped on 270 field assertions across 36 policy blocks from three realized-$k = 3$ trajectories, although the registered path and objective contracts held. We preserve the pinned failure record and a locked adjudication: the adjudicated validator waives only messages with the exact `same-policy stream mismatch:` prefix, then reruns every registered hash, design, path, cache-accounting, aggregation, and bootstrap check. Neither timing rows nor numerical summaries were changed.

Absolute timing uses the same two-stage aggregation as the table: a median over five measured repeats within each trajectory, followed by the median over 128 trajectories; warmup is excluded. At nominal 1%/2%, Segment-Delta candidate-solver/total time falls from 0.638/0.639 s and 1.171/1.177 s under scan-local caching to 0.229/0.230 s and 0.320/0.322 s under rolling reuse. Corresponding localized-exhaustive totals are 0.665/1.348 s and 0.264/0.500 s, respectively. Candidate-solver time includes interval construction, cache operations, and candidate comparison; the instrumentation does not support a post-hoc construction-versus-pure-search split.

Nominal rate (%)	Source evals. saved (%)	SD cache speedup	EX cache speedup	EX/SD time rolling cache	Peak entries median [P5,P95]
1	79.31 [75.15,83.07]	2.809 [2.627,2.880]	2.526	1.103 [1.089,1.151]	7,452 [6,580,7,840]
2	88.13 [86.45,89.80]	3.621 [3.540,3.705]	2.661	1.542 [1.496,1.616]	7,759 [7,434,7,888]

Table 18: Formal rolling two-generation cache timing on the frozen 128-trajectory MLSimp cohort. Bracketed trajectory-bootstrap 95% intervals are reported for source reduction, the Segment-Delta cache-policy effect, and the matched-rolling solver comparison; the localized-exact cache-policy effect is a point estimate. Peak-entry brackets are percentiles. Source reduction is work weighted and, within each exact solver, equals source evaluations saved by previous-generation hits. Both solvers use the same cache policy and reproduce the frozen complete path, terminal tuple, round count, and objective. SD denotes Segment-Delta and EX denotes localized exhaustive. Cache speedup is scan-local time divided by rolling-cache time. Timing covers complete refinement only.

J EXACT COMPILER PSEUDOCODE

Algorithm 2 One exact Segment-Delta step

Build/cache the interval costs requested by the current state.
Partition discarded points by their current anchor gap in one ordered two-pointer scan.
if the objective is additive **then**
    Compile deletion deltas and each gap's best insertion delta.
    Score one allowed far representative per deletion and all near pairs.
else
    Compile the four largest records over all gaps and four smallest split records over nonempty gaps.
    Evaluate repeated-floor and unique-critical far classes, scan every near pair, and take their global minimum as $\tau^\star$.
    Recover threshold-eligible far witnesses and compare them with the best near witness.
end if
Return the lowest-objective candidate, breaking ties by its complete resulting tuple.

K ARTIFACT SCOPE AND AI-USE STATEMENT

The anonymous archive supplies runnable exact compilers, independent exhaustive references, tests, a synthetic example and aggregate evidence with hash-based replay. Raw trajectories, neural checkpoints and the full training pipeline are excluded; the archive supports partial experimental reproduction.

Generative AI assisted manuscript/protocol review, result interpretation and source checking, terminal-residual audit design and implementation, reporting code, illustrations and document/language editing. Checks included frozen-source comparisons, differential tests, reaggregation and rendered document inspection. The authors remain responsible for the submission.