EVOGEN-HARNESS: LEARNING WHERE AND HOW TO EVOLVE IMAGE-GENERATION HARNESSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern text-to-image (T2I) systems can be improved without modifying generator parameters by adapting the external system around frozen generators. However, existing approaches typically optimize a predefined dimension, such as prompts, routing, or workflows, restricting the space in which generation failures can be corrected. Allowing multiple generator-external responsibilities to evolve provides a broader adaptation space, but introduces a new challenge: visual feedback reveals *what failed*, but not *where* persistent evolution should occur or *how* this space should be explored efficiently. We introduce EVOGEN-HARNESS, a generator-agnostic framework for multi-responsibility image-generation harness evolution, together with TRACE (**Tra**jectory-Relative Attribution and **C**oordinated **E**volution). TRACE aggregates evidence across stochastic executions, uses failure attribution as a search prior to focus candidate updates, and progressively reattributes residual failures to coordinate evolution across responsibilities, while NO-PATCH and held-out validation prevent unnecessary or harmful updates. Across GenEval2, T2I-CompBench++, and WISE, EVOGEN-HARNESS improves over the strongest evaluated baselines by **+0.2633**, **+0.0720**, and **+0.0752**, respectively, while achieving **87.9–91.4%** attribution recall, **94.8%** NO-PATCH accuracy, and only **1.9%** regression. These results demonstrate that attribution-guided multi-responsibility evolution can substantially enhance frozen T2I systems beyond single-dimension adaptation.

1 INTRODUCTION

Text-to-image (T2I) generation has advanced rapidly, yet even strong diffusion and flow-based models still struggle with compositional requirements such as object counting, attribute binding, spatial relations, and multi-object interactions (Rombach et al., 2022; Esser et al., 2024; Ghosh et al., 2023; Huang et al., 2025; Cho et al., 2024). Recent studies show that these failures can be mitigated without modifying the generator itself. RePrompt and VisualPrompter improve prompt construction through reasoning and visual feedback (Wu et al., 2026a;b); DiffAgent and GenArtist coordinate generators and external tools (Zhao et al., 2024; Wang et al., 2024a); and OctoT2I learns tool knowledge for multi-round generator routing (Jiang et al., 2026). More generally, harness engineering suggests that persistent model-external components can themselves become objects of optimization (Lin et al., 2026). Together, these advances establish generator-external adaptation as a promising way to improve frozen T2I systems.

However, existing approaches largely optimize a predefined adaptation dimension, or a small fixed subset of them. Prompt-based methods rewrite textual inputs, routing methods reconsider generators, while workflow- and skill-based methods modify predefined procedures. Such specialization can be effective, but also restricts the space in which a generation failure can be corrected. For example, repeated prompt refinement cannot directly repair inaccurate tool knowledge or a flawed reusable procedure, while generator switching cannot correct misleading long-term experience. This motivates a broader view: instead of committing to one adaptation dimension in advance, an image-generation agent should be able to improve across multiple persistent responsibilities.

Broadening the adaptation space, however, introduces a new challenge: *where should evolution occur, and how should this larger space be explored efficiently?* As illustrated in Fig. 1, consider a request for a fox on a bicycle followed by *three white ducks*, while only two ducks are generated. The

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



(a) Prior methods optimize predefined adaptation targets. **(b) Same failure, different adaptation decisions.**

Figure 1: **Motivation.** Prior image-generation methods typically optimize a predefined adaptation dimension, restricting the available correction space. Allowing multiple harness responsibilities to evolve provides a broader adaptation space, but introduces a new attribution problem: the same visual failure may require different persistent changes, or no update at all.

same visible failure may arise because the count requirement is represented incorrectly, the system holds inaccurate capability knowledge, a reusable procedure or run-time decision is inadequate, previous experience is misleading, or simply because stochastic generation produced an unlucky sample. Thus, visual feedback reveals *what failed*, but not necessarily *what should change*. Trying all possible harness modifications would be both expensive and unreliable. Multi-responsibility evolution therefore requires both *automatic attribution* and *guided search*, rather than brute-force trial and error.

To address this challenge, we introduce EVOGEN-HARNESS, a generator-agnostic framework for multi-responsibility evolution around frozen image generators. We operationalize persistent generator-external decisions through five functional responsibilities: **Policy** for visual requirements, **Tools** for available capabilities, **Skills** for reusable procedures, **Middleware** for run-time orchestration, and **Memory** for cross-task experience. These responsibilities define a practical adaptation space rather than an exhaustive taxonomy. To evolve this space efficiently, we introduce TRACE (**Tra**jectory-Relative Attribution and **C**oordinated **E**volution). TRACE aggregates constraint-level evidence across stochastic executions, attributes failures to plausible responsibilities, and uses this attribution as a search prior to prune unlikely edits. Candidate updates are evaluated relative to the current harness under matched generation conditions; after each accepted update, residual failures are re-attributed, allowing evolution to progressively move across responsibilities. NO-PATCH avoids unnecessary updates, while held-out validation prevents harmful changes from persisting.

We evaluate EVOGEN-HARNESS on three complementary T2I benchmarks, where it consistently improves over the strongest evaluated baselines, and further analyze attribution, preservation, efficiency, and cross-generator generalization.

Our contributions are summarized as follows:

- We propose EVOGEN-HARNESS, a **generator-agnostic framework for multi-responsibility image-generation harness evolution**. By exposing multiple persistent generator-external responsibilities for adaptation, it moves beyond optimization confined to a predefined dimension while keeping the underlying image generator frozen.
- We introduce **TRACE**, a trajectory-relative attribution and coordinated evolution mechanism. TRACE uses LLM-based harness attribution as a search prior, prunes the heterogeneous edit space, evaluates candidate updates under matched stochastic conditions, and progressively re-attributes residual failures to enable efficient cross-responsibility evolution.
- Extensive experiments show that EVOGEN-HARNESS improves over the strongest evaluated baselines by **+0.2633**, **+0.0720**, and **+0.0752** on GenEval2, T2I-CompBench++, and WISE, respectively. It further achieves **87.9–91.4%** attribution recall, **94.8%** NO-PATCH accuracy, and only **1.9%** regression, while generalizing across frozen T2I backbones.

2 Related Work

2.1 Prompt Optimization and Visual Feedback

Prompt optimization improves T2I generation while keeping the generator fixed. Early methods learn or search for model-preferred prompts (Hao et al., 2023; Cao et al., 2023), while recent approaches increasingly exploit visual feedback. RePrompt learns reasoning-augmented prompt refinement from image-level rewards (Wu et al., 2026a), VisualPrompter performs targeted refinement from missing visual concepts (Wu et al., 2026b), and IterComp learns composition-aware feedback for iterative improvement (Zhang et al., 2025). Iterative self-correction is also explored in prior work (Wu et al., 2024; Yang et al., 2024b). These methods demonstrate the value of visual feedback, but mainly optimize a predefined target—the prompt or current generation. In contrast, EvoGen-Harness determines which persistent generator-external responsibility should evolve.

2.2 Agentic and Self-Evolving Image Generation

Agentic T2I systems broaden adaptation beyond prompts through planning, tools, routing, and reusable experience. LayoutGPT and LLM-grounded Diffusion use LLMs for explicit visual or layout planning (Feng et al., 2023; Lian et al., 2023), while RPG, CompAgent, Ranni, Marmot, MCCD, and M3 improve complex generation through regional planning, decomposition, or multi-agent correction (Yang et al., 2024a; Wang et al., 2024b; Feng et al., 2024; Sun et al., 2025; Li et al., 2025; Yang et al., 2026). Other agentic systems perform generator selection, tool use, and stateful routing (Zhao et al., 2024; Wang et al., 2024a; Jiang et al., 2026), while recent methods evolve workflows, skills, or visual experience (Chen et al., 2026a; Li et al., 2026; Chen et al., 2026b). Despite increasing adaptivity, these approaches generally predefine *what* is improved. Our work instead determines both *where* and *how* persistent evolution should proceed.

2.3 Agent Harness Evolution

Recent work extends self-improvement to the model-external harness. Agentic Harness Engineering evolves heterogeneous harness components through observability-driven feedback (Lin et al., 2026), while Self-Harness improves operating harnesses through weakness mining, modification, and regression validation (Zhang et al., 2026b). JIT-Agent further synthesizes and self-evolves task-adaptive harnesses on demand (Zhang et al., 2026a). These studies establish harness evolution as a general agentic paradigm. EvoGen-Harness focuses on stochastic image generation, where TRACE aggregates trajectory-relative visual evidence, guides search over multiple responsibilities, and progressively re-attributes residual failures to coordinate their evolution.

3 Method

We present EvoGen-Harness, a framework that enables frozen image-generation systems to evolve through a persistent external harness. Instead of modifying generator parameters, TRACE (**Tra**jectory-Relative Attribution and **C**oordinated **E**volution) identifies which harness component should evolve from visual execution evidence and iteratively refines it through feedback-driven updates.

3.1 Multi-Responsibility Visual Harness

Motivated by recent advances in self-evolving image generation, we identify five persistent responsibilities that define the evolution space: **Policy**, **Tools**, **Skills**, **Middleware**, and **Memory**.

