StitchEdit: Stitching Depth Priors into Image Editors via Per-Layer Gradient Probing

Qin Guo **Dan Xu***
The Hong Kong University of Science and Technology
qguoam@connect.ust.hk danxu@cse.ust.hk

Abstract

Text-guided image editing with diffusion models struggles with spatial edits (*e.g.* manipulating objects, changing viewpoints, manipulating occlusions) that depend on the 3D structure understanding of a scene. Marigold and follow-up work demonstrate that a pretrained diffusion backbone already harbors strong geometric priors that can be activated for depth estimation, yet how to redirect these priors toward editing remains unresolved. Through per-layer gradient probing, we discover that the editing and depth objectives are compatible in early transformer layers but actively conflict in late layers, explaining why naive depth-conditioned editing either ignores geometry or degrades visual quality. We present STITCHEDIT, which first activates the backbone's geometric capability via a lightweight depth-denoising adapter, then selectively fine-tunes this adapter for editing using a probe-guided per-layer learning rate schedule that preserves depth knowledge where it helps and prunes it where it conflicts. On the proposed *StitchEditBench*, a benchmark of 150 geometry-dependent edits spanning spatial transformation, occlusion manipulation, and viewpoint change, STITCHEDIT consistently surpasses both instruction-based and depth-conditioned baselines on perceptual quality and geometric fidelity.

1 Introduction

Moving an object closer should make it appear larger. Inserting furniture should respect occlusion boundaries. Changing a viewpoint should transform the entire depth field. These spatial edits are second nature to humans yet remain difficult for text-guided editing models [1, 8, 15, 18], which manipulate 2D appearance without explicit access to scene geometry. The 3D knowledge required for such edits is already present inside modern diffusion models in latent form. Marigold [10] and follow-up work [2, 7, 6, 11] show that a single diffusion backbone can be turned into a strong depth estimator with minimal adaptation, indicating that scene geometry is already represented internally as a byproduct of large-scale image generation. We build on this premise and ask not whether to use depth, but *how to transfer* the backbone's own depth capability toward the editing task.

The most natural strategy is to simply train the backbone for editing from text instructions and image-pairs. However, without explicit or implicit geometric guidance, the resulting model falls back on 2D pattern matching and cannot enforce spatial consistency, as shown in the *Nano-Banana* column of Figure 1. Another strategy is to first activate the backbone's depth capability and then continue training for spatial editing. However, this sequential transfer also fails: the editing gradient overwrites the geometric knowledge that the depth adapter encodes, and the resulting model (*Depth-Init* in Figure 1) performs no better than one trained for editing from scratch. The question, then, is *which layers* can be safely fine-tuned for editing without losing the geometric features that make depth-aware editing possible.

*Corresponding author. Project page: https://guoqincode.github.io/StitchEdit/

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

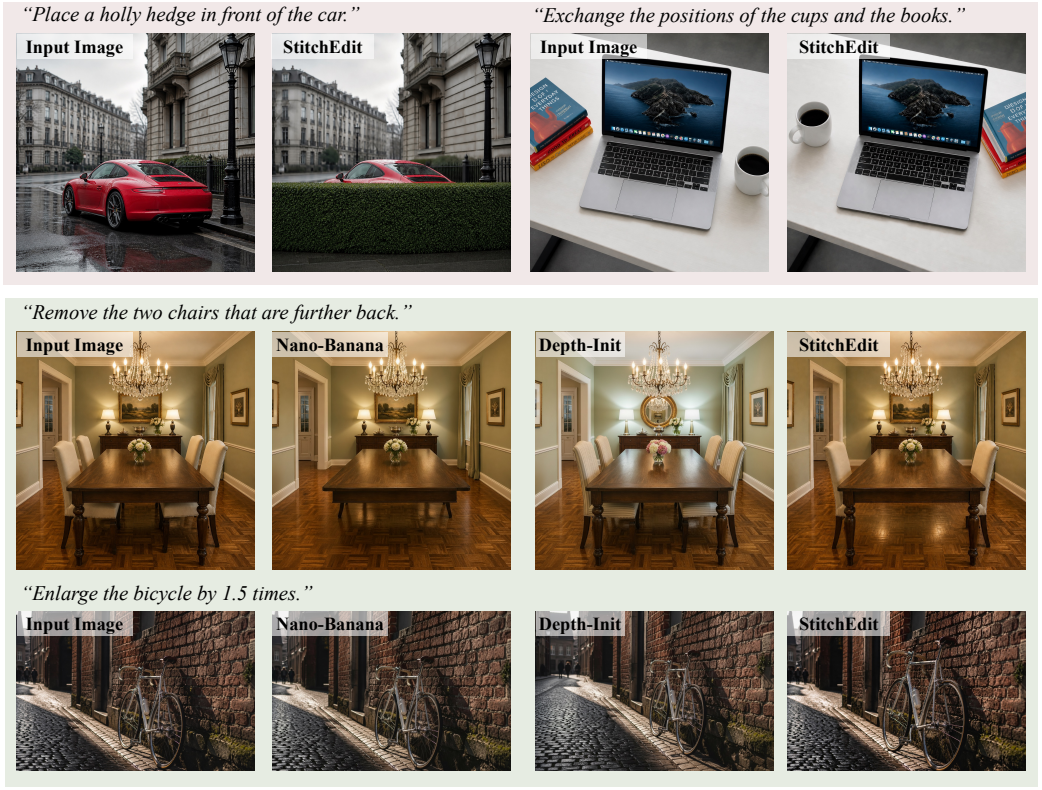


Figure 1: **Top:** Challenging 3D-aware edits generated by STITCHEDIT conditioned on text instructions. The model accurately handles complex object repositions and occlusions while strictly preserving visual quality. **Bottom:** Qualitative comparison against baselines. Nano-Banana [4] manipulates 2D appearance without explicit scene geometry, while uniformly fine-tuning the depth adapter (*Depth-Init*) leads to catastrophic forgetting, causing structural failures.

