VLTR: Vision-Language Tool Reasoning for Instruction-Guided Image Editing

Yike Wang[1], Yitao Yu[1], Shaohua Sun[2], and Ping Luo[1⋆]

[1] The University of Hong Kong, Hong Kong SAR, China
u3641127@connect.hku.hk, yitao_yu2024@connect.hku.hk, pluo@cs.hku.hk
[2] EPFL, Lausanne, Switzerland
shaohua.sun@epfl.ch

Abstract. Multi-tool agents for instruction-guided image editing remain brittle under open-loop pipelines, where early routing errors propagate unchecked through the entire workflow. We propose VLTR (Vision-Language Tool Reasoning), a training-free framework that reformulates editing as closed-loop tool reasoning over a directed acyclic graph (DAG) of atomic primitives. A generative decomposer first converts each natural-language instruction into executable primitives. A Bayes-UCB router then adaptively selects tools by combining contextual statistics with semantic priors, balancing exploration and exploitation online. After each execution, a heteroscedastic verifier produces a calibrated quality–uncertainty signal (q, σ^2) through inverse-variance fusion of three complementary assessment tiers. A verifier-guided scheduler uses this joint signal to retry, reroute, or replan only the failed subgraph while preserving validated ancestors, turning full-pipeline restarts into efficient local repair. On PIE-Bench++, VLTR achieves the best overall VLM ranking among all evaluated baselines. On MagicBrush, it delivers a 4.8× runtime speedup over the strongest multi-tool competitor with no quality loss. Pilot evaluations on GEdit-Bench and RISEBench further confirm generalization to challenging edits.

Keywords: Instruction-guided image editing · Closed-loop reasoning · Heteroscedastic uncertainty · Multi-tool agents

1 Introduction

Text-driven image editing has advanced rapidly with diffusion-based and multimodal models such as InstructPix2Pix, ControlNet, FLUX.1, and Stable Diffusion 3 [2, 3, 12, 33]. These models achieve strong synthesis quality when the instruction is concrete, but they remain unreliable for abstract, multi-step requests such as "make this photo more cinematic" or "keep the beautiful silhouette vibe." Such instructions must be decomposed into executable primitives, matched to heterogeneous tools, and verified online while preserving already-correct regions. As shown in Fig. 1, current systems struggle because they either treat editing as

⋆ Corresponding author.

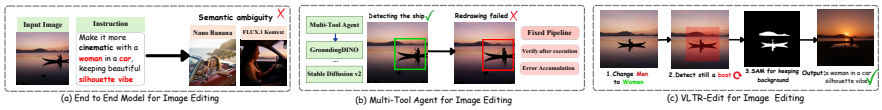

Fig. 1: Comparison on abstract semantic preservation. Given the instruction to transform the scene while "keeping the beautiful silhouette vibe," end-to-end models and fixed multi-tool pipelines fail to preserve the abstract effect, while VLTR succeeds through adaptive routing with verification-driven iteration.

a single forward pass or commit to a fixed tool chain before observing execution failures.

The core difficulty is not only visual generation but *instructive reasoning*. A user instruction can be semantically ambiguous, multiple tools may appear plausible for the same primitive, and failures are often local rather than global. Existing multi-tool systems [6, 23, 25, 26] offer modularity, yet they still rely largely on static planning and uniform-confidence assumptions. Once an early decision is wrong, the rest of the pipeline inherits that error. This motivates a closed-loop formulation in which tool selection and execution are continuously revised according to online feedback.

We address this problem with **VLTR** (Vision-Language Tool Reasoning), a training-free framework that casts instruction-guided editing as closed-loop tool reasoning over executable primitive states. A planner first decomposes the instruction into a directed acyclic graph of atomic primitives. A Bayes-UCB router then adaptively selects tools for each primitive by combining contextual statistics with semantic compatibility signals when needed. After each execution, a heteroscedastic verifier produces a quality–uncertainty pair (q, σ^2) from three complementary assessment tiers. An uncertainty-aware scheduler uses this signal to accept, retry, reroute, or replan only the failed subgraph, rather than restarting the whole pipeline. This closes the loop between reasoning and execution and turns uncertainty from a passive confidence score into an active control signal.

Beyond this closed-loop architecture, we ground routing, verification, and scheduling in principled statistical decision-making. Specifically, the verifier adopts inverse-variance fusion, interpretable as a BLUE-style estimator under a heteroscedastic Gaussian assumption; the router is instantiated with Bayes-UCB rather than a heuristic score; and the decision thresholds are guided by a Bayesian-risk criterion. We further broaden the empirical study with compute-matched comparisons, cost-benefit analysis, calibration statistics, and additional benchmark pilots.

Our contributions are:

- **Closed-loop tool reasoning for image editing.** We formulate instruction-guided editing as closed-loop search over primitive–tool states, enabling per-node interventions instead of fixed pipelines.

– **Principled routing and verification.** We combine a Bayes-UCB router with a heteroscedastic verifier whose inverse-variance fusion yields a statistically grounded quality–uncertainty signal rooted in BLUE-style estimation.
– **Uncertainty-aware local repair.** We derive verifier-guided scheduling decisions from joint quality–uncertainty feedback, and use a DAG representation to retry, reroute, or replan only the failed subgraph while preserving validated edits.

2 Related Work

2.1 Text-to-Image Editing

Text-driven image editing aims to modify images according to natural language instructions while preserving structural coherence and visual realism [2,3,21,34]. Diffusion-based methods such as InstructPix2Pix, Imagic, and ControlNet [2, 11, 33] have established strong baselines for direct instruction following, while newer systems such as SmartEdit, MGIE, and InsightEdit improve language understanding with multimodal large language models [5, 8, 28]. Training-free pipelines such as Zero [18] further show that competitive editing can be achieved without retraining. Recent diffusion transformers including FLUX.1 and Stable Diffusion 3 further improve synthesis quality [3, 12]. However, most of these methods still assume a single editing model or a fixed editing recipe, making them less reliable for abstract instructions that require several coordinated operations.

