Self-Corrected Image Generation with Explainable Latent Rewards

Supplementary Material

Contents

A. Method Details

A.1. Transformer-Based Corrector Architecture

The Understanding-Guided Reinforcement Corrector (URC) is implemented as a lightweight transformer that operates on the latent feature grid of the frozen text-to-image model. Given a latent representation $z_0 \in \mathbb{R}^{C\times H\times W}$ and the prompt embedding e_p, URC predicts a residual $\Delta_\theta(z_0, e_p)$ applied before decoding.

Latent Tokenization. We flatten the spatial dimensions and treat each spatial location as a token: $Z_0 = \text{reshape}(z_0) \in \mathbb{R}^{(HW)\times C}$. 2D sine–cosine positional encodings are added to retain spatial structure.

Prompt Conditioning. The prompt embedding $e_p \in \mathbb{R}^d$ is projected to the latent dimension via $e'_p = W_p e_p$ and integrated using Feature-wise Linear Modulation (FiLM):

$$Z'_0 = \gamma(e'_p) \odot Z_0 + \beta(e'_p),$$

where γ and β are produced from e'_p through small MLPs. This allows the transformer to modulate latent tokens according to the semantic content of the prompt.

Transformer Layers. URC consists of six transformer blocks, each containing multi-head self-attention with 8 heads, cross-attention to prompt tokens, feedforward networks (hidden size $4C$), and pre-norm residual connections. Cross-attention explicitly injects linguistic structure, including object categories, colors, and relational phrases.

Residual Prediction. The output of the transformer is projected via a linear layer W_o back to latent dimensionality, reshaped to form the residual:

$$\Delta_\theta(z_0, e_p) \in \mathbb{R}^{C\times H\times W}.$$

Despite its transformer architecture, URC remains compact ($\leq$ 15M parameters), ensuring it refines semantics without overpowering the frozen generator.

A.2. Transformer-Based Latent Reward Projection

The latent reward projector R_ϕ is a transformer that maps corrected latent activations and prompt embeddings to interpretable rewards approximating image-level feedback.

Input Construction. R_ϕ receives the corrected latent tokens $Z_c \in \mathbb{R}^{(HW)\times C}$, the prompt token embeddings $\{e_{p,i}\}$, and the CMD-derived global semantic vector $g_{\text{cmd}} \in \mathbb{R}^d$, which is appended as an extra token:

$$X_0 = [Z_c; e_{p,1}; \ldots; e_{p,T}; g_{\text{cmd}}].$$

Transformer Design. R_ϕ has four transformer layers with 8-head multi-head attention, alternating self-attention and cross-attention, feedforward networks of size $4C$, and rotary positional embeddings for prompt tokens.

Reward Heads. After the transformer, the updated semantic token g'_{cmd} is passed through three linear layers to produce the latent reward vector:

$$r_{\text{latent}} = \left[W_{\text{count}} g'_{\text{cmd}},\ W_{\text{color}} g'_{\text{cmd}},\ W_{\text{pos}} g'_{\text{cmd}}\right] \in \mathbb{R}^3,$$

corresponding to counting, color, and position sub-rewards.

Training Objective. The projector is trained to minimize the L2 distance between predicted latent rewards and image-level task-specific rewards:

$$\mathcal{L}_{\text{proj}} = \sum_{i=1}^{3} \| r^{(i)}_{\text{latent}} - r^{(i)}_{\text{image}} \|_2^2,$$

enabling gradient-based optimization of URC even when the original image-level reward is non-differentiable.

B. Additional Qualitative Results

B.1. Text-to-Image Generation

Additional qualitative results for text-to-image generation using our model are shown in Fig.1. The prompts used for text-to-image generation, starting from top-left and going row-wise from left to right, are as follows:

1. A person walking alone on a quiet street at sunset.
2. A bowl of fresh fruit sitting on a kitchen counter.
3. A dog lying on a couch in a cozy living room.
4. A car parked beside a forest road in the morning.
5. A cup of coffee on a wooden table near a window.
6. A small boat floating on a calm lake at dawn.
7. A cyclist riding through a city park.
8. A marketplace stall filled with colorful vegetables.
9. A cat sitting on a windowsill looking outside.
10. A person reading a book in a quiet café.
11. A train passing through a snowy landscape.
12. A street food vendor cooking at night.

