Appendices

Contents

A. Reference Velocity Field Formulation

We derive the reference velocity field in Eq. (4) from the Euler discretization of the rectified flow ODE. The update rule of Euler discretization is:

$$Z_{t_{i-1}} = Z_{t_i} + (t_{i-1} - t_i)\, v_\theta(Z_{t_i}, t_i, \mathbf{c}), \tag{17}$$

which provides a finite-difference approximation of the instantaneous velocity:

$$v_\theta(Z_{t_i}, t_i, \mathbf{c}) = \frac{Z_{t_{i-1}} - Z_{t_i}}{t_{i-1} - t_i}. \tag{18}$$

For brevity, we denote $v_{t_j} \equiv v_\theta(Z_{t_j}, t_j, \mathbf{c})$. Applying the update rule iteratively from step i down to step 1 yields:

$$Z_{t_{i-1}} - Z_{t_i} = (t_{i-1} - t_i)\, v_{t_i}, \tag{19}$$

$$Z_{t_{i-2}} - Z_{t_{i-1}} = (t_{i-2} - t_{i-1})\, v_{t_{i-1}}, \tag{20}$$

$$\vdots$$

$$Z_{t_0} - Z_{t_1} = (t_0 - t_1)\, v_{t_1}. \tag{21}$$

Summing over all remaining steps from i down to 1 produces the telescoping relation:

$$Z_{t_0} - Z_{t_i} = \sum_{j=1}^{i} (t_{j-1} - t_j)\, v_{t_j}. \tag{22}$$

Since the terminal state Z_{t_0} corresponds to the context image Z_0^{ref}, we have:

$$Z_0^{\text{ref}} - Z_{t_i} = \sum_{j=1}^{i} (t_{j-1} - t_j)\, v_{t_j}. \tag{23}$$

We approximate the velocity field to be constant over the remaining interval $[t_0, t_i]$, setting it equal to v_{ti}^{ref}. Under this constant-velocity approximation:

$$Z_0^{\text{ref}} - Z_{t_i} = \sum_{j=1}^{i} (t_{j-1} - t_j)\, v_{t_i}^{\text{ref}} = (t_0 - t_i)\, v_{t_i}^{\text{ref}}, \tag{24}$$

which directly yields:

$$v_{t_i}^{\text{ref}} = \frac{Z_0^{\text{ref}} - Z_{t_i}}{t_0 - t_i}, \tag{25}$$

which recovers Eq. (4) from the main text. This expression represents the average velocity required to traverse from Z_{t_i} to the reference image Z_0^{ref} over the remaining time $(t_0 - t_i)$. This formulation is analogous to the average velocity approach proposed in MeanFlow [16]. Geometrically, this induces a straight-line trajectory in the latent space that preserves the high-frequency features of the context image.

B. VLM-Based Evaluation Metric Details

This section details our VLM-based evaluation metrics, including instruction following, visual consistency, and perceptual quality.

Instruction Following. For each editing turn k, we evaluate how well the generated image $X^{[k+1]}$ follows the corresponding instruction $p^{[k]}$ by comparing it against the previous image $X^{[k]}$. We employ GPT-4o as the evaluator with a carefully designed prompt that instructs the model to focus exclusively on instruction adherence. The evaluation prompt consists of three components: (1) a message emphasizing objective evaluation, (2) the editing instruction $p^{[k]}$, and (3) the source image $X^{[k]}$ and the edited image $X^{[k+1]}$. The model returns a score in $[0, 1]$, where 1 indicates perfect instruction fulfillment and 0 indicates complete failure. The evaluation prompt is specifically designed to assess instruction adherence in isolation, explicitly instructing the evaluator to disregard visual quality.

Visual Consistency. Our consistency metric comprises subject consistency and background consistency. For subject consistency, instead of detecting all objects indiscriminately, we first leverage GPT-4o to infer which subjects should be present after each editing operation. Specifically, we provide GPT-4o with: (1) the original image $X^{[1]}$, and (2) the cumulative editing instructions $\{p^{[1]}, p^{[2]}, \ldots, p^{[k]}\}$. The model reasons about the editing semantics and outputs a list of expected subject categories $\mathcal{S}^{[k+1]} = \{s_1, s_2, \ldots, s_n\}$ (e.g., "person", "blue car", "tree"). This inference-based approach enables editing-aware subject tracking, as opposed to naive object detection.

For each inferred subject category $s_i \in \mathcal{S}^{[k+1]}$, we use it as the text prompt for GroundingDINO [31] to localize the corresponding object instance in images $X^{[1]}$, $X^{[k]}$, and $X^{[k+1]}$. We set the box confidence threshold to 0.25, the text matching threshold to 0.25, and enable `keep_top1_per_label` to retain only the highest-confidence detection per subject category, thereby reducing false positives. We denote the set of successfully detected object instances as $\mathcal{O}$.

We compute consistency for object instances that are successfully detected in both source and target images. Specifically, we measure: (1) original-to-current consistency: consistency of objects present in both $X^{[1]}$ and $X^{[k+1]}$, and (2) previous-to-current consistency: consistency of objects present in both $X^{[k]}$ and $X^{[k+1]}$.

For each common object instance $o \in \mathcal{O}$, we extract its region-of-interest (ROI) based on the detected bounding box and compute two types of features: (1) DINOv2 features: We resize each ROI to 224×224 pixels and feed it to DINOv2 [39]. Consistency is measured via cosine similarity:

$$\text{sim}_{\text{DINOv2}}(o) = \frac{\mathbf{f}_{\text{src}}(o) \cdot \mathbf{f}_{\text{tgt}}(o)}{\|\mathbf{f}_{\text{src}}(o)\| \, \|\mathbf{f}_{\text{tgt}}(o)\|}, \tag{26}$$

where $\mathbf{f}_{\text{src}}(o)$ and $\mathbf{f}_{\text{tgt}}(o)$ are DINOv2 features of object o in the source and target images, respectively. (2) L1 pixel distance: We compute the normalized L1 distance between the resized ROI patches (with pixel values normalized to $[0, 1]$) and convert it to a similarity score:

$$\text{sim}_{\text{L1}}(o) = 1 - \frac{1}{HWC} \sum_{h,w,c} |P_{\text{src}}^{(o)}(h,w,c) - P_{\text{tgt}}^{(o)}(h,w,c)|, \tag{27}$$

where $P_{\text{src}}^{(o)}$ and $P_{\text{tgt}}^{(o)}$ are the RGB patches of object o with spatial dimensions $H \times W$ and $C = 3$ color channels.

For each similarity type, the subject consistency score is computed by averaging over all common object instances:

$$\text{Consistency}_{\text{subject}}^{\text{type}} = \frac{1}{|\mathcal{O}|} \sum_{o \in \mathcal{O}} \text{sim}_{\text{type}}(o), \tag{28}$$

where $\mathcal{O}$ denotes the set of common object instances detected in both source and target images, and type $\in \{\text{DINOv2}, \text{L1}\}$.