2.2 Multi-Tool Vision Systems

Large language models have enabled agent-based systems that coordinate specialized tools for complex visual tasks [6,23,25,26]. VisProg generates executable programs for visual reasoning [6], HuggingGPT provides general cross-model planning [23], and GenArtist applies MLLM-based orchestration to image generation and editing [25]. Recent multimodal foundation models such as GPT-4V [17] and planning paradigms such as ReAct [31] further strengthen tool-using agents in open-ended visual tasks. Related work on adaptive expert routing such as MoVA [37] likewise shows that modular systems can outperform monolithic models when expert selection is handled well. Yet most existing tool orchestrators remain largely static after planning: they seldom verify intermediate outputs, calibrate tool confidence, or selectively repair only the failing branch. Our work focuses precisely on this online control problem.

2.3 Multi-Step Reasoning and Deliberative Search

Recent work studies multi-step reasoning through branching, revision, and search over intermediate states [1,4,14,16,24,30,36]. These methods show that explicit deliberation can outperform single-pass decision making when intermediate outcomes can be evaluated and revised. In instruction-guided image editing, however, the states are executable primitive–tool configurations whose quality must

be judged by external visual feedback rather than text-only self-evaluation. We therefore position VLTR as a vision-language tool reasoning framework with verifier-guided scheduling, adaptive routing, and DAG-based local repair, while treating this line of deliberative search work only as broad related background rather than the central framing of the method.

3 Method

VLTR is a training-free framework for instruction-guided image editing that treats execution as *closed-loop tool reasoning*. Instead of fixing a tool pipeline upfront, VLTR expands one primitive at a time, executes it, evaluates the result, and uses the returned quality–uncertainty signal to decide whether to continue, retry, reroute, or replan. This formulation combines four ingredients: a structured primitive decomposition, an adaptive Bayes-UCB router, a heteroscedastic verifier, and a verifier-guided scheduler with DAG-based local repair.

3.1 Overview

Given a source image $I_s \in \mathbb{R}^{3\times H\times W}$ and a natural-language instruction $\mathcal{L}$, our goal is to produce an edited image I_e that satisfies user intent while preserving unaffected regions. VLTR proceeds through four stages (Fig. 2):

$$(I_s, \mathcal{L}) \xrightarrow{\text{Understand}} P \xrightarrow{\text{Route}} \mathcal{S} \xrightarrow{\text{Verify}} (q, \sigma^2) \xrightarrow{\text{Repair}} I_e \tag{1}$$

where $P = \{p_1, \ldots, p_M\}$ is a set of atomic primitives arranged as a directed acyclic graph. The search states are executable tuples of the form $(p_i, T_k, \text{params})$ and must be judged by external visual verifiers rather than self-evaluation. The resulting uncertainty estimate is therefore central to both assessment and control.

3.2 Stage 1: Primitive Decomposition

The first stage converts the input instruction into atomic editing primitives that define the search space. Each primitive is represented as

$$p_i = (\text{type}_i, \text{target}_i, \text{description}_i, \text{order}_i) \tag{2}$$

where type_i follows the PIE-Bench++ taxonomy, target_i specifies the entity or region to edit, description_i retains fine-grained semantics needed by downstream tools, and order_i encodes dependencies. The output is a DAG rather than a chain, which later allows VLTR to re-execute only failed nodes and their descendants.

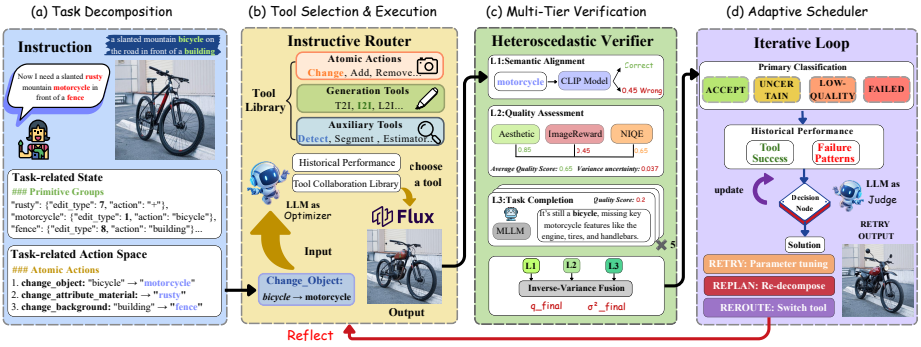

Fig. 2: VLTR overview. The instruction is decomposed into atomic primitives, routed to candidate tools, verified through three complementary assessment tiers, and corrected through retry, reroute, or replan decisions. The DAG representation allows local repair while preserving validated edits.

3.3 Stage 2: Adaptive Tool Routing

The router expands each primitive by selecting a tool from the library of generation, editing, and auxiliary models. The candidate set is first pruned according to primitive type and required inputs, which reduces the action space from the full tool library to a small subset $\mathcal{T}_{p_i}$ of plausible options.

