2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

HINT-3D: Human-in-the-Loop Interactive Test-Time Adaptation for 3D Segmentation

Oday Jamaleddine, Daniel Asmar, and Imad H. Elhajj

***Abstract*—We present HINT-3D, a human-in-the-loop test-time adaptation framework for 3D semantic segmentation. A few corrective clicks are converted into region masks by a promptable 3D interface (PointSAM). These masks supervise stability-aware updates to a pretrained backbone at inference. We persist the updates so later scenes start from improved weights, enabling cumulative learning. The wrapper is backbone-agnostic: it requires only logits, a mask-to-index bridge, plus access to a small trainable parameter set; we instantiate it on KPConv, RandLA-Net, and Point Transformer v1. On S3DIS Area-5, HINT-3D delivers strong effort-accuracy gains within a scene, consistent zero-click improvements across scenes, and reduced Expected Calibration Error (ECE), while maintaining responsiveness with head-only updates and uncertainty-gated training. We report mIoU versus saved masks, cross-scene transfer, ECE, latency, and class-specific corrections on common indoor failure modes.**

I. Introduction

Interactive segmentation has matured in 2D vision through point and box prompts and promptable models [1], [2], with recent variants that propagate masks in video using streaming memory [3]. Bringing the same usability to 3D point clouds is compelling but nontrivial. Point sets are sparse and cluttered, small objects fragment easily (board–wall, table–floor), and models trained offline often suffer deployment shift caused by sensor changes, lighting, and novel layouts. These issues arise in indoor navigation, augmented reality (AR) and virtual reality (VR), digital twins, and mobile robotics. We ask a deployment-critical question: can a pretrained 3D segmenter learn safely from a few human clicks at test time and retain those gains across future scenes drawn from the same closed label set?

We propose **HINT-3D**, a human-in-the-loop test-time adaptation framework for 3D semantic segmentation. A user issues positive and negative clicks on a mis-segmented region, a promptable 3D module (PointSAM) converts the clicks into a coherent region mask [4], and a stability-aware hybrid objective performs a short burst of updates during inference. Adapted weights are then persisted so subsequent scenes begin with improved predictions before any new clicks, realizing cumulative learning across scenes. A representative sequence is shown in Fig. 1, from the raw point cloud on the left, through user clicks and the PointSAM mask in the middle, to the refined segmentation produced by HINT-3D on the right. The full correction loop and data flow are detailed in the Method section (Fig. 2).

O. Jamaleddine, D. Asmar, and I. H. Elhajj are with the Vision and Robotics Lab, Maroun Semaan Faculty of Engineering and Architecture, American University of Beirut, 1107 2020, Riad El Solh, Beirut, Lebanon; email: oj07@aub.edu.lb, da20@aub.edu.lb, ie05@aub.edu.lb.

Design at a glance. To keep updates safe and fast, HINT-3D uses uncertainty-gated, head-only adaptation driven by a hybrid correction and stabilization objective, with the end-to-end loop detailed in the Method section and illustrated qualitatively in Fig. 1.

Scope. The interface is generic in that it requires only per-location logits and a mask-to-index bridge, but in this work we evaluate only on the indoor S3DIS benchmark, which stresses fine boundaries and small objects.

Contributions

- A wrapper that turns multi-click PointSAM masks into region-level supervision and performs stability-aware updates during inference, with persistence across scenes.
- A click-only protocol with uncertainty gating that avoids learning from unreliable regions, evaluated on indoor S3DIS.
- Model-agnostic validation on KPConv, RandLA-Net, and Point Transformer v1 under a single code path.
- An evaluation recipe for interactive 3D test-time adaptation on S3DIS reporting effort–accuracy, cumulative zero-click gains, calibration, latency, and qualitative fixes.

II. Related Work

Point-cloud semantic segmentation spans point-based models like KPConv [5], efficiency oriented designs such as RandLA-Net [6], attention on point neighborhoods via Point Transformer [7], and sparse voxel CNNs [8], all achieve strong offline accuracy yet degrade under deployment shift.

Interactive segmentation is mature in 2D, where clicks/boxes refine masks in real time [2] and foundation-style prompting broadened scope [1]. In 3D, recent outdoor works show click-driven segmentation [9], [10], and promptable 3D modules such as PointSAM map sparse clicks to region masks [4]. Kontogianni et al. [11] demonstrated interactive instance segmentation in 3D point clouds by combining sparse user scribbles with graph-cut optimization, while AGILE3D [12] introduced an agile framework for interactive 3D semantic segmentation that leverages hierarchical features and efficient region proposals.

However, most interactive approaches keep weights frozen and confine improvements within a scene. In video, prompt propagation maintains temporal consistency without updating parameters, so gains do not persist [3].