At evolution round t, we represent the persistent generator-external state as

$$\mathcal{H}_t = \big(\Pi_t,\, \mathcal{T}_t,\, \mathcal{S}_t,\, \mathcal{W}_t,\, \mathcal{M}_t\big), \tag{1}$$

where **Policy** Π specifies visual requirements, **Tools** $\mathcal{T}$ describe available capabilities, **Skills** $\mathcal{S}$ encode reusable procedures, **Middleware** $\mathcal{W}$ controls run-time orchestration, and **Memory** $\mathcal{M}$ stores cross-task experience. We denote the editable responsibilities by $\mathcal{R} = \{\Pi, \mathcal{T}, \mathcal{S}, \mathcal{W}, \mathcal{M}\}$. Together, they define what the system should achieve, what capabilities it can use, how generation should be executed, and what experience should persist.

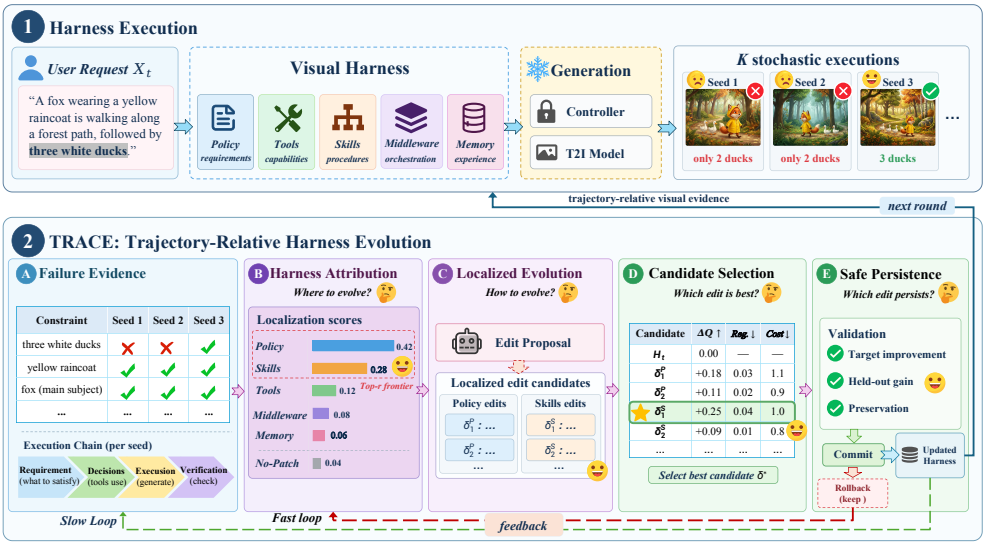


Figure 2: **Overview of EvoGen-Harness.** The image generator remains frozen while its persistent generator-external harness evolves. TRACE aggregates trajectory-relative visual evidence, localizes which harness responsibilities should be explored, proposes responsibility-conditioned edits with a fixed LLM, and retains only updates that pass matched evaluation and held-out validation.

For request x_t, the current harness is executed under K stochastic generation conditions $\{\omega_k\}_{k=1}^{K}$:

$$\left(I_t^{(k)}, \tau_t^{(k)}\right) = \text{Execute}\left(x_t; \mathcal{H}_t, \mathcal{G}, \omega_k\right), \qquad k = 1, \ldots, K, \tag{2}$$

where $\mathcal{G}$ is the frozen generation system, $I_t^{(k)}$ is the generated image, and $\tau_t^{(k)}$ records the harness decisions and executions that produced it. The stochastic condition ω_k captures generation randomness, such as a random seed.

3.2 Trajectory-Relative Harness Attribution

A single failed image is insufficient evidence for persistent evolution: the same harness may succeed under another stochastic execution. We therefore decompose x_t into fine-grained visual constraints $\mathcal{C}_t = \{c_{t,1}, \ldots, c_{t,m}\}$ and aggregate their outcomes across the K executions. Following structured visual evaluation (Cho et al., 2024), we construct

$$\mathcal{E}_t = \left\{ \left(c_{t,j}, \tau_t^{(k)} \downarrow c_{t,j}, v_{t,j}^{(k)} \right) \right\}_{\substack{j=1,\ldots,m \\ k=1,\ldots,K}}, \tag{3}$$

where $\tau_t^{(k)} \downarrow c_{t,j}$ denotes the trajectory decisions relevant to constraint $c_{t,j}$, and $v_{t,j}^{(k)}$ is its visual verification result. Thus, $\mathcal{E}_t$ connects each visual requirement to the decisions and outcomes that may explain its success or failure.

LLM-based Harness Localizer. Given $\mathcal{E}_t$ and the current harness, a fixed LLM serves as the *Harness Localizer*. Using a structured localization prompt, it assigns a score $s_{\text{loc}}(z \mid \mathcal{E}_t, \mathcal{H}_t)$ to each $z \in \mathcal{R} \cup \{\varnothing\}$, where $\varnothing$ denotes NO-PATCH. Rather than committing immediately to one responsibility, TRACE retains a compact search frontier:

$$\mathcal{R}_t^{(r)} = \underset{z \in \mathcal{R}}{\text{Top-}r}\ s_{\text{loc}}\left(z \mid \mathcal{E}_t, \mathcal{H}_t\right), \tag{4}$$

where r controls the number of plausible responsibilities retained. NO-PATCH is selected when the evidence is better explained by generation stochasticity, verifier uncertainty, or insufficient support for a persistent update.

The Harness Localizer remains fixed throughout evolution and requires no additional training. Controlled single-responsibility interventions are used only to evaluate localization accuracy in Sec. 4.4. Visual verification therefore identifies *what failed*, while harness localization determines *where evolution should be explored*.

3.3 TRACE: ATTRIBUTION-GUIDED PROGRESSIVE EVOLUTION

Starting from $\mathcal{H}^{(0)} = \mathcal{H}_t$, TRACE explores only the responsibilities retained by the Harness Localizer. For each responsibility z, let Δ_z contain at most M localized candidate edits:

$$\Delta_\ell = \bigcup_{z \in \mathcal{R}_\ell^{(r)}} \Delta_z, \qquad \Delta_z = \{\delta_{z,1}, \ldots, \delta_{z,M}\}, \tag{5}$$

where $\delta_{z,i}$ is the i-th candidate edit associated with responsibility z. Depending on the selected responsibility, an edit may revise requirement interpretation, capability knowledge, reusable procedures, run-time orchestration, or persistent experience.

Responsibility-conditioned proposal. A fixed LLM serves as the edit proposer. It receives the current harness state, trajectory-relative evidence, and a responsibility-specific edit schema, and generates Δ_z while being restricted to responsibility z. The localizer and proposer may share the same underlying LLM but use separate prompts and structured output schemas. Thus, localization determines *where* to evolve, whereas proposal determines *how* that responsibility may be modified.

Trajectory-relative evaluation. Because image generation is stochastic, independently sampled outputs can confound edit quality with generation variance. Let $Q(x; \mathcal{H}, \omega)$ denote the visual evaluation score obtained by executing harness $\mathcal{H}$ on request x under stochastic condition ω. Each candidate is compared with its parent harness under the same K generation conditions:

$$\begin{aligned} \Delta Q_\ell(\delta) = \frac{1}{K} \sum_{k=1}^{K} \Big[& Q\Big(x_t; \mathcal{H}^{(\ell)} \oplus \delta, \omega_k\Big) \\ & - Q\Big(x_t; \mathcal{H}^{(\ell)}, \omega_k\Big) \Big]. \end{aligned} \tag{6}$$

Here $\oplus$ denotes applying an edit to the current harness. Matched stochastic conditions make $\Delta Q_\ell(\delta)$ reflect the effect of the edit rather than random variation in generation.

Progressive evolution. Within the attribution-pruned search space, TRACE ranks candidate edits by their relative improvement while penalizing preservation regression and additional execution cost:

$$\begin{aligned} U_\ell(\delta) &= \Delta Q_\ell(\delta) - \lambda R_{\text{pres}}(\delta) - \mu C(\delta), \\ \delta_\ell^* &= \arg\max_{\delta \in \Delta_\ell} U_\ell(\delta), \qquad \mathcal{H}^{(\ell+1)} = \mathcal{H}^{(\ell)} \oplus \delta_\ell^*. \end{aligned} \tag{7}$$

Here $R_{\text{pres}}(\delta)$ measures degradation of previously successful behavior, $C(\delta)$ measures additional execution cost, and $\lambda, \mu \geq 0$ control their relative penalties. Attribution therefore guides *where to search*, while execution feedback determines *which edit to prefer*.

After each accepted edit, the updated harness is executed again and residual failures are re-localized. If another persistent weakness remains, TRACE continues from $\mathcal{H}^{(\ell+1)}$. The resulting evolution path

$$\boldsymbol{\pi} = (\delta_0^*, \ldots, \delta_{L-1}^*)$$

may therefore traverse multiple responsibilities, where L is the maximum evolution depth.

Evolution stops when NO-PATCH is selected, no candidate yields positive utility, or the evolution budget is exhausted. The resulting path is retained only if it improves the triggering and held-out cases while keeping preservation regression below tolerance ϵ; otherwise the original harness $\mathcal{H}_t$ is restored.

Overall, TRACE separates four complementary decisions: visual verification determines *what failed*, the Harness Localizer determines *where to evolve*, the LLM proposer determines *how to evolve*, and validation determines *what persists*. See Appendix B.1 for details.