For background consistency, when the editing instruction does not involve background modification (as determined by intent parsing), we evaluate background preservation using GPT-4o. The evaluation protocol consists of two stages: (1) we explicitly instruct the model to focus only on background regions and ignore foreground objects identified during subject consistency evaluation, and (2) the model assesses background consistency across four dimensions, including *layout* (preservation of spatial arrangement), *texture* (consistency of material properties), *lighting* (color temperature and shadow consistency), and *artifacts* (absence of boundary artifacts or distortions).

Perceptual Quality. We assess the visual quality of each generated image $X^{[k+1]}$ using two complementary quality models. We obtain GPT-4o-based scores by prompting the model to evaluate six dimensions, including *aesthetics*, *realism*, *sharpness*, *exposure*, *artifacts*, and *composition*, each scored in $[0, 1]$, along with an overall quality score. We then compute HPSv3 [35] scores as a learned perceptual quality metric.

C. Additional Qualitative Comparisons

Figures 15 to 20 present additional qualitative comparisons across diverse multi-turn editing scenarios, including complex background modifications, fine-grained attribute changes, action and pose alterations, object manipulations and style transfers. These extended results consistently reveal the limitations of existing approaches. Specifically, Qwen-Image and FLUX.1 Kontext exhibit progressive quality degradation with accumulated artifacts and body deformations. VINCIE, Seedream 4.0, and Bagel fail to preserve facial details and introduce significant visual artifacts. While MTC maintains reasonable image quality, it demonstrates limited instruction-following capability for complex edits. Nano Banana achieves competitive performance but suffers from noticeable color shifts across editing iterations. Across all scenarios, our method achieves the superior overall performance, successfully balancing accurate instruction following, robust subject consistency, and high perceptual quality.

D. Additional Quantitative Results

To provide a more comprehensive evaluation, we present results on three additional metrics: PSNR, SSIM [54], and DINO-Sim [39]. PSNR measures pixel-level reconstruction quality between the edited and original images. SSIM evaluates structural information preservation, including luminance, contrast, and texture patterns. DINO-Sim measures high-level semantic consistency via cosine similarity between DINO features. All metrics are computed between the edited image $X^{[k+1]}$ at turn k and the original unedited image $X^{[1]}$.

Table 2 presents the cumulative mean of PSNR, SSIM, and DINO-Sim across progressive editing turns for all compared methods. FreqEdit demonstrates substantial and consistent improvements when integrated with different base models. Both FLUX.1 Kontext and Qwen-Image exhibit significant gains across all three metrics when equipped with FreqEdit. As discussed in Section 5.2, MTC achieves high PSNR values due to its limited ability to execute editing instructions, resulting in edited images that remain largely unchanged from the original. Notably, Qwen-Image

Table 2. **Additional quantitative results using PSNR, SSIM and DINO-Sim.** We report cumulative averages computed from turn 1 through each specified turn (1, 4, 7, 10) across 10 sequential edits. Bold indicates the best results, and underlined values denote the second-best results.

Methods	Turn 1			Turn 4			Turn 7			Turn 10		
	PSNR↑	SSIM↑	DINO-Sim↑	PSNR↑	SSIM↑	DINO-Sim↑	PSNR↑	SSIM↑	DINO-Sim↑	PSNR↑	SSIM↑	DINO-Sim↑
Bagel	21.615	0.839	0.938	16.953	0.672	0.850	14.216	0.544	0.769	12.505	0.454	0.695
MTC	17.346	0.528	0.860	16.139	0.483	0.835	15.618	0.466	0.816	**15.291**	0.456	0.799
Nano Banana	20.243	0.662	0.963	16.096	0.512	**0.924**	14.210	0.446	0.873	12.983	0.403	0.816
Seedream 4.0	19.885	0.737	0.957	16.916	0.603	0.878	14.749	0.514	0.790	13.259	0.458	0.706
VINCIE	17.648	0.687	0.881	14.766	0.587	0.790	13.095	0.510	0.717	12.121	0.459	0.658
FLUX.1 Kontext + FreqEdit	21.886	0.844	0.961	17.649	0.707	0.906	15.105	0.591	0.850	13.519	0.514	0.792
FLUX.1 Kontext	18.998	0.724	0.953	15.214	0.597	0.884	13.228	0.510	0.817	11.942	0.446	0.746
Qwen-Image + FreqEdit	**23.548**	**0.892**	0.962	**19.349**	**0.798**	0.920	**16.577**	**0.706**	**0.877**	14.791	**0.632**	**0.823**
Qwen-Image	17.765	0.713	**0.965**	14.397	0.602	0.909	12.737	0.533	0.845	11.694	0.485	0.784

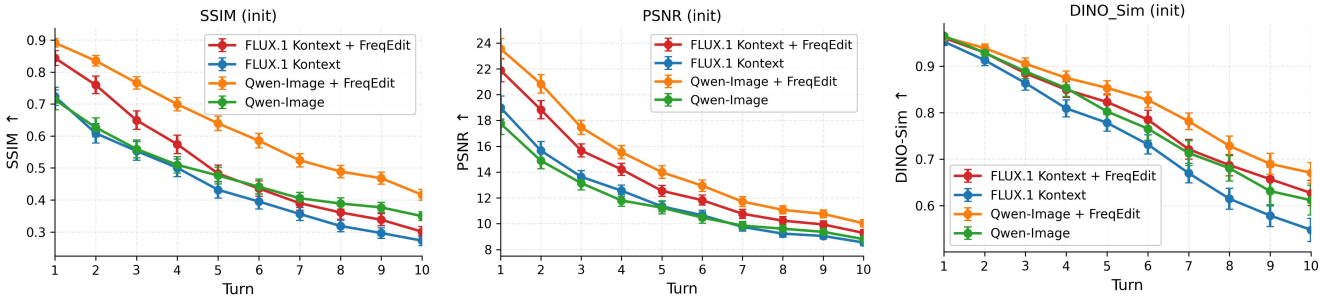

Figure 7. **Per-turn metrics across 10 sequential editing steps.** We report SSIM (left), PSNR (middle), and DINO-Sim (right) at each turn k, all computed by comparing the edited image $X^{[k+1]}$ with the original image $X^{[1]}$. For clarity, we only show our FreqEdit-enhanced models and their corresponding base models. FreqEdit consistently improves preservation of both low-level details (PSNR, SSIM) and high-level semantics (DINO-Sim) relative to the original image across all editing turns for both base models.

+ FreqEdit achieves the best results across most metrics and editing turns.

We further provide quantitative results at each individual editing turn relative to the original image in Figure 7. As shown, FreqEdit-equipped models consistently maintain higher similarity scores at every turn, with improvements ranging from 1–3 dB in PSNR, 0.05–0.15 in SSIM, and 0.05–0.1 in DINO-Sim across different turns. Importantly, FreqEdit-equipped models not only achieve higher absolute scores but also exhibit more stable degradation patterns throughout the editing sequence, demonstrating superior consistency preservation during iterative editing.