We then model tool selection as a contextual multi-armed bandit. For tool T_k selected at step t, the raw reward combines quality and uncertainty:

$$\tilde{r}_t(T_k, x_t) = \max\big(0,\ q_t - \lambda_u \sigma_t^2\big), \tag{3}$$

where x_t is the primitive context and q_t, σ_t^2 come from the verifier. Because $q_t \in [0,1]$ by construction and $\lambda_u \sigma_t^2 \geq 0$, the clamped reward satisfies $\tilde{r}_t \in [0,1]$, which allows it to be interpreted as a fractional success under a Beta-Bernoulli observation model. Each tool maintains a Beta posterior, updated online as

$$\alpha_k \leftarrow \alpha_k + \tilde{r}_t, \qquad \beta_k \leftarrow \beta_k + (1 - \tilde{r}_t). \tag{4}$$

The base routing policy is Bayes-UCB [10]:

$$T^* = \arg\max_{T_k \in \mathcal{T}_{p_i}} Q_{0.95}(\mathrm{Beta}(\alpha_k, \beta_k))\,, \tag{5}$$

which balances exploitation and exploration through an upper posterior quantile instead of a heuristic score. When the top tools remain statistically ambiguous, we query an LLM for semantic compatibility scores and fuse them with the statistical score via a fixed convex combination:

$$s_{\text{final}}(T_k, p_i) = w_{\text{stat}}\, s_{\text{stat}} + w_{\text{LLM}}\, s_{\text{LLM}}, \qquad w_{\text{stat}} = 0.6,\ w_{\text{LLM}} = 0.4, \tag{6}$$

where the weights are fixed scalars selected on a held-out validation split. Unlike the verifier's inverse-variance fusion (Eq. 8), these two signals have no well-defined per-source variances, so this is a learned combination rather than a statistical estimator. It keeps routing cheap when the bandit is confident, invoking semantic reasoning only when needed.

3.4 Stage 3: Heteroscedastic Verifier

VLTR performs inline verification after every primitive execution. The verifier produces tier-wise scores (q_i, σ_i^2) from three complementary signals: CLIP-based semantic alignment [20], an expert ensemble composed of ImageReward [27], aesthetic prediction, and NIQE [15], and repeated VLM judgments that estimate reasoning uncertainty. Tier 1 uses a fixed calibrated variance, while Tier 2 uses disagreement among normalized expert scores:

$$\sigma_2^2 = \mathrm{var}\{q_{\text{ImageReward}}, q_{\text{Aesthetic}}, q_{\text{NIQE}}\}. \tag{7}$$

Tier 3 uses the sampling variance of repeated VLM judgments.

We then fuse the three tiers with inverse-variance weighting:

$$q = \sum_{i=1}^{3} w_i q_i, \qquad w_i = \frac{1/\sigma_i^2}{\sum_j 1/\sigma_j^2}. \tag{8}$$

Under a heteroscedastic Gaussian observation model, Eq. 8 coincides with the best linear unbiased estimator (BLUE) for that model; we use it as a principled, BLUE-style weighting rather than a claim of optimality under the true (unknown) noise distribution. To account for inter-tier disagreement, we augment the fused uncertainty as

$$\sigma_{\text{final}}^2 = \frac{1}{\sum_{i=1}^{3} 1/\sigma_i^2} + \lambda \cdot \mathrm{var}\{q_1, q_2, q_3\}, \tag{9}$$

where $\lambda = 0.5$ in our default setting. When the tiers agree, the second term vanishes and the estimator reduces to classical inverse-variance fusion; when they disagree, the additional term prevents overconfident acceptance. The fused pair (q, σ^2) and the tier-wise diagnostics are both passed to the scheduler.

3.5 Stage 4: Verifier-Guided Scheduler and DAG Repair

The uncertainty-aware scheduler turns the verifier output into control decisions. The Bayesian risk criterion defines the decision form:

$$R(\text{accept} \mid q, \sigma^2) = (1 - q) + C_{\text{uncertain}}\sigma^2, \tag{10}$$

We select $\tau_q = 0.6$ and $\tau_u = 0.15$ on a held-out validation split. The primary classification is then

$$\text{Primary} = \begin{cases} \texttt{ACCEPT} & \text{if } q \geq \tau_q \wedge \sigma^2 \leq \tau_u \\ \texttt{UNCERTAIN} & \text{if } q \geq \tau_q \wedge \sigma^2 > \tau_u \\ \texttt{LOW-QUALITY} & \text{if } q < \tau_q \wedge \sigma^2 > \tau_u \\ \texttt{FAILED} & \text{if } q < \tau_q \wedge \sigma^2 \leq \tau_u. \end{cases} \tag{11}$$

`ACCEPT` freezes the result and advances to the next primitive. `UNCERTAIN` triggers **RETRY**, where the same tool is re-executed with refined parameters.

`LOW-QUALITY` triggers **REROUTE**, where the router switches to a different candidate tool. Persistent `FAILED` outcomes trigger **REPLAN**, sending the instruction and accumulated failure evidence back to the decomposer. Because primitives form a DAG, accepted ancestors are cached and only the failed node and its descendants are re-executed. This adaptive local repair policy allows VLTR to improve quality while avoiding the computational overhead of repeated full pipeline restarts.

4 Experiments

4.1 Experimental Setup

Datasets and benchmarks. We evaluate primarily on **PIE-Bench++** [7], an extension of PIE-Bench [9], which contains 700 images across 9 representative editing categories, including object addition/removal, attribute modification, background replacement, and style transfer. Each sample provides source–target pairs and a detailed editing instruction; the ground-truth masks are used only for evaluation. We reserve a 400-case subset for the calibration analysis. We further test cross-dataset generalization on **MagicBrush** [32], a multi-turn editing benchmark. To evaluate out-of-domain robustness, we construct 200-sample evaluation splits from **GEdit-Bench** [13] and **RISEBench** [35], which focus on challenging generative edits and reasoning-informed visual transformations, respectively.

Baselines. We compare VLTR against three groups of methods: end-to-end diffusion editors (InstructPix2Pix [2], SDXL-Inpainting [19], FLUX.1-Kontext-Dev [12]), multi-tool systems (RPG [29], GenArtist [25], SmartEdit [8]), and ablated variants of our own pipeline.