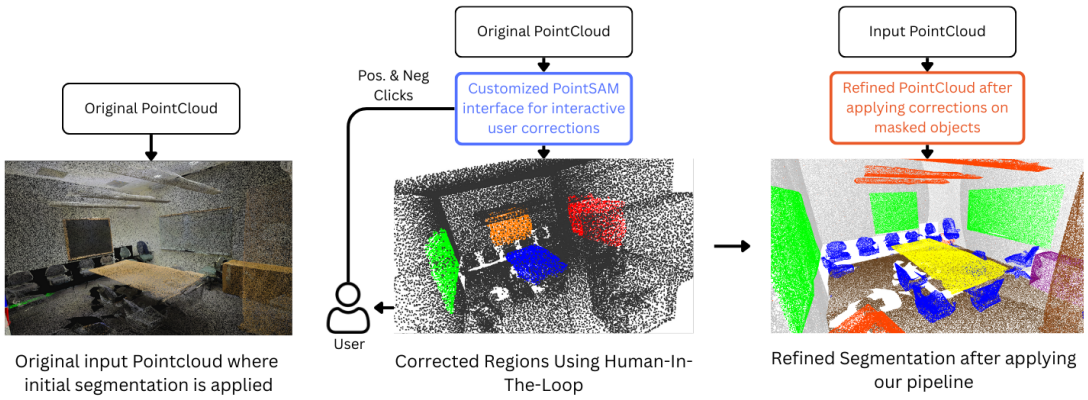

Fig. 1. Interactive correction with HINT-3D. Left: original input point cloud. Middle: user guidance via positive and negative clicks produces a PointSAM region mask. Right: refined segmentation after a stability-aware, head-only test-time update. Example shown with a total of 15 corrective clicks.

Active learning for 3D segmentation strategically selects informative samples to minimize annotation cost. Core-set selection [13] and entropy-based querying [14] have been applied to point clouds [15], [16], yet these methods assume offline retraining on accumulated labels rather than real-time weight updates during deployment.

Test-time adaptation (TTA) adapts without source data, commonly via entropy minimization or statistic updates [17], and continual TTA emphasizes stability/forgetting [18]. Calibration under shift is a parallel concern, with ECE quantifying confidence–accuracy gaps [19].

Positioning. We connect interactivity and TTA: promptable 3D clicks supervise small, stability-aware test-time updates that are uncertainty-gated and *persist* across scenes. Unlike interactive methods that freeze weights [11], [12] or active learning that requires offline retraining [13], [14], HINT-3D performs lightweight head-only adaptation at inference time, enabling cumulative learning across scenes. The wrapper is model-agnostic (KPConv, RandLA-Net, Point Transformer), targeting cumulative gains under indoor shift unaddressed by prior interactive or TTA work in isolation.

III. Method

A. Overview

HINT-3D performs human-in-the-loop test-time adaptation for 3D semantic segmentation. A pretrained backbone produces initial per-point logits and probabilities for an input point cloud. The user supplies a few positive and negative clicks on mis-segmented regions, and a promptable 3D module (PointSAM) converts the clicks into a coherent region mask. A reliability gate abstains when supervision is uncertain, and otherwise a hybrid training objective takes a short optimizer burst on a small trainable subset (classification head or lightweight adapters). The updated weights are persisted so later scenes start from a stronger initialization, which enables cumulative learning. An overview of the loop is shown in Fig. 2, and the step-by-step procedure is summarized in Algorithm 1.

B. Design Objectives and Constraints

HINT-3D targets deployment-time correction with three goals: safety under shift, responsiveness with a small parameter budget, and persistence of improvements. These goals motivate head or adapters-only updates, explicit stabilization on confident non-masked anchors, and a replay buffer that lightly rehearses recent masks. While the loop may appear hand-crafted, each component follows a standard principle (supervised correction, stability via divergence bounds, and uncertainty gating) and its contribution is validated through ablations.

C. Problem Setup and Notation

A scene is $\mathcal{P} = \{x_i\}_{i=1}^{M}$ with $x_i \in \mathbb{R}^3$ and optional attributes. A pretrained backbone f_θ outputs per-location logits and probabilities:

$$z_0 = f_\theta(\mathcal{P}) \in \mathbb{R}^{M\times C}, \quad p_0 = \mathrm{softmax}(z_0), \tag{1}$$

where M is the number of prediction locations and C the number of classes. During interactive correction, we denote the probabilities before an update as p_{θ^-} (pre-update snapshot) and after an update as p_θ (post-update). The class index is denoted by $c \in \{1, \ldots, C\}$.

D. From Multi-Clicks to Region Masks

At round r the user provides positive clicks C^+ and negative clicks C^- on a mis-segmented *region* (the intended object or part to be corrected). A promptable 3D segmenter (PointSAM) maps the clicks to a *proposal* $\widehat{M} \subseteq \{1, \ldots, M\}$, which is PointSAM's initial mask hypothesis based on the click locations and local geometry. After user preview and optional refinement, the finalized *mask* $M^\star$ is approved—this mask defines the spatial extent of the supervision signal for the subsequent update. If K regions are saved in a round, $\Omega = \bigcup_{k=1}^{K} M_k^\star$. Optimization is triggered once per saved mask.

Positive clicks indicate points that should be included in the target region, while negative clicks refine the region boundary by indicating points that should be excluded from the mask proposal—they guide PointSAM to suppress leakage or

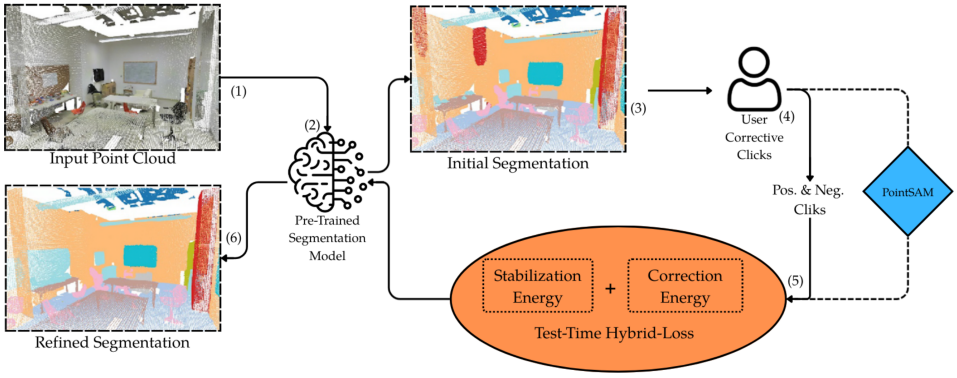

Fig. 2. Interactive correction with HINT-3D. From the input point cloud, a pretrained segmenter yields the initial prediction. User clicks (positive and negative) define an interactive mask via PointSAM. A confidence gate abstains on unreliable supervision, and otherwise a hybrid objective performs two Adam steps on head or adapters. We persist the updated weights, yielding a refined segmentation now and a higher zero-click starting point for subsequent scenes.

over-segmentation within the local neighborhood, rather than globally erasing labels.