We answer this with a per-layer gradient probe that measures, at each transformer layer, the cosine alignment between the editing gradient and the depth gradient. This gradient probing profile is strongly positive in early transformer layers, decays through the middle, and turns negative in late layers (Figure 3). The mechanism behind the sign flip is the well-documented division of labor inside diffusion transformers [21, 17, 19]: early layers carry the spatial layout that organizes a scene, where editing and depth make compatible demands, while late layers synthesize the textures and materials that render it, where the depth signal conflicts with the appearance variation editing seeks. The implication for editing is direct: the depth adapter should be fine-tuned only where the probe is positive, and pruned entirely from the layers where it would actively degrade geometric knowledge.

We propose STITCHEDIT, a two-stage framework that turns this analysis into a concrete training strategy. We first activate the geometric capability of the backbone with a lightweight depth-denoising adapter, confirming that the backbone already encodes rich geometric structure. We then selectively fine-tune the resulting adapter for editing using a per-layer learning rate schedule derived from the gradient probing profile: compatible layers train at full rate, transition layers at reduced rate, and conflicting layers are pruned outright. Because no additional gating networks or auxiliary losses are needed, the method is simple to implement and adds no overhead at inference. On the proposed *StitchEditBench*, a benchmark of 150 geometry-dependent edits spanning spatial transformation, occlusion manipulation, and viewpoint change, STITCHEDIT consistently outperforms both instruction-based and depth-initialized baselines while preserving generation quality.

In summary, this paper makes three contributions. **First**, we introduce per-layer gradient probing, a layer-wise measurement of cross-objective compatibility, and show that it reveals the structural cause of depth-knowledge degradation during editing fine-tuning. **Second**, we propose STITCHEDIT, whose probe-guided training schedule selectively fine-tunes only the layers where depth and editing objectives are compatible, requiring no complex inference-time mechanisms. **Third**, we construct

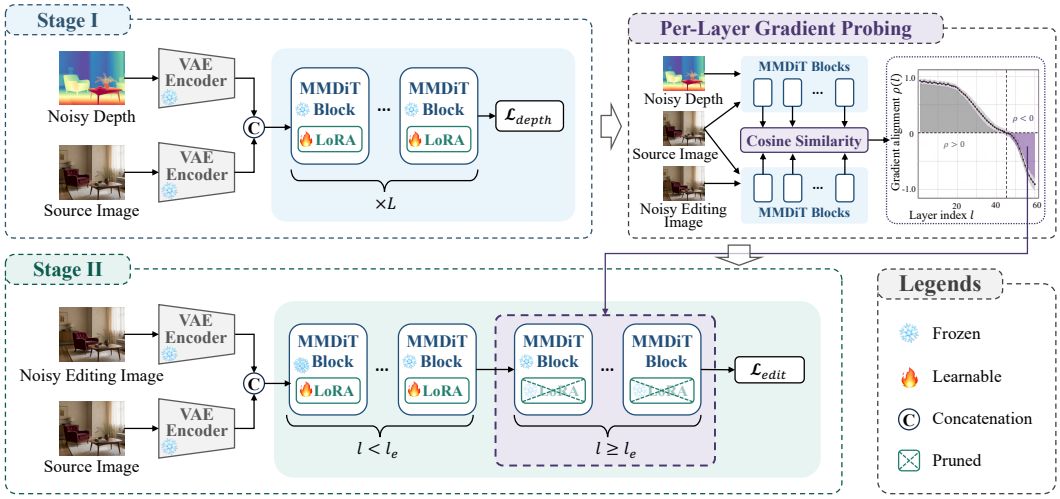


Figure 2: **Overview of StitchEdit. Stage 1**: Depth activation via a frozen-backbone LoRA adapter. **Per-layer gradient probing** reveals the layer-wise compatibility between the depth and editing objectives. **Stage 2**: The probing profile fundamentally determines both the LoRA pruning boundary and the per-layer learning rate schedule for the editing fine-tuning.

StitchEditBench, a 150-case benchmark of geometry-dependent edits, and show that StitchEdit consistently outperforms instruction-based and depth-initialized baselines.

2 Related Work

Geometry-aware image editing. Text-guided editing methods [1, 8, 15] operate on 2D features and lack the geometric understanding needed for spatially consistent edits. A line of recent work injects geometry through external estimators combined with explicit 3D warping [13], which introduces warping artifacts and propagates depth errors into the output, while studies of depth conditioning [25] show that models simply ignore injected geometric cues whenever conditioning does not compel their use. StitchEdit differs in two respects. First, geometry is not injected from outside but read out of the same backbone via a lightweight adapter, so depth and editing share a representational space and require no cross-modal alignment. Second, the resulting depth adapter is selectively fine-tuned for editing using a probe-derived per-layer schedule rather than applied uniformly, which preserves geometric knowledge where it helps and removes it where it conflicts with editing. Related work GeoEdit [26] tackles geometric editing through 3D reconstruction and in-context inpainting with effects-sensitive attention; our method instead reads geometry out of the backbone's own representations without external 3D reconstruction.