B.2. Editing Tasks

The prompts used to get results for the image editing task Figure 2 Figure 3 Figure 4, starting from the top-left and going row-wise from left to right, are as follows:

1. Add a small backpack resting on the ground next to the bicycle.
2. Add a small cushion under the cat.
3. Turn the baker into a man.
4. Add a small cup next to the pitcher.
5. Add a small dog walking next to the couple.
6. Remove half the people from the image, from the crosswalk.
7. Change the kite's color to bright red.
8. Place a small cooler next to the fisherman, near his feet.
9. Add a line of people near the food truck.
10. Add a straw to the glass.
11. Add an umbrella above the table.
12. Add a plate of vegetables on the grill.
13. Add a notebook next to the laptop.
14. Add a small trail sign beside the path.
15. Add a small seashell next to the sneakers.
16. Add a distant mountain in the background scenery.
17. Add a small bench next to the bus stop.
18. Add a single cloud in the sky above the mountain.
19. Add a small water bottle on the ground next to the person.
20. Add a few blueberries on the plate beside the pancakes.
21. Add a folded blanket at the side of the bed.
22. Add an open guitar case on the ground in front of him.
23. Add a price tag to one of the flower pots.
24. Remove the plant pots from the row.

C. Supervised Data Generation

Training prompts were generated using large language models to cover three semantic domains: color, position, and object count. For each domain, 10k candidate prompts were generated using multiple models (BLIP3o, SD3, and DEV). To ensure high-quality and semantically accurate prompts, all outputs were manually reviewed and filtered, retaining only those that correctly captured the intended attributes. Among the models tested, BLIP3o, SD3, and DEV consistently produced the most reliable and coherent prompts, showing strong consistency in describing colors, spatial relations, and object counts, whereas other models occasionally produced ambiguous or incomplete descriptions.

Rather than using a separate testing set, the supervised dataset is used to guide the generator itself: for each validated prompt, the model generates multiple images using different random seeds. This self-generation process, supervised by the validated dataset, provides diverse latent representations and ensures coverage of the semantic domains, without requiring an explicit test split. The resulting dataset thus serves both as a source of training supervision and a reference for controlled evaluation during model development.

D. Implementation and Reproducibility

D.1. Training Configuration

We train using AdamW with a learning rate of 1×10^{-4}, batch size 8 per GPU, PPO clipping ratio 0.2, gradient clipping 1.0, and a cosine learning rate schedule. Experiments are conducted on H100 80GB GPUs.

D.2. Hyperparameters

The URC transformer has 6 layers, R_ϕ has 4 layers, embedding size 1024, residual scale $\alpha = 0.8$, and task-weight modulation range $[0.5, 2.0]$.

E. Additional Experiments

This section provides additional quantitative experiments complementing the analyses in the main paper.

E.1. Process-Level Validation and Attribution

Causal Evidence. In Sec. 4.3.1, we demonstrate that improvements are directly driven by our formulation via three validation tests: (i) **Spatial Causal Link:** Masking high-activation LAM regions leads to a 6.3% drop in CLIP-Score, proving these regions are where alignment is fixed; (ii) **Signal-to-Gain Correlation:** A Spearman correlation of ρ=0.71 between token contribution magnitudes and reward gains confirms that latent rewards directly drive the correction; (iii) **Consistency:** A Jaccard similarity of 0.68