Label source (click-only). Users never type class names. Each $i \in M^\star$ inherits $y_i = \arg\max_c p_{\theta^-}(i, c)$ from the pre-update snapshot p_{θ^-}.

E. Mask-to-Index Bridge

Prediction domains differ across backbones. Point models may predict on subsampled points, and sparse voxel models predict on active voxels. We transfer $M^\star$ with nearest-neighbor assignment for point decimation or via the union of voxels covering masked points for sparse grids [6]–[8]. This bridge is the only backbone-specific component, while the loop and objective remain unchanged.

F. Confident Non-Masked Anchors

To prevent drift on untouched content, we preserve predictions on confident, non-masked points taken from the pre-update snapshot. We refer to these as *anchors*—points outside all corrected regions where the model is already confident, which serve as stable reference predictions to prevent catastrophic forgetting during the update. We define the anchor index set as

$$\mathcal{G}(\tau) = \{\, i \notin \Omega \mid \max_c \, p_{\theta^-}(i, c) \geq \tau \,\}, \tag{2}$$

where $p_{\theta^-}(i, c)$ is the pre-update class probability for class c at prediction location i, and Ω is the union of indices inside all saved masks. The set $\mathcal{G}(\tau)$ therefore contains non-masked locations whose most confident class exceeds a confidence threshold τ. We set $\tau = 0.7$ by default, selected from an ablation over $\tau \in [0.6, 0.8]$ that balances coverage against reliability (lower τ anchors more points but risks anchoring errors, higher τ is safer but may under-constrain the update; see Table III). Confidence can be computed via maximum softmax probability (optionally after temperature calibration), entropy, or Monte Carlo dropout uncertainty [19], [20].

G. Stability-Aware Hybrid Loss

Let $p_\theta = \mathrm{softmax}(f_\theta(\mathcal{P}))$ denote post-update probabilities. We minimize

$$\mathcal{L} = \lambda_{\text{corr}} \sum_{i \in \Omega} \mathrm{CE}\big(p_\theta(i), y_i\big) + \lambda_{\text{stab}} \sum_{i \in \mathcal{G}(\tau)} d\big(p_\theta(i), p_{\theta^-}(i)\big), \tag{3}$$

where d is the divergence between post- and pre-update distributions (we use KL divergence unless otherwise noted, and also report MSE in ablations). By Pinsker's inequality, $D_{\mathrm{KL}}(p\|q) \geq \frac{1}{2}\|p - q\|_1^2$, so the stabilization term bounds the L_1 drift on anchors and acts as a trust region around p_{θ^-} [21].

In our experiments, we set $\lambda_{\text{corr}} = 1.0$ and $\lambda_{\text{stab}} = 0.4$. These values were selected from preliminary sweeps to balance accuracy gains with calibration and latency, sensitivity analyses are summarized in Table III.

H. Distance-Decay Weighting and Class Balance

Large masks may include uncertain borders. We reweigh the correction term with a radial decay from the mask click centroid (computed as the mean position of all positive clicks):

$$w_i = \exp\Big(-\frac{\|x_i - c_{\text{centroid}}\|_2^2}{2\sigma^2}\Big), \quad \mathrm{CE}(p_\theta(i), y_i) \leftarrow w_i \, \mathrm{CE}(\cdot), \tag{4}$$

with σ set to the 75th percentile of within-mask distances. For rare classes we optionally apply class-balanced weights $\alpha_{y_i} \propto 1/\mathrm{freq}(y_i)$ inside Ω.

I. Uncertainty Gating

Not all corrective masks are reliable—users may click on ambiguous boundaries, reflective surfaces, or regions where PointSAM's proposal leaks beyond the intended target. To avoid learning from such unreliable supervision, we introduce an *uncertainty gate* that abstains from updating when the mask is deemed untrustworthy, requesting an additional clarifying click instead. We abstain (request another click) if either: (i) more than 60% of points in $M^\star$ have $\max_c p_{\theta^-}(i, c) < \tau$, or (ii) mean entropy $H(p_{\theta^-}(i))$ over $M^\star$ exceeds 1.2 bits:

$$H(p) = -\sum_{c=1}^{C} p_c \log p_c. \tag{5}$$

This trades false updates for a small extra click burden and is robust to mild miscalibration [22], [23].

J. Adapters and Parameter-Efficient Updates

We update a small parameter subset ($< 3\%$ of total). For KPConv and RandLA-Net we tune the final classifier MLP, and for Point Transformer v1 we tune the penultimate and final linear layers of the classifier head. Low-rank adapters are compatible but not required, and we include their parameter counts in Table II.

K. Update Scope, Step Count, and Persistence

We use Adam at 10^{-4} for **exactly two steps** per saved mask (not per click). We then persist θ and rehearse a small buffer of the most recent $K{=}10$ masks with mix fraction $\rho{=}0.2$ [24]. Two steps balance stability and speed (see ablations).