Implementation details. VLTR orchestrates 26 core tools spanning generation and editing, listed in Table 1. For each primitive, the candidate set $\mathcal{T}_{p_i}$ is constructed by selecting tools whose declared primitive categories and required inputs match the primitive type, yielding 3–6 plausible options for Bayes-UCB routing. Supporting infrastructure and quality-assessment models are listed in the supplementary. A GPT-4V-based agent performs primitive decomposition and ambiguity resolution. The verifier combines CLIP ViT-B/32, ImageReward-v1.0, the LAION Aesthetic Predictor v2, NIQE, and repeated VLM judgments. The scheduler is initialized with $\tau_q = 0.6$ and $\tau_u = 0.15$, derived from the Bayesian risk criterion in Eq. 10, and then refined online during search. Unless otherwise stated, we use 50 diffusion steps with guidance scales $w_{\text{src}} = 1.0$ and $w_{\text{tgt}} = 7.5$. We further evaluate open-source Qwen-VL decomposers fine-tuned with SFT and GRPO [22] (Supp. §A), showing that they can run the same downstream VLTR pipeline with competitive but lower decomposition accuracy than GPT-4V, suggesting that VLTR is not tied to a single decomposer.

Metrics. We report content-preservation metrics (Structure Distance, background PSNR/SSIM), edit-fidelity metrics (CLIP-Text similarity, ImageReward, Aesthetic score), and an overall VLM-based ranking weighted by instruction

Table 1: VLTR tool inventory. The 26 core tools used by the Bayes-UCB router; each primitive indexes 3–6 matching tools. Supporting infrastructure and quality-assessment models are listed in the supplementary.

Tool	Function
Generation backbones (6)	
FLUX.1-dev	Text-to-image
Stable Diffusion 3	Text-to-image
SDXL	Text-to-image
PixArt-α	Text-to-image
BoxDiff	Layout-to-image
LMD	Layout-to-image
Object-level editing (6)	
FLUX.1-Fill	Object addition / fill
AnyDoor	Object insertion
FLUX.1-Kontext	Object removal / replace
LaMa	Object removal
SDXL-Inpainting	Local object editing
PowerPaint V2	Local object editing
Attribute-level editing (6)	
InstructPix2Pix	Attribute editing
DiffEdit	Attribute editing
IC-Light	Color / illumination
ControlNet-Recolor	Color editing
DreamMat	Material editing
ControlNet-Pose	Pose editing
Image-level editing (3)	
LayerDiffusion	Background editing
InST	Style transfer
InstantStyle	Style transfer
Specialized editing (5)	
BrushNet	Precise region control
MagicBrush	Instruction editing
TextDiffuser	Text rendering
BLIP-Diffusion	Single-subject cust.
λ-ECLIPSE	Multi-subject cust.

completeness, semantic correctness, and visual quality. For the calibration analysis we additionally report false accept rate (FAR), false reject rate (FRR), and expected calibration error (ECE).

4.2 Main Results on PIE-Bench++

Table 2 summarizes the primary PIE-Bench++ evaluation. VLTR achieves the best overall VLM ranking (1.8) and leads on five of seven metrics, demonstrating that closed-loop orchestration simultaneously improves both edit fidelity and structural preservation. We highlight three comparison dimensions below.

VLTR vs. end-to-end models. End-to-end editors trade structural integrity for global coherence: FLUX.1-Kontext-Dev, the strongest, incurs Structure Distance 101.5 (3.6× VLTR's 28.5) for lack of region isolation. VLTR's mask-aware execution instead yields the best BG-PSNR (25.20) and BG-SSIM (0.855) while improving CLIP-T to 0.672, removing the preservation–fidelity trade-off.

VLTR vs. multi-tool systems. Among multi-tool methods, GenArtist preserves structure (SD 29.0) but sacrifices fidelity (Reward 0.374); VLTR matches its preservation (SD 28.5) with 2.0× the Reward. RPG maximizes aesthetics (Reward 0.805) but wrecks structure (SD 154.0); VLTR attains 94% of its Reward at only 19% of its Structure Distance. SmartEdit is balanced but trails VLTR on all preservation metrics and on VLM Rank (2.9 vs. 1.8).

Table 2: Quantitative evaluation on PIE-Bench++.

Method	Preservation			Fidelity			Overall
	SD↓	BG-PSNR↑	BG-SSIM↑	CLIP-T↑	Reward↑	Aesthetic↑	VLM Rank↓
End-to-End Models							
InstructPix2Pix	64.4	19.57	0.766	0.656	0.544	6.232	5.3
SDXL-Inpainting	78.8	18.04	0.702	<u>0.663</u>	0.657	6.361	6.2
FLUX.1-Kontext-Dev	101.5	22.27	0.802	0.658	0.721	6.407	3.5
Multi-Tool Systems							
RPG (VisProg)	154.0	12.35	0.588	0.658	**0.805**	**6.873**	3.8
SmartEdit	71.5	<u>24.99</u>	<u>0.842</u>	0.654	0.612	6.782	<u>2.9</u>
GenArtist	<u>29.0</u>	23.32	0.785	0.640	0.374	6.554	4.5
VLTR (Ours)	**28.5**	**25.20**	**0.855**	**0.672**	<u>0.758</u>	<u>6.785</u>	**1.8**

SD: Structure Distance ($\times 10^3$); BG-PSNR/SSIM: preservation on unedited regions; ↓: lower is better.

VLM-judge preference ranking. The VLM Rank gap between VLTR (1.8) and the next-best method SmartEdit (2.9) is the largest pairwise difference in the table. We treat VLM Rank as *complementary* semantic evidence rather than a human study: the same conclusion is supported by the six GPT-4V-independent metrics in Table 2 (SD, BG-PSNR, BG-SSIM, CLIP-T, ImageReward, Aesthetic), on which VLTR also leads or ties. This advantage arises because VLTR's uncertainty-aware scheduling catches partial failures—high CLIP-T but incomplete execution—that a quality-only assessment would accept.