E. User Study Details

We conduct a comprehensive user study to evaluate the perceptual quality of our method against seven baseline approaches. This section provides detailed information on our study protocol and scoring mechanism.

We design an online questionnaire-based user study where participants rank multiple methods based on their editing results. Participants are instructed to comprehensively consider three key aspects when making their judgments: *aesthetics* (visual appeal), *instruction following* (accuracy in executing given instructions), and *consistency* (preservation of unedited regions and object identity across turns). For each question in the survey, participants are shown: 1) the source (unedited) image $X^{[1]}$, 2) edited results at three intermediate turns: $X^{[5]}$, $X^{[8]}$, and $X^{[11]}$, and 3) all editing instructions $\{p^{[1]}, p^{[2]}, \ldots, p^{[10]}\}$ applied sequentially. We collect a total of 60 completed survey responses.

Given the cognitive difficulty of ranking all 9 methods simultaneously, we adopt a random sampling strategy. For each question, we randomly select 7 out of 9 methods and present their results to participants, who then rank these methods from best (rank 1) to worst (rank 7) based on overall quality. This choice of 7 methods balances cognitive load with questionnaire efficiency and reliability.

To obtain a unified score for each method, we convert the collected rankings to a 9-point scale. Let n_k denote the number of times a method receives rank k (where $k \in \{1, 2, \ldots, 7\}$). We assign descending weights to each rank using the weight vector:

$$\mathbf{w} = [9, 8, 7, 6, 5, 4, 3], \tag{29}$$

where rank 1 receives weight 9 and rank 7 receives weight

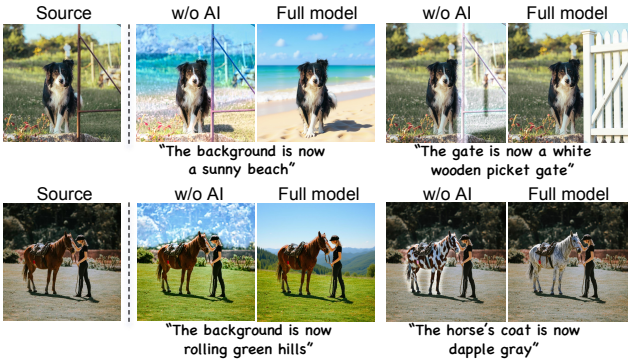

Figure 8. **Additional ablation results for Adaptive Injection (AI).** The adaptive injection strategy modulates injection strength based on semantic correspondence between editing and reference velocity fields. When using uniform injection (w/o AI), semantically modified regions suffer from over-preservation, resulting in incomplete transformations for beach backgrounds, white gates, rolling hills, and horse coat patterns.

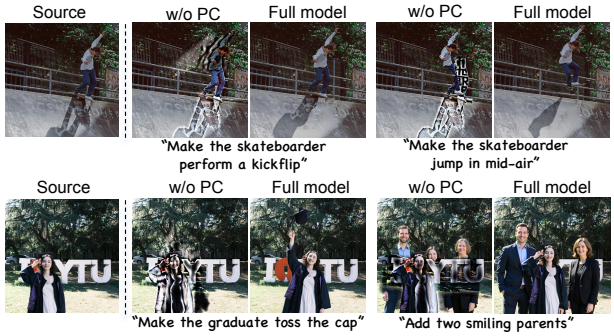

Figure 9. **Additional ablation results for Path Compensation (PC).** Without PC, high injection strength introduces ghosting artifacts where conflicting visual elements from both editing and reference velocity fields manifest simultaneously (e.g., duplicated skateboarder and graduate poses, and ghost-like parent figures). Our path compensation strategy eliminates these artifacts while maintaining strong high-frequency injection.

3. The final score S for a given method is computed as:

$$S = \frac{9n_1 + 8n_2 + 7n_3 + 6n_4 + 5n_5 + 4n_6 + 3n_7}{\sum_{k=1}^{7} n_k}. \quad (30)$$

This formulation ensures that methods consistently ranked higher receive proportionally higher scores, with the weight assignment reflecting relative preference while mapping rankings to an intuitive 9-point scale.

F. Additional Ablation Study

To validate the effectiveness of each component in our framework, we conduct ablation studies by systematically removing individual components. We evaluate three variants: without adaptive injection strategy, without path compensation mechanism, and without quality guidance. In addition, we analyze the sensitivity of FreqEdit to its three key hyperparameters (α_0, γ, and λ), varying each individually while fixing the others to their default values.

Figure 10. **Additional ablation results for Quality Guidance (QG) on FLUX.1 Kontext.** After several editing iterations, noise artifacts accumulate progressively in the generated images. The native model exhibits severe noise degradation (column 2), while adding QG to the native model effectively suppresses noise accumulation (column 3). When combined with our wavelet-based injection framework, QG further enhances visual fidelity by eliminating residual noise artifacts (comparing columns 4 and 5).

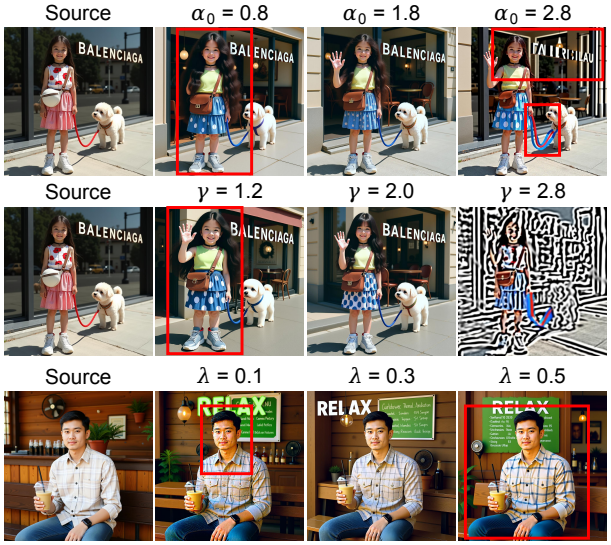

Figure 11. **Hyperparameter sensitivity analysis on FLUX.1 Kontext.** Each row varies one hyperparameter while fixing the others to their default values ($\alpha_0 = 1.6$, $\gamma = 2.0$, $\lambda = 0.3$); red boxes highlight failure regions. Insufficient α_0 leads to subject deformation, while excessive α_0 causes editing failure due to reference velocity dominance. γ jointly controls injection strength and spatial map contrast: too small causes subject deformation; too large over-amplifies high-frequency components, corrupting image structure. For λ, too small leaves residual noise artifacts, while too large introduces over-saturation.

Adaptive Injection. Figure 8 presents qualitative comparisons validating our adaptive injection strategy (Sec-

tion 4.3). Without adaptive injection, uniform injection strength often leads to over-preservation artifacts in semantically modified regions. In the top row, the model overly preserves the original background and gate geometry, preventing full transition to a sunny beach scene (column 2) and inhibiting the reconstruction of a white wooden picket gate (column 4). Similarly, in the bottom row, large portions of the original background remain intact instead of forming rolling green hills (column 2), and the horse's coat retains much of its initial appearance, failing to exhibit a coherent dapple-gray pattern (column 4). In contrast, our adaptive injection approach enables faithful editing in modified regions while preserving details elsewhere.