4 EXPERIMENTS

We evaluate EVOGEN-HARNESS in terms of generation quality, harness attribution, and generalization. Beyond measuring performance across diverse T2I benchmarks, we examine whether EVOGEN-HARNESS can correctly localize persistent failures, avoid unnecessary updates, and produce reusable improvements on held-out cases and consistent gains across frozen T2I backbones.

Table 1: **Quantitative evaluation on the GenEval2 benchmark.** We report atom-level Soft-TIFA AM across five visual skills and prompt-level Soft-TIFA GM as the overall score. The best and second-best results are marked in **bold red** and underlined blue, respectively. ↑ indicates higher is better.

Type	Method	Object	Attribute	Count	Position	Verb	Overall↑
T2I Models	FLUX.1-dev	0.8754	0.6734	0.5311	0.3928	0.2150	0.2115
	Janus-Pro	0.9102	0.7213	0.4834	0.5217	0.2056	0.1878
	Flow-GRPO	0.9175	0.7438	0.6032	0.5233	0.2541	0.2198
	Qwen-Image	0.9730	0.8412	0.6857	0.6083	0.3752	0.3483
	Z-Image	0.9755	0.7851	0.6232	0.6041	0.2419	0.3034
	Nano Banana	0.9722	0.8981	0.6830	**0.7044**	**0.4532**	0.4456
Prompt Opt.	NeuroPrompts	0.8433	0.6421	0.5008	0.3712	0.2045	0.1976
	Promptist	0.8533	0.6510	0.5122	0.3819	0.2134	0.2056
	BeautifulPrompt	0.5713	0.4406	0.3567	0.2523	0.1432	0.1134
	VisualPrompter	0.9321	0.7223	0.5614	0.4350	0.2412	0.2552
Agentic	Idea2Img	0.7114	0.4016	0.2450	0.1967	0.0145	0.0586
	GenArtist	0.6598	0.3813	0.2535	0.2516	0.0945	0.0556
	ChatGen	0.5274	0.3045	0.2011	0.1569	0.0269	0.0388
	OctoT2I	0.9429	0.7602	0.5381	0.5506	0.2099	0.2895
	Ours	**0.9799**	**0.9469**	**0.9053**	0.6418	0.3851	**0.7089**

4.1 Experimental Settings

Datasets and metrics. We evaluate on three complementary T2I benchmarks: GenEval2 (Kamath et al., 2025), T2I-CompBench++ (Huang et al., 2025), and WISE (Niu et al., 2026). We follow the official evaluation protocol of each benchmark, reporting Soft-TIFA AM for skill-wise analysis and Soft-TIFA GM for overall performance on GenEval2, the official compositional metrics for T2I-CompBench++, and WiScore for WISE. For TRACE-specific analyses, we additionally report attribution and evolution statistics, whose definitions are provided in Appendix A.1. Efficiency is measured by average wall-clock inference time and the numbers of generator and LLM/VLM calls per prompt.

Baselines. We compare EvoGen-Harness with representative methods from three categories. For T2I generation, we include FLUX.1-dev (Black Forest Labs, 2024), Janus-Pro (Chen et al., 2025), Flow-GRPO (Liu et al., 2026), Qwen-Image (Wu et al., 2025), Z-Image (Cai et al., 2025), and Gemini 2.5 Flash Image (Nano Banana) (Google, 2025). For prompt optimization, we compare with NeuroPrompts (Rosenman et al., 2024), Promptist (Hao et al., 2023), BeautifulPrompt (Cao et al., 2023), and VisualPrompter (Wu et al., 2026b). For agentic image generation, we include Idea2Img (Yang et al., 2024b), GenArtist (Wang et al., 2024a), ChatGen (Jia et al., 2025), and OctoT2I (Jiang et al., 2026). Unless otherwise specified, we evaluate all baselines using their official implementations and recommended configurations.

Implementation details. We use FLUX.1-dev as the frozen image generator, GPT-4.1 for harness localization and edit proposal, and OWLv2-base with NVILA-Lite-2B-Verifier for visual verification. Full settings are provided in Appendix B. All experiments are conducted on 2 NVIDIA A100 (80 GB) GPUs.

4.2 Main Results

Table 1 evaluates whether EvoGen-Harness improves fine-grained instruction following. Our method achieves an overall Soft-TIFA GM score of **0.7089**, outperforming both the strongest standalone T2I model (**0.4456**) and all prompt-optimization and agentic baselines. The gain is particularly large on counting, improving from 0.6857 to **0.9053**. This suggests that multi-responsibility

Table 2: **Quantitative evaluation results on T2I-CompBench++.** The best and second-best results are marked in **bold red** and underlined blue, respectively. ↑ indicates higher is better.

Method	Color	Shape	Texture	2D-Spatial	3D-Spatial	Numeracy	Non-Spatial	Complex	Average↑
T2I Models									
FLUX.1-dev	0.7572	0.5066	0.6300	0.2700	0.3992	0.6165	0.3065	0.3628	0.4811
Janus-Pro	0.5145	0.3323	0.4069	0.1566	0.2753	0.4406	0.3137	0.3806	0.3526
Flow-GRPO	0.8379	0.6130	0.7236	0.5447	0.4471	0.6752	0.3195	0.3741	0.5669
Qwen-Image	0.8392	0.5941	0.7568	0.4512	0.4631	0.7705	0.3126	0.3944	0.5727
Z-Image	0.8413	0.6078	0.7512	0.3822	0.4412	0.6676	0.3181	0.4201	0.5537
Nano Banana	**0.8728**	0.6420	0.7855	0.5057	0.5221	0.7923	0.3254	0.4239	0.6087
Prompt Optimization									
NeuroPrompts	0.6700	0.4650	0.5200	0.1850	0.3750	0.5600	0.2860	0.3150	0.4220
Promptist	0.6936	0.4827	0.5437	0.1997	0.3936	0.5874	0.2912	0.3334	0.4407
BeautifulPrompt	0.5610	0.3940	0.4530	0.1630	0.3080	0.4380	0.3040	0.3100	0.3664
VisualPrompter	0.7700	0.5150	0.6400	0.2850	0.4150	0.6550	0.3100	0.3850	0.4969
Agentic Methods									
Idea2Img	0.4897	0.3442	0.3911	0.1473	0.3407	0.4584	0.2895	0.2945	0.3444
GenArtist	0.4786	0.5492	0.5711	0.4716	0.2895	0.5674	0.2826	0.3723	0.4478
ChatGen	0.3619	0.3062	0.2767	0.1073	0.2641	0.3860	0.2931	0.2644	0.2825
OctoT2I	0.8410	0.6283	0.7259	0.5256	0.3872	0.7320	0.3182	0.3582	0.5646
Ours	0.8671	**0.7120**	**0.7913**	**0.6524**	**0.6125**	**0.8344**	**0.4312**	**0.5451**	**0.6807**

Table 3: **Quantitative evaluation on the WISE benchmark.** We report world-knowledge consistency across six domains and the overall score. The best and second-best results are marked in **bold red** and underlined blue, respectively. ↑ indicates higher is better.

Method	Culture	Time	Space	Biology	Physics	Chemistry	Overall↑
Qwen-Image	0.6275	0.5250	0.5583	0.3417	0.4833	0.2500	0.5100
Nano Banana	0.7000	0.5900	0.6700	0.4500	0.5800	0.4000	0.6028
VisualPrompter	0.5800	0.4500	0.5900	0.2300	0.4300	0.2900	0.4708
OctoT2I	0.5350	0.4160	0.5090	0.2620	0.5130	0.3140	0.4557
Ours	**0.7589**	**0.6712**	**0.7160**	**0.5889**	**0.6651**	**0.4788**	**0.6780**

evolution is especially effective when generation failures involve explicit visual constraints that cannot be reliably resolved through a single predefined adaptation dimension.

Table 2 further evaluates complex compositions with multiple interacting requirements. EvoGen-Harness achieves the best average score of **0.6807** (**+0.0720**), leading in seven of eight dimensions and showing consistent gains as compositional difficulty increases.

On WISE (Table 3), our method reaches **0.6780** (**+0.0752**) and achieves the best result across all six domains. Together, these results show that EvoGen-Harness is not limited to a single failure type, generalizing from fine-grained visual constraints to complex compositions and world-knowledge consistency.

4.3 Generalization and Efficiency

Generalization across frozen generators. A central goal of EvoGen-Harness is to improve the external system without relying on the parameters of a particular image generator. Table 4 therefore applies the same framework to three frozen backbones: FLUX.1-dev, Qwen-Image, and Janus-Pro. Compared with VisualPrompter, EvoGen-Harness improves the overall GenEval2 score by **+0.4537**, **+0.3750**, and **+0.4674**, respectively. The gain is positive for every reported visual skill on all three backbones. This consistency supports the generator-agnostic nature of generator-external harness evolution rather than an improvement tied to a particular T2I model. Additional qualitative cross-generator examples are provided in Appendix B.7.