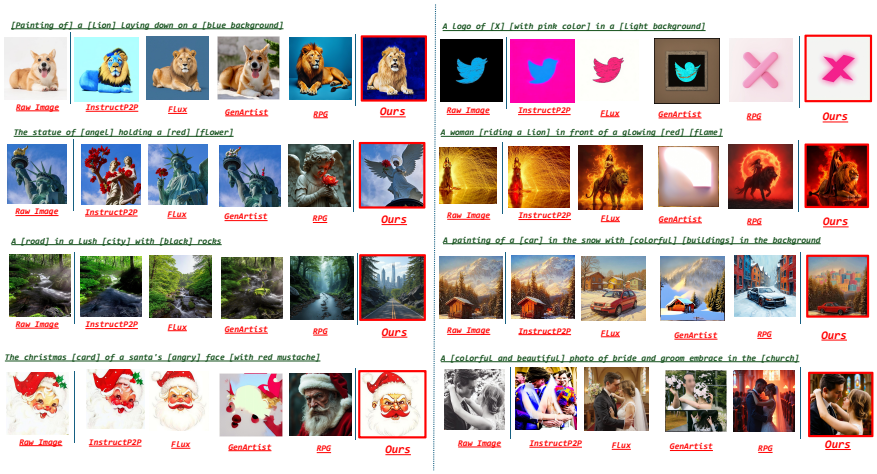

Fig. 3: Qualitative comparison on PIE-Bench++. Eight representative tasks spanning object transformation, attribute modification, background replacement, and style transfer. VLTR (rightmost, red box) consistently produces more faithful edits while better preserving unedited regions.

Qualitative analysis. Figure 3 presents visual comparisons across eight representative editing tasks. We highlight two cases that expose systematic failure modes of existing methods.

Multi-attribute coordination ("painting of a lion on a blue background"): baselines each miss a constraint—InstructPix2Pix distorts the subject, FLUX.1-Kontext-Dev misses the painting style, and RPG drops the blue background. VLTR decomposes the instruction into three independently verified primitives (subject replacement, style transfer, background fill) and satisfies all three.

Abstract semantic preservation ("colorful and beautiful photo of bride and groom in the church"): end-to-end models oversaturate (InstructPix2Pix) or hallucinate elements (FLUX.1-Kontext-Dev), and GenArtist leaves the image largely unchanged. VLTR's scheduler flags the high inter-tier disagreement and triggers targeted refinement that enhances richness without structural degradation.

4.3 Cross-Dataset Validation

To verify that the gains observed on PIE-Bench++ are not benchmark-specific, we evaluate VLTR on three additional datasets that differ in task format, difficulty, and domain distribution.

MagicBrush. Table 3 reports results on MagicBrush [32], a multi-turn editing benchmark requiring iterative refinement across consecutive instructions. VLTR achieves the highest CLIP-T (0.669) and ImageReward (0.752) while using only 5.7 tool switches—roughly half of RPG's 10.8 and 38% fewer than GenArtist's 9.2—demonstrating that the closed-loop reasoning loop maintains quality advantages even under distribution shift. We report aggregate MagicBrush results and do not stratify by turn count; accordingly we do not claim a uniform advantage on longer sequences, and long-horizon planning remains future work.

Table 3: Evaluation on MagicBrush.

Method	CLIP-T↑	Reward↑	Switch↓
SDXL-Inpainting	0.613	0.632	–
InstructPix2Pix	0.642	0.523	–
GenArtist	0.628	0.382	9.2
RPG	0.654	<u>0.748</u>	10.8
VLTR (Ours)	**0.669**	**0.752**	**5.7**

Table 4: Results on GEdit-Bench and RISEBench.

Benchmark	Metric	Base.	VLTR
GEdit-Bench	G_{SC} ↑	7.03	**7.69**
	G_{PQ} ↑	6.72	**6.95**
RISEBench	Causal%↑	5.3	**8.8**
	Spatial%↑	13.0	**19.0**

†GEdit: Step1X-Edit [13]

‡RISE: FLUX.1-Kontext-Dev [12]

Figure 4 shows representative multi-turn cases: VLTR's subgraph-level backtracking preserves cross-turn identity coherence where feedforward baselines accumulate drift across rounds.

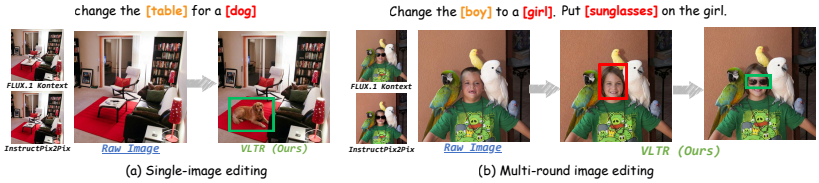

Fig. 4: Qualitative comparison on MagicBrush. (a) Single-image editing: "change the table for a dog." (b) Multi-round editing: "change the boy to a girl; put sunglasses on the girl." Red boxes highlight failure regions; green boxes indicate successful edits.

GEdit-Bench and RISEBench. We further assess out-of-domain robustness on GEdit-Bench [13] and RISEBench [35], which target challenging generative edits and reasoning-informed causal transformations, respectively. Since neither benchmark defines a fixed evaluation split, we randomly sample 200 images from each and apply the same inference pipeline used for PIE-Bench++ without any task-specific adaptation. For fair comparison, we re-evaluate Step1X-Edit [13] on GEdit-Bench and FLUX.1-Kontext-Dev on RISEBench using the identical 200-sample splits. As shown in Table 4, VLTR improves G_{SC} by 9.4% and G_{PQ} by 3.4% over Step1X-Edit on GEdit-Bench, and raises Causal % from 5.3 to 8.8 (+3.5 pp) and Spatial % from 13.0 to 19.0 (+6.0 pp) over FLUX.1-Kontext-Dev [12] on RISEBench, confirming that the structured decompose–verify–reroute loop transfers effectively to harder, out-of-distribution editing scenarios.

4.4 Component Ablation

Table 5: Progressive component integration on PIE-Bench++.