Path Compensation. Figure 9 demonstrates the effectiveness of our path compensation strategy (Section 4.4). Without path compensation, high injection strength leads to ghosting artifacts where visual elements from both editing and reference velocity fields appear simultaneously. The kickflip scene (top row, second column) shows duplicated skateboarder poses with overlapping limbs, the mid-air jump (fourth column) produces ghost figures, the cap toss (bottom row, second column) exhibits duplicated body configurations, and the parent addition (fourth column) displays semi-transparent overlapping figures. These results confirm that path compensation is crucial for stabilizing the editing trajectory and preventing ghosting artifacts.

Quality Guidance. The quality guidance mechanism (Section 4.5) addresses noise accumulation across editing iterations, particularly for models like FLUX.1 Kontext. Figure 10 demonstrates its effectiveness through two representative examples after multiple editing turns. We compare four configurations: native model baseline, native model with quality guidance, our framework without quality guidance (w/o QG), and our complete model. The native FLUX.1 Kontext (column 2) exhibits pronounced noise artifacts, particularly in skin textures. Applying quality guidance alone (column 3) substantially reduces these artifacts by blending the editing velocity with an auxiliary velocity from the original high-quality image during final denoising steps. When integrated into our complete framework, quality guidance (column 5) further refines visual quality compared to the variant without it (column 4), producing clean results that maintain both high-frequency details and low noise levels.

Hyperparameter Sensitivity. We analyze the sensitivity of FreqEdit to its three key hyperparameters on FLUX.1 Kontext, varying each individually while keeping the others fixed at their default values ($\alpha_0 = 1.6$, $\gamma = 2.0$, $\lambda = 0.3$). As shown in Figure 11, α_0 and γ jointly govern the magnitude and spatial distribution of high-frequency feature injection, while λ independently controls the quality guidance strength applied during the final denoising steps.

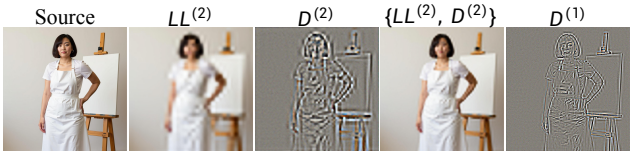

Figure 12. **Latent-space frequency alignment.** Images are reconstructed by retaining only the specified frequency band(s) from the 2-level DWT of the latent representation and decoding via VAE. $LL^{(2)}$ captures global structure and color; $D^{(2)}$ encodes coarser-scale edges and textures; $\{LL^{(2)}, D^{(2)}\}$ together yield a sharp reconstruction approximating the full image; $D^{(1)}$ captures the finest details such as hair strands and fabric textures. The consistent correspondence confirms that latent-space DWT reliably reflects image-space frequency structure.

α_0 controls the overall injection magnitude in Eq. (12). When α_0 is too small, the injected high-frequency signal is insufficient to counteract progressive degradation, leading to subject deformation in later editing turns. Conversely, when α_0 is too large, the reference velocity dominates the editing velocity, causing editing failure where the intended semantic changes are suppressed. Our method is robust for $\alpha_0 \in [1.5, 2.5]$.

γ jointly controls the overall injection strength and the contrast of the spatially-adaptive injection map (Eq. (12)). A small γ yields both weaker injection and a low-contrast map that fails to sufficiently differentiate between semantically stable and modified regions, resulting in subject deformation due to inadequate high-frequency reinforcement. A large γ amplifies injection strength and sharpens spatial boundaries, which can suppress desired semantic transformations and introduce visible transition artifacts. Our method is robust for $\gamma \in [1.5, 2.5]$.

λ controls the blending strength of the quality guidance mechanism (Eq. (16)), which is applied during the final denoising steps to suppress accumulated noise artifacts. When λ is too small, residual noise artifacts remain visible in the output, particularly in fine-grained regions such as skin textures. When λ is too large, the auxiliary velocity exerts excessive influence, causing over-saturation in the output image. Our method is robust for $\lambda \in [0.2, 0.4]$.

G. Latent-Space Frequency Alignment

FreqEdit operates on high-frequency components extracted from the latent velocity field via DWT. A key assumption underlying this design is that frequency decomposition in the latent space exhibits consistent behavior with that in the image space. To empirically validate this assumption, we apply 2-level DWT to the latent representation of a source image and selectively reconstruct images from individual frequency bands via VAE decoding.

As shown in Figure 12, the reconstructed images re-

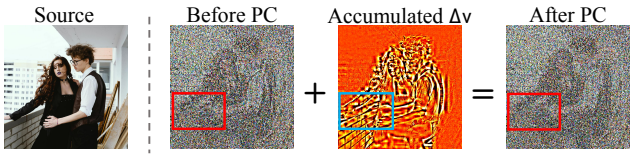

Figure 13. **Visualization of path compensation at an intermediate denoising step.** The corrected latent Z_{t_i} after high-frequency injection exhibits residual structure conflicting with the editing objective (red box). The trajectory buffer $B = \sum(t_{i-1} - t_i) \cdot \Delta v_{t_i}$, visualized as a heatmap, carries a structured correction signal in the corresponding region (blue box) that counteracts this residual structure. Adding $\mathcal{B}$ to the latent yields a clean latent aligned with the intended edit. Please zoom in for details.

veal a clear correspondence between latent-space frequency bands and their visual counterparts in image space. The second-level low-frequency component $LL^{(2)}$ captures the global structure and coarse appearance of the scene, including the overall layout, color distribution, and subject silhouette, albeit with noticeable blurring. The second-level high-frequency component $D^{(2)}$ encodes coarser-scale structural edges and textural patterns, such as body contours and clothing boundaries, without retaining global color information. Combining $LL^{(2)}$ and $D^{(2)}$ recovers a substantially sharper reconstruction that approximates the full image, confirming that these two levels together account for the dominant visual content. The first-level high-frequency component $D^{(1)}$ captures the finest-grained details, including hair strands, fabric textures, and subtle edge structures. These observations confirm that latent-space frequency components behave consistently with their image-space counterparts, providing a principled justification for performing high-frequency feature injection directly in the latent space.

H. Path Compensation Visualization

While Section 4.4 establishes the mathematical equivalence between the path compensation trajectory and the editing velocity, we provide here a step-level visualization to intuitively demonstrate how path compensation resolves the semantic conflict introduced by high-frequency injection during denoising.

