# RESTORESTEER: ADAPTIVE FLOW CONTROL FOR FAITHFUL IMAGE RESTORATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Flow-based image editors provide strong generative priors for image restoration, but perceptually plausible outputs can deviate from the scene supported by a degraded observation, limiting restoration fidelity. We regard a restoration as faithful when it preserves scene information supported by the observation without reproducing the degradation itself. The available evidence and safely editable quantities change along the sampling trajectory: early states permit changes to scene formation but expose only coarse evidence; intermediate states provide more reliable boundaries while geometry remains editable; and late states support precise residual measurements but tolerate only small corrections. We introduce RestoreSteer, a training-free controller that matches the correction target and magnitude to these sampling phases. RouteCal redirects early scene formation using observation-supported structure and coarse appearance; StructAlign corrects persistent geometric disagreement after reliable boundaries emerge; and ObsRefine optimizes a bounded late latent residual through the frozen decoder and forward degradation model. Each stage evaluates bounded candidates before modifying the sampling state. RestoreSteer requires neither backbone retraining nor a fixed pseudo-clean target obtained by inverting the observation. Across six restoration tasks on Qwen-Image-Edit and FLUX.2, RestoreSteer improves PSNR, SSIM, and LPIPS over native sampling in every evaluated task-backbone setting. Component ablations further show complementary contributions from the three sampling phases.

## 1 INTRODUCTION

Flow matching models learn continuous transports from simple noise to complex image distributions (Lipman et al., 2023; Liu et al., 2023). Recent image-editing variants make this prior available for restoration through a degraded image and a natural-language instruction (Meng et al., 2022; Brooks et al., 2023). They can reconstruct realistic texture and semantic content without training a separate network for every corruption. Yet a generative prior answers what an image could look like, not necessarily what the particular observation supports. During sampling, a model may choose the wrong crop or object layout, preserve corruption from its condition, or hallucinate details in weakly constrained regions.

These fidelity errors are trajectory dependent. Before coarse content settles, a small structural error can redirect the entire recovery route. In the middle of sampling, object boundaries become stable enough to compare but remain editable. Near the endpoint, large spatial changes risk disrupting established content, motivating a focus on residual observation inconsistency. A fixed projection applied at every time ignores this changing controllability. Conversely, a single endpoint correction offers limited scope for revising a scene layout established earlier in sampling.

Restoration fidelity denotes the preservation of scene information supported by the degraded observation, rather than pixelwise reproduction of the degradation itself. Forward-observation consistency provides an operational criterion: when a restored image is passed through the degradation model, it should agree with the input within measurement noise and model uncertainty. Existing diffusion and flow inverse solvers pursue this consistency using a known degradation operator, its adjoint, or a pseudoinverse (Chung et al., 2023; Wang et al., 2023; Martin et al., 2025; Wickremasinghe et al., 2026). These methods demonstrate the potential of pretrained generative priors for solving inverse problems. However, their extension to general image editors requires handling imperfect observa-



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.

tion models and controlling the effects of operator-based corrections. Real degradations are often only approximately characterized, and directly injecting the output of an inverse routine can transmit its artifacts into the generative trajectory. RestoreSteer therefore treats the observation model as corrective evidence while leaving the frozen flow prior responsible for reconstructing content that the observation does not determine.

We address this phase-dependent controllability by treating restoration as adaptive observation control over a frozen flow trajectory. At selected sampling steps, RestoreSteer decodes the model's current endpoint estimate and compares it with the degraded input through stage-specific evidence. The observation determines what evidence remains unexplained, while the current sampling phase determines which variables are used for correction. This separation allows the frozen flow prior to reconstruct content that is not identified by the observation, while using measured evidence to constrain generation.

The progression of controllable quantities motivates the three components of RestoreSteer. Before global layout stabilizes, *Route Calibration* (RouteCal) uses observation-supported structure and coarse appearance to redirect scene formation. Once boundaries have emerged but geometry remains editable, *Structural Alignment* (StructAlign) corrects persistent spatial disagreement. After content and geometry have largely formed, *Observation Refinement* (ObsRefine) differentiates a forward observation objective through the frozen decoder and optimizes a bounded latent correction. Their task-dependent benefits are evaluated empirically rather than assumed to be universally additive.

The components operate in different control spaces but share a conservative execution rule. RestoreSteer proposes a bounded stage-appropriate action and evaluates it before committing a sampling-state update. RouteCal scores decoded previews using a cached velocity, whereas StructAlign and ObsRefine re-evaluate candidate states at the same sampling time. Rejection preserves the arriving state and previously accepted interventions.

Our contributions are as follows:

- We propose RestoreSteer, a training-free controller that improves observation-supported restoration fidelity by adapting the correction target and magnitude along a frozen flow trajectory.
- We develop a phase-adaptive correction algorithm that combines early scene calibration, intermediate structural alignment, and bounded late latent optimization, with each intervention evaluated before it modifies the live sampling state.
- We separate shared trajectory control from task-specific observation models and backbone interfaces, enabling training-free transfer across restoration tasks and flow editors. Experiments on Qwen-Image-Edit and FLUX.2 across six tasks show consistent improvements in reference fidelity and perceptual similarity over native sampling.

## 2 Related Work

**Flow matching and rectified-flow editing.** Flow matching and rectified flow learn vector fields to transport samples between distributions (Lipman et al., 2023; Liu et al., 2023). Conditional editors like Qwen-Image-Edit (Wu et al., 2025) and FLUX.2 (Black Forest Labs, 2025) fuse these priors with image and language conditioning. Pretrained rectified-flow models have also been adapted for real-image inversion and semantic editing (Rout et al., 2025; Kulikov et al., 2025), but focus on semantic transformation rather than fidelity to a prescribed degraded observation.

**Generative inverse solvers.** Diffusion-based inverse solvers incorporate measurements through likelihood or manifold-constrained gradients (Chung et al., 2023; 2022), linear degradation structure (Kawar et al., 2022; Wang et al., 2023), or pseudo-inverse guidance (Song et al., 2023). For flow models, D-Flow optimizes source noise through generation (Ben-Hamu et al., 2024), PnP-Flow alternates data-fidelity updates and flow-path reprojection (Martin et al., 2025), and FlowSteer applies operator-aware corrections along sampling (Wickremasinghe et al., 2026).

**Generative and instruction-guided image restoration.** Diffusion-based restoration methods adapt pretrained priors through restoration-specific training, conditioning, or learned enhancement modules (Wang et al., 2024; Lin et al., 2024; Wu et al., 2024; Yang et al., 2024; Yu et al., 2024).

Prompt- and instruction-guided methods learn task-aware representations for unified restoration (Potlapalli et al., 2023; Conde et al., 2024; Luo et al., 2024). ResFlow instead learns a continuous reversal of the degradation process from paired data (Qin et al., 2025).