Variant	Preservation		Fidelity		
	SD↓	BG-PSNR↑	CLIP-T↑	Reward↑	VLM Rank↓
V1: Random Baseline	142.3	15.82	0.639	0.346	5.2
V2: + Decomposition	76.6	22.68	0.650	0.515	3.8
V3: + Verifier	65.2	22.85	0.651	0.521	3.2
V4: + Scheduler & Router	**28.5**	**25.20**	**0.672**	**0.758**	**1.8**

Table 5 isolates the contribution of each component through progressive integration.

Decomposition is foundational (V1→V2). Structured instruction parsing provides the single largest improvement: SD drops 46% (142.3→76.6) and BG-PSNR rises 43%, because mask-guided editing confines modifications to targeted regions. Without decomposition, no downstream component can compensate.

The verifier eliminates catastrophic failures (V2→V3). Inline verification further reduces SD by 15% and improves VLM Rank from 3.8 to 3.2. Notably, CLIP-T improves by only 0.1%—the verifier's primary value is not raising the ceiling but removing the floor: it detects low-quality outputs and triggers tool switches before errors propagate through the DAG.

The scheduler and router deliver the largest final gain (V3→V4). Combining Bayes-UCB routing with uncertainty-aware scheduling produces a dramatic leap: SD plummets 56% (65.2→28.5), reaching parity with GenArtist; ImageReward surges 45%; and VLM Rank improves from 3.2 to 1.8, surpassing all external baselines. This confirms that adaptive tool selection and joint quality–uncertainty scheduling create synergistic effects—the online learning loop turns per-step verifier signals into progressively better routing decisions. V4 matches GenArtist's preservation (SD 28.5 vs. 29.0) while far exceeding its fidelity (CLIP-T +5%, Reward 2×), showing that adaptive orchestration need not sacrifice ambition for safety.

Router strategy. Within V4, Bayes-UCB achieves the highest average reward (0.8245) among four routing strategies while maintaining the lowest cross-seed variance (0.079 vs. 0.156 for random).

4.5 Uncertainty Calibration and Decision Making

This section establishes that VLTR's uncertainty estimates are well-calibrated, reflect genuine task difficulty, and lead to better scheduling decisions.

Table 6: Calibration of the heteroscedastic verifier. Results on the 400-case validation subset.

Method	FAR↓	FRR↓	Intervene↓	Quality↑	ECE↓
Fixed-Variance	11.8%	8.2%	2.8	0.76	0.237
Quality-Only	17.3%	3.8%	1.4	0.72	0.288
Variance-Only	10.2%	7.1%	2.5	0.77	0.186
VLM-Only	9.4%	5.9%	2.2	0.78	0.154
VLTR-Hetero	**7.3%**	**4.6%**	**2.1**	**0.81**	**0.142**

Verifier design comparison. Table 6 compares five verification strategies. Quality-Only (no uncertainty) has the highest FAR (17.3%), accepting outputs where metrics superficially agree but execution is incomplete. Fixed-Variance reduces FAR to 11.8% but applies a uniform threshold across all tasks. VLM-Only improves further (FAR 9.4%, ECE 0.154) but lacks multi-source precision. Our full heteroscedastic verifier achieves FAR of 7.3% (38% reduction vs. Fixed-Variance) and ECE of 0.142 (40% reduction), confirming that input-dependent uncertainty enables genuinely calibrated quality assessment.

The learned uncertainty is not arbitrary. On the validation set, σ^2 correlates with prompt complexity ($r = 0.68$), mask coverage ($r = 0.54$), and inter-expert

disagreement ($r = 0.72$), assigning higher uncertainty to genuinely harder edits. It also exhibits intuitive category-level structure: color adjustments yield mean $\sigma_2^2 = 0.008$ and FAR of 3.2%, while scene composition—the most complex category—shows $\sigma_2^2 = 0.035$ and FAR of 9.8%. This 4.4× variance difference translates to a 3× FAR difference, demonstrating task-adaptive thresholding.

The scheduler thresholds follow the Bayesian risk criterion in Eq. 10. A validation grid search selects $(\tau_q, \tau_u) = (0.6, 0.15)$, yielding 82% success versus 79% for a stricter $\tau_q = 0.7$; outputs that violate the risk rule exhibit 3.7× higher failure rates. On the routing side, Bayes-UCB is stable across five seeds (std 0.079) compared with random selection (0.156).

Figure 5 visualizes the same trend at the per-step level: success degrades monotonically as σ^2 increases, and the high-uncertainty regime ($\sigma^2 > 0.15$) contains 26% of sampled outputs but 68% of failures. This pattern motivates joint quality–uncertainty decisions instead of quality-only thresholds.

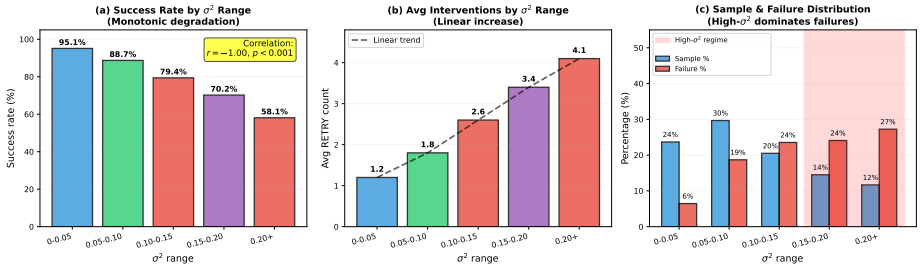

Fig. 5: Confidence-stratified analysis. Higher uncertainty is strongly associated with lower success rates and more interventions.

From calibration to decisions. Well-calibrated uncertainty is only useful if it leads to better scheduling outcomes. We test this by comparing the full scheduler against a quality-only variant on 100 difficult PIE-Bench++ cases whose first execution either has low quality or high uncertainty.

Table 7: Decision effectiveness across (q, σ^2) regimes.