Figure 13 visualizes the latent state at an intermediate denoising timestep t_i. Before applying path compensation, the corrected latent Z_{t_i} exhibits residual structure from the high-frequency injection that conflicts with the editing objective, as highlighted by the red box. The trajectory buffer B (Eq. (14)) accumulates the per-step velocity divergence weighted by the timestep interval:

$$B \leftarrow B + (t_{i-1} - t_i) \cdot \Delta v_{t_i}, \quad \text{where} \quad \Delta v_{t_i} = v_{t_i}^{\text{edit}} - v_{t_i}^{\text{corr}}.$$

As shown in the heatmap, B carries a structured correction signal in the region (blue box) corresponding to the residual structure highlighted in the red box, counteracting the conflict introduced by high-frequency injection. Adding B to the latent effectively suppresses this residual structure, yielding a clean latent aligned with the intended edit. This visualization confirms that path compensation periodically realigns the denoising trajectory with the editing objective while preserving the high-frequency information injected in preceding steps.

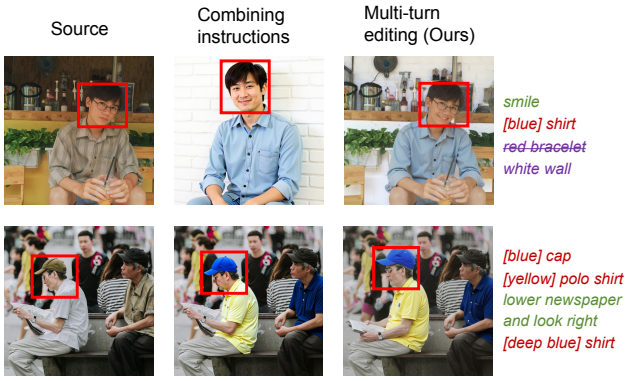

Figure 14. **Comparison between combining instructions and multi-turn editing.** Red boxes highlight subject regions for identity comparison. Combining all instructions into a single prompt causes unintended subject identity drift and background changes, despite these regions not being targeted by any instruction. Our multi-turn approach applies instructions incrementally, preserving subject identity while accurately executing each individual edit.

I. Multi-Turn Editing vs. Combined Prompts

A natural question arises as to whether multi-turn editing can be replaced by combining all instructions into a single prompt, which would eliminate the need for iterative processing. While combining instructions may work in simple cases, we argue that multi-turn editing is both practically necessary and technically preferable for the following reasons.

First, real-world creative workflows are inherently iterative. Users typically generate multiple candidates at each editing step and select the most satisfactory result before proceeding to the next modification. This interactive process cannot be replicated by a single-pass prompt. Second, many practical scenarios involve secondary creative work on existing images, *e.g.*, refining a previously generated AIGC result, which is by nature a multi-turn process. Third, prior works have established multi-turn image editing as an independent research challenge [41, 62], and the FLUX.1 Kontext technical report [27] explicitly identifies multi-turn editing as a key capability to develop.

From a technical standpoint, combining multiple editing instructions into a single prompt substantially increases the

semantic complexity of the generation task, making it difficult for the model to disentangle and faithfully execute each individual operation while simultaneously preserving subject identity. As shown in Figure 14, combining instructions often leads to significant subject identity drift: in the top row, the subject's facial appearance and surrounding background change substantially; in the bottom row, the left subject (red box) undergoes unintended appearance alterations. In contrast, our multi-turn editing approach applies instructions incrementally, enabling the model to maintain consistent subject identity across all editing turns while accurately executing each individual instruction.

J. More Discussion on Limitations and Future Work

Dependency on source image quality. FreqEdit relies on the high-frequency details present in the context image to compensate for progressive degradation across editing turns. When the initial source image is of low quality or already lacks fine-grained details, the high-frequency components available for injection are inherently limited, which constrains the effectiveness of our preservation mechanism. This limitation is fundamental to any reference-based injection approach and is not specific to our method.

Out-of-distribution behavior. Our method modifies the velocity field at inference time without any model fine-tuning, which may steer the denoising trajectory into regions that lie outside the model's training distribution. The adaptive injection strategy (Section 4.3) and path compensation mechanism (Section 4.4) are specifically designed to mitigate such deviations by spatially modulating injection strength, periodically realigning the trajectory with the editing objective. Nevertheless, these mechanisms reduce rather than eliminate the risk of out-of-distribution behavior, and edge cases may still arise for inputs with unusual appearance or highly atypical editing instructions.

Reduced effectiveness for large-area edits. The adaptive injection map derives spatial injection weights from the divergence between the editing and reference velocity fields, which provides reliable region discrimination when edits are localized. When a single instruction targets a large portion of the image, *e.g.*, a global style transfer, the divergence map loses discriminative power and the injection strength becomes less spatially precise. We note, however, that multi-turn editing workflows typically decompose complex modifications into a sequence of incremental changes, each affecting a limited region, which naturally mitigates this limitation in practice.

Potential for training-based extensions. As a training-free framework, FreqEdit may not be optimized for specific editing types or distributions. The high-frequency preservation principles introduced in this work may nonetheless serve as a useful foundation for future training-based approaches, for instance by incorporating a frequency-aware consistency objective into the training loss, or by using our method to generate high-quality multi-turn training pairs for supervised fine-tuning.

K. Multi-Turn Editing Instruction Generation

We design a comprehensive instruction template (see Figure 21) to guide a VLM in generating structured image editing instructions. The template instructs the VLM to: (1) generate a detailed initial description capturing attributes such as color, shape, material, and spatial arrangement; (2) produce exactly 10 semantically coherent editing instructions that simulate a realistic editing workflow.

Each editing instruction is constrained to execute one primary operation from nine predefined categories: *Subject Addition*, *Subject Removal*, *Subject Replacement*, *Background Change*, *Portrait Beautification*, *Color Alteration*, *Material Modification*, *Motion Change*, and *Style Transfer*. To ensure editability while preserving image identity, we enforce strict syntax rules: (i) mandatory preservation clauses specifying unchanged attributes, (ii) prohibition of ambiguous pronouns requiring precise descriptive references, (iii) a single-subject preservation rule preventing direct modifications to solitary primary objects.

To enable fair comparison with text-to-image editing baselines that require explicit source and target descriptions rather than editing instructions (*e.g.*, inversion-based methods like MTC [62]), we design a secondary VLM instruction template for sequential description transformation (see Figure 22). Given the initial image description and the sequence of 10 editing instructions generated by the previous template, this instruction guides the VLM to produce 11 complete, standalone textual descriptions through iterative application. Starting from the initial description (description$[0]$), each editing instruction $p^{[i]}$ is applied sequentially to the previous description (description$[i-1]$) to generate the next state (description$[i]$). Crucially, each output description is self-contained and cumulative, incorporating all modifications from $p^{[1]}$ through $p^{[i]}$ while preserving unmodified attributes from the previous state. The resulting sequence enables the construction of 10 consecutive source-target pairs: $(\text{descriptions}[i], \text{descriptions}[i+1])$ for $i = 0$ to 9, providing inversion-based editing models with the paired text descriptions required for their inference pipeline.