Inference efficiency. Table 5 examines whether the performance gain simply comes from additional sampling. EvoGen-Harness uses 1.4 generator calls and 2.1 LLM/VLM calls per prompt, close to OctoT2I's 1.3 and 2.0 calls, while achieving a substantially higher GenEval2 score (**0.7089**

Table 4: **Generalization across frozen T2I backbones on GenEval2.** We apply the same generator-external adaptation methods to different frozen T2I backbones and we report Soft-TIFA AM for skill-wise analysis and Soft-TIFA GM for overall performance. Δ denotes the absolute gain of EvoGen-Harness over baseline on the corresponding backbone. The best results within each backbone are marked in **bold**. ↑ indicates higher is better.

Method	Object	Attribute	Count	Position	Verb	Overall ↑
FLUX.1-dev	0.8754	0.6734	0.5311	0.3928	0.2150	0.2115
+ Promptist	0.8533	0.6510	0.5122	0.3819	0.2134	0.2056
+ VisualPrompter	0.9321	0.7223	0.5614	0.4350	0.2412	0.2552
+ **Ours**	**0.9799**	**0.9469**	**0.9053**	**0.6418**	**0.3851**	**0.7089**
Δ *over VisualPrompter*	+0.0478	+0.2246	+0.3439	+0.2068	+0.1439	+0.4537
Qwen-Image	0.9730	0.8412	0.6857	0.6083	0.3752	0.3483
+ Promptist	0.9484	0.8132	0.6613	0.5914	0.3724	0.3386
+ VisualPrompter	0.9853	0.8650	0.7060	0.6355	0.3961	0.3844
+ **Ours**	**0.9956**	**0.9742**	**0.9365**	**0.7689**	**0.5106**	**0.7594**
Δ *over VisualPrompter*	+0.0103	+0.1092	+0.2305	+0.1334	+0.1145	+0.3750
Janus-Pro	0.9102	0.7213	0.4834	0.5217	0.2056	0.1878
+ Promptist	0.8872	0.6973	0.4662	0.5072	0.2041	0.1826
+ VisualPrompter	0.9511	0.7630	0.5168	0.5549	0.2321	0.2328
+ **Ours**	**0.9855**	**0.9547**	**0.8957**	**0.7178**	**0.3777**	**0.7002**
Δ *over VisualPrompter*	+0.0344	+0.1917	+0.3789	+0.1629	+0.1456	+0.4674

Table 5: **Efficiency comparison on GenEval2.** We report generation quality, average inference time, and the numbers of generator and LLM/VLM calls per prompt. ↑ / ↓ indicate higher / lower is better.

Method	Score ↑	Time (s) ↓	Gen. Calls ↓	LLM/VLM ↓
VisualPrompter	0.2552	61.8	2.0	4.0
Idea2Img	0.0586	453.2	12.0	8.0
GenArtist	0.0556	117.3	3.2	4.6
OctoT2I	0.2895	10.0	1.3	2.0
Ours	**0.7089**	42.6	1.4	2.1

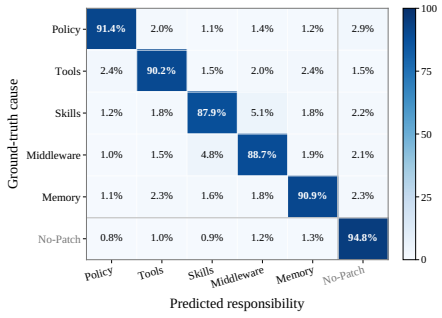


Figure 3: **Harness attribution across persistent failure causes.**

vs. 0.2895). OctoT2I remains faster in wall-clock latency (10.0 s vs. 42.6 s), whereas EvoGen-Harness provides a markedly higher generation score under a similar model-call budget. Compared with VisualPrompter, our method is both faster (42.6 s vs. 61.8 s) and substantially more accurate (0.7089 vs. 0.2552). Thus, the improvement cannot be explained by simply invoking more models or sampling more images.

The search analysis in Appendix B.2 further supports this result: Trace retains a GenEval2 score of 0.7089, close to full-space search (0.7112), while reducing the number of evaluated candidate edits from 20 to 8 and evolution-time generator calls from 80 to 32.

4.4 Attribution and Evolution Analysis

Harness localization. We evaluate the Harness Localizer on held-out controlled interventions, where one responsibility is perturbed while the remaining factors are kept fixed. As shown in Fig. 3, attribution recall reaches **87.9–91.4%** across the five editable responsibilities, while No-Patch achieves **94.8%**. The strong diagonal structure indicates that trajectory-relative evidence provides sufficient signal to distinguish different persistent failure sources. Importantly, the high No-Patch accuracy shows that Trace can also separate persistent harness weaknesses from stochastic generation failures, avoiding unnecessary evolution.

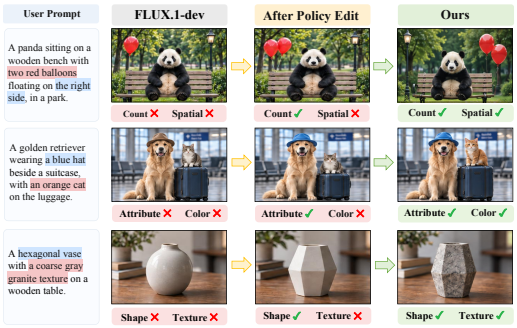


Figure 4: **Progressive visual repair with TRACE.** TRACE progressively repairs multiple visual constraints through successive harness updates while keeping the generator frozen.

Table 6: **Evolution across harness responsibilities.**

Responsibility	Attr. Recall ↑	Update Share	Commit Rate	Held-out Gain ↑
Policy	91.4%	27.8%	74.2%	+0.151
Tools	90.2%	14.6%	69.8%	+0.097
Skills	87.9%	23.4%	76.5%	+0.182
Middleware	88.7%	21.2%	72.1%	+0.139
Memory	90.9%	13.0%	67.4%	+0.089

Table 7: **Component ablation of TRACE.**

Variant	GenEval2 ↑	Attr. F1 ↑	Held-out Gain ↑	Regression ↓
w/o Traj. Aggregation	0.6580	0.816	+0.111	5.1%
w/o Attribution Guidance	0.6070	0.906	+0.086	6.8%
w/o Matched Evaluation	0.6390	0.906	+0.101	7.6%
w/o Progressive Re-local.	0.6250	0.906	+0.091	4.4%
w/o NO-PATCH	0.6420	0.753	+0.103	10.7%
w/o Safe Validation	0.6780	0.906	+0.119	13.9%
TRACE (Full)	**0.7089**	**0.906**	**+0.140**	**1.9%**

Multi-responsibility evolution. Accurate localization is useful only if it leads to different adaptation behaviors. Table 6 shows that accepted updates span all five responsibilities, with update shares of **13.0–27.8%** and positive held-out gains of **+0.089** to **+0.182**. Policy is updated most frequently, whereas Skills yields the largest held-out gain, suggesting that update frequency and transferable benefit capture different aspects of harness evolution. The update distribution therefore does not collapse to a single predefined target; instead, the responsibility to evolve changes with the observed failure. Figure 4 further visualizes this progressive behavior: successive harness updates resolve different constraints of the same failure, while the underlying image generator remains frozen. Additional compound-failure results and responsibility-specific examples are provided in Appendices B.3 and B.6.

4.5 ABLATION STUDY

Evidence and guided evolution. As shown in Table 7, removing trajectory aggregation reduces GenEval2 from **0.7089** to 0.6580 and Attribution F1 from **0.906** to 0.816, while unmatched evaluation increases regression to 7.6%. Removing attribution guidance causes the largest performance drop to 0.6070, and disabling progressive re-localization further reduces the score to 0.6250. These results show that reliable trajectory evidence is critical not only for localizing persistent failures, but also for narrowing and redirecting the subsequent evolution search.

Selective persistence. NO-PATCH and safe validation control which changes become persistent. Removing them increases regression from **1.9%** to 10.7% and 13.9%, respectively. The full TRACE therefore achieves the strongest balance between generation quality and persistent improvement, reaching **0.7089** GenEval2 and **+0.140** held-out gain with only **1.9%** regression. These results show that effective harness evolution depends on reliable localization, progressive search, and selective retention working together. Additional sensitivity and proposal-LLM robustness results are provided in Appendices B.4 and B.5.

5 CONCLUSION

We introduce EVOGEN-HARNESS, a generator-agnostic framework for multi-responsibility evolution around frozen image generators. Its core mechanism, TRACE, uses trajectory-relative visual evidence to guide search over persistent harness responsibilities and progressively coordinates updates through re-attribution and regression-safe validation. Across GenEval2, T2I-CompBench++, and WISE, EVOGEN-HARNESS consistently improves generation quality while achieving reliable attribution, low regression, and consistent improvements across frozen T2I backbones. These results suggest that selectively evolving the external harness provides a practical path toward more capable and self-improving image-generation systems without modifying generator parameters.

AI USE STATEMENT

We used generative AI tools to assist with literature search, the identification of relevant references, the improvement and modification of software code, and limited language polishing of this disclosure. All AI-assisted code modifications were reviewed and verified by the authors for correctness. Generative AI tools were not used for research ideation, methodology or experimental design, result interpretation, or scientific figure preparation. AI-identified references were independently and carefully checked against original sources by the authors. The authors take full responsibility for the final content and accuracy of this work.