Regime	N	Quality-only	VLTR	Δ
HQ-LU	28	96%	96%	+0pp
HQ-HU	23	65%	87%	+22pp
LQ-HU	31	48%	77%	+29pp
LQ-LU	18	33%	61%	+28pp
Overall	100	63%	82%	+19pp

Table 7 reveals that the largest gains occur in the ambiguous regimes. **Row 2 (HQ-HU):** These 23 cases have acceptable quality ($q \geq 0.6$) but high inter-tier

disagreement ($\sigma^2 > 0.15$), where CLIP appears good but VLM detects incomplete execution. Quality-only accepts many of these (65%), while VLTR triggers targeted RETRY and reaches 87% (+22pp). **Row 3 (LQ-HU):** Low quality with high uncertainty indicates tool mismatch; VLTR's REROUTE outperforms blind retry by 29pp. **Row 4 (LQ-LU):** Low quality with low uncertainty indicates primitive-level failure; REPLAN outperforms parameter-only tuning by 28pp.

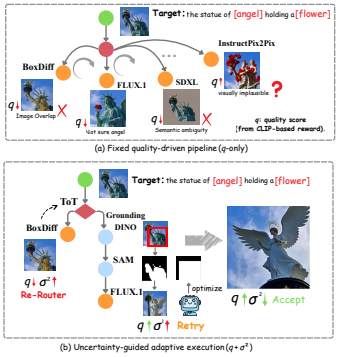

Fig. 6: Comparison between quality-only and uncertainty-guided schedulers in VLTR.

Figure 6 illustrates this on a concrete case. For the angel statue edit, BoxDiff produces a visually plausible output that receives an acceptable CLIP score, but the three verifier tiers disagree substantially—CLIP approves while VLM detects missing details. The quality-only scheduler false-accepts this output, whereas VLTR classifies it as UNCERTAIN, reroutes through GroundingDINO and SAM for precise localization, and achieves verified success after parameter refinement with FLUX.1-Kontext-Dev.

λ **sensitivity.** The inter-tier disagreement weight λ in Eq. 9 controls the balance between intra-tier and inter-tier uncertainty. Performance remains within $\pm 2\%$ for $\lambda \in [0.3, 0.7]$, with the optimum at $\lambda = 0.5$ (ECE $= 0.142$). At $\lambda = 0$, ECE rises to 0.183 due to under-estimated uncertainty when tiers conflict; at $\lambda = 1.0$, the system becomes over-conservative (ECE $= 0.172$, 30% RETRY rate).

4.6 Compute and Cost-Benefit Analysis

Here *best-of-5* draws five independent samples from FLUX.1-Kontext-Dev and reports the candidate maximizing the shared verifier score, matching VLTR's tool-call budget (5.0 vs. 5.7 calls). Under comparable compute (Table 8), VLTR reaches essentially the same CLIP-T as this best-of-5 baseline while avoiding independent brute-force sampling. Together with the preservation and VLM-rank gains in Table 2, this indicates that VLTR's improvements come from decomposition, verification, and targeted rerouting rather than simply evaluating many more candidates.

Table 8: Compute comparison on PIE-Bench++. VLTR reports the standard setting from Table 2; baselines use the same verifier for candidate selection.

Method	CLIP-T↑	Tool Calls↓	Iters↓
FLUX.1-Kontext-Dev (Single-pass)	0.658	1.0	1.0
FLUX.1-Kontext-Dev + Best-of-5	0.671	5.0	1.0
VLTR (Ours)	0.672	5.7	2.4

Table 9: Efficiency–quality profile on MagicBrush.

Method	CLIP-T↑	ImageReward↑	Tool Switch↓	GPU Avg. (MB)↓	Time (s)↓
SDXL-Inpainting	0.613	0.632	–	11620	4.9
InstructPix2Pix	0.642	0.523	–	5410	42.3
GenArtist	0.628	0.382	9.2	1340	121.7
RPG	0.654	0.748	10.8	1090	466.9
VLTR (Ours)	**0.669**	**0.752**	**5.7**	**3525**	**97.4**

Table 9 further clarifies the efficiency–quality trade-off on MagicBrush. VLTR achieves the best CLIP-T and ImageReward among multi-tool systems while reducing tool switches to 5.7 (vs. 10.8 for RPG and 9.2 for GenArtist), indicating that higher quality is not obtained through excessive tool hopping.

The time metric highlights the same point. Measured on a single Tesla V100 (32GB), VLTR performs multi-step reasoning yet remains markedly faster than prior multi-tool systems (97.4s vs. 466.9s for RPG and 121.7s for GenArtist) because its deliberation is not purely serial: topologically independent primitives can be processed in parallel, and DAG-level local repair reuses validated intermediate states instead of restarting the whole pipeline. End-to-end editors are still faster in absolute latency, but they achieve this by giving up robustness and edit fidelity under multi-turn instructions.

5 Conclusion

We presented VLTR, a training-free framework that reformulates instruction-guided image editing as closed-loop tool reasoning over executable primitive states. Its central idea is to make uncertainty actionable: the Bayes-UCB router uses it for exploration, the heteroscedastic verifier uses it for calibrated quality estimation, and the verifier-guided scheduler uses it to decide when to retry, reroute, or replan. Combined with a DAG representation, this yields adaptive local repair instead of full-pipeline restart.

Experiments on PIE-Bench++, MagicBrush, GEdit-Bench, and RISEBench show that VLTR improves edit fidelity and preservation while remaining compute-efficient, achieving the best overall VLM ranking on PIE-Bench++ and a 4.8× speedup over the strongest multi-tool competitor on MagicBrush. More broadly, closed-loop tool reasoning turns local verification signals into reliable execution and can extend to other editing and generation domains.