Example of Generated Editing Instructions. We provide a representative example of the structured editing instructions generated by our VLM template. Given an input image depicting a young girl with a white dog on a leash in front of a storefront (see Figure 16), the VLM produces the

following sequence of 10 editing prompts:

1. *The young girl with long dark hair is now raising her hand in a friendly wave towards the camera, while maintaining her same facial expression, clothing, and position.*
2. *The bright red leash is now a vibrant royal blue color, preserving the appearance of the young girl with long dark hair and the fluffy white dog.*
3. *The young girl's white platform sneakers are now made of a shiny, silver glitter material, keeping the same shape and style of the shoes.*
4. *A luxury paper shopping bag with rope handles, in a pastel pink color, is now placed on the sidewalk next to the young girl, preserving the appearance of the young girl with long dark hair and the fluffy white dog.*
5. *The patterns on the dress worn by the young girl with long dark hair are changed; the top is now solid light yellow, and the bottom tier has a blue and white polka dot pattern, preserving the girl's identity and the style of the dress.*
6. *Change the background from a modern storefront to the exterior of a charming Parisian café with a bistro table and chairs visible through the window, keeping the exact same camera angle, position, and framing, and preserving the girl and the dog.*
7. *Enhance the hair of the young girl with long dark hair to be more voluminous and styled with soft, flowing waves, while preserving her facial features and clothing.*
8. *Replace the white circular pouch at the girl's waist with a small, rectangular brown leather crossbody bag, keeping the rest of her outfit and appearance the same.*
9. *The young girl with long dark hair is now gently patting the top of the fluffy white dog's head with her hand, keeping her smiling expression and overall pose.*
10. *Transform the entire image's aesthetic into a bright, colorful watercolor painting style, preserving the recognizable features of the girl, the dog, and the café background.*

Figure 15. **Additional qualitative comparison (1/6).** We compare our method against several state-of-the-art methods, including FLUX.1 Kontext [27], Qwen-Image [55], Seedream 4.0 [46], Nano Banana [18], MTC [62], VINCIE [41], and Bagel [9]. Please zoom in for a better view.

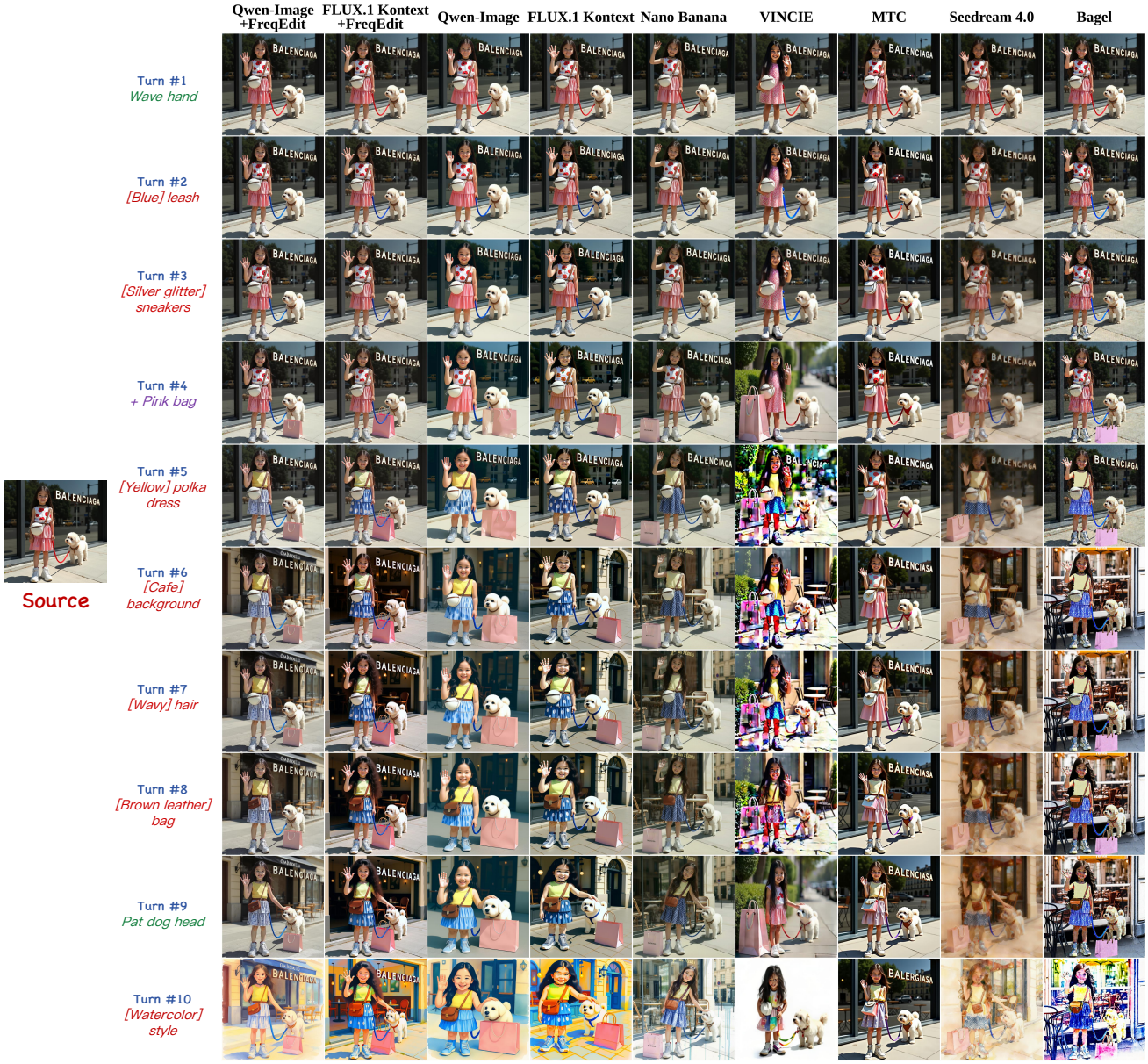

Figure 16. **Additional qualitative comparison (2/6).** We compare our method against several state-of-the-art methods, including FLUX.1 Kontext [27], Qwen-Image [55], Seedream 4.0 [46], Nano Banana [18], MTC [62], VINCIE [41], and Bagel [9]. Please zoom in for a better view.