Layer-wise structure of diffusion transformers. Activating geometric capability inside a diffusion backbone has been studied for depth alone: Marigold [10], DepthFM [5], and the Lotus family [7, 6] repurpose generation models for depth, with parameter-efficient [9] and unified [22, 11] variants preserving generation; however, all of these treat depth and generation as *parallel* tasks and none use the activated depth to guide editing. A complementary line of work characterizes the internal hierarchy of diffusion transformers: REPA [21] shows that aligning only early transformer layers with external visual encoders outperforms uniform alignment, since unconstrained late layers retain detail-synthesis capacity, and DDT [17] and SP-DiT [19] confirm this layer-wise division of labor in decoupled and prompted DiT architectures. StitchEdit extends this hierarchy from generation and perception to geometry-aware editing: a LoRA-activated depth adapter provides geometric knowledge without error propagation, per-layer gradient probing identifies where this knowledge is compatible with editing, and probe-guided selective fine-tuning transfers depth capability to editing while preserving both geometry and generation quality.

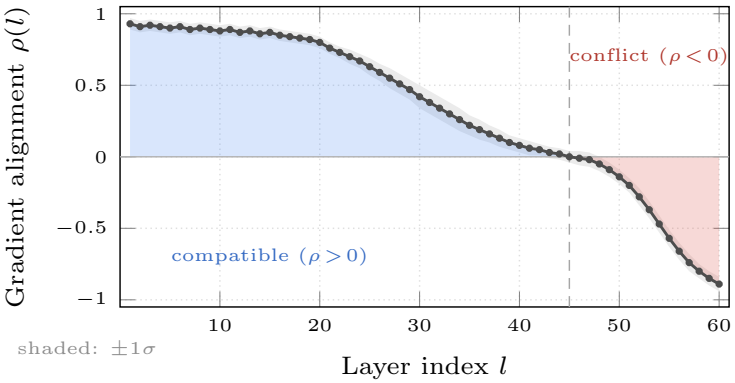


Figure 3: **Per-layer gradient probing profile $\rho(l)$.** The probe value saturates near 1 in early layers, decays through the intermediate layers, and drops below zero in late layers, indicating that the editing gradient actively interferes with depth knowledge. The shaded band denotes ± 1 standard deviation across the calibration set. This empirical profile directly dictates the pruning boundary l_e and explicitly guides the per-layer learning rate schedule in Stage 2.

3 Method

Figure 2 provides an overview of StitchEdit. We first introduce the preliminaries (§3.1), then present the depth-activation stage (§3.2), the per-layer gradient probe that reveals the compatibility structure (§3.3), and ultimately the probe-guided editing fine-tuning (§3.4).

3.1 Preliminaries

We build upon Qwen-Image-Edit [18], a Multi-Modal Diffusion Transformer (MMDiT) backbone that maintains separate text and image branches interacting through joint attention. For a noisy latent $\mathbf{z}_t$ at timestep $t \in [0, 1]$, the model is trained to predict the flow-matching velocity [12]:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{t,\mathbf{z}_0,\boldsymbol{\epsilon}} \left[\| v_\theta(\mathbf{z}_t, \mathbf{c}_t, \mathbf{c}, t) - (\boldsymbol{\epsilon} - \mathbf{z}_0) \|^2 \right], \tag{1}$$

where $\mathbf{z}_t = (1-t)\mathbf{z}_0 + t\boldsymbol{\epsilon}$, $\mathbf{z}_0$ is the clean latent, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ represents the sampled noise. The conditioning terms $\mathbf{c}_t$ and $\mathbf{c}$ denote the text prompt and the optional reference image, respectively. Given the velocity prediction v_θ, the estimated clean latent can be recovered via $\hat{\mathbf{z}}_0 = \mathbf{z}_t - t \cdot v_\theta$.

Functional hierarchy of transformer layers. A consistent empirical observation in diffusion transformers [21, 17, 19] is that early layers encode the global spatial layout that structuralizes a scene, whereas later layers focus on rendering fine-grained textures and materials. This functional division of labor provides the structural basis for the per-layer gradient probing profile we analyze in §3.3, and serves as the core motivation for our proposed layer-wise training strategy.

3.2 Stage 1: Depth Activation via LoRA

Stage 1 activates the latent depth capability of the backbone through a lightweight LoRA adapter. We condition the backbone on the source image and train it to denoise toward the depth map. Both the image and the depth are encoded by the shared VAE encoder ($\mathbf{z}^I = \text{Enc}(\mathbf{I})$, $\mathbf{z}_0^d = \text{Enc}(\mathbf{d})$); the noisy depth latent $\mathbf{z}_t^d$ is concatenated with the clean image latent $\mathbf{z}^I$ along the channel dimension and fed into the model. LoRA adapters [9] are inserted into the query, key, and value projections of each MMDiT block, keeping the backbone explicitly frozen:

$$\mathcal{L}_{\text{depth}} = \mathbb{E}_{t,\mathbf{z}_0^d,\boldsymbol{\epsilon}} \left[\left\| v_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}\big([\mathbf{z}^I, \mathbf{z}_t^d], t\big) - (\boldsymbol{\epsilon} - \mathbf{z}_0^d) \right\|^2 \right], \tag{2}$$

where $[\cdot, \cdot]$ denotes channel concatenation. At inference, the depth latent is recovered by the same denoising process and decoded by the shared VAE decoder, requiring no additional specific decoder. Freezing the backbone strictly preserves the generation-relevant representations upon which Stage 2 relies.