# 3 BACKGROUND

## 3.1 FROZEN FLOW EDITOR AND ITS CURRENT CLEAN PREDICTION

A conditional flow editor generates an image by evolving a latent state. Let $y$ denote the degraded input, $q$ the restoration task, and $c_q(y)$ the image-and-instruction condition given to the editor. At sampling step $k$, the frozen model evaluates its velocity field at the live latent $z_k$:

$$v_k = v_\theta(z_k, t(\sigma_k); c_q(y)) \,, \tag{1}$$

where $\sigma_k$ is the scheduler noise level, $t(\sigma_k)$ converts it to the model's time convention, and the pretrained parameters $\theta$ remain fixed. The scheduler uses $v_k$ to advance the actual sampling trajectory.

The same state also provides a prediction of the clean image toward which the current trajectory is heading:

$$\hat{z}_{0,k} = F_k(z_k) = P_k(z_k, v_k), \qquad \hat{x}_{0,k} = \mathcal{G}(\hat{z}_{0,k}), \tag{2}$$

where $P_k$ is the backbone-specific conversion from the current state and velocity to a clean latent, and $\mathcal{G}$ is the frozen VAE decoder (Kingma & Welling, 2014; Rombach et al., 2022). The map $F_k$ combines a model query at the fixed noise level $\sigma_k$ with this endpoint conversion: $F_k(z) = P_k(z, v_\theta(z, t(\sigma_k); c_q(y)))$, without advancing the scheduler. Under the rectified-flow convention used here, $P_k(z, v) = z - \sigma_k v$ (Liu et al., 2023; Esser et al., 2024). We call $\hat{x}_{0,k}$ the *current clean-image prediction*. It is the endpoint presently implied by $z_k$, not a separate completed sampling run.

The distinction between $z_k$ and $\hat{z}_{0,k}$ is important. The former is the live state from which generation continues; the latter is a prediction that can be decoded and inspected before sampling finishes. A useful change to the predicted endpoint affects the final restoration only after it has been mapped back to the live state. During controlled sampling, the reference at step $k$ is always computed from the state arriving at that step, including all interventions accepted earlier.

## 3.2 OBSERVATION CONSISTENCY AS CORRECTIVE EVIDENCE

Restoration assumes that an unknown clean image $x^\star$ produced the input through a task-dependent degradation process,

$$y = \mathcal{D}_q(x^\star; \eta_q) + n, \tag{3}$$

where $\mathcal{D}_q$ is the forward degradation model, $\eta_q$ contains its parameters, and $n$ represents measurement noise and unmodeled effects. For a candidate restoration $x$, we can ask whether degrading it again reproduces the measured input:

$$E_q(x; y) = \ell_q(\mathcal{D}_q(x; \eta_q), y) \,, \tag{4}$$

where $\ell_q$ accounts for measurement noise and model uncertainty. We call $E_q$ the forward-observation error. For example, super-resolution compares the downsampled restoration with the observed LR image.

Different clean images can explain the same observation, while exact matching can reproduce noise or compensate for model error (Wang et al., 2023; Kawar et al., 2022). RestoreSteer therefore uses observation evidence to constrain the prediction and the frozen prior to recover undetermined content. The sampling phase determines which evidence guides correction and how strongly the live state is modified.

# 4 METHOD

## 4.1 OVERVIEW: MATCH THE CORRECTION TO THE SAMPLING PHASE

RestoreSteer matches its correction target and magnitude to the sampling phase: RouteCal redirects early scene formation, StructAlign corrects intermediate geometric disagreement, and ObsRefine

**Algorithm 1 RestoreSteer adaptive intervention controllers.** Each active controller constructs a latent direction, evaluates bounded trial states, and commits only a candidate satisfying its phase-specific criteria. Detailed objectives, candidate sets, and thresholds are defined in the following sections and appendix.

**RouteCal: Early**

**1. Reference**
Use the cached native velocity:

$$h_k^- = z_k - \sigma_k v_k^-,$$
$$x_k^{\mathrm{ref}} = \mathcal{G}(h_k^-).$$

**2. Direction**

$$x_k^p = \mathcal{P}_q(\mathcal{W}(x_k^{\mathrm{ref}}; \phi_k^\star), y),$$
$$\Delta_k = \mathcal{E}(x_k^p) - \mathcal{E}(x_k^{\mathrm{ref}}).$$

Select $d_k$ from bounded scalings of $\Delta_k$ and the residual-refinement iterates.

**3. Trial**
For each tested live-state coefficient $c$, form

$$z_k' = z_k + c d_k,$$
$$x_k^{\mathrm{prev}} = \mathcal{G}\left(h_k^- + \frac{c}{1-\sigma_k} d_k\right).$$

**4. Check and commit**
Retain only trials satisfying $J_{q,k}(x_k^{\mathrm{prev}}; y) \le J_{q,k}(x_k^{\mathrm{ref}}; y) - \epsilon_R$. If a trial is selected and passes the checks, set $z_k \leftarrow z_k'$; otherwise retain $z_k$.

**StructAlign: Middle**

**1. Reference**
Use a fresh same-time prediction:

$$\hat{z}_{0,k} = F_k(z_k),$$
$$\hat{x}_{0,k} = \mathcal{G}(\hat{z}_{0,k}).$$

**2. Direction**

$$x_k^p = \mathcal{P}_q(\mathcal{W}(\hat{x}_{0,k}; \phi_k^\star), y),$$
$$\Delta_k = \mathcal{E}(x_k^p) - \mathcal{E}(\hat{x}_{0,k}).$$

Select $d_k$ from bounded scalings of $\Delta_k$.

**3. Trial**
For each tested endpoint strength $\alpha$, form

$$z_k' = z_k + (1-\sigma_k)\alpha d_k,$$
$$\widetilde{x}_{0,k} = \mathcal{G}(F_k(z_k')).$$

**4. Check and commit**
Select a trial using its realized joint score and the applicable structural and response checks. If a trial is selected and passes the checks, set $z_k \leftarrow z_k'$; otherwise retain $z_k$.

**ObsRefine: Late**

**1. Reference**
Use a fresh same-time prediction:

$$\hat{z}_{0,k} = F_k(z_k),$$
$$\hat{x}_{0,k} = \mathcal{G}(\hat{z}_{0,k}).$$

**2. Direction**
Optimize the bounded residual objective in Eq. 8. The selected eligible Adam iterate becomes $d_k$.

**3. Trial**
For each tested endpoint strength $\alpha$, form

$$z_k' = z_k + (1-\sigma_k)\alpha d_k,$$
$$\widetilde{x}_{0,k} = \mathcal{G}(F_k(z_k')).$$

**4. Check and commit**
Require reduced observation error, bounded structural change, and retained detail. If a trial is selected and passes the checks, set $z_k \leftarrow z_k'$; otherwise retain $z_k$.

reduces late forward-observation error. Algorithm 1 summarizes their intervention-step updates. At designated intervention steps after native scheduler updates, the active controller constructs a bounded correction, evaluates it, and commits an accepted update; otherwise, sampling continues from the arriving state. RouteCal uses cached-velocity previews, whereas StructAlign and ObsRefine verify realized predictions through fresh same-time model queries. Native sampling then resumes until the final latent is decoded. Detailed schedules are given in Appendix A.7.