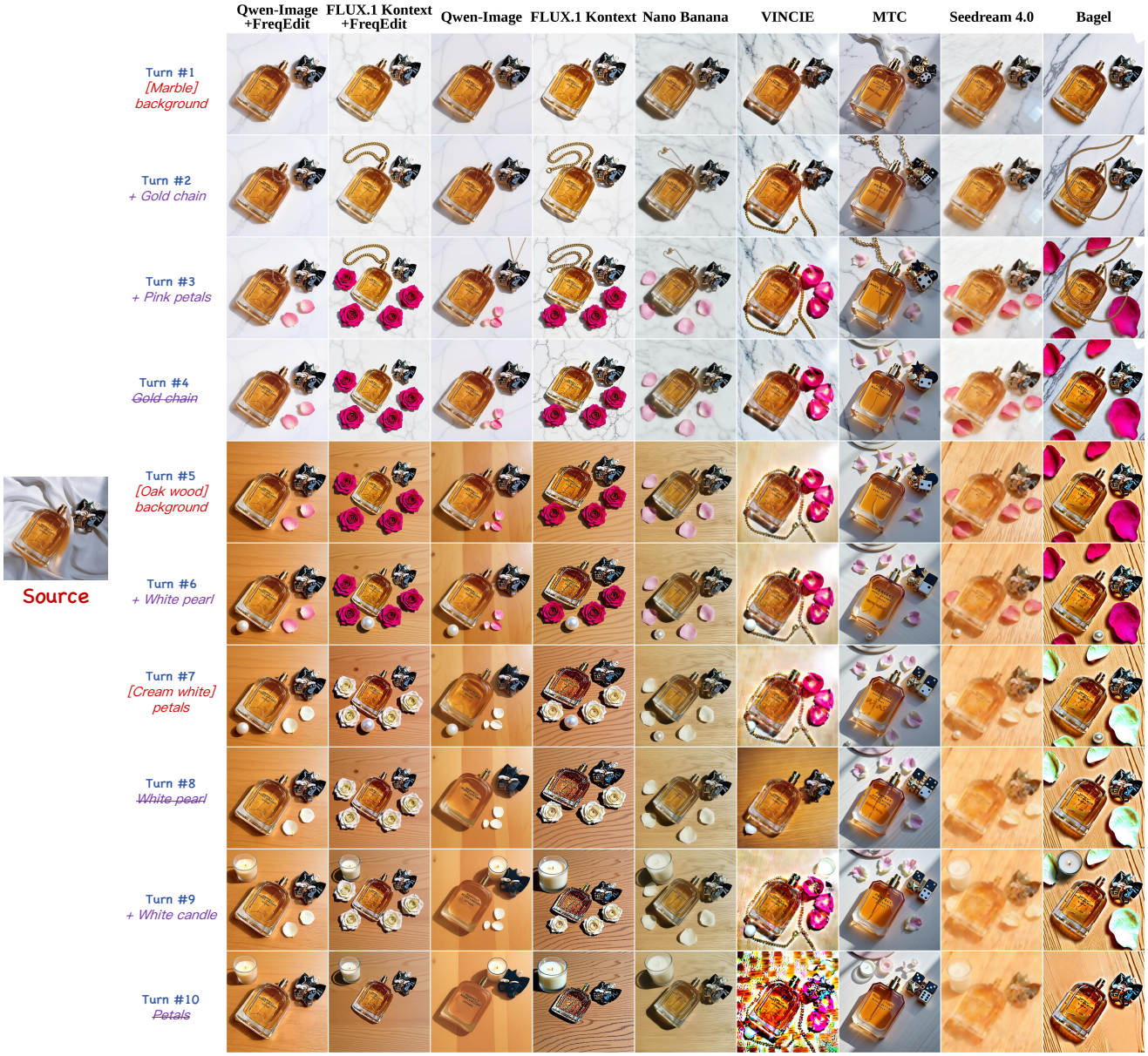

Figure 17. **Additional qualitative comparison (3/6).** We compare our method against several state-of-the-art methods, including FLUX.1 Kontext [27], Qwen-Image [55], Seedream 4.0 [46], Nano Banana [18], MTC [62], VINCIE [41], and Bagel [9]. Please zoom in for a better view.

Figure 18. **Additional qualitative comparison (4/6).** We compare our method against several state-of-the-art methods, including FLUX.1 Kontext [27], Qwen-Image [55], Seedream 4.0 [46], Nano Banana [18], MTC [62], VINCIE [41], and Bagel [9]. Please zoom in for a better view.

Figure 19. **Additional qualitative comparison (5/6).** We compare our method against several state-of-the-art methods, including FLUX.1 Kontext [27], Qwen-Image [55], Seedream 4.0 [46], Nano Banana [18], MTC [62], VINCIE [41], and Bagel [9]. Please zoom in for a better view.

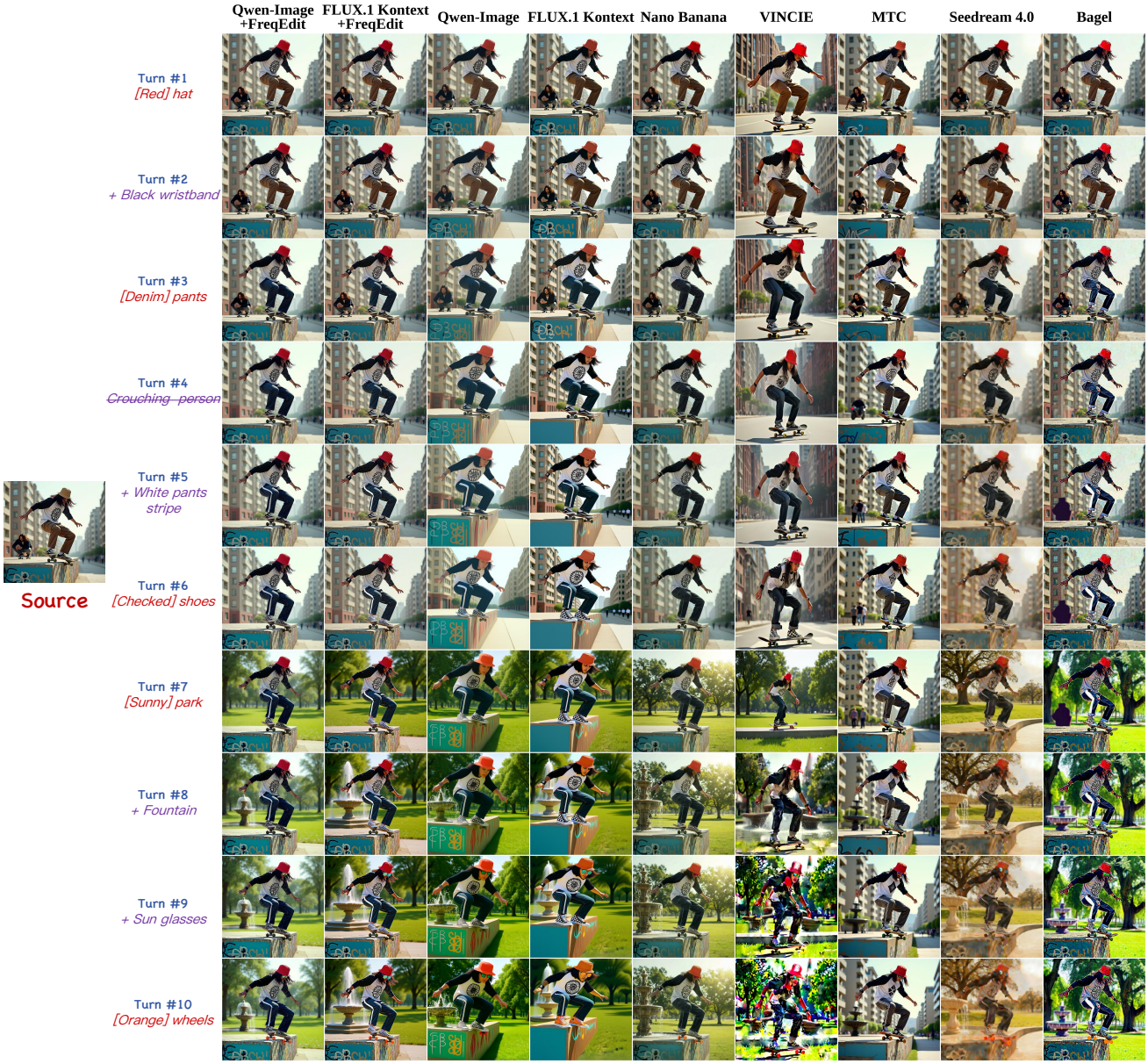

Figure 20. **Additional qualitative comparison (6/6).** We compare our method against several state-of-the-art methods, including FLUX.1 Kontext [27], Qwen-Image [55], Seedream 4.0 [46], Nano Banana [18], MTC [62], VINCIE [41], and Bagel [9]. Please zoom in for a better view.