3.3 Per-Layer Gradient Probing

Following Stage 1, the LoRA adapter encodes rich geometric priors across all layers. However, naively continuing the training process exclusively on the editing objective risks catastrophic forgetting of this knowledge, particularly in layers where the two objectives diverge. To systematically identify these conflict zones, we introduce a per-layer gradient probe.

At each layer l, we quantify the compatibility between the editing and depth objectives via the cosine similarity of their respective gradients:

$$\rho(l) = \frac{\langle \nabla_{\boldsymbol{\theta}^l} \mathcal{L}_{\text{flow}}, \nabla_{\boldsymbol{\theta}^l} \mathcal{L}_{\text{depth}} \rangle}{\|\nabla_{\boldsymbol{\theta}^l} \mathcal{L}_{\text{flow}}\| \, \|\nabla_{\boldsymbol{\theta}^l} \mathcal{L}_{\text{depth}}\|}, \tag{3}$$

which is computed on a calibration set of editing pairs, utilizing the frozen backbone and the depth LoRA initialized from Stage 1. Unlike traditional hidden-state probes that extract properties from a single objective, our gradient probe explicitly measures the interference between two competing objectives at a given layer. Using a first-order Taylor expansion, we demonstrate that updating layer l with the editing gradient preserves depth knowledge only when $\rho(l) > 0$. Conversely, when $\rho(l) < 0$, training for editing actively increases the depth loss, thereby degrading geometric features:

$$\Delta \mathcal{L}_{\text{depth}} \approx -\eta \, \|\nabla_{\boldsymbol{\theta}^l} \mathcal{L}_{\text{flow}}\| \, \|\nabla_{\boldsymbol{\theta}^l} \mathcal{L}_{\text{depth}}\| \, \rho(l), \tag{4}$$

where η denotes the learning rate.

Empirically (Figure 3), we observe that $\rho(l)$ saturates near 1 in the early MMDiT layers, gradually decays through the intermediate layers, and ultimately crosses zero in the late layers. The shaded confidence band narrows at both extremes, confirming that alignment and conflict are consistent properties of their respective layer groups, while the transition zone exhibits greater variance across calibration pairs. This trajectory reveals three distinct regimes: **(1) Compatible zone** ($l \leq l_{\text{s}}$): $\rho \approx 1$; objectives reinforce each other. **(2) Transition zone** ($l_{\text{s}} < l < l_{\text{e}}$): ρ decays toward zero; objectives become increasingly orthogonal. **(3) Conflict zone** ($l \geq l_{\text{e}}$): $\rho < 0$; editing actively degrades depth knowledge. This gradient profile provides a principled basis for determining both the pruning boundary and the per-layer learning rate schedule in Stage 2.

3.4 Stage 2: Probe-Guided Editing Fine-tuning

We continue training the Stage 1 LoRA for the editing task, modulating the per-layer training intensity according to the probing profile. This ensures that geometric priors are preserved where they are synergistic with editing, and rigorously pruned where they induce conflict.

LoRA pruning. We remove the LoRA adapters from all layers $l \geq l_{\text{e}}$, where $\rho(l) < 0$. These layers revert to the frozen backbone behavior, ensuring that the appearance-synthesizing capacity of the late layers is not corrupted by depth-trained weights whose gradients explicitly oppose the editing objective.

Per-layer learning rate schedule. For the retained layers ($l < l_{\text{e}}$), we modulate the learning rate using a probe-derived cosine multiplier:

$$\eta(l) = w(l) \cdot \eta_0, \quad w(l) = \mathbb{1}[l \leq l_{\text{s}}] + \cos\!\left(\tfrac{\pi}{2} \cdot \tfrac{l - l_{\text{s}}}{l_{\text{e}} - l_{\text{s}}}\right) \mathbb{1}[l_{\text{s}} < l < l_{\text{e}}], \tag{5}$$

such that layers with high ρ train at the full rate, while transition layers receive progressively attenuated rates. The boundary l_{s} marks the onset of decay in the gradient probing profile.

Training objective. The editing model takes as input the clean source-image latent $\mathbf{z}^I$ and the noisy target-edit latent $\mathbf{z}_t^e$, concatenated along the channel dimension, together with the text instruction $\mathbf{c}_t$:

$$\mathcal{L}_{\text{depth}} = \mathbb{E}_{t, \mathbf{z}_0^e, \boldsymbol{\epsilon}} \left[\left\| v_{\boldsymbol{\theta} + \Delta \boldsymbol{\theta}'}([\mathbf{z}^I, \mathbf{z}_t^e],\, \mathbf{c}_t,\, t) - (\boldsymbol{\epsilon} - \mathbf{z}_0^e) \right\|^2 \right], \tag{6}$$

where $\Delta \boldsymbol{\theta}'$ denotes the LoRA parameters after pruning. Our design is intentionally minimal: there is no dual forward pass, no per-layer gating network, and no auxiliary geometric loss. The probe-guided schedule alone optimally balances geometry and editing.