L. Complexity and Memory Footprint

Let n_h be the head parameter count and B the points per pass. Each round does one PointSAM proposal, an $\mathcal{O}(BC)$ forward, and an $\mathcal{O}(Bn_h)$ backward; head gradients dominate memory. Since $n_h \ll |\theta_{\text{enc}}|$, backward cost and memory remain small, and latency grows roughly linearly with the number of saved masks.

M. Robustness to Noisy Clicks

Noisy borders or out-of-class clicks can induce drift. HINT-3D mitigates this via abstention when masks are uncertain, distance-decay weighting, stabilization at confident non-masked anchors, and rehearsal that tempers recency bias.

IV. Implementation Details

Preprocessing and augmentation. Scenes are normalized to zero-mean, unit-variance per axis. If the dataset provides per-point RGB colors, we rescale the raw 8-bit values (0–255) to $[0, 1]$, otherwise only geometry (XYZ) is used. We follow the standard S3DIS splits [25] and apply point jittering, random rotations about gravity, and color jitter during backbone training.

Backbones and heads. KPConv (final $1{\times}1$ classifier), RandLA-Net (last classifier MLP), Point Transformer v1 (penultimate and final linear classifier). Encoders are frozen during interaction.

Optimization. We use Adam with a learning rate of 10^{-4} and perform exactly two gradient steps for each saved mask. Updates are gated by confidence: only applied when mask reliability exceeds $\tau = 0.7$ and mean entropy is below 1.2 bits. To reduce drift, a replay buffer of the most recent $K = 10$ masks is maintained, and 20% of them are mixed into each update so that the model rehearses past corrections while adapting to the new one.

Click collection. Our results use human clicks. When simulating clicks in pilot experiments, we sample next-click locations along the largest connected error component and alternate positive and negative clicks until IoU within the region stabilizes.

Algorithm 1: HINT-3D interactive test-time adaptation loop

Input: scene $\mathcal{P}$, pretrained f_θ, threshold τ, weights $\lambda_{\text{corr}}, \lambda_{\text{stab}}$
Output: updated parameters θ

$p_{\theta^-} \leftarrow \mathrm{softmax}(f_\theta(\mathcal{P}))$
User clicks C^+, C^- and approves $M^\star$ via PointSAM
Map $M^\star$ and assign $y_i \leftarrow \arg\max_c p_{\theta^-}(i, c)$ for $i \in M^\star$
Build anchors $\mathcal{G}(\tau)$ and compute optional w_i
If the gate fires, request another click and restart
Compute (3) with w_i and take **two** optimizer steps on head or adapters
Persist θ and optionally update the replay buffer

TABLE I
S3DIS SPLIT AND TYPICAL POINT COUNTS USED IN OUR PIPELINE.

Area	Rooms	Avg points/room	Used for
1,2,3,4,6	200+	1.2M–1.6M	Train
5	68	1.3M	Test

TABLE II
KEY HYPERPARAMETERS AND HEAD PARAMETER COUNTS.

Backbone	Trainable params	Steps/round	λ_{stab}
KPConv	≈1.1M (head)	2	0.4
RandLA-Net	≈0.9M (head)	2	0.4
Point Transformer v1	≈1.3M (head)	2	0.4

Hardware and runtime. All experiments are run on a single 24GB GPU paired with a 16-core CPU, using a single process and no data parallelism. A "round" denotes one saved mask followed by two optimizer steps on the intersecting blocks only. The head-only update keeps the additional gradient memory under 1.0 GB across all backbones, and the total footprint remains within a 24 GB device for the block sizes used in Section V. Because we re-run only blocks that intersect the saved mask, latency scales approximately with the touched region size rather than the full-scene point count.

Dataset statistics. The S3DIS split and typical point counts used in our pipeline are summarized in Table I, and the trainable head sizes together with the update settings used throughout are listed in Table II.

V. Experiments

A. Dataset and Protocol

We evaluate on S3DIS [25] under the standard Area-5 protocol: train on Areas 1,2,3,4,6 and test on Area 5. All results are single-scene semantic segmentation without multi-view voting.

Comparison to active learning and TTA baselines. Direct quantitative comparison to active learning methods [13], [14] is not straightforward because those approaches assume offline retraining on accumulated labels with dataset-level query budgets, whereas HINT-3D performs live, per-scene corrections without access to the training set. Similarly, standard TTA

baselines [17], [26] rely on unsupervised entropy minimization or batch-norm statistics updates, while our framework uses supervised region-level corrections from human clicks. We position HINT-3D as a complementary approach that bridges interactivity and test-time adaptation rather than a replacement for either paradigm. Developing unified benchmarks that fairly compare interactive correction effort against active learning annotation cost and unsupervised TTA performance remains an open challenge for the community.

B. Backbones and Setup

We instantiate HINT-3D on three backbones: KPConv [5] (point convolution), RandLA-Net [6] (random sampling with local aggregation), and Point Transformer v1 [7] (attention on point neighborhoods). Scenes are mean-centered. When RGB is available (e.g., S3DIS), raw 0–255 values are scaled to $[0, 1]$ (divide by 255) to match the scale of normalized coordinates and improve optimization, if RGB is absent, only geometry is used. Inference runs on overlapping blocks of 1.0–1.5×10^5 points with 50% overlap, and during interaction we re-run only blocks intersecting the saved mask.

Encoders are frozen. The trainable set is limited to the classifier head (under 3% of parameters): KPConv updates the final 1×1 linear, RandLA-Net the last classifier MLP, and Point Transformer v1 the penultimate and final linear layers (see Table II). The mask-to-index bridge uses nearest-neighbor assignment for decimated point sets (KPConv, RandLA-Net), and maps masks to active tokens with back-projection by nearest neighbor for Point Transformer.