ETHICS STATEMENT

This work uses publicly available text-to-image models and academic benchmarks and does not involve newly collected human-subject data or private personal information. Our study focuses on improving the reliability and adaptability of image-generation systems. As with general-purpose image-generation technology, the resulting methods could potentially be applied to generate misleading or harmful visual content; such deployment is outside the scope of this work and should follow appropriate model-use policies and safeguards.

REPRODUCIBILITY STATEMENT

We provide the information needed to reproduce our method and experiments throughout the paper and appendix. The complete EVOGEN-HARNESS and TRACE formulation is described in Sec. 3, while datasets, baselines, evaluation protocols, and computational resources are reported in Sec. 4. Appendix A.1 defines the auxiliary evaluation metrics, and the appendix further provides implementation details, the complete TRACE algorithm, search-efficiency analysis, and hyperparameter sensitivity results.

REFERENCES

Black Forest Labs. FLUX, 2024. URL https://github.com/black-forest-labs/flux. GitHub repository.

Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Aiming Hao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Yuming Jiang, et al. Z-image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.

Tingfeng Cao, Chengyu Wang, Bingyan Liu, Ziheng Wu, Jinhui Zhu, and Jun Huang. Beautifulprompt: Towards automatic prompt engineering for text-to-image synthesis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1–11, 2023.

Harold Haodong Chen, Zhiyu Hou, Wen-Jie Shu, Weilin Ruan, Yingjie Xu, Litao Guo, and Ying-Cong Chen. Genrouter: Unified workflow routing for agentic image generation, 2026a.

Sixiang Chen, Zhaohu Xing, Tian Ye, Xinyu Geng, Yunlong Lin, Jianyu Lai, Xuanhua He, Fuxiang Zhai, Jialin Gao, and Lei Zhu. Genevolve: Self-evolving image generation agents via tool-orchestrated visual experience distillation, 2026b.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

Jaemin Cho, Yushi Hu, Jason Baldridge, Roopal Garg, Peter Anderson, Ranjay Krishna, Mohit Bansal, Jordi Pont-Tuset, and Su Wang. Davidsonian scene graph: Improving reliability in fine-grained evaluation for text-to-image generation. In *The Twelfth International Conference on Learning Representations*, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers

for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. volume 36, pp. 18225–18250, 2023.

Yutong Feng, Biao Gong, Di Chen, Yujun Shen, Yu Liu, and Jingren Zhou. Ranni: Taming text-to-image diffusion for accurate instruction following. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4744–4753. IEEE, 2024.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems*, volume 36, pp. 52132–52152, 2023.

Google. Gemini 2.5 Flash Image (Nano Banana). Google AI for Developers, 2025. URL `https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash-image`.

Yaru Hao, Zewen Chi, Li Dong, and Furu Wei. Optimizing prompts for text-to-image generation. volume 36, pp. 66923–66939, 2023.

Kaiyi Huang, Chengqi Duan, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench++: An enhanced and comprehensive benchmark for compositional text-to-image generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(5):3563–3579, 2025.

Chengyou Jia, Changliang Xia, Zhuohang Dang, Weijia Wu, Hangwei Qian, and Minnan Luo. Chatgen: Automatic text-to-image generation from freestyle chatting. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13284–13293. IEEE, 2025.

Xu Jiang, Bin Chen, Gehui Li, Yule Duan, Ronggang Wang, and Jian Zhang. Octot2i: A self-evolving agentic text-to-image router. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 31628–31638, June 2026.

Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. Geneval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025.

Mingcheng Li, Xiaolu Hou, Ziyang Liu, Dingkang Yang, Ziyun Qian, Jiawei Chen, Jinjie Wei, Yue Jiang, Qingyao Xu, and Lihua Zhang. Mccd: Multi-agent collaboration-based compositional diffusion for complex text-to-image generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13263–13272. IEEE, 2025.

Zongxia Li, Dawei Liu, Fuxiao Liu, Yuhang Zhou, Xiyang Wu, Jingxi Chen, Jing Xie, Xiaomin Wu, and Lichao Sun. COMFYCLAW: Self-evolving skill harnesses for image generation workflows, 2026.

Long Lian, Boyi Li, Adam Yala, and Trevor Darrell. Llm-grounded diffusion: Enhancing prompt understanding of text-to-image diffusion models with large language models. *arXiv preprint arXiv:2305.13655*, 2023.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Zhiheng Xi, Xuanjing Huang, Hang Yan, Zhenhua Han, Tao Gui, and Yu-Gang Jiang. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses, 2026.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. volume 38, pp. 40783–40818, 2026.

Yuwei Niu, Munan Ning, Mengren Zheng, Weiyang Jin, Bin Lin, Peng Jin, Jiaqi Liao, Chaoran Feng, Fanqing Meng, Kun-Peng Ning, Bin Zhu, and Li Yuan. WISE: World knowledge-informed semantic evaluation for text-to-image generation. In *Forty-third International Conference on Machine Learning*, 2026.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Shachar Rosenman, Vasudev Lal, and Phillip Howard. Neuroprompts: An adaptive framework to optimize prompts for text-to-image generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pp. 159–167, 2024.

Jiayang Sun, Hongbo Wang, Jie Cao, Huaibo Huang, and Ran He. Marmot: Multi-agent reasoning for multi-object self-correcting in improving image-text alignment. *arXiv e-prints*, pp. arXiv–2504, 2025.

Zhenyu Wang, Aoxue Li, Zhenguo Li, and Xihui Liu. Genartist: Multimodal llm as an agent for unified image generation and editing. In *Advances in Neural Information Processing Systems*, volume 37, pp. 128374–128395, 2024a.

Zhenyu Wang, Enze Xie, Aoxue Li, Zhongdao Wang, Xihui Liu, and Zhenguo Li. Divide and conquer: Language models can plan and self-correct for compositional text-to-image generation. *arXiv preprint arXiv:2401.15688*, 2024b.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report, 2025.

Mingrui Wu, Lu Wang, Pu Zhao, Fangkai Yang, Jianjin Zhang, Jianfeng Liu, Yuefeng Zhan, Weihao Han, Hao Sun, Jiayi Ji, et al. Reprompt: Reasoning-augmented reprompting for text-to-image generation via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 14030–14057, 2026a.

Shiyu Wu, Mingzhen Sun, Weining Wang, Yequan Wang, and Jing Liu. Visualprompter: Semantic-aware prompt optimization with visual feedback for text-to-image synthesis. In *International Conference on Learning Representations*, volume 2026, pp. 30578–30600, 2026b.

Tsung-Han Wu, Long Lian, Joseph E Gonzalez, Boyi Li, and Trevor Darrell. Self-correcting llm-controlled diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6327–6336. IEEE, 2024.

Bangji Yang, Ruihan Guo, Jiajun Fan, Chaoran Cheng, and Ge Liu. M3: High-fidelity text-to-image generation via multi-modal, multi-agent and multi-round visual reasoning. *arXiv preprint arXiv:2602.06166*, 2026.

Ling Yang, Zhaochen Yu, Chenlin Meng, Minkai Xu, Stefano Ermon, and Bin Cui. Mastering text-to-image diffusion: Recaptioning, planning, and generating with multimodal llms. In *Forty-first International Conference on Machine Learning*, 2024a.

Zhengyuan Yang, Jianfeng Wang, Linjie Li, Kevin Lin, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. Idea2img: Iterative self-refinement with gpt-4v for automatic image design and generation. In *European Conference on Computer Vision (ECCV)*, pp. 167–184, 2024b.

Guibin Zhang, Leo Lu, Fangzhou Xie, Kang Zhu, Junhao Wang, Zhifei Xie, Zhaochen Yu, Zihang Liu, Zhongxiang Sun, Qiankun Li, et al. Jit-agent: Scaling harness intelligence via just-in-time harness evolution. *arXiv preprint arXiv:2608.25593*, 2026a.

Hangfan Zhang, Shao Zhang, Kangcong Li, Chen Zhang, Yang Chen, Yiqun Zhang, Lei Bai, and Shuyue Hu. Self-harness: Harnesses that improve themselves, 2026b.

Xinchen Zhang, Ling Yang, Guohao Li, Yaqi Cai, Yong Tang, Yujiu Yang, Mengdi Wang, Bin CUI, et al. Itercomp: Iterative composition-aware feedback learning from model gallery for text-to-image generation. In *International Conference on Learning Representations*, volume 2025, pp. 31968–31988, 2025.

Lirui Zhao, Yue Yang, Kaipeng Zhang, Wenqi Shao, Yuxin Zhang, Yu Qiao, Ping Luo, and Rongrong Ji. Diffagent: Fast and accurate text-to-image api selection with large language model. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6390–6399. IEEE, 2024.

A APPENDIX

A.1 EVALUATION METRICS

We follow the official metrics and evaluation protocols for GenEval2, T2I-CompBench++, and WISE. Here we define only the auxiliary metrics used to analyze TRACE's attribution and persistent evolution.