3.5 StitchEditBench: Task Taxonomy and Data Construction

We introduce **StitchEditBench**, a comprehensive benchmark organized around geometry-aware edits. Each sample is formulated as a 5-tuple: $(\mathbf{I}_{\text{src}}, \mathbf{c}_t, \mathbf{I}_{\text{tgt}}, \mathbf{D}_{\text{src}}, \mathbf{D}_{\text{tgt}})$, representing the source image, text instruction, target image, source depth, and target depth, respectively.

Task taxonomy. We categorize geometric edits into three types: **(1) Spatial Transformation (ST)** repositions objects in 3D space (translate, rotate, rescale), requiring adjustments for scale and shadows; **(2) Occlusion Manipulation (OM)** alters visibility relationships (insert, remove, reorder), requiring rigorous depth ordering and occluded-region completion; and **(3) Viewpoint Change (VC)** applies camera transformations, requiring global structural transformation of the depth field.

Data construction & Statistics. We synthesize paired data by chaining a Large Language Model planner (Gemini [3]) and an image generator (Nano-Banana [4]). Specifically, Gemini is prompted with a source image and a category label to produce a structured edit specification alongside a natural-language instruction $\mathbf{c}_t$. The same Gemini model acts as a secondary verifier to filter out non-compliant geometric transformations, yielding a high-precision pool. The depth maps $(\mathbf{D}_{\text{src}}, \mathbf{D}_{\text{tgt}})$ are generated by our Stage 1 depth model. The training set comprises 10K pairs (4K ST, 3.5K OM, 2.5K VC). The evaluation benchmark contains 150 meticulously curated cases drawn from a strictly held-out, non-overlapping source-image pool.

4 Experiments

4.1 Experimental Setup

Implementation Details. Our framework is built upon Qwen-Image-Edit [18], a DiT backbone comprising $L = 60$ blocks. We attach LoRA adapters (rank $r = 64$, $\alpha = 128$) to the query, key, and value projections of each MMDiT block. All images are processed at 1024×1024 resolution with classifier-free guidance scale 4.0. All experiments are conducted on 8 NVIDIA H100 GPUs. **Stage 1** trains the depth LoRA for 50K steps at a batch size of 32 using AdamW (learning rate 1×10^{-4}, weight decay 1×10^{-2}, 1K-step warmup) on the 3M image-depth pairs curated by Lotus-2 [6]. **Per-layer gradient probing** is computed on a 1K-pair calibration subset drawn from the Stage 2 training pool, yielding the critical boundaries $l_{\text{s}} = 20$ and $l_{\text{e}} = 45$. **Stage 2** consequently prunes the LoRA from layers $l \geq 45$ and continues fine-tuning on our constructed 10K editing pairs. This stage trains for 100K steps at a batch size of 16 with AdamW (learning rate 2×10^{-5}, weight decay 1×10^{-2}, 500-step warmup), strictly applying the probe-guided schedule formulated in Equation (5).

Benchmark and Metrics. We evaluate our method on the proposed *StitchEditBench*. Perceptual quality is measured by *LPIPS* [24]. Geometric fidelity is evaluated via the *Depth Consistency Score (DCS)*, defined as the Pearson correlation between the target depth and the depth of the edited output (estimated by Depth Anything V2 [20]). Instruction following is measured by *CLIP-Dir* [14], which calculates the cosine similarity between the CLIP embedding change of the image pair and the instruction embedding. Additionally, we report a *User Preference* score, aggregating rankings from 50 human participants across all test cases.

Baselines. We benchmark against two distinct families of methods. *Instruction-based editors* rely solely on the source image and textual instructions, represented by the state-of-the-art closed-source model Nano-Banana [4] and the open-source model Qwen-Image-Edit [18]. *Depth-initialized editors* benefit from our Stage 1 depth pre-training. We define *Uniform FT* as a rigorous ablation baseline where all LoRA layers are trained uniformly at a constant learning rate, devoid of our probe guidance.

4.2 Main Results

Table 1 reports the quantitative comparison across all 150 test cases. STITCHEDIT consistently achieves the best performance across all metrics. Notably, while Nano-Banana and Qwen-Image-Edit yield visually plausible outputs, they score poorly on DCS, confirming that semantic text supervision alone is insufficient to enforce strict geometric consistency. Uniform FT serves as the most revealing depth-initialized baseline: despite inheriting the same depth-pre-trained adapter, uniformly updating all layers leads to catastrophic forgetting of geometric knowledge in conflict zones, degrading the DCS to 0.80. Conversely, the probe-guided schedule of STITCHEDIT optimally preserves geometric priors where they cooperate with editing and prunes them where they interfere, unlocking the best



Figure 4: **Qualitative comparison.** StitchEdit maintains strict geometric consistency while preserving exceptional visual quality. Instruction-based editors frequently fail to respect 3D spatial structures, and Uniform FT suffers from depth degradation.

Table 1: **Main results on *StitchEditBench*.** Uniform FT starts from the same depth-pre-trained adapter as StitchEdit but trains all layers uniformly, leading to the loss of geometric knowledge. StitchEdit resolves this via probe-guided selective training.