Test-time optimization uses Adam at 1×10^{-4} for exactly two steps per saved mask, with KL stabilization on confident non-masked anchors (threshold τ=0.7) and abstention when mean entropy over the mask exceeds 1.2 bits. A small replay buffer (most recent K=10 masks, mix ρ=0.2) mitigates recency bias. Using three diverse backbones shows the wrapper transfers without backbone-specific redesign beyond the bridge and exposed head.

VI. Ablation Study

We ablate update scope, stabilization (weight and divergence), gating thresholds, optimizer step count, and light rehearsal (Table III). The goal is to balance accuracy, calibration, and latency for deployment.

Update scope. Moving from head-only updates to head+last encoder block yields a small mIoU lift ($73.6 \rightarrow 74.2$) but worsens calibration (ECE $0.13 \rightarrow 0.15$) and adds latency ($540\,\text{ms} \rightarrow 610\,\text{ms}$). Given this speed–stability trade-off, head-only is the default.

Stabilization term. Removing stabilization (λ_{stab}=0) slightly reduces mIoU ($73.6 \rightarrow 73.1$) and substantially harms calibration (ECE $0.13 \rightarrow 0.20$), confirming the anchor's role in preventing drift. KL and MSE behave similarly at the same weight; we keep KL at λ_{stab}=0.4 for consistency.

Uncertainty gate. A looser gate (τ=0.6, entropy 1.0) gives a modest mIoU bump (73.8) with ECE 0.14 and slightly lower latency, while a stricter gate (τ=0.8, entropy 1.5) trades accuracy (72.9) for a minor calibration gain (ECE 0.13). We adopt mid-strength thresholds (τ=0.7, entropy 1.2) to avoid learning from unreliable masks without over-asking for clicks.

Step count. One step underfits (72.8, ECE 0.14); three steps add compute (590 ms) with negligible benefit (73.7, ECE 0.13). Two steps offer the best balance (73.6, ECE 0.13, 540 ms).

Replay. Mixing a small fraction of recent masks (ρ=0.2) improves calibration (ECE $0.15 \rightarrow 0.13$) at no observed latency penalty, reducing recency bias during online updates.

Takeaways. For interactive use we recommend: head-only updates, KL stabilization at λ_{stab}=0.4, mid-strength gate (τ=0.7, entropy $\approx$ 1.2 bits), two optimizer steps per saved mask, and a tiny replay buffer (K=10, ρ=0.2). This configuration maximizes accuracy gains from early interactions while keeping calibration and latency within practical bounds.

TABLE III
Ablations on S3DIS Area-5 (KPConv).

Setting	mIoU (%)	ECE	Total ms
Head-only (KL, $\lambda_{\text{stab}} = 0.4$)	73.6	0.13	540
Head+last block (same)	74.2	0.15	610
$\lambda_{\text{stab}} = 0.0$ (no stabilization)	73.1	0.20	520
MSE (0.4)	73.3	0.14	540
Gate: $\tau = 0.6$, entropy $= 1.0$	73.8	0.14	520
Gate: $\tau = 0.8$, entropy $= 1.5$	72.9	0.13	560
Steps: 1	72.8	0.14	480
Steps: **2** (default)	73.6	0.13	540
Steps: 3	73.7	0.13	590
Replay $\rho = 0.0$	73.2	0.15	540
Replay $\rho = 0.2$ (default)	73.6	0.13	540

TABLE IV
Baseline (0-click) mIoU on S3DIS Area 5.

Backbone	KPConv	RandLA-Net	Point Transformer v1
mIoU (%)	67.1	63.0	70.4

VII. Discussion

Zero-click baselines for each backbone are reported in Table IV; all interaction results are measured relative to these starting points.

Effort vs. accuracy. The largest gains arrive with the first one to three saved masks, after which returns saturate (Figure 4). This is consistent with the qualitative progression in Figure 5: early masks typically resolve the dominant error regions (for example, board-wall or table-floor), while later masks target residual, smaller fragments. Per-class effects concentrate on classes where fragmentation is common (Table V).

Cumulative gains. Persisting updates raises the zero-click starting point on later scenes without incurring catastrophic forgetting (Table VI and Figure 3). The effect is strongest after the first adapted room and tapers as more rooms are incorporated, which indicates that the head-only updates capture reusable corrections while remaining stable.

TABLE V
PER-CLASS IOU (%) ON DIFFICULT CLASSES IN AREA 5. R0: 0 MASKS, R3: 3 MASKS.

Backbone	Class	R0	R3	Δ
KPConv	board	42.0	58.5	+16.5
KPConv	table	55.0	69.0	+14.0
KPConv	chair	60.5	72.0	+11.5
RandLA-Net	board	38.5	52.0	+13.5
RandLA-Net	table	50.0	63.0	+13.0
RandLA-Net	chair	56.0	66.5	+10.5
Point Transf. v1	board	45.0	59.5	+14.5
Point Transf. v1	table	58.0	70.0	+12.0
Point Transf. v1	chair	62.0	72.5	+10.5

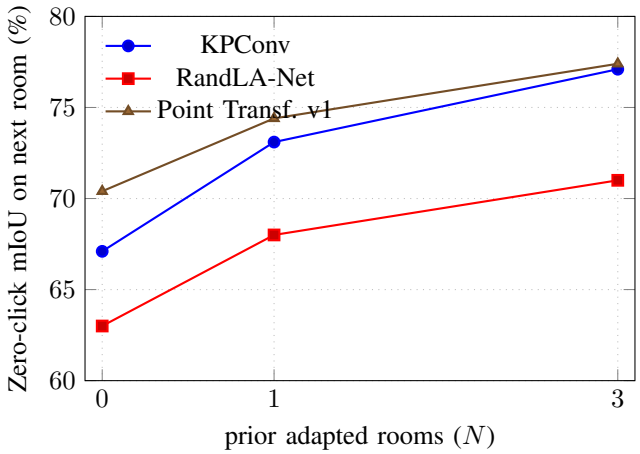

Fig. 3. Cumulative learning: zero-click mIoU on room $N+1$ after adapting on N rooms.