### 4.2 What the controller measures

The observation contains both useful scene evidence and task-specific corruption. RestoreSteer therefore does not use raw RGB distance as a single objective at every phase. Early and middle control require a structural reading of the input, whereas late control requires the remaining error under the forward degradation model.

For early and middle control, $L_{\mathrm{stru}}(x; y)$ compares locally normalized intensity and band responses together with edge magnitude and orientation. The comparison is weighted by observation-derived edge reliability and aggregated over spatial regions, with additional emphasis on the highest-error regions so that a small misaligned object is not hidden by a large aligned background. The appearance score $A_{q,k}(x; y)$ uses the task-relevant quantity that remains reliable in the observation: coarse image content for denoising and deblurring, atmospheric or exposure agreement for dehazing and low-light restoration, the measured low-resolution image for super-resolution, and luminance for colorization. After an intervention has been accepted, $A_{q,k}$ can additionally protect appearance agreement established by earlier stages. Candidate actions are ranked by

$$J_{q,k}(x; y) = \frac{L_{\mathrm{stru}}(x; y)}{s_{\mathrm{str}}} + \frac{A_{q,k}(x; y)}{s_{\mathrm{app}}}, \tag{5}$$

where the positive scales $s_{\mathrm{str}}$ and $s_{\mathrm{app}}$ are fixed from initial reference scores so that neither term dominates only because of its numerical units.

For late control, $E_q(x; y)$ from Eq. 4 measures whether the candidate reproduces the observation after applying the task's forward model. The implementation evaluates this residual at several spatial scales and ignores magnitudes below an input-estimated uncertainty floor.

Accordingly, RouteCal and StructAlign rank candidates with $J_{q,k}$, whereas ObsRefine uses $E_q$ together with structure and detail-preservation checks. Appendix A.1 provides the exact descriptors and task-specific terms.

### 4.3 ROUTECAL: REDIRECTING EARLY SCENE FORMATION

Early in sampling, errors in object position and coarse appearance can propagate through generation, while fine details remain unstable. RouteCal adjusts the layout and coarse appearance of the current prediction before sampling proceeds, first aligning predictions to the input, then correcting appearance differences using task-specific information. At its early intervention step, RouteCal uses the cached clean-latent reference

$$h_k^- = z_k - \sigma_k v_k^-, \qquad x_k^{\mathrm{ref}} = \mathcal{G}(h_k^-),$$

where $v_k^-$ is the velocity cached from the preceding native scheduler update.

For spatial alignment, we use a coordinate map $\phi$ that combines bounded translation, rotation, and scale changes with a smooth local deformation (Jaderberg et al., 2015; Ashburner, 2007; Dalca et al., 2019). Let $\mathcal{W}(x;\phi)$ denote the warped version of an input image $x$. We optimize

$$\min_{\phi} \; L_{\mathrm{stru}}\big(\mathcal{W}(x_k^{\mathrm{ref}};\phi); y\big) + \lambda_{\mathrm{warp}} R(\phi), \tag{6}$$

where $R$ penalizes large or rapidly varying deformations, with weight $\lambda_{\mathrm{warp}}$. Among geometrically valid candidates, we select the smallest tested warp that retains a prescribed fraction of the best structural improvement. This limits unnecessary displacement when the input contains corruption. Denoting the selected map by $\phi_k^\star$, we obtain the aligned prediction $x_k^w = \mathcal{W}(x_k^{\mathrm{ref}};\phi_k^\star)$.

Spatial alignment corrects where image content appears, but does not ensure that its brightness, color, or coarse structure agrees with the observed scene. The appearance adjustment addresses these remaining differences before they influence later generation. Because the input also contains corruption, directly matching its RGB values would encourage the prediction to reproduce the degradation.

We next adjust the aligned prediction's coarse appearance using task-specific information from the observation. This includes coarse intensity and color patterns for denoising and deblurring, an exposure-adjusted anchor for low-light restoration, an atmospheric-model restoration estimate for dehazing, and measured low-resolution or grayscale content for super-resolution and colorization. These quantities guide the proposal's coarse appearance while the aligned prediction supplies the image content. We denote the adjustment by $\mathcal{P}_q$ and the resulting image proposal by $x_k^p = \mathcal{P}_q(x_k^w, y)$.

The proposal describes how we want the current prediction to change; it is not returned as the restored image. To obtain a correction in latent space, we encode both the proposal and the original prediction and take their difference:

$$\Delta_k = \mathcal{E}(x_k^p) - \mathcal{E}(x_k^{\mathrm{ref}}), \tag{7}$$

where $\mathcal{E}$ is the frozen VAE encoder. This difference includes both the spatial alignment and appearance adjustment.

The encoding difference $\Delta_k$ provides the initial latent direction. RouteCal evaluates bounded scalings of this direction with $J_{q,k}$ and denotes the best initial candidate by $d_0$. Starting from $d_0$, it optimizes an additional bounded residual through the frozen decoder, as specified in Eq. 42. The lowest-scoring candidate among the bounded scalings and residual-refinement iterates becomes $d_k$. Algorithm 1 then evaluates the corresponding early live-state writes through the cached-velocity preview. Appendix A.4 gives the tested fractions, refinement settings, and update bounds.

### 4.4 STRUCTALIGN: CORRECTING OBSERVABLE AND EDITABLE GEOMETRY

StructAlign begins only after object boundaries have become reliable enough to compare. At this phase, geometry is no longer merely an uncertain early cue, but it can still be changed without reconstructing the entire scene. The controller repeats RouteCal's alignment–appearance–encoding pipeline on the current prediction and uses the resulting $d_k$ as a candidate structural action; it does not repeat RouteCal's early residual-refinement stage. Each candidate is instead written to a trial sampling state and the frozen editor is queried again at the same noise level. Selection is therefore based on the geometry and appearance actually predicted after the intervention.

A second alignment is attempted only when reliable edges in the resulting prediction still disagree with the fixed observation. If activated, the second alignment is accepted only when the resulting

image prediction reduces the RGB distance to its image proposal and satisfies the signed edge-agreement criterion.