Attribution metrics. Let $\mathcal{R} = \{\Pi, \mathcal{T}, \mathcal{S}, \mathcal{W}, \mathcal{M}\}$ denote the five editable harness responsibilities. For each responsibility $z \in \mathcal{R}$, we compute precision and recall as

$$P_z = \frac{\mathrm{TP}_z}{\mathrm{TP}_z + \mathrm{FP}_z}, \qquad R_z = \frac{\mathrm{TP}_z}{\mathrm{TP}_z + \mathrm{FN}_z}. \tag{8}$$

We report macro-averaged attribution F1 over the editable responsibilities:

$$\mathrm{F1}_{\mathrm{attr}} = \frac{1}{|\mathcal{R}|} \sum_{z \in \mathcal{R}} \frac{2 P_z R_z}{P_z + R_z}. \tag{9}$$

For the class-wise attribution results in Fig. 3, we additionally report the fraction of examples from each ground-truth responsibility that are correctly attributed.

The abstention class $\varnothing$ is evaluated separately. Its NO-PATCH accuracy is

$$\mathrm{Acc}_{\mathrm{NP}} = \frac{\mathrm{TP}_{\varnothing}}{N_{\varnothing}}, \tag{10}$$

where $N_{\varnothing}$ is the number of ground-truth NO-PATCH examples. This corresponds to the diagonal entry of the NO-PATCH row in the row-normalized confusion matrix.

Held-out gain. To measure whether an accepted update transfers beyond the examples that triggered it, we compare the evolved harness with its parent harness on a disjoint held-out set $\mathcal{D}_{\mathrm{held}}$. For matched stochastic conditions $\Omega = \{\omega_1, \ldots, \omega_K\}$, define

$$\bar{Q}(x; \mathcal{H}) = \frac{1}{K} \sum_{k=1}^{K} Q(x; \mathcal{H}, \omega_k), \tag{11}$$

where Q is the corresponding visual evaluation score. Held-out Gain is then

$$\mathrm{Gain}_{\mathrm{held}} = \frac{1}{|\mathcal{D}_{\mathrm{held}}|} \sum_{x \in \mathcal{D}_{\mathrm{held}}} \left[\bar{Q}(x; \mathcal{H}_{t+1}) - \bar{Q}(x; \mathcal{H}_t) \right]. \tag{12}$$

A positive value therefore indicates that the accepted update improves previously unseen cases rather than only repairing the triggering example.

Regression. We measure regression on a preservation set $\mathcal{D}_{\mathrm{pres}}$ containing cases that are successfully handled before evolution. Let

$$S(x; \mathcal{H}) = \mathbb{I}\left[\bar{Q}(x; \mathcal{H}) \geq \theta_x \right] \tag{13}$$

denote whether harness $\mathcal{H}$ successfully handles case x under the corresponding evaluation threshold θ_x. Defining

$$\mathcal{D}^{+}_{\mathrm{pres}} = \{ x \in \mathcal{D}_{\mathrm{pres}} : S(x; \mathcal{H}_t) = 1 \},$$

the regression rate is

$$\mathrm{Regression} = \frac{100}{|\mathcal{D}^{+}_{\mathrm{pres}}|} \sum_{x \in \mathcal{D}^{+}_{\mathrm{pres}}} \mathbb{I}\left[S(x; \mathcal{H}_{t+1}) = 0 \right]. \tag{14}$$

Thus, regression measures the percentage of previously successful cases that become unsuccessful after a persistent harness update.

Responsibility statistics. For Table 6, *Update Share* is the fraction of all committed updates assigned to a given responsibility, while *Commit Rate* is the fraction of proposed updates for that responsibility that pass validation and are retained. These statistics describe how evolution is distributed across the harness and are not treated as optimization objectives.

Efficiency metrics. We report average wall-clock inference time per prompt together with the numbers of generator and LLM/VLM calls. For the search-efficiency analysis, Candidate Edits denotes the average number of candidate harness modifications actually evaluated per task.

B Implementation Details

Benchmark and evaluation settings. We use the official releases and evaluation pipelines for all three benchmarks. For GenEval2, we evaluate on the official 800 prompts and report Soft-TIFA following the official aggregation protocol. For T2I-CompBench++, we follow the official evaluation setup, including the prescribed sampling and fixed-seed protocol, and report the eight official compositional metrics. For WISE, we use the latest verified benchmark release and its official WiScore evaluation pipeline.

Benchmark evaluation is separated from the stochastic executions used internally by TRACE. Unless otherwise specified, TRACE uses $K = 4$ stochastic executions for trajectory-relative evidence collection. Benchmark labels, official scores, and test annotations are never exposed to the harness localizer, edit proposer, or candidate-selection procedure.

System configuration. Unless otherwise specified, we use FLUX.1-dev as the frozen image generator and GPT-4.1 as the fixed LLM for both harness localization and responsibility-conditioned edit proposal. For fair comparison, all baseline methods are evaluated using their official implementations and recommended default settings. The Harness Localizer and Edit Proposer use separate prompts and structured output schemas. OWLv2-base with NVILA-Lite-2B-Verifier provides visual verification, while a deterministic execution controller executes the evolved harness during deployment, coordinating requirement parsing, tool invocation, generation procedures, and verification. All models and controllers remain fixed throughout evolution; only the persistent generator-external harness is updated. Benchmark labels, official benchmark scores, and test annotations are never provided to the localization, proposal, or candidate-selection procedure.

TRACE configuration. Unless otherwise specified, we use an attribution frontier size of $r = 3$, $K = 4$ stochastic executions for trajectory-relative evidence, at most $M = 3$ candidate edits per responsibility, and a maximum evolution depth of $L = 3$. We set the preservation and execution-cost coefficients to $\lambda = 0.5$ and $\mu = 0.05$, respectively. The Harness Localizer ranks the five editable responsibilities together with NO-PATCH directly from trajectory-relative evidence and requires no additional training. All candidate edits are evaluated against their parent harness under matched stochastic conditions.

Validation and compute. Target and held-out validation use at most three examples each, while the preservation set contains at most two examples. We use a regression tolerance of $\epsilon = 0.01$ and at least two validation seeds. The evolution loop allows up to two proposal attempts when an edit is invalid or cannot be executed. All main experiments are conducted on 2 NVIDIA A100 (80 GB) GPUs. Reported LLM/VLM calls include harness-localization, edit-proposal, and visual-verification calls.

Evolution data and evaluation separation. EVOGEN-HARNESS performs no additional parameter training; all image generators, the harness localizer, the edit proposer, visual verifiers, and the deterministic controller remain fixed throughout evolution, and only the persistent generator-external harness is updated. We construct a dedicated evolution corpus from PartiPrompts (P2), independent of the benchmarks used for final evaluation. After de-duplication and benchmark-overlap filtering, we form three pairwise-disjoint subsets: an evolution set $\mathcal{D}_{\text{evo}}$ containing 500 prompts, a held-out validation set $\mathcal{D}_{\text{val}}$ containing 100 prompts, and a preservation set $\mathcal{D}_{\text{pres}}$ containing 100 prompts. During each evolution step, at most three examples from $\mathcal{D}_{\text{val}}$ and at most two examples from $\mathcal{D}_{\text{pres}}$ are used for held-out and preservation validation, respectively.

Table 8: Operational representation of the five persistent harness responsibilities. The examples describe the editable state rather than the parameters of the frozen generator.

Responsibility	Persistent representation	Editable contents
Policy	Visual requirement and constraint rules	Requirement interpretation, constraint priority, and criteria used to represent explicit visual requirements such as count, attribute, relation, and action constraints.
Tools	Capability registry for available generation and auxiliary tools	Capability and limitation descriptions, applicable scenarios, and tool-specific knowledge used by the harness. Tool implementations and model parameters are never modified.
Skills	Reusable procedural templates	Multi-step generation routines, prompt templates, decomposition strategies, and reusable task-solving procedures.
Middleware	Run-time orchestration configuration	Routing, verification, retry, termination, and tool-invocation rules. The deterministic base controller itself remains fixed.
Memory	Persistent cross-task experience records	Reusable summaries of successful and failed executions, recurring failure patterns, and previously validated corrective experience.

To prevent evaluation contamination, we remove exact and near-duplicate prompts overlapping with the official evaluation prompts of GenEval2, T2I-CompBench++, and WISE before constructing these splits. The evolution, held-out validation, and preservation sets are therefore disjoint from all final benchmark evaluation sets. Benchmark test prompts, labels, annotations, and official evaluation scores are never exposed to harness localization, edit proposal, candidate selection, or validation. After evolution, the resulting harness is frozen and used unchanged for final benchmark evaluation; no benchmark test prompt is used to update the harness or can influence subsequent test examples.

Harness representations and edit operators. The five harness responsibilities are instantiated as explicit and independently editable persistent records. Their concrete roles and editable contents are summarized in Table 8. This decomposition makes the heterogeneous harness state operational: an edit is localized to one responsibility and cannot directly modify the remaining responsibilities or the frozen generation system.

Formally, each candidate edit can be represented at the harness interface as

$$\delta = (z, o, k, v_{\text{old}}, v_{\text{new}}) \tag{15}$$

where $z \in \{\Pi, T, S, W, M\}$ denotes the selected responsibility, $o \in \{\textsc{Add}, \textsc{Replace}, \textsc{Delete}\}$ is the edit operation, k identifies the affected record or field, and v_{old} and v_{new} denote its states before and after editing. Applying an edit yields

$$H' = H \oplus \delta \tag{16}$$

where only the state associated with z is changed. Edits that modify another responsibility, the frozen generator, or the deterministic base controller are rejected as invalid.