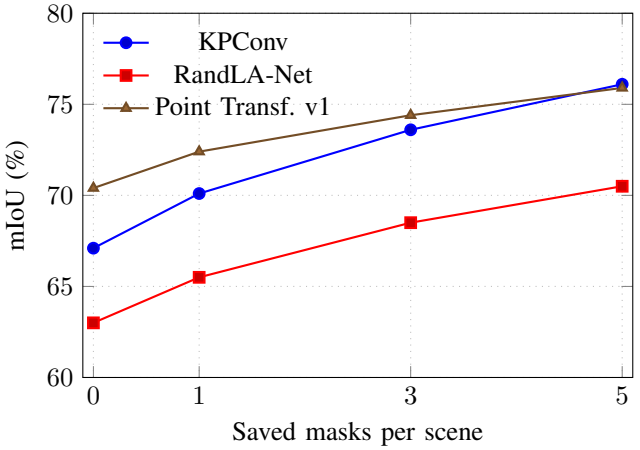

Fig. 4. Effort–accuracy: mIoU vs. number of saved masks per scene. Early rounds yield the largest gains then saturate.

TABLE VI
ZERO-CLICK MIOU (%) ON A NEW AREA 5 ROOM AFTER ADAPTING ON N PRIOR ROOMS.

Backbone	N=0	N=1	N=3
KPConv	67.1	73.1	77.1
RandLA-Net	63.0	68.0	71.0
Point Transformer v1	70.4	74.4	77.4

TABLE VII
CALIBRATION (ECE) AND RUNTIME ON S3DIS AREA-5.

ECE (lower is better)					**Latency per round (ms)**				
Backbone	0c	1m	3m	5m	Backbone	PS	Fwd	Bwd+Opt	Tot
KPConv	0.20	0.16	0.13	0.12	KP	250	180	110	540
RandLA-Net	0.22	0.18	0.15	0.14	RL	220	130	80	430
PT v1	0.18	0.15	0.13	0.12	PT	280	240	140	660

Calibration and safety. ECE decreases with interaction across all backbones (Table VII), which suggests that the stabilization term on confident non-masked anchors plus uncertainty gating avoids reinforcing wrong labels. In practice, abstentions introduce a modest extra click burden but prevent harmful updates when masks are noisy or ambiguous.

Decoupling uncertainty gating and anchors. The uncertainty gate and confident non-masked anchors serve distinct purposes: the gate decides whether to apply an update at all (filtering unreliable supervision), while anchors constrain drift on untouched content during accepted updates. In our ablations (Table III), removing stabilization ($\lambda_{\text{stab}} = 0$) substantially harms calibration (ECE 0.13 $\rightarrow$ 0.20) even when gating remains active, confirming that anchors provide complementary protection against drift. A full factorial ablation isolating gating thresholds from anchor selection independently would strengthen claims about their individual contributions and remains valuable future work.

Update scope, latency, and stability. Head-only adaptation offers the best speed and stability trade-off because it localizes capacity to the decision layer, bounds backprop cost, and limits drift. Ablations (Table III) show that updating deeper blocks can nudge mIoU slightly higher but at the expense of latency and calibration. Two optimizer steps per saved mask are a robust default, while one step underfits and three steps offer marginal benefit.

Sensitivity to hyperparameters. Performance is stable for $\tau \in [0.6, 0.8]$ and for either KL or MSE in the stabilization term (Table III). Removing stabilization raises short-term accuracy in edited regions but degrades global calibration (higher ECE), which supports the need for an anchor term when adapting online.

Failure modes and mitigations. Hard cases include reflective or transparent surfaces, thin structures, and heavy occlusion where PointSAM proposals may leak. The gate prevents learning from such masks, and distance-decay weighting further down-weights uncertain borders. A lightweight spatial prior such as a short-range CRF could reduce leakage and is a promising extension.

VIII. LIMITATIONS AND FUTURE WORK

This study focuses on closed-set, indoor semantic segmentation and demonstrates human-in-the-loop test-time adaptation using lightweight head-level updates. While the results are encouraging, several limitations remain that point to natural directions for extension.