Changing the live state can also make the editor respond differently from the requested clean-endpoint change. After an accepted structural update, StructAlign stores the difference between the re-evaluated clean latent and the requested endpoint. At the next correction opportunity, it uses a bounded negative of this stored difference to propose a correction to the current endpoint. The controller tests this correction by querying the editor again at the same noise level. The update is accepted only if the actual endpoint change better matches the requested correction and the component orthogonal to the requested direction does not exceed a prescribed fraction of the total change. Otherwise, the sampling state is left unchanged. This step compensates for the editor's response to the earlier update; it does not construct a new image target. Appendix A.5 provides the correction bound and response checks.

### 4.5 ObsRefine: reducing late forward-observation error

Once content and geometry have largely formed, ObsRefine reduces the remaining observation error by optimizing a bounded residual $\delta$ around the current clean latent $\hat{z}_{0,k}$. Prior methods guide diffusion with measurement gradients (Chung et al., 2023) or optimize source noise by differentiating through a flow trajectory (Ben-Hamu et al., 2024). ObsRefine instead updates only the residual $\delta$ at the current intervention step through the frozen decoder and forward degradation model, while keeping $\hat{z}_{0,k}$ and all model parameters fixed. We approximately minimize the following constrained objective:

$$\delta_k^\star = \arg \min_{\|\delta\|_{\mathrm{rms}} \le r_{\max}} \Big\{ E_q(\mathcal{G}(\hat{z}_{0,k} + \delta); y) + \lambda_z \langle \delta^2 \rangle + \lambda_x \langle |\mathcal{G}(\hat{z}_{0,k} + \delta) - \hat{x}_{0,k}| \rangle \Big\}, \tag{8}$$

where $\langle \cdot \rangle$ denotes the mean over all tensor entries. The observation term measures how well the corrected image reproduces $y$ after applying the forward degradation model. The second term penalizes the mean squared latent correction, while the third penalizes the mean absolute image change relative to the current prediction $\hat{x}_{0,k}$. The weights $\lambda_z$ and $\lambda_x$ balance these penalties against observation fidelity. Unlike these soft penalties, the RMS constraint places an explicit upper bound on the correction magnitude.

We use four to eight Adam iterations (Kingma & Ba, 2015) to update $\delta$, keeping $\hat{z}_{0,k}$ and all model parameters fixed. Intermediate iterates are evaluated for observation-error reduction, and the selected residual becomes the correction direction $d_k$. We screen scaled residuals for observation-error reduction, structural consistency, and detail retention. Eligible proposals are then tested through same-time model queries, and the sampling state is updated only when the realized prediction passes the same checks. Appendix A.6 provides the exact criteria.

### 4.6 Executing a correction in the live sampler

Each phase constructs a clean-latent correction direction $d_k$. For StructAlign and ObsRefine, a tested endpoint strength $\alpha$ gives $a_k = \alpha d_k$, which is transported to the live state under rectified-flow interpolation as

$$z'_k = z_k + (1 - \sigma_k) a_k. \tag{9}$$

Here $z_k$ is the arriving state and $z'_k$ the trial state. RouteCal instead writes directly in live-state coordinates,

$$z'_k = z_k + c d_k, \tag{10}$$

equivalent for $\sigma_k < 1$ to $a_k = [c/(1 - \sigma_k)] d_k$. Thus, $\alpha$ scales the endpoint correction, whereas $c$ scales the live-state displacement.

RouteCal evaluates trials with the cached-velocity preview described in Appendix A.4. StructAlign and ObsRefine instead re-query the frozen editor at the same sampling time:

$$\widetilde{z}_{0,k} = F_k(z'_k), \qquad \widetilde{x}_{0,k} = \mathcal{G}(\widetilde{z}_{0,k}). \tag{11}$$

Because changing $z_k$ also changes the model velocity, the realized endpoint $\widetilde{z}_{0,k}$ need not equal the requested $\hat{z}_{0,k} + a_k$. These stages therefore accept or reject candidates using their realized predictions rather than the nominal endpoint proposal alone.

For all three controllers, an accepted trial replaces $z_k$ by $z'_k$; otherwise the arriving state is retained. The checks apply only at the current intervention step and do not certify the final output after subsequent flow steps.

**Backbone adaptation.** Transferring RestoreSteer to another flow editor changes only its sampling interface: the mapping between scheduler noise and model time, the endpoint conversion $P_k$, latent packing, frozen VAE calls, and phase-to-step mapping. Changing restoration tasks instead changes the observation comparisons, task proposal, and forward degradation model. The observable–editable phase division and propose–evaluate–commit interface remain unchanged; neither type of transfer requires training the backbone or a restoration adapter.

# 5 EXPERIMENTS

We evaluate whether phase-adaptive observation control improves restoration fidelity over native sampling, whether these gains transfer across frozen flow editors and restoration tasks, and how the three control phases contribute. The evaluation uses two frozen editors and six restoration tasks; no backbone or restoration adapter is trained.

## 5.1 EXPERIMENTAL SETUP

**Datasets.** We construct a fixed benchmark of 300 input–reference pairs, with 50 pairs per task. Denoising uses CBSD68 images with additive Gaussian noise ($\sigma = 25$, 8-bit scale). Dehazing uses 25 indoor and 25 outdoor pairs from RESIDE SOTS (Li et al., 2019). Low-light restoration uses the REED test split (Wang et al., 2025), pairing each $-3$EV image with its normal-exposure counterpart. Super-resolution uses $\times 4$ camera-captured pairs from RealSR V3 (Cai et al., 2019), and deblurring uses blurred–sharp frame pairs from GoPro (Nah et al., 2017). For colorization, we form grayscale inputs from 50 FlowSteer100 images (Wickremasinghe et al., 2026). Dataset sources, subset construction, and preprocessing are documented in Appendix C.1.

**Backbones and competing methods.** We compare native sampling, PnP-Flow (Martin et al., 2025), FlowSteer (Wickremasinghe et al., 2026), and RestoreSteer on Qwen-Image-Edit-2511 and FLUX.2-klein-base-9B, with frozen editors and VAEs. Both baselines are adapted to conditional latent space while preserving their published update procedures. We extend their original task configurations to dehazing and low light using atmospheric scattering and illumination models, respectively. Within each backbone, methods share inputs, instructions, output sizes, and evaluation metrics. Adaptation details appear in Appendix C.2.

**Evaluation and implementation.** We report mean per-image RGB PSNR, SSIM (Wang et al., 2004), and AlexNet LPIPS (Zhang et al., 2018). PSNR and SSIM measure pixel-level and structural fidelity to the paired clean reference, while LPIPS measures perceptual feature distance. Together, these metrics test whether stronger reconstruction fidelity is accompanied by improved rather than degraded perceptual similarity. We average each metric over the 50 images in the corresponding task. Native sampling and the trajectory controllers use 40 steps on Qwen and 50 on FLUX.2. PnP-Flow uses its separate 100-iteration schedule with five noise samples per iteration. RestoreSteer places RouteCal early, StructAlign in the middle, and ObsRefine at eight late intervention steps, with the indices adapted to each backbone's schedule. Implementation settings, recorded inference times, and evaluation protocols are detailed in Appendices C.3, C.4, and C.5, respectively.