The responsibility-conditioned proposer receives the current value of the selected responsibility, the trajectory-relative evidence, and its permitted edit schema. It proposes localized modifications only within this schema. NO-PATCH is an attribution decision rather than a persistent harness component and therefore does not modify H.

Examples of localized harness edits. The following examples illustrate how an observed visual failure is converted into a responsibility-specific persistent modification.

Example 1: Policy edit for an exact-count failure. Suppose the request requires "three white ducks", but repeated executions produce only two ducks. After trajectory-relative attribution selects Policy, the proposer may revise the corresponding requirement rule as follows:

> **Before:** Explicit numerical descriptions are represented together with other object attributes during requirement parsing.

> **After:** Explicit cardinality expressions (e.g., "three ducks") must be represented as hard count constraints. The required cardinality is preserved during prompt construction and explicitly checked during visual verification.

This edit changes only the persistent interpretation of numerical requirements; the image generator, tool inventory, and execution controller remain unchanged.

Example 2: Middleware edit for repeated verification failure. Consider a case in which the generated image violates a spatial or counting constraint and the current execution terminates immediately after verification. If the residual failure is attributed to Middleware, a possible edit is:

> **Before:** Terminate the current generation procedure after the first verification pass.
>
> **After:** When verification detects an unresolved explicit constraint and the execution budget remains available, trigger one targeted retry using the verifier feedback. Terminate when all required constraints pass verification or when the retry budget is exhausted.

The edit modifies the orchestration policy without changing the verifier, generator, or underlying controller implementation.

Other responsibilities are edited through the same localized interface. For example, a **Tools** edit may correct an inaccurate capability record when repeated evidence shows that a generator is unreliable for exact-count requests; a **Skills** edit may replace a direct-generation routine with a validated procedure that first makes spatial relations explicit and then verifies them after generation; and a **Memory** edit may store a cross-task experience stating that explicit cardinality preservation and post-generation counting consistently repaired a recurring family of failures. All such edits are retained only after the matched and held-out validation described in Sec. 3.3.

B.1 TRACE: PROGRESSIVE HARNESS EVOLUTION

Algorithm 1 summarizes the complete TRACE procedure. Trajectory-relative localization first narrows the heterogeneous harness space to a small set of plausible responsibilities. A fixed LLM then proposes localized edits only within this frontier, and matched execution feedback determines which edit should be applied. After each accepted edit, residual failures are re-localized, allowing the evolution path to move across responsibilities when necessary.

Search complexity. At each evolution step, the Harness Localizer retains at most r responsibilities, each contributing at most M candidate edits. Therefore, TRACE evaluates at most

$$\mathcal{O}(LrM) \tag{17}$$

candidate edits over L evolution steps. Since each candidate is evaluated under K matched stochastic conditions, the corresponding generation complexity is

$$\mathcal{O}(LrMK). \tag{18}$$

This replaces combinatorial enumeration over multi-responsibility edit sequences with a bounded attribution-guided search, while progressive re-localization still allows the evolution path to traverse different responsibilities.

B.2 SEARCH EFFICIENCY ANALYSIS

We examine whether TRACE reduces unnecessary exploration while preserving generation quality. We compare three search strategies under the same frozen generator, LLM proposer, candidate budget, stochastic execution budget, maximum evolution depth, and matched-evaluation protocol. *Full-Space Search* explores all editable responsibilities at each step; *Hard Top-1* explores only the responsibility with the highest localization score; and TRACE retains a Top-r localization frontier and progressively re-localizes residual failures after each accepted update.

Full-space search achieves only a marginally higher score than TRACE (0.7112 vs. 0.7089), but requires substantially more candidate evaluations and generator calls. Hard Top-1 is the most efficient strategy, yet its lower score (0.6815) shows the risk of committing prematurely to a single responsibility. In contrast, TRACE preserves multiple plausible responsibilities in a compact Top-r frontier, providing a favorable balance between search efficiency and generation quality while allowing subsequent evolution to be redirected through progressive re-localization.

Algorithm 1 TRACE: Trajectory-Relative Attribution and Coordinated Evolution

Require: Current harness $\mathcal{H}_t$, frozen generator $\mathcal{G}$, frontier size r, stochastic executions K, candidate budget M, maximum depth L

$\widetilde{\mathcal{H}} \leftarrow \mathcal{H}_t$
$\boldsymbol{\pi} \leftarrow [\,]$
for $\ell = 1, \ldots, L$ **do**
  Collect trajectory-relative evidence $\mathcal{E}_\ell$ from K stochastic executions under $\widetilde{\mathcal{H}}$
  Obtain localization scores $s_{\mathrm{loc}}(z \mid \mathcal{E}_\ell, \widetilde{\mathcal{H}})$ from the LLM-based Harness Localizer
  if NO-PATCH is preferred **then**
    break
  end if
  $\mathcal{R}_\ell^{(r)} \leftarrow \text{Top-}r_{z \in \mathcal{R}} s_{\mathrm{loc}}(z \mid \mathcal{E}_\ell, \widetilde{\mathcal{H}})$
  Generate at most M localized edits Δ_z for each $z \in \mathcal{R}_\ell^{(r)}$ using the LLM proposer
  $\Delta_\ell \leftarrow \bigcup_{z \in \mathcal{R}_\ell^{(r)}} \Delta_z$
  for each $\delta \in \Delta_\ell$ **do**
    Evaluate $\Delta Q_\ell(\delta)$ under the same K stochastic conditions
    Compute $U_\ell(\delta)$
  end for
  $\delta_\ell^* \leftarrow \arg\max_{\delta \in \Delta_\ell} U_\ell(\delta)$
  if $U_\ell(\delta_\ell^*) \leq 0$ **then**
    break
  end if
  $\widetilde{\mathcal{H}} \leftarrow \widetilde{\mathcal{H}} \oplus \delta_\ell^*$
  $\boldsymbol{\pi} \leftarrow \boldsymbol{\pi} \parallel \delta_\ell^*$
end for
if $\widetilde{\mathcal{H}}$ passes held-out and preservation validation **then**
  return $\widetilde{\mathcal{H}}$
else
  return $\mathcal{H}_t$
end if

Table 9: **Search efficiency of TRACE.** Top-r localization substantially reduces exploration relative to full-space search while preserving generation quality. Candidate edits and generator calls are averaged per task. ↑ / ↓ indicate higher / lower is better.

Search Strategy	GenEval2 ↑	Candidate Edits ↓	Gen. Calls ↓	Evol. Time (s) ↓
Full-Space Search	0.7112	20.0	80.0	2386
Hard Top-1	0.6815	**4.0**	**16.0**	**486**
TRACE Top-r	0.7089	8.0	32.0	964

Evolution-time cost accounting. We distinguish deployment-time inference cost from the one-time cost of harness evolution. Table 5 reports the former, i.e., the average cost of executing the current harness for a test request. In contrast, Table 9 reports the computation incurred by TRACE when a persistent update is triggered, including candidate evaluation under matched stochastic conditions.

All search-efficiency statistics are averaged per evolution task. The reported generator calls include candidate evaluation rollouts, while the evolution time covers localization, proposal, candidate evaluation, and validation. Since accepted harness updates persist across subsequent requests, this evolution cost is amortized over future executions and is not included in the per-prompt inference cost reported in Table 5.

Table 10: Robustness of TRACE under paired-responsibility failures. Each example contains two simultaneously injected persistent causes. Cause Recall@3 measures coverage of the two ground-truth causes by the initial Top-3 localization frontier; Both Recovered measures whether both causes are identified over progressive re-localization; and Repair Success measures whether both injected failure effects are resolved after evolution. ↑ indicates higher is better.

Compound Causes	Cause Recall@3 ↑	Both Recovered ↑	Repair Success ↑
Policy + Tools	96.1%	92.4%	86.8%
Policy + Skills	94.8%	90.1%	85.7%
Policy + Middleware	94.6%	89.5%	84.3%
Policy + Memory	95.5%	91.3%	85.1%
Tools + Skills	94.1%	88.7%	83.6%
Tools + Middleware	93.8%	87.9%	82.4%
Tools + Memory	95.0%	90.4%	83.2%
Skills + Middleware	91.9%	83.8%	79.6%
Skills + Memory	93.6%	87.1%	82.8%
Middleware + Memory	94.0%	88.2%	81.9%
Average	**94.3%**	**88.9%**	**83.5%**

B.3 ROBUSTNESS TO COMPOUND FAILURES

The attribution analysis in Sec. 4.4 uses controlled single-responsibility interventions to obtain unambiguous ground-truth failure causes. In realistic generation pipelines, however, a failure may arise from multiple harness responsibilities simultaneously. We therefore conduct an additional controlled stress test to evaluate the robustness of TRACE under compound failure causes.

Paired-responsibility interventions. We construct compound failures by simultaneously injecting two independently defined persistent perturbations into different harness responsibilities. We evaluate all ten unordered pairs among $\{\Pi, T, S, W, M\}$. For each compound case, the request, frozen generator, stochastic conditions, and all remaining harness responsibilities are kept unchanged. This construction isolates the effect of interacting persistent causes while preserving exact ground-truth attribution.