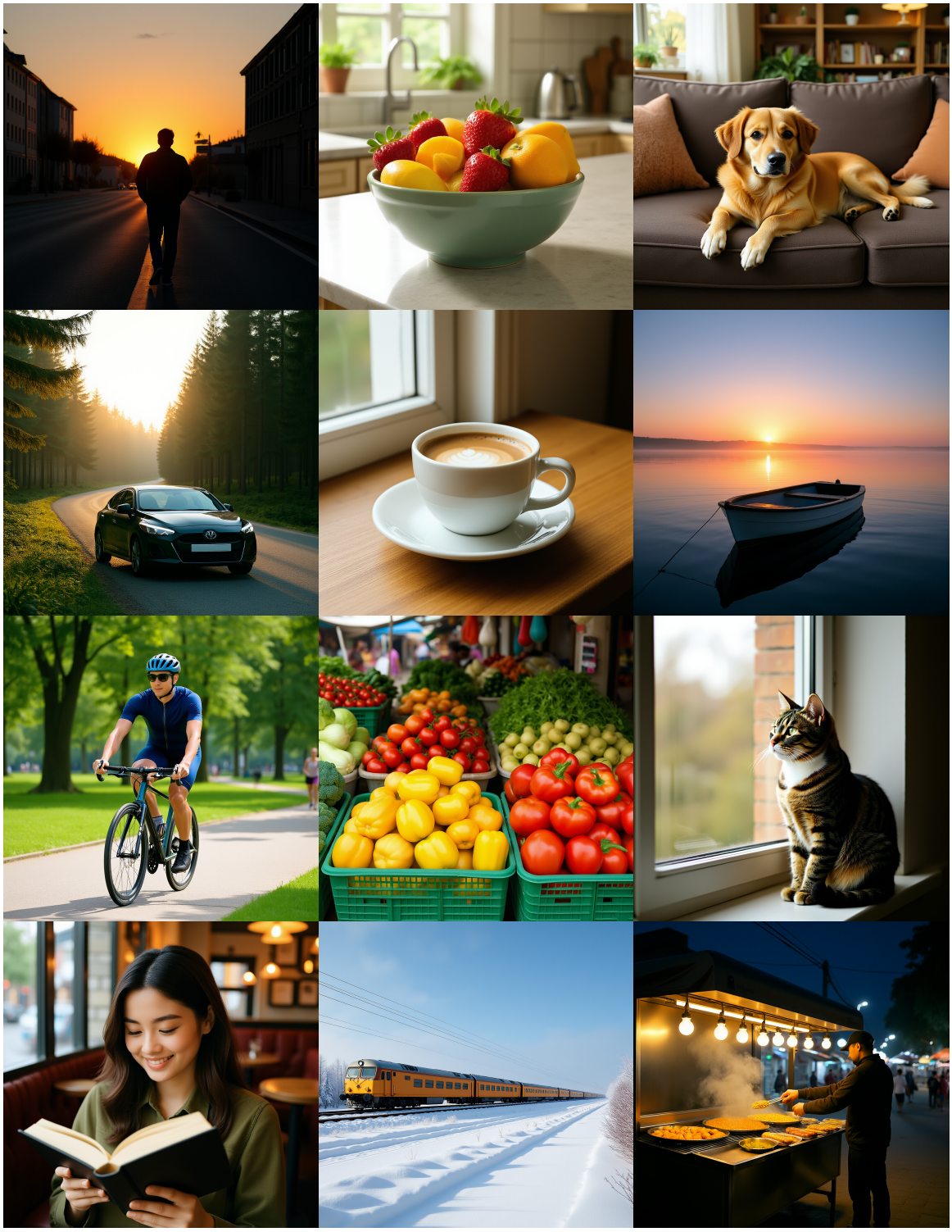

Figure 1. **Qualitative results for text to image generation.**

across prompts shows stable, predictable correction patterns.

Incremental Dynamics. To quantify process-level dynamics, we analyzed semantic trajectories on a subset of 500 prompts from GenEval by sampling latent sub-rewards at irregular intervals through our differentiable projector R_Φ. We observe a 26.34% average reward increase between denoising steps 7 and 38, with a high Pearson correlation