## 5.2 QUANTITATIVE COMPARISON

Table 1 shows RestoreSteer consistently improves PSNR, SSIM, and LPIPS over native sampling across all six tasks on both backbones without retraining. Simultaneous PSNR/SSIM gains indicate higher reference fidelity, while lower LPIPS confirms improved perceptual similarity, not just reduced pixel error. On Qwen, RestoreSteer outperforms FlowSteer on all metrics for dehazing, super-resolution, deblurring, and colorization, with notable PSNR gains for super-resolution and deblurring. Versus PnP-Flow, it improves all metrics on denoising, dehazing, low-light restoration, and deblurring, and achieves lower LPIPS across all six tasks, highlighting superior perceptual fidelity.

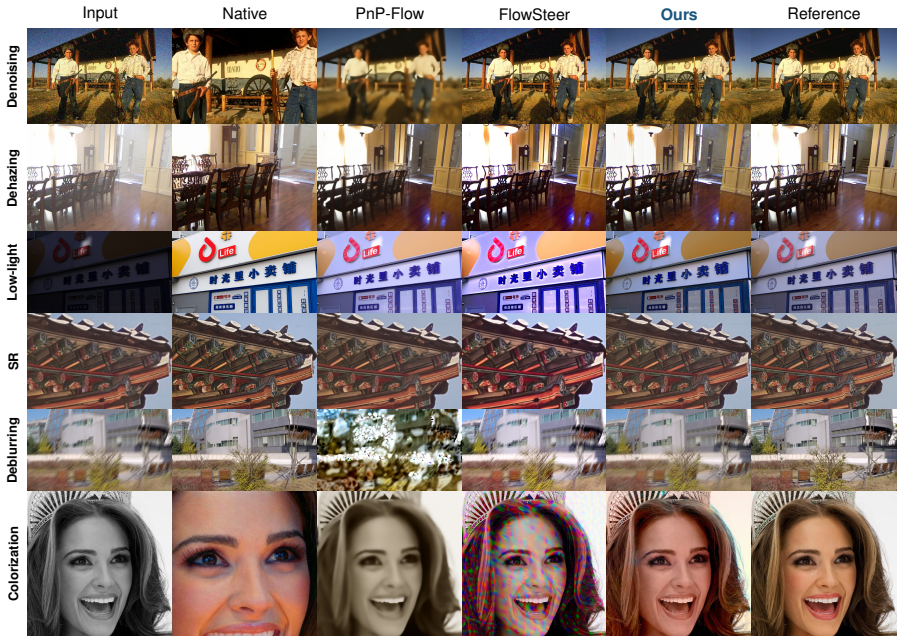



Figure 1: **Qualitative comparison across six restoration tasks.** Rows show denoising, dehazing, low-light restoration, super-resolution, deblurring, and colorization, from top to bottom. Columns show the shared input, native editor, PnP-Flow, FlowSteer, Ours (RestoreSteer), and paired reference. All examples use Qwen-Image-Edit. Saved final outputs are assembled without additional restoration or contrast adjustment, preserving their aspect ratios.

Table 1: Quantitative comparison across six restoration tasks, with 50 images per task. Red bold and blue indicate the best and second-best values within each backbone. Here "+" denotes applying the listed method on top of the corresponding native editor. PnP-Flow and FlowSteer follow their published update procedures and are adapted to the corresponding frozen editor; implementation details are given in Appendix C.2.

| Method | Denoising | | | Dehazing | | | Low light | | | Super-resolution | | | Deblurring | | | Colorization | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ |
| **Qwen-Image-Edit** | | | | | | | | | | | | | | | | | | |
| Native | 15.15 | 0.335 | 0.408 | 13.06 | 0.360 | 0.475 | 13.54 | 0.446 | 0.314 | 17.19 | 0.452 | 0.356 | 18.39 | 0.482 | 0.284 | 10.58 | 0.261 | 0.760 |
| + PnP-Flow | 23.78 | 0.572 | 0.638 | 17.68 | 0.725 | 0.331 | 15.08 | 0.634 | 0.279 | **25.65** | **0.692** | 0.390 | 13.54 | 0.361 | 0.761 | **22.50** | **0.698** | 0.620 |
| + FlowSteer | **25.47** | **0.711** | **0.262** | 17.01 | 0.737 | 0.197 | 13.89 | **0.680** | 0.282 | 19.38 | 0.542 | 0.650 | 19.16 | 0.621 | 0.382 | 17.25 | 0.486 | 0.658 |
| + Ours | 24.59 | 0.690 | 0.309 | **19.11** | **0.762** | **0.183** | **17.13** | 0.678 | **0.275** | 24.72 | 0.673 | **0.289** | **24.34** | **0.722** | **0.229** | 19.39 | 0.662 | **0.366** |
| **FLUX.2** | | | | | | | | | | | | | | | | | | |
| Native | 17.99 | 0.668 | 0.222 | 13.60 | 0.578 | 0.350 | 15.11 | 0.663 | 0.262 | 15.01 | 0.368 | 0.448 | 18.00 | 0.495 | 0.369 | 18.16 | 0.694 | 0.266 |
| + PnP-Flow | 20.43 | 0.469 | 0.788 | 18.05 | 0.695 | 0.416 | 17.82 | 0.664 | 0.319 | **20.99** | 0.504 | 0.489 | 18.34 | 0.541 | 0.629 | 13.28 | 0.451 | 0.792 |
| + FlowSteer | 20.83 | 0.503 | 0.420 | 16.55 | 0.700 | 0.192 | 14.62 | 0.653 | 0.304 | 12.14 | 0.105 | 1.143 | 15.67 | 0.251 | 0.884 | 11.73 | 0.259 | 0.690 |
| + Ours | **25.97** | **0.783** | **0.177** | **20.01** | **0.807** | **0.134** | **17.99** | **0.748** | **0.209** | 20.77 | **0.526** | **0.359** | **21.04** | **0.579** | **0.312** | **23.95** | **0.886** | **0.160** |

On FLUX.2, RestoreSteer surpasses FlowSteer on all 18 task-metric combinations and PnP-Flow on 17 of 18. While PnP-Flow achieves marginally higher super-resolution PSNR, RestoreSteer yields better SSIM and substantially lower LPIPS for that task.

### 5.3 Qualitative comparison

Figure 1 illustrates spatial, appearance, and detail recovery across six tasks. RestoreSteer retains more clothing and building detail than PnP-Flow in denoising, and better preserves the reference dining-room layout than native Qwen in dehazing. For low light, it restores storefront visibility with brightness and colors closer to the reference than the brighter native output.