Dataset scope and generalization. All experiments were conducted on S3DIS under the Area-5 protocol. Although the wrapper is model-agnostic, our claims do not extend

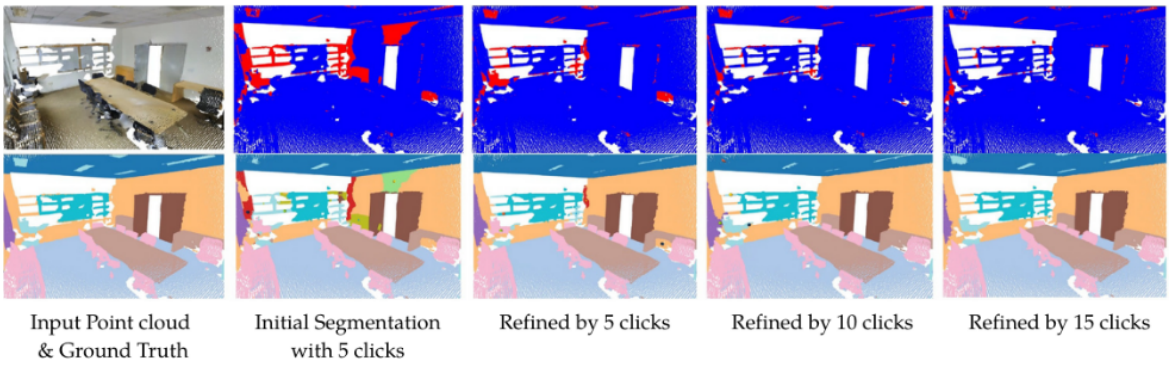

Fig. 5. Interactive segmentation process. Top: input point cloud and error maps. Bottom: ground truth, initial segmentation, and refined results as corrective clicks are added (5, 10, 15). Colored dots mark corrective clicks (total 15).

to outdoor LiDAR or cross-dataset transfer. Evaluating on benchmarks such as SemanticKITTI and nuScenes-lidarseg which differ in sampling density, motion artifacts, and class priors is an important next step toward establishing robustness across sensing conditions [27], [28].

Closed label set. The protocol assumes a fixed taxonomy, with each region inheriting the dominant pre-update class. This reduces user effort but risks propagating errors when the base prediction is confidently incorrect. A promising extension is to incorporate optional class disambiguation under low consensus, or to adopt open-vocabulary heads that align point features with text prompts, following recent advances in CLIP-based segmentation [29]–[34].

Reliance on PointSAM and thresholds. The framework depends on PointSAM for region proposals and on fixed uncertainty thresholds for abstention. These mechanisms can fail on thin structures or reflective surfaces, where leakage is common. More principled uncertainty estimates (ensembles, Monte Carlo dropout), energy-based gating, or spatial priors such as lightweight CRFs could provide stronger safeguards [19], [20], [22], [35]. Alternative mask generation strategies such as geometric clustering (DBSCAN, region growing) or learned grouping could reduce reliance on promptable models, though we selected PointSAM for its demonstrated click efficiency and boundary quality on cluttered indoor scenes. Systematic comparison of clustering-based alternatives against promptable segmentation under matched click budgets would clarify trade-offs and is a natural direction for future work.

Update scope and capacity. Our adaptation is restricted to classifier heads or shallow adapters. This ensures responsiveness but limits the ability to correct deeper encoder biases. Parameter-efficient strategies that reach into later encoder stages for example low-rank adapters or distilled auxiliary blocks may increase capacity while retaining stability [21], [36], [37].

Safety, calibration, and drift. Improvements in calibration on S3DIS indicate safer updates, but we make no claim of robustness under severe distribution shift, class imbalance, or adversarial clicks. Incorporating calibration-aware objectives (e.g., Dirichlet calibration, temperature scaling), conservative regularization (trust-region or Fisher penalties), and stronger rehearsal beyond a small buffer would help mitigate these risks [19], [38]–[41].

Human effort and UI assumptions. We measured clicks and latency but did not conduct a user study. The fixed order of corrections and stopping criteria may not reflect real-world practice. Controlled studies of user effort, responsiveness, and error taxonomy would provide more reliable guidance for interface design, including multi-region batching or active-learning based region prioritization [13], [14].

Cumulative learning protocol. Our evaluation followed a single fixed room order. Order effects and long-horizon stability remain open questions. Randomized protocols, replay scheduling, or periodic consolidation sweeps could strengthen claims of continual adaptation [17], [18], [24], [26], [42].
Temporal and multi-view context. The current framework processes static scenes. Incorporating temporal windows or multi-view fusion has the potential to stabilize both PointSAM proposals and adaptation targets without incurring large runtime costs [8], [43].

Computational envelope. Experiments were conducted on a single GPU. Deployment on embedded or resource-constrained platforms will require careful study of memory and latency. Mixed-precision training, 8-bit optimizers, and compact adapter designs are viable strategies, but their trade-offs merit systematic evaluation [37], [44], [45].

Interaction policy. Finally, the correction loop is currently hand-crafted: region selection, gating, and two optimization steps. A learned controller that adapts these decisions with reinforcement or active learning, under safety constraints, could make the framework more efficient and adaptive [14], [46].

IX. Conclusion

We presented HINT-3D, a human-in-the-loop framework that turns PointSAM region masks into stability-aware test-time updates and persists those updates across scenes. The design is intentionally simple, relying on click-only supervision, uncertainty-gated training, and head- or adapter-level updates, so that corrections are fast, localized, and safe to apply during deployment.

On S3DIS Area-5, HINT-3D delivers three consistent outcomes across distinct backbones (KPConv, RandLA-Net, Point

Transformer v1). First, within a scene, a small number of saved masks yields sizable mIoU gains, with the largest improvements arriving in the first interaction rounds and then saturating. Second, because updates are persisted, later scenes start from a stronger zero-click baseline, demonstrating cumulative learning without catastrophic drift. Third, calibration improves alongside accuracy, as Expected Calibration Error decreases with more interactions, showing that the hybrid objective and confident non-masked anchors reduce overconfident mistakes.

These benefits come with practical latency and a bounded parameter budget. Restricting adaptation to the classifier head or lightweight adapters keeps wall-clock cost low while preserving stability, and abstention under high uncertainty prevents harmful updates when supervision is unreliable. The wrapper's model agnostic interface logits, a mask-to-index bridge, and access to a small trainable set enabled the same loop to run across all three backbones without bespoke redesign.