References

1. Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Gianinazzi, L., Gajda, J., Lehmann, T., Podstawski, M., Niewiadomski, H., Nyczyk, P., Hoefler, T.: Graph of thoughts: Solving elaborate problems with large language models. Proceedings of the AAAI Conference on Artificial Intelligence **38**, 17682–17690 (2024)
2. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 18392–18402 (2023)
3. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Rombach, R.: Scaling rectified flow transformers for high-resolution image synthesis. In: Proceedings of the 41st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 235, pp. 12606–12633. PMLR (2024)
4. Feng, X., Wan, Z., Wen, M., McAleer, S., Wen, Y., Zhang, W., Wang, J.: Alphazero-like tree-search can guide large language model decoding and training. arXiv preprint arXiv:2309.17179 (2023)
5. Fu, T.J., Hu, W., Du, X., Wang, W.Y., Yang, Y., Gan, Z.: Guiding instruction-based image editing via multimodal large language models. In: The Twelfth International Conference on Learning Representations (2024)
6. Gupta, T., Kembhavi, A.: Visual programming: Compositional visual reasoning without training. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 14953–14962 (2023)
7. Huang, M., Cai, J., Jia, S., Lokhande, V.S., Lyu, S.: Paralleledits: Efficient multi-aspect text-driven image editing with attention grouping. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 37 (2024)
8. Huang, Y., Xie, L., Wang, X., Yuan, Z., Shan, Y., Chen, C.: Smartedit: Exploring complex instruction-based image editing with multimodal large language models. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2024)
9. Ju, X., Zeng, A., Bian, Y., Liu, S., Xu, Q.: Direct inversion: Boosting diffusion-based editing with 3 lines of code. arXiv preprint arXiv:2310.01506 (2023)
10. Kaufmann, E., Cappé, O., Garivier, A.: On bayesian upper confidence bounds for bandit problems. In: Artificial intelligence and statistics. pp. 592–600. PMLR (2012)
11. Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., Irani, M.: Imagic: Text-based real image editing with diffusion models. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 6007–6017 (2023)
12. Labs, B.F.: Flux. https://github.com/black-forest-labs/flux (2024), accessed: 2026-06-30
13. Liu, S., Han, Y., Xing, P., Wu, C., Huang, J., Fu, S., et al.: Step1x-edit: A practical framework for general image editing. arXiv preprint arXiv:2504.17761 (2025)
14. Long, J.: Large language model guided tree-of-thought. arXiv preprint arXiv:2305.08291 (2023)
15. Mittal, A., Soundararajan, R., Bovik, A.C.: Making a "completely blind" image quality analyzer. IEEE Signal Processing Letters **20**(3), 209–212 (2013), core NIQE paper - Natural Image Quality Evaluator
16. Mo, S., Xin, M.: Tree of uncertain thoughts reasoning for large language models. 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) pp. 12742–12746 (2024)

17. OpenAI: Gpt-4v(ision) system card. https://openai.com/index/gpt-4v-system-card/ (2023), accessed: 2026-06-30
18. Parmar, G., Singh, K.K., Zhang, R., Li, Y., Lu, J., Zhu, J.Y.: Zero-shot image-to-image translation. In: ACM SIGGRAPH 2023 Conference Proceedings (2023)
19. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023)
20. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. International conference on machine learning pp. 8748–8763 (2021)
21. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition pp. 10684–10695 (2022)
22. Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Zhang, M., Li, Y., Wu, Y., Guo, D.: Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300 (2024)
23. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., Zhuang, Y.: Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. Advances in Neural Information Processing Systems **36** (2023)
24. Singh, C., Morris, J., Rush, A.M., Gao, J., Deng, Y.: Tree prompting: Efficient task adaptation without fine-tuning. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. pp. 6253–6267 (2023)
25. Wang, Z., Li, A., Li, Z., Liu, X.: Genartist: Multimodal llm as an agent for unified image generation and editing. Advances in Neural Information Processing Systems **37**, 128374–128395 (2024)
26. Wu, C., Yin, S., Qi, W., Wang, X., Tang, Z., Duan, N.: Visual chatgpt: Talking, drawing and editing with visual foundation models. arXiv preprint arXiv:2303.04671 (2023)
27. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation. Advances in Neural Information Processing Systems **36** (2023)
28. Xu, Y., Kong, J., Wang, J., Pan, X., Lin, B., Liu, Q.: Insightedit: Towards better instruction following for image editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2694–2703 (2025)
29. Yang, L., Yu, Z., Meng, C., Xu, M., Ermon, S., Cui, B.: Mastering text-to-image diffusion: Recaptioning, planning, and generating with multimodal llms. In: Forty-first International Conference on Machine Learning (2024)
30. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T.L., Cao, Y., Narasimhan, K.: Tree of thoughts: Deliberate problem solving with large language models. Advances in Neural Information Processing Systems **36** (2023)
31. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y.: React: Synergizing reasoning and acting in language models. International Conference on Learning Representations (2023)
32. Zhang, K., Mo, L., Chen, W., Sun, H., Su, Y.: Magicbrush: A manually annotated dataset for instruction-guided image editing. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 36, pp. 28808–28826 (2023)
33. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. Proceedings of the IEEE/CVF International Conference on Computer Vision pp. 3836–3847 (2023)

34. Zhang, Z., Han, L., Ghosh, A., Metaxas, D.N., Ren, J.: Sine: Single image editing with text-to-image diffusion models. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 6027–6037 (2023)
35. Zhao, X., Zhang, P., Tang, K., Zhu, X., Li, H., Chai, W., Zhang, Z., Xia, R., Zhai, G., Yan, J., Yang, H., Yang, X., Duan, H.: Envisioning beyond the pixels: Benchmarking reasoning-informed visual editing. arXiv preprint arXiv:2504.02826 (2025)
36. Zhou, A., Yan, K., Shlapentokh-Rothman, M., Wang, H., Wang, Y.X.: Language agent tree search unifies reasoning acting and planning in language models. arXiv preprint arXiv:2310.04406 (2023)
37. Zong, Z., Ma, B., Shen, D., Song, G., Shao, H., Jiang, D., Li, H., Liu, Y.: Mova: Adapting mixture of vision experts to multimodal context. Advances in Neural Information Processing Systems **37**, 103305–103333 (2024)