Method	LPIPS↓	DCS↑	CLIP-Dir↑	User Pref.↑
Instruction-based				
Nano-Banana [4]	0.13	0.74	0.78	16%
Qwen-Image-Edit [18]	0.14	0.71	0.76	14%
Depth-initialized				
Uniform FT	0.14	0.80	0.76	12%
StitchEdit (Ours)	**0.10**	**0.92**	**0.81**	**58%**

trade-off between visual quality and geometric fidelity. The 58% User Preference score further corroborates this: human raters consistently favor outputs that respect spatial plausibility, and the 0.12-point DCS gap between Uniform FT and StitchEdit (0.80 vs. 0.92) confirms that the late-layer conflict zone identified by our probe is the primary source of geometric forgetting.

Per-category analysis. The superiority of StitchEdit scales proportionally with the geometric complexity of the edit: relative to the strongest instruction-based baseline, our method reduces

LPIPS by 0.03 on Spatial Transformation (ST), 0.04 on Viewpoint Change (VC), and 0.02 on Occlusion Manipulation (OM). The DCS performance gap remains consistently substantial across all categories (Table 2). The largest DCS improvement appears on VC (0.90 vs. 0.69 for Nano-Banana), where global depth-field transformation demands coherent geometric reasoning across the entire image. Figure 4 provides representative qualitative comparisons: instruction-based editors frequently disregard scale transformations and occlusion logic; Uniform FT partially corrupts depth constraints; whereas STITCHEDIT synthesizes geometrically rigorous and visually compelling edits.

Table 2: **Per-category breakdown on StitchEditBench.** All three categories require geometric understanding; gains are consistent across spatial transformation, occlusion manipulation, and viewpoint change.

	ST (60)		OM (50)		VC (40)	
Method	LPIPS↓	DCS↑	LPIPS↓	DCS↑	LPIPS↓	DCS↑
Nano-Banana [4]	0.14	0.71	0.13	0.76	0.16	0.69
Qwen-Image-Edit [18]	0.15	0.68	0.14	0.73	0.17	0.66
Uniform FT	0.15	0.78	0.14	0.81	0.16	0.77
STITCHEDIT	**0.11**	**0.91**	**0.11**	**0.91**	**0.12**	**0.90**

4.3 Ablation Studies

Training strategy. Table 3 validates the efficacy of our probe-guided schedule against alternative strategies. Direct LoRA fine-tuning without Stage 1 depth pre-training yields the poorest DCS (0.70), confirming that explicit geometric initialization is indispensable. Uniform FT improves DCS to 0.80 but still suffers from geometric forgetting in the late layers. Top-K Freeze retains depth LoRA only in the first K layers and removes it elsewhere; its 0.87 DCS shows that a hard cutoff discards useful geometric information in the transition zone. Random Schedule assigns each layer a uniformly sampled learning rate multiplier, serving as a controlled baseline to verify that the probe's layer ordering carries genuine signal rather than benefiting from mere variance in training intensity; its 0.83 DCS confirms this. A heuristic Fixed Cosine decay captures a significant portion of the benefit (0.90 DCS), validating the existence of the three-regime structure, but its fixed boundaries cannot adapt to the actual probe profile. Ultimately, our dynamically computed **Probe-Guided schedule** achieves the apex performance (0.92 DCS, 0.10 LPIPS) by precisely calibrating the learning intensity to the layer-wise objective compatibility.

Table 3: **Ablation on training strategy.** Probe-guided selective fine-tuning significantly outperforms both uniform training and heuristic schedules. Direct fine-tuning validates the absolute necessity of Stage 1 geometric initialization.

Strategy	Depth Init	LPIPS↓	DCS↑	CLIP-Dir↑
Direct LoRA	×	0.17	0.70	0.74
Uniform FT	✓	0.14	0.80	0.76
Top-K Freeze	✓	0.12	0.87	0.78
Fixed Cosine	✓	0.11	0.90	0.80
Random Schedule	✓	0.13	0.83	0.77
Probe-Guided (Ours)	✓	**0.10**	**0.92**	**0.81**

Explicit vs. implicit geometric conditioning. A natural alternative to our implicit depth-prior approach is to supply an explicit depth map through a ControlNet-style adapter [23]. We evaluate two variants: *ControlNet-Depth ($\mathbf{D}_{src}$)* conditions the backbone on the *source* depth map, and *ControlNet-Depth ($\mathbf{D}_{tgt}$)* conditions on the *target* depth map. The latter serves as an **oracle upper bound**, as the target depth is strictly unavailable at inference time.

Table 4: **Explicit vs. implicit conditioning.**

Method	LPIPS↓	DCS↑	CLIP-Dir↑
CN-D ($\mathbf{D}_{\text{src}}$)	0.17	0.72	0.62
CN-D ($\mathbf{D}_{\text{tgt}}$)†	0.12	0.90	0.79
Ours	**0.10**	**0.92**	**0.81**

†Oracle: target depth unavailable.