Structured Prompt Template for Multi-Turn Image Editing Instruction Generation

Imagine that you are a top-tier AI Prompt Engineer. You are an expert in image analysis, JSON data structures, and a specific, rigorous English syntax for generating image editing prompts. Your output must adhere 100% to the syntax and formatting rules defined in this directive.

Your task is to analyze a user-provided image (or its textual description) and output a single, syntactically perfect JSON object. This object will contain an initial description of the image, a descriptive tag, and an array of 10 distinct English editing prompts that follow a strict, sequential logic.

You must generate the editing prompts from the following guidelines:
1. Analyze the image first. Identify all key subjects, elements, the environment, and the composition. Assign specific, descriptive names to key subjects for subsequent reference (*e.g.*, "the woman with short black hair", "the snowy mountain on the left").
2. Generate a comprehensive initial prompt (`init_prompt`). For every significant object, subject, and background element, describe its key attributes including precise color, shape, material, texture, and position. Do not mention the real name of entities.
3. Generate a tag in the format `[Formatted_Filename]_[Subject]_[Scene]`. Format numbers to three digits with leading zeros (*e.g.*, input '1.png' results in '001').
4. Generate exactly 10 editing prompts. Each must be unique and adhere to the syntax rules below. Avoid random, disconnected edits.
5. The 10 prompts must be logical and thematically cohesive, simulating a real editor's workflow.

An editing prompt should follow these rules:
1. Execute only one primary operation from: `Subject Addition`, `Subject Removal`, `Subject Replacement`, `Background Change`, `Portrait Beautification`, `Color Alteration`, `Material Modification`, `Motion Change`, `Style Transfer`.
2. Single Subject Preservation Rule: If the image contains only one primary subject (*e.g.*, product photography), you are strictly forbidden from making any direct changes to that subject. Edits can modify the background, add/remove surrounding objects, and adjust lighting only.
3. Mandatory Preservation Clause: Every prompt must end with an explicit preservation clause using: *while maintaining...*, *preserving...*, or *keeping...* (*e.g.*, "while maintaining her same facial features, hairstyle, and overall appearance").
4. No Pronouns: Strictly forbidden from using ambiguous pronouns like he, she, it, they, this, or that. Always use specific descriptive names.
5. Extreme Specificity: Use precise descriptions (*e.g.*, "a bright scarlet red T-shirt" instead of "a red shirt").
6. Compositional Control: When changing backgrounds, explicitly lock camera viewpoint: *"...while keeping the exact same camera angle, position, and framing"*.
7. Style Constraint: You are prohibited from generating prompts that create a sci-fi or dark/gloomy aesthetic. The "Style Transfer" operations must only be placed between steps 7 and 10.

Output Format: You should output a JSON file to include the following information:
`tag`: formatted filename with subject and scene
`init_prompt`: comprehensive photorealistic description of the initial image
`editing_prompts`: array of exactly 10 editing instructions following all syntax rules
Your output should be a JSON file in one row (without any format), which looks like:

```
{"tag": "001_Woman_By_Wall", "init_prompt": "A full-body photo of a woman
with short black hair standing in front of a brick wall, wearing a blue
t-shirt and jeans.", "editing_prompts": ["The woman's blue t-shirt is
now a bright scarlet red, while maintaining her same facial features,
hairstyle, and overall appearance.", ..., "The entire image is transformed
into a Claymation style, while keeping the same person and the overall
composition."]}
```

Figure 21. **VLM prompt template for generating structured image editing instructions.**

Structured Prompt Template for Image Description Transformation

Imagine that you are a meticulous prompt engineer. You should take an initial Base Description and a sequence of Editing Instructions, then generate a complete list of transformed image descriptions.

Your task is to transform the base description sequentially by applying each editing instruction one at a time, where each output becomes the input for the next step.

You must follow this transformation workflow:
1. Start with `description_0` = `init_prompt` (the initial base description).
2. For each editing instruction $p^{[i]}$ (where i ranges from 1 to 10):
- Take the previous description `description_i-1` as the base.
- Apply only the single editing instruction $p^{[i]}$ while preserving all unmodified details.
- Generate `description_i` as the new complete, standalone description.

3. Output a dictionary containing the tag and list of all 11 descriptions.

Key principles:
- Each description must be complete and standalone (not incremental changes).
- Perfectly preserve all details from the previous step that are not modified by the current edit.
- Apply edits cumulatively: `description_i` includes all changes from $p^{[1]}$ through $p^{[i]}$.

Input Format:
Tag: `<tag>`
Initial Description: `<init_prompt>`
Editing Instructions (in order):
1. `<editing_instruction_1>`
2. `<editing_instruction_2>`
...
10. `<editing_instruction_10>`

Example:
Input:
Tag: "001_Woman_By_Wall"
Initial: "A full-body photo of a woman with short black hair standing in front of a brick wall, wearing a blue t-shirt and jeans."
Edits: ["t-shirt now red, maintaining...", "hair now wavy, preserving...", ...]
Output:

```
{"tag": "001_Woman_By_Wall",
  "descriptions": [
    "A full-body photo of a woman with short black hair standing in front of
a brick wall, wearing a blue t-shirt and jeans.",
    "A full-body photo of a woman with short black hair standing in front of
a brick wall, wearing a red t-shirt and jeans.",
    "A full-body photo of a woman with wavy black hair standing in front of
a brick wall, wearing a red t-shirt and jeans.",
    ...
  ]}
```

Output Format: A JSON object with tag and descriptions array:

```
{"tag": "<tag>", "descriptions": ["<description_0>", "<description_1>",
..., "<description_10>"]}
```

Figure 22. **VLM prompt template for sequential description transformation.** The system applies all editing instructions in one pass, generating a dictionary containing the tag and a list of 11 descriptions (the initial state followed by 10 transformed states). Source–target pairs are then constructed as consecutive descriptions, i.e., (`descriptions[i]`, `descriptions[i+1]`) for $i = 0$ to 9, for use in inversion-based image editing models (e.g., MTC [62]).