In super-resolution, RestoreSteer better preserves roof geometry than FlowSteer. In deblurring, it recovers clearer building boundaries and vegetation than PnP-Flow while maintaining scene com-




Figure 2: **Corrections at RouteCal and StructAlign intervention steps.** Each row shows one restoration example. From left to right: input, reference, predictions before and after RouteCal, predictions before and after StructAlign, and final outputs of the native editor and RestoreSteer. Red boxes highlight local structural changes.

Table 2: Component ablation on Qwen, using the same 50 images and manifest-index seeds. Each variant removes only the named component and keeps the remaining settings unchanged. Red bold and blue indicate the best and second-best values for each metric.

| Method | Denoising | | | Dehazing | | | Low light | | | Super-resolution | | | Deblurring | | | Colorization | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ | PSNR↑ | SSIM↑ | LPIPS↓ |
| w/o ObsRefine | 22.32 | 0.610 | 0.333 | 18.74 | 0.741 | 0.190 | 14.45 | 0.581 | 0.302 | 23.95 | 0.657 | 0.296 | 23.80 | 0.708 | 0.226 | 18.58 | 0.632 | **0.365** |
| w/o StructAlign | 19.42 | 0.467 | 0.332 | 15.00 | 0.477 | 0.361 | 15.32 | 0.580 | 0.267 | 21.60 | 0.589 | **0.274** | 22.67 | 0.662 | 0.225 | 13.64 | 0.357 | 0.734 |
| w/o RouteCal | 24.40 | 0.682 | **0.284** | 18.83 | 0.740 | 0.211 | **18.06** | **0.700** | **0.233** | 24.05 | 0.646 | 0.282 | 23.49 | 0.688 | **0.218** | 18.92 | 0.637 | 0.386 |
| Full RestoreSteer | **24.59** | **0.690** | 0.309 | **19.11** | **0.762** | **0.183** | 17.13 | 0.678 | 0.275 | **24.72** | **0.673** | 0.289 | **24.34** | **0.722** | 0.229 | **19.39** | **0.662** | 0.366 |

position. For colorization, it preserves the portrait framing altered by native Qwen and produces plausible colors without FlowSteer's strong color artifacts.

**Phase-specific corrections.** Figure 2 illustrates the intermediate effects of RouteCal and StructAlign by showing clean endpoint predictions before and after each intervention, rather than only the final restoration. RouteCal produces broader changes to scene formation and coarse appearance at the early intervention step, whereas StructAlign makes more localized geometric corrections after object boundaries have become reliable, as indicated by the highlighted regions. The final columns show the corresponding restoration outputs.

### 5.4 Component Ablation

We individually remove RouteCal, StructAlign, or ObsRefine from the Qwen configuration, holding images, seeds, and remaining settings fixed; Table 2 compares the full controller with the three two-component variants. The ablation validates each component's contribution: removing StructAlign degrades PSNR and SSIM across all tasks; removing ObsRefine also reduces both metrics universally, confirming the value of late refinement; removing RouteCal lowers performance on five tasks, verifying early calibration's role. The full model achieves the highest PSNR and SSIM on five of six tasks.

## 6 Conclusion

We present RestoreSteer, a training-free, phase-adaptive controller for faithful image restoration with frozen flow editors. RestoreSteer improves observation-supported restoration fidelity by matching early scene calibration, intermediate structural alignment, and late residual refinement to the evidence and editability available along the sampling trajectory. Each component evaluates bounded corrections before modifying the live state, allowing the frozen generative prior to recover uncertain content while constraining scene properties supported by the observation. Experiments across six tasks and two backbones show consistent improvements in PSNR, SSIM, and LPIPS without retraining. Component ablations confirm the central roles of intermediate alignment and late refinement, together with the task-dependent benefits of early calibration.

## REFERENCES


John Ashburner. A fast diffeomorphic image registration algorithm. *NeuroImage*, 38(1):95–113, 2007.

Jonathan T. Barron. A general and adaptive robust loss function. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4331–4339, 2019.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-Flow: Differentiating through flows for controlled generation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 3462–3483, 2024.

Black Forest Labs. FLUX.2: Frontier visual intelligence. Black Forest Labs technical release, 2025.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18392–18402, 2023.

Jianrui Cai, Hui Zeng, Hongwei Yong, Zisheng Cao, and Lei Zhang. Toward real-world single image super-resolution: A new benchmark and a new model. In *IEEE/CVF International Conference on Computer Vision*, pp. 3086–3095, 2019.

Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of the IEEE International Conference on Image Processing*, volume 2, pp. 168–172, 1994.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Hyungjin Chung, Jeongsol Kim, Michael T. McCann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations*, 2023.

Marcos V. Conde, Gregor Geigle, and Radu Timofte. InstructIR: High-quality image restoration following human instructions. In *European Conference on Computer Vision*, pp. 1–21, 2024.

Adrian V. Dalca, Guha Balakrishnan, John Guttag, and Mert R. Sabuncu. Unsupervised learning of probabilistic diffeomorphic registration for images and surfaces. *Medical Image Analysis*, 57: 226–236, 2019.

David L. Donoho and Iain M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633, 2024.

Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011.

Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1397–1409, 2013.

Michal Irani and Shmuel Peleg. Improving resolution by image registration. *CVGIP: Graphical Models and Image Processing*, 53(3):231–239, 1991.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. FlowEdit: Inversion-free text-based editing using pre-trained flow models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19721–19730, 2025.

Edwin H. Land and John J. McCann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61(1):1–11, 1971.

Anat Levin, Dani Lischinski, and Yair Weiss. Colorization using optimization. *ACM Transactions on Graphics*, 23(3):689–694, 2004.

Boyi Li, Wenqi Ren, Dengpan Fu, Dacheng Tao, Dan Feng, Wenjun Zeng, and Zhangyang Wang. Benchmarking single-image dehazing and beyond. *IEEE Transactions on Image Processing*, 28 (1):492–505, 2019.

Xinqi Lin, Jingwen He, Ziyan Chen, Zhaoyang Lyu, Bo Dai, Fanghua Yu, Yu Qiao, Wanli Ouyang, and Chao Dong. DiffBIR: Toward blind image restoration with generative diffusion prior. In *European Conference on Computer Vision*, pp. 430–448, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Ziwei Luo, Fredrik K. Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B. Schön. Controlling vision-language models for multi-task image restoration. In *International Conference on Learning Representations*, 2024.

David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pp. 416–423, 2001.

Ségolène Martin, Anne Gagneux, Paul Hagemann, and Gabriele Steidl. PnP-Flow: Plug-and-play image restoration with flow matching. In *International Conference on Learning Representations*, 2025.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 257–265, 2017.

Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Shahbaz Khan. PromptIR: Prompting for all-in-one blind image restoration. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Haina Qin, Wenyang Luo, Libin Wang, Dandan Zheng, Jingdong Chen, Ming Yang, Bing Li, and Weiming Hu. Reversing flow for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7545–7558, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Semantic image inversion and editing using rectified stochastic differential equations. In *The Thirteenth International Conference on Learning Representations*, 2025.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023.

Bo Wang, Huiyuan Fu, Zhiye Huang, Siru Zhang, Xin Wang, and Huadong Ma. From abyssal darkness to blinding glare: A benchmark on extreme exposure correction in real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7666–7675, 2025.

Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin C. K. Chan, and Chen Change Loy. Exploiting diffusion prior for real-world image super-resolution. *International Journal of Computer Vision*, 132(12):5929–5949, 2024.

Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *International Conference on Learning Representations*, 2023.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Deep retinex decomposition for low-light enhancement. In *British Machine Vision Conference*, 2018.

Tharindu Wickremasinghe, Chenyang Qi, Harshana Weligampola, Zhengzhong Tu, and Stanley H. Chan. FlowSteer: Conditioning flow field for consistent image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Findings*, pp. 5106–5116, 2026.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Rongyuan Wu, Tao Yang, Lingchen Sun, Zhengqiang Zhang, Shuai Li, and Lei Zhang. SeeSR: Towards semantics-aware real-world image super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25456–25467, 2024.

Tao Yang, Rongyuan Wu, Peiran Ren, Xuansong Xie, and Lei Zhang. Pixel-aware stable diffusion for realistic image super-resolution and personalized stylization. In *European Conference on Computer Vision*, pp. 74–91, 2024.

Fanghua Yu, Jinjin Gu, Zheyuan Li, Jinfan Hu, Xiangtao Kong, Xintao Wang, Jingwen He, Yu Qiao, and Chao Dong. Scaling up to excellence: Practicing model scaling for photo-realistic image restoration in the wild. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25669–25680, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

## AI ASSISTANCE DISCLOSURE

We used large language models to assist with language polishing and limited bibliographic search. They were not used to design the methodology, generate experimental data, or determine the reported results and conclusions. All AI-assisted material was carefully reviewed, verified, and revised by the authors.

## A CONTROLLER DETAILS

These definitions expand the Method section and describe the selected controller configuration. All losses are inference-time criteria; ground truth is used only for evaluation.

### A.1 OBSERVATION AND STRUCTURE OBSERVERS

**Late observation objective.** Let $\widehat{\eta}_q(y)$ denote fixed input-derived degradation parameters and $\Pi_s$ area average pooling by factors $s \in \mathcal{S} = \{1, 2, 4\}$. With $[b]_+ = \max(b, 0)$ and $\rho(a) = \sqrt{a^2 + 10^{-6}} - 10^{-3}$ (Charbonnier et al., 1994; Barron, 2019), define

$$\begin{aligned} r_s(x; y) &= \Pi_s(\mathcal{D}_q(x; \widehat{\eta}_q(y))) - \Pi_s(y), \\ E_q(x; y) &= \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \left\langle \rho\left( \frac{[|r_s(x; y)| - u(y)]_+}{u(y)} \right) \right\rangle. \end{aligned} \tag{12}$$

Here $\langle \cdot \rangle$ averages all tensor entries. To estimate the uncertainty floor, use arithmetic grayscale $\bar{Y}(y)$ and the reflection-padded $3 \times 3$ box average $\mathcal{B}_3$:

$$h_y = \bar{Y}(y) - \mathcal{B}_3 \bar{Y}(y), \qquad u(y) = \mathrm{clip}_{[0.01, 0.20]} \frac{\mathrm{median}_p \, |h_y(p) - \mathrm{median}_{p'} \, h_y(p')|}{0.67448975}. \tag{13}$$

This scalar is shared across scales and fixed during optimization. Images are converted to $[0, 1]$ and clipped before applying the forward map. The six maps and their parameter estimates are specified in Appendix A.3. These models are proxies for the benchmark degradations; the dead zone tolerates small residuals but does not remove model mismatch. No inverse image target enters this objective.

**Regional structure.** The descriptor $C$ contains normalized intensity and band responses; $M$ and $g$ are normalized edge magnitude and stabilized gradient direction, and $w_y$ is an input-reliability weight. Their construction is given in Appendix A.2. At a pixel $p$, define

$$\begin{aligned} d_p = w_y(p) \Bigg[ \frac{1}{n_C} \sum_{j=1}^{n_C} \rho_c(C_j(x)_p - C_j(y)_p) + \lambda_m \rho_c(M(x)_p - M(y)_p) \\ + \lambda_g (1 - g(x)_p^\top g(y)_p) \{M(x)_p + M(y)_p\}/2 \Bigg], \end{aligned} \tag{14}$$

where $n_C = 2$, $\rho_c(a) = \sqrt{a^2 + 10^{-6}}$, $\lambda_m = 0.35$, and $\lambda_g = 0.1$. With $\bar{d}_i$ denoting the mean in region $i$, the score is

$$L_{\mathrm{stru}}(x; y) = \frac{1}{N} \sum_{i=1}^{N} \bar{d}_i + \frac{1}{|\mathcal{H}|} \sum_{i \in \mathcal{H}} \bar{d}_i. \tag{15}$$

The grid has $9 \times 16$ regions and $\mathcal{H}$ contains its highest-error 20% (rounded to 29 regions). No additional border term is used in the selected configuration.

**Early/middle appearance score.** RouteCal and StructAlign combine $L_{\mathrm{stru}}$ with an appearance score $A_{q,k}$, as in Eq. 5. Let $A_h$ be the initial anchoring criterion and $A_t$ the later task-specific appearance criterion. Their complete task definitions are given in Appendix A.3. For noise, blur, haze, and low light, $A_h$ measures agreement with the coarse anchor used to construct the initial proposal; SR and colorization compare the measured LR or grayscale image directly. The later criteria measure noise-tolerant residuals, coarse-band agreement, LR or grayscale agreement, and scattering or re-exposure residuals. These are distinct from the late objective $E_q$.

**History-dependent appearance protection.** Let $x_{\text{init}}$ be the decoded reference at the first intervention attempt. Fix

$$s_h = \max(A_h(x_{\text{init}}; y), 10^{-3}), \qquad s_t = \max(A_t(x_{\text{init}}; y), 10^{-3}). \tag{16}$$

Initially, $A_{q,k}(x; y) = A_h(x; y)/s_h$. After an initial intervention has been accepted, let $r_h = A_h(x_{\text{ref}}; y)/s_h$ at the current intervention step and maintain $r_{\text{best}}$, initialized to one, as the smallest normalized anchoring score recorded at previous references or accepted endpoints, including the current reference. Define

$$w_h = \text{clip}_{[0,1]} \frac{r_h - r_{\text{best}}}{\max(1 - r_{\text{best}}, 10^{-6})}, \qquad A_{q,k}(x; y) = (1 - w_h)\frac{A_t(x; y)}{s_t} + w_h \frac{A_h(x; y)}{s_h}. \tag{17}$$

The weight is fixed during the current candidate comparison. The outer scales in $J_{q,k}$ are independently fixed from the initial reference as $s_{\text{str}} = \max(L_{\text{str}}(x_{\text{init}}; y), 10^{-3})$ and $s_{\text{app}} = \max(A_{q,k_{\text{init}}}(x_{\text{init}}; y), 10^{-3})$. Thus, $s_h, s_t$ normalize the appearance criteria internally, whereas $s_{\text{app}}$ balances their combined score against structure. The same history weight also protects the base image proposal as specified in Eq. 37.