Table 4 reveals the core limitation of explicit conditioning. Conditioning on the source depth rigidly locks the spatial layout to the initial geometry, causing the model to resist the structural changes demanded by the edit instruction. This leads to a severe degradation in both instruction following (0.62 CLIP-Dir) and geometric fidelity (0.72 DCS, measured against the *target* depth). Even the oracle variant, which receives the ground-truth target depth, achieves only 0.90 DCS and 0.79 CLIP-Dir, still underperforming STITCHEDIT (0.92, 0.81). This performance gap arises because rigid ControlNet conditioning excessively constrains the network's appearance-synthesis capacity, whereas STITCHEDIT's implicit prior leaves the backbone free to jointly optimize geometry and visual quality. In particular, the oracle variant demonstrates that even perfect geometric input cannot compensate for the loss of representational flexibility: the ControlNet signal competes with the backbone's own features at every layer, whereas our probe-guided LoRA cooperates with the backbone in early layers and simply stays silent in late ones.

Schedule boundary sensitivity. Table 5 varies l_s and l_e around the probed defaults. Performance peaks at $(20, 45)$ and degrades when the compatible region is too narrow ($l_\text{s} = 10$: insufficient geometric capacity) or too broad ($l_\text{s} = 30$: conflicting layers are trained, degrading depth knowledge). The optimal $l_\text{s} = 20$ aligns with the REPA [21] finding that roughly the first third of layers carry spatial structure, suggesting a shared structural boundary in diffusion transformers. Notably, the $(30, 55)$ setting maintains high DCS (0.92) but suffers a sharp CLIP-Dir drop to 0.75, indicating that training conflict-zone layers preserves depth at the cost of editing fidelity—precisely the trade-off our probe is designed to avoid.

Table 5: **Sensitivity to l_s, l_e boundaries.**

l_s	l_e	LPIPS↓	DCS↑	CLIP-Dir↑
10	35	0.12	0.86	0.78
15	40	0.11	0.89	0.79
20	**45**	**0.10**	**0.92**	**0.81**
25	50	0.11	0.91	0.78
30	55	0.13	0.92	0.75

Decay function. We evaluate three mathematical decay functions for the transition zone ($l_\text{s} < l < l_\text{e}$). As shown in Table 6, the Cosine decay formulation achieves the best balance (0.10 LPIPS, 0.92 DCS), outperforming both step-function (0.13, 0.88) and linear decay (0.11, 0.90). Its slow-fast-slow continuous profile strategically concentrates parameter updates at the stable regime extremes while rapidly traversing the ambiguous middle layers.

Table 6: **Decay function shape.**

Decay	LPIPS↓	DCS↑	CLIP-Dir↑
Step	0.13	0.88	0.78
Linear	0.11	0.90	0.79
Cosine	**0.10**	**0.92**	**0.81**

Depth estimation capacity. Beyond its primary role in facilitating editing, the Stage 1 depth LoRA alone achieves competitive depth estimation performance (Table 7). On the NYU Depth V2 dataset [16], our model attains 5.4 AbsRel, slightly outperforming dedicated depth estimators (*e.g.* Marigold [10]: 5.5). Notably, Qwen-Image-Edit's 12.2 AbsRel reveals that the backbone's latent geometric knowledge is poorly activated without dedicated depth training; the Stage 1 LoRA, despite its lightweight parameter footprint, reduces the error by more than half (5.4 vs. 12.2), confirming that a small adapter suffices to unlock the backbone's depth capability.

Table 7: **Depth estimation (NYU V2).**

Method	AbsRel↓	δ_1↑	Edit
Marigold	5.5	0.964	×
Depth Any. V2	4.3	0.981	×
Qwen-IE	12.2	0.865	✓
Ours (S1)	**5.4**	**0.966**	✓

5 Conclusion and Limitations

We presented STITCHEDIT to solve the challenge of spatial manipulations in text-guided image editing. Through per-layer gradient probing, we found that depth and editing objectives cooperate in early transformer layers but actively conflict in late ones. This insight drives a selective fine-tuning schedule that preserves geometric knowledge where it helps and prunes it where it harms. As a result, STITCHEDIT achieves strict spatial consistency without the rigidity of explicit depth conditioning.

Limitations. STITCHEDIT currently relies on natural-language instructions to specify spatial edits, which limits the precision of geometric control. For instance, the user cannot prescribe an exact scale factor or rotation angle; the model interprets approximate intent from text. Extending the framework with explicit geometric parameters (*e.g.* rotation angles, translation offsets) to enable fine-grained spatial control is a promising direction for future work.

Acknowledgements

The work is supported in part by the Early Career Scheme of the Research Grants Council (RGC) of the Hong Kong SAR under grant No. 26202321, ITF PRP/046/24FX, the Science & Technology Cooperation Program of Shandong under grant No. SDST26EG01, SAIL Research Project, and HKUST Zeekr Collaborative Research Fund.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800*, 2023.

[2] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In *ECCV*, 2024.

[3] Google DeepMind. Gemini: A family of highly capable multimodal models. `https://gemini.google/overview/`, 2025.

[4] Google DeepMind. Nano-banana: Image generation and editing in gemini. `https://gemini.google/overview/image-generation/`, 2025.

[5] Ming Gui, Johannes S Fischer, Ulrich Prestel, Pingchuan Ma, Dmytro Kotovenko, Olga Grebenkova, Stefan Andreas Baumann, Vincent Tao Hu, and Björn Ommer. Depthfm: Fast monocular depth estimation with flow matching. *arXiv preprint arXiv:2403.13788*, 2024.