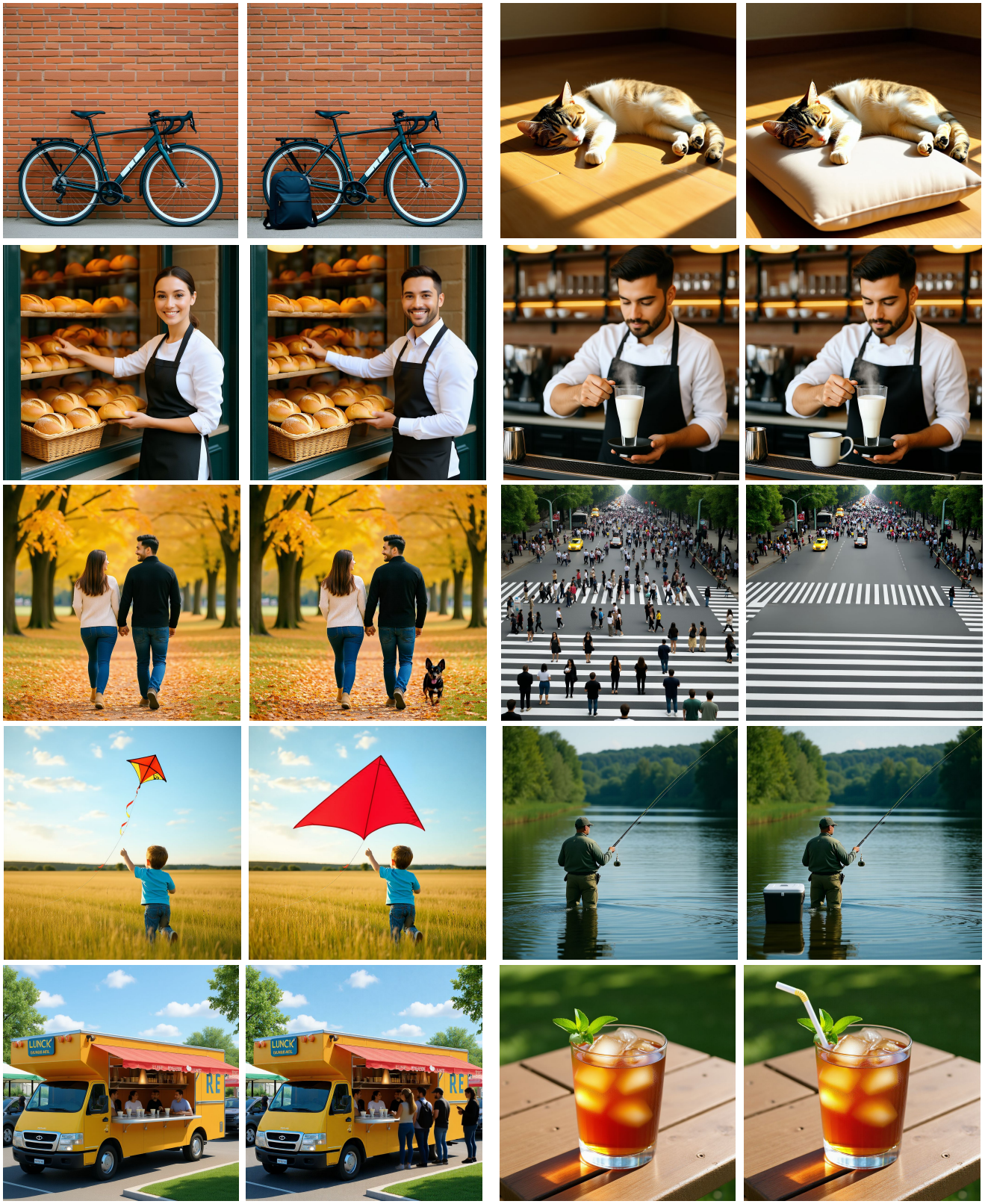

Figure 2. Qualitative editing results-1.