Let $Z_i^\star = \{z_i^{(1)}, z_i^{(2)}\}$ denote the two ground-truth responsibilities for example i, and let $\mathcal{R}_i^{(3)}$ denote the initial Top-3 localization frontier. We measure initial localization quality using

$$\text{CauseRecall@3} = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\mathcal{R}_i^{(3)} \cap Z_i^\star\right|}{|Z_i^\star|}. \tag{19}$$

Because TRACE re-localizes residual failures after each accepted edit, we additionally report **Both Recovered**, the fraction of cases for which both ground-truth responsibilities are identified over the complete progressive evolution path. Finally, **Repair Success** measures the fraction of compound cases for which both injected failure effects are successfully resolved after evolution.

Results. As shown in Table 10, TRACE remains robust when two persistent failure causes are present simultaneously. The initial Top-3 frontier covers 94.3% of ground-truth causes on average, indicating that attribution remains reliable despite competing evidence from multiple responsibilities. More importantly, progressive re-localization recovers both causes in 88.9% of compound cases, showing that TRACE does not require an entire generation failure to originate from a single responsibility.

The final repair success rate reaches 83.5%. The gap between cause recovery and successful repair indicates that correctly identifying both responsibilities does not always guarantee that the proposed edits fully eliminate the corresponding visual failures. The most challenging combination is Skills + Middleware, with a repair success rate of 79.6%, suggesting stronger interaction between reusable procedures and run-time orchestration. Nevertheless, performance remains consistent across all ten responsibility pairs, supporting the ability of progressive re-localization to handle compound persistent failures.

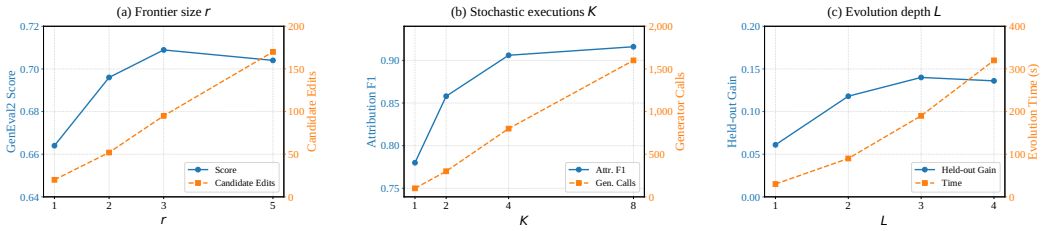


Figure 5: **Sensitivity analysis of TRACE. (a)** A larger attribution frontier initially improves GenEval2 performance by retaining more plausible responsibilities, but yields diminishing gains while increasing the number of candidate edits. **(b)** Increasing K improves attribution reliability by reducing the influence of stochastic generation, with gains saturating as generator calls increase. **(c)** Increasing L improves held-out performance by enabling residual failures to be re-attributed across responsibilities, while deeper evolution provides limited additional benefit at higher cost.

Table 11: **Robustness to different LLMs on GenEval2.** Only the proposal LLM is replaced while all other components and budgets remain fixed.

Proposal LLM	GenEval2 ↑	Held-out Gain ↑	Commit Rate ↑	Regression ↓
Qwen3.5-27B	0.6996	+0.133	70.6%	2.3%
GPT-5.5	**0.7164**	**+0.146**	**73.2%**	**1.7%**
GPT-4.1 (Default)	0.7089	+0.140	72.0%	1.9%

Importantly, this compound-failure setting directly tests the motivation for coordinated multi-responsibility evolution: when multiple persistent failure sources coexist, effective evolution must remain open to different intervention sites rather than committing to a single predefined responsibility. The strong *Both Recovered* and *Repair Success* results show that progressive re-localization enables TRACE to redirect evolution across responsibilities as residual failures change, rather than collapsing to a fixed adaptation target.

B.4 SENSITIVITY ANALYSIS

We study the sensitivity of TRACE to three search parameters: the attribution frontier size r, the number of stochastic executions K, and the maximum evolution depth L. All remaining settings follow Appendix B.

As shown in Fig. 5, performance improves rapidly as the search frontier, trajectory evidence, and evolution depth increase, but the gains become marginal beyond moderate budgets. We therefore use $r = 3$, $K = 4$, and $L = 3$ as the default configuration, which provides a favorable balance between evolution quality and computational cost.

B.5 ROBUSTNESS TO DIFFERENT PROPOSAL LLMS

We further examine whether EVOGEN-HARNESS depends on a particular proposal LLM. We replace only the responsibility-conditioned proposal operator while keeping the Harness Localizer, image generator, verifier, candidate budget, stochastic seeds, and validation protocol unchanged. We compare the default GPT-4.1 with the open-weight Qwen3.5-27B and the stronger GPT-5.5.

Table 11 shows that EVOGEN-HARNESS remains effective across different proposal LLMs. Qwen3.5-27B retains performance close to the default GPT-4.1 setting, while GPT-5.5 provides a modest further improvement. These results indicate that the effectiveness of EVOGEN-HARNESS primarily comes from TRACE and its attribution-guided, regression-safe evolution procedure, rather than from a particular proposal LLM.



Figure 6: **Responsibility-specific harness evolution.** Representative Policy, Skills, Tools, and Memory updates repair distinct persistent failure modes while the image generator remains frozen.



Figure 7: **Cross-generator adaptation.** Representative qualitative results of EVOGEN-HARNESS on FLUX.1-dev, Qwen-Image, and Janus-Pro. Each image generator remains frozen while its generator-external harness is evolved.

B.6 Additional Responsibility-Specific Evolution Examples

Figure 6 complements the aggregate responsibility statistics in Table 6 with representative evolution cases. The examples illustrate how different persistent failure sources induce different localized harness updates rather than a uniform correction.

Specifically, the Policy example strengthens an explicit counting constraint, the Skills example revises a reusable procedure for spatial composition, the Tools example corrects capability knowledge associated with attribute binding, and the Memory example updates reusable experience for an action–object relation. In all cases, the underlying image generator remains frozen; only the responsibility selected by TRACE is modified. Middleware updates are characterized quantitatively in Table 6 and illustrated through the concrete edit example in Appendix B.

B.7 ADDITIONAL CROSS-GENERATOR EXAMPLES

Table 4 quantitatively shows that EVOGEN-HARNESS consistently improves three frozen T2I backbones. Figure 7 provides representative qualitative examples that complement these results.

For each backbone, we apply the same EVOGEN-HARNESS framework while keeping the image generator fixed and evolving only its external harness. The evolved harness is therefore adapted separately to each frozen generator; we do not assume zero-shot transfer of a single evolved harness across backbones. The examples span counting, spatial placement, and multi-object relations, showing that the benefit of harness evolution is not tied to the rendering behavior of a particular T2I model.

C HARNESS LOCALIZER PROMPT

The Harness Localizer in TRACE is implemented as a fixed LLM-based attribution module. It receives trajectory-relative evidence and the sanitized current harness state, and outputs responsibility-level scores over five editable harness responsibilities and a NO-PATCH option. The complete system prompt is provided below.

Harness Localizer Prompt

You are the Harness Localizer in TRACE, a fixed LLM role in an attribution-guided evolution system for visual generation agents.
Your task is to identify where a persistent model-external harness weakness should be investigated. You perform attribution only: visual verification determines what failed, while you determine which harness responsibilities could explain the failure.
You do not propose edits, select candidates, validate patches, modify the generator, or claim that a repair will succeed.
The harness contains five independent responsibilities:
1. POLICY Policy defines how user requests are interpreted into visual requirements, constraints, evidence rules, and preservation requirements. Assign higher scores when failures indicate incorrect requirement representation or constraint interpretation.
2. TOOLS Tools describe available generation and verification capabilities, limitations, and usage conditions. Assign higher scores when evidence suggests incorrect capability knowledge or missing tool information.
3. SKILLS Skills represent reusable generation procedures, including decomposition, prompt construction, layout planning, object binding, and verification-aware procedures. Assign higher scores when reusable procedures are insufficient for similar requests.
4. MIDDLEWARE Middleware controls runtime orchestration, including routing, skill selection, retries, verification, and execution ordering. Assign higher scores when available resources are sufficient but the execution strategy is inappropriate.
5. MEMORY Memory stores reusable cross-task experience, recurring failure patterns, and validated corrective knowledge. Assign higher scores when failures indicate missing transferable experience.
6. NO-PATCH NO-PATCH indicates that persistent evolution is not justified. Assign higher scores when failures are better explained by stochastic generation, ambiguous intent, verifier uncertainty, or insufficient evidence.
Perform attribution following these principles:
(1) Identify the violated visual constraint rather than only the observed symptom.
(2) Consider repeated evidence across stochastic executions.
(3) Attribute failures to the earliest responsibility-specific decision that could plausibly explain the failure.
(4) Distinguish requirement interpretation, capability knowledge, reusable procedures, runtime orchestration, and persistent experience.
(5) Preserve multiple plausible causes when evidence supports interacting responsibilities.

(6) Prefer NO-PATCH when persistent evolution is not supported.
Return independent scores between 0 and 1 for all responsibilities. The scores represent evidence strength rather than probabilities and do not need to sum to one.
Return only one valid JSON object:

```
{
  "policy": score,
  "tools": score,
  "skills": score,
  "middleware": score,
  "memory": score,
  "no_patch": score,
  "rationale": "brief evidence-based explanation"
}
```

The rationale should explain the strongest responsibility, mention uncertainty when applicable, and justify whether NO-PATCH is preferred.