[6] Jing He, Haodong Li, Mingzhi Sheng, and Ying-Cong Chen. Lotus-2: Advancing geometric dense prediction with powerful image generative model. *arXiv preprint arXiv:2512.01030*, 2025.

[7] Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Liu, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. *arXiv preprint arXiv:2409.18124*, 2024.

[8] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

[10] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *CVPR*, 2024.

[11] Hongkai Lin, Dingkang Liang, Mingyang Du, Xin Zhou, and Xiang Bai. More than generation: Unifying generation and depth estimation via text-to-image diffusion models. In *NeurIPS*, 2025.

[12] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[13] Rishubh Parihar, Sachidanand VS, and R Venkatesh Babu. Zero-shot depth aware image editing with diffusion models. In *CVPR*, 2025.

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[15] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[16] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[17] Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. *arXiv preprint arXiv:2504.05741*, 2025.

[18] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

[19] Gangwei Xu, Haotong Lin, Hongcheng Luo, Xianqi Wang, Jingfeng Yao, Lianghui Zhu, Yuechuan Pu, Cheng Chi, Haiyang Sun, Bing Wang, Guang Chen, Hangjun Ye, Sida Peng, and Xin Yang. Pixel-perfect depth with semantics-prompted diffusion transformers. *arXiv preprint arXiv:2510.07316*, 2025.

[20] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv preprint arXiv:2406.09414*, 2024.

[21] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *ICLR*, 2025.

[22] Jingyang Zhang, Shiwei Li, Yuanxun Lu, Tian Fang, David Neil McKinnon, Yanghai Tsin, Long Quan, and Yao Yao. Jointnet: Extending text-to-image diffusion for dense distribution modeling. In *ICLR*, 2024.

[23] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023.

[24] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[25] Shihua Zhang, Qiuhong Shen, Shizun Wang, Tianbo Pan, and Xinchao Wang. Make geometry matter for spatial reasoning. *arXiv preprint arXiv:2603.26639*, 2024.

[26] Shuo Zhang, Wenzhuo Wu, Huayu Zhang, Jiarong Cheng, Xianghao Zang, Chao Ban, Hao Sun, Zhongjiang He, Tianwei Cao, Kongming Liang, and Zhanyu Ma. Geometric image editing via effects-sensitive in-context inpainting with diffusion transformers. In *ICLR*, 2026.

A Per-Layer Gradient Probing Details

The probing profile $\rho(l)$ is computed on a 1K-pair calibration subset drawn from the Stage 2 training pool. For each pair, we compute $\nabla_{\boldsymbol{\theta}^l}\mathcal{L}_{\text{flow}}$ and $\nabla_{\boldsymbol{\theta}^l}\mathcal{L}_{\text{depth}}$ at every MMDiT block using the frozen backbone and the Stage 1 LoRA, then take the per-layer cosine similarity and average across the calibration set. Probing takes $\sim$30 minutes on a single H100 and is run once before Stage 2 training; the resulting curve (Figure 3) determines l_{s} and l_{e}.

B Broader Impact Statement

StitchEdit enables geometrically plausible editing for creative applications, VR/AR, and content generation. As with all editing methods, potential misuse for generating misleading content exists. We encourage responsible use and development of detection methods for edited images.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state three contributions (per-layer gradient probing, probe-guided selective fine-tuning, StitchEditBench), all of which are supported by the experiments in §4 and appendix.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present formal theorems. The first-order gradient analysis in Equation (4) is a standard Taylor expansion used as motivation, not a formal theorem.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All hyperparameters, training schedules, data construction pipelines, and evaluation metrics are detailed in §4 and the appendix. The probing boundary values (l_s, l_e) and the per-layer learning rate schedule are fully specified.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code and the StitchEditBench dataset will be released upon acceptance. The paper provides sufficient detail for independent reimplementation.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: §4 specifies the optimizer, learning rates, batch sizes, warmup steps, training steps, and data splits for both stages. The per-layer schedule boundaries are derived from the probing profile and reported explicitly.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We report single-run results following standard practice in the image editing literature. The probing profile is averaged over a 1K calibration set. The user study collects preferences from 50 participants.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: All experiments use 8 NVIDIA H100 GPUs as stated in §4. Probing takes approximately 30 minutes on a single H100.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics. The broader impact statement in the appendix discusses potential misuse.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The appendix includes a broader impact statement discussing positive applications (creative tools, VR/AR) and potential negative impacts (misleading content generation).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The method fine-tunes an existing publicly available model (Qwen-Image-Edit) and does not introduce new risks beyond those of the base model. The StitchEditBench dataset contains synthetic editing pairs, not personal data.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models and datasets used are properly cited. Qwen-Image-Edit is used under its public license. Baseline methods are properly attributed.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: StitchEditBench is described in detail in §3.5, including the data construction pipeline, statistics, and category definitions. The dataset will be released with documentation upon acceptance.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: The user preference study involves 50 participants who rank editing outputs. Participants were shown all method outputs for each test case and asked to rank them by overall quality and geometric correctness.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The user study involves only preference ranking of edited images with no risk to participants. No personal data was collected.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Gemini is used as the planner and verifier in the StitchEditBench data construction pipeline (§3.5), and Nano-Banana (a Gemini-based image editor) generates target images. Both are non-standard components of the data pipeline and are fully described.