Overall, HINT-3D offers a viable approach for deploying 3D segmentation under distribution shift: minimal operator effort, immediate within-scene correction, and improvements that carry forward across future scenes. In ongoing work, we aim to extend the protocol to outdoor LiDAR and open-vocabulary settings, and to explore learned controllers that allocate clicks and adapt step sizes under explicit safety constraints.

ACKNOWLEDGMENTS

This work was supported by the DIDYMOS-XR Horizon Europe project (grant number 101092875–DIDYMOS-XR, www.didymos-xr.eu), and the Research Board at the American University of Beirut.

REFERENCES

[1] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023.

[2] K. Sofiiuk, I. Petrov, and A. Konushin, "F-brs: Rethinking backpropagation refinement for interactive segmentation," in *CVPR*, 2020, pp. 8623–8632.

[3] D. Rao, X. Zhang, P. Li, Z. Dong *et al.*, "Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[4] Y. Zhou, J. Gu, T. Y. Chiang, F. Xiang, and H. Su, "Point-SAM: Promptable 3d segmentation model for point clouds," *arXiv preprint arXiv:2406.17741*, 2024.

[5] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *ICCV*, 2019, pp. 6411–6420.

[6] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *CVPR*, 2020, pp. 11 105–11 114.

[7] H. Zhao, J. Jia, V. Koltun *et al.*, "Point transformer," in *ICCV*, 2021, pp. 16 259–16 268.

[8] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *CVPR*, 2019, pp. 3075–3084.

[9] H. Wu *et al.*, "Clickformer: Interactively annotating 3d objects with clicks," in *ICCV*, 2023, pp. 12 484–12 495.

[10] L. Xu *et al.*, "Crsnet: Category-level residual structure network for point cloud segmentation," in *CVPR*, 2023, pp. 12 814–12 824.

[11] T. Kontogianni, M. Radovic, C. Aliaga, M. Pollefeys, and F. Yu, "Interactive object segmentation in 3d point clouds," in *ICRA*, 2023, pp. 8834–8841.

[12] Y. Yue, Y. Chen, X. Zhou, and V. A. Prisacariu, "Agile3d: Attention guided interactive multi-object 3d segmentation," in *ICLR*, 2024.

[13] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *ICLR*, 2018.

[14] B. Settles, *Active Learning Literature Survey*. University of Wisconsin-Madison, 2009.

[15] H. Shi, J. Lin, C. Choy, A. Dai, and Y. Furukawa, "Weakly supervised segmentation on outdoor 4d point clouds with temporal matching and spatial graph propagation," in *CVPR Workshop on Autonomous Driving*, 2022.

[16] J. Zhang and B. Kang, "Active learning for deep object detection via probabilistic modeling," in *ICCV*, 2021, pp. 10 264–10 273.

[17] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell, "Tent: Fully test-time adaptation by entropy minimization," in *ICLR*, 2021.

[18] S. Niu, H. Chen *et al.*, "A comprehensive survey on test-time adaptation," *arXiv preprint arXiv:2301.11037*, 2023.

[19] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *ICML*, 2017, pp. 1321–1330.

[20] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *ICML*, 2016, pp. 1050–1059.

[21] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[22] W. Liu, X. Wang, J. D. Owens, and Y. Li, "Energy-based out-of-distribution detection," in *NeurIPS*, 2020.

[23] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016.

[24] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *CVPR*, 2017, pp. 2001–2010.

[25] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3d semantic parsing of large-scale indoor spaces," in *CVPR*, 2016.

[26] S. Niu *et al.*, "Efficient test-time adaptation by entropy minimization and online self training," in *NeurIPS*, 2022.

[27] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *ICCV*, 2019.

[28] H. Caesar, V. Bankiti *et al.*, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020.

[29] A. Radford, J. W. Kim, C. Hallacy *et al.*, "Learning transferable visual models from natural language supervision," in *ICML*, 2021.

[30] G. Ghiasi *et al.*, "Scaling open-vocabulary image segmentation with image-level labels," in *ECCV*, 2022.

[31] K. Zhou, J. Yang, C. C. Loy, and Z. Liu, "Extract free dense labels from clip," in *ECCV*, 2022.

[32] K. Liang *et al.*, "Lseg: Language-driven semantic segmentation," in *NeurIPS*, 2022.

[33] S. Peng *et al.*, "Openscene: 3d scene understanding with open vocabularies," in *CVPR*, 2023.

[34] X. Chen *et al.*, "Openmask3d: Open-vocabulary 3d instance segmentation," in *ICCV*, 2023.

[35] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *NeurIPS*, 2011.

[36] N. Houlsby *et al.*, "Parameter-efficient transfer learning for nlp," in *ICML*, 2019.

[37] E. J. Hu *et al.*, "Lora: Low-rank adaptation of large language models," 2022.

[38] M. Kull, T. Silva Filho, and P. Flach, "Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration," in *NeurIPS*, 2017.

[39] J. Kirkpatrick *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[40] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *NeurIPS*, 2017.

[41] A. Chaudhry *et al.*, "Tiny episodic memories in continual learning," *arXiv preprint arXiv:1902.10486*, 2019.

[42] J. Wang *et al.*, "Continual test-time domain adaptation," in *CVPR*, 2022.

[43] G. Qian *et al.*, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," in *NeurIPS*, 2022.

[44] P. Micikevicius *et al.*, "Mixed precision training," 2018, iCLR 2018.

[45] T. Dettmers, "8-bit optimizers via block-wise quantization," 2022.

[46] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.