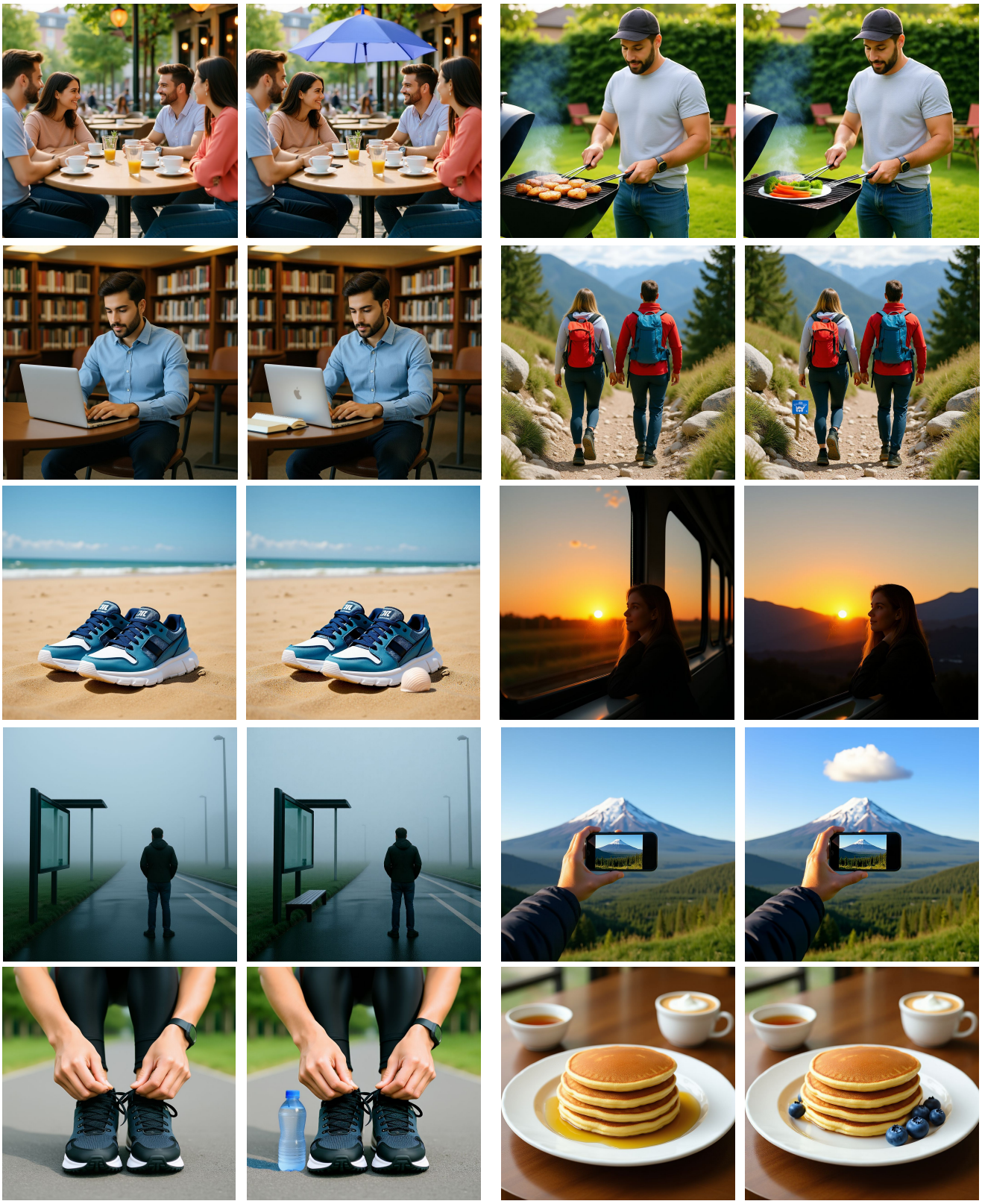

Figure 3. Qualitative editing results-2.

Figure 4. Qualitative editing results-3.

Table 1. Ablation study of reward components.

Method Variant	GenEval (%)	DPG-Bench (%)
Full xLARD Framework	**81.24**	**86.72**
w/o Counting Reward	76.45 (−4.79)	83.10 (−3.62)
w/o Color Reward	78.92 (−2.32)	84.55 (−2.17)
w/o Position Reward	79.15 (−2.09)	84.88 (−1.84)

(r=0.827) between these incremental gains and final performance. Specifically, Counting rewards stabilize rapidly within the first 18 steps to establish structural layout, while Color and Position rewards provide continuous refinement through step 43. This temporal analysis confirms that xLARD performs active, multi-stage steering throughout the generative process rather than a singular post-hoc correction.

E.2. Sub-Reward Analysis and Ablation

Our modular design is a deliberate choice to ensure interpretability and avoid the black-box nature of aggregate rewards. While the sub-rewards are specific, they address the most frequent failure modes in T2I models and provide a framework that is easily extendable to other attributes. The ablation of reward roles is reported in Table 1.

E.3. Comparison with Plug-and-Play Guidance Baseline

For latent editing comparisons, we include a Plug-and-Play Guidance (PPGD) baseline adapted to our architecture; most existing latent editing methods are U-Net–based and do not generalize to unified multimodal transformers. On GenEval and DPG-Bench, PPGD achieves 77.04% and 83.54%, below our method (81.29% and 86.45%). These results provide additional comparative context against recent SOTA techniques.

E.4. Confidence-Based Modulation (CMD) Analysis

The Confidence Head ω is a lightweight MLP trained via PPO to predict sub-reward reliability. Rather than static weights, it acts as a dynamic gating mechanism to suppress irrelevant signals. Our analysis shows removing CMD causes a 3.3% GenEval drop due to "gradient interference," where unmodulated rewards compete for latent updates.

E.5. Additional Technical Details

Reward Projection. The differentiable projector R_ϕ does not backpropagate through the decoder. Instead, it is trained via supervised regression to approximate non-differentiable image-level rewards, enabling stable gradient flow entirely within latent space during corrector optimization.

Corrector Behavior. The corrector Δ_θ is explicitly constrained to produce small-magnitude residual updates via the scaling factor α and PPO regularization, ensuring localized semantic refinement rather than latent overwriting or re-generation.

Counting Implementation. Connected-component analysis is applied after adaptive thresholding and morphological filtering of token attention maps, which removes spurious

activations and yields stable object-count estimates across prompts.

E.6. Runtime and Memory

Training takes 4 h/epoch, inference matches base-generator runtime as mentioned in the paper discussion, and peak memory is approximately 72 GB at batch size 8. Reference-prompt construction details are provided in the supplementary material.