### A.2 Construction of the structural descriptors

This subsection specifies $C$, $M$, $g$, and $w_y$ in Eq. 14; the regional aggregation in Eq. 15 is unchanged. Let $\mathcal{B}_s$ denote an $s \times s$ box average with reflection padding, and let $Y(x) = 0.299x_R + 0.587x_G + 0.114x_B$. Structural analysis first box-smooths each image on its own pixel grid, converts it to $[0, 1]$ with clipping, and extracts luminance. The smoothing size is 7 for denoising, deblurring, and dehazing, and 5 for low light, super-resolution, and colorization. For low light, luminance is replaced by $\log(Y + 0.02)$. The resulting scalar image is bilinearly resized to a common analysis canvas whose long side is at most 512 pixels, preserving the candidate's aspect ratio with a minimum of 16 pixels per dimension. For SR, the input luminance is resized from the LR grid to this same canvas. Denote the resulting scalar field by $\ell_x$.

**Locally normalized intensity and band responses.** Define

$$\begin{aligned} f_x &= \mathcal{B}_5 \ell_x, & c_x &= \mathcal{B}_{13} \ell_x, \\ \mu_x &= \mathcal{B}_{17} c_x, & s_x &= \sqrt{\mathcal{B}_{17}[(c_x - \mu_x)^2] + 10^{-4}}, \\ n_x &= \tanh((c_x - \mu_x)/s_x), & b_x &= \tanh((f_x - c_x)/s_x), \\ C(x) &= (n_x, \tfrac{1}{2} b_x). \end{aligned} \tag{18}$$

Thus, $C$ has two channels. Both compare structure at fixed coordinates after suppressing absolute brightness and local contrast differences.

**Edges and input reliability.** Use forward differences of $n_x$, with the final difference replicated at the image boundary, to obtain $\nabla n_x$. Set

$$\begin{aligned} r_x &= \sqrt{\|\nabla n_x\|_2^2 + 10^{-6}}, & g(x) &= \nabla n_x / r_x, \\ M(x) &= \tanh\left(\frac{r_x}{\mathcal{B}_9 r_x + 0.02}\right), \\ u_y &= \text{clip}_{[0.05,2]}\left(\frac{M(y)}{\text{median}_p\, M(y)_p + 10^{-4}}\right), \\ w_y &= 0.5 + 0.5\, \text{clip}_{[0,1]}\left(\frac{u_y}{\max(\langle u_y \rangle, 10^{-6})}\right). \end{aligned} \tag{19}$$

The reliability weight depends only on the input. In the structural discrepancy, $\rho_c(a) = \sqrt{a^2 + 10^{-6}}$. The edge-support factor $(M(x) + M(y))/2$ is held fixed when differentiating the directional term. The effective direction weight in Eq. 14 is $\lambda_g = 0.1$. Regional means are formed by adaptive average pooling to $9 \times 16$; the hard-region mean uses the largest 29 of these 144 values.

### A.3 TASK-SPECIFIC OBSERVATION AND APPEARANCE OPERATORS

The definitions below supply the task-dependent quantities used in the main algorithm: $\mathcal{P}_q$ constructs an image proposal after alignment, $A_h$ and $A_t$ supply the initial and later appearance criteria, and $\mathcal{D}_q$ supplies the late forward-observation error $E_q$. Their outputs enter the existing latent correction and acceptance rules; they do not replace the restored image returned by the sampler.

**Image conventions and common operators.** In this subsection images are represented in $[0,1]$; $[x]_{01}$ denotes componentwise clipping to this range. The encoder and decoder interfaces convert to and from their native range. Let $\bar{Y}(x) = (x_R + x_G + x_B)/3$ denote arithmetic grayscale, distinct from the luminance $Y$ used for structure and exposure estimation. For appearance scores, define $\rho(a) = \sqrt{a^2 + 10^{-6}} - 10^{-3}$ and average over all entries. Let $D_f$ be area reduction to $(\max(2, \lfloor H/f \rfloor), \max(2, \lfloor W/f \rfloor))$, $U_f$ bilinear interpolation back to the original grid, and $S_f = U_f D_f$. Bilinear and bicubic interpolation use `align_corners=False`. For same-resolution tasks, the $A_q$ and $B_q$ in Eq. 41 are $D_{f_q}$ and $U_{f_q}$.

**Initial appearance anchoring.** For noise, blur, haze, and low light, the initial proposal $\mathcal{P}_q^h(x, y)$ is exactly the clipped construction in Eq. 41, with eight back-projection updates (Irani & Peleg, 1991). Its task choices are

| $q$ | Noise | Blur | Haze | Low light |
|---|---|---|---|---|
| $f_q$ | 4 | 8 | 2 | 4 |
| $b_q(y)$ | $y$ | $y$ | $b_{\text{haze}}(y)$ | $b_{\text{low}}(y)$. |

(20)

The two nontrivial anchors are defined below. Their initial appearance criterion compares coarse measurements, not the full proposal image:

$$A_h(x; y) = \left\langle \rho\big(D_{f_q} x - D_{f_q} b_q(y)\big) \right\rangle. \tag{21}$$

SR and colorization use their directly measured quantities instead, as specified in their task paragraphs.

**Denoising.** The late map is $\mathcal{D}_{\text{noise}}(x) = x$; noise tolerance is supplied by $u(y)$ in Eq. 12. The middle appearance score instead uses an input-estimated statistical noise radius. For each channel, estimate (Donoho & Johnstone, 1994)

$$\widehat{\sigma}_c = \max\left(0.005, \frac{\text{median}_p\, |(K * y_c)_p|}{0.6745}\right), \qquad K = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}, \tag{22}$$

using valid convolution. Set

$$v(x; y) = \sqrt{\max\left(\frac{1}{3}\sum_c \mathcal{B}_{17}[(x_c - y_c)^2/\widehat{\sigma}_c^2], 10^{-8}\right)}, \qquad A_t(x; y) = \langle \rho([v(x; y) - 1.25]_+) \rangle. \tag{23}$$

The middle proposal uses local regression to attenuate input-correlated high-frequency components away from strong coarse edges. Define

$$\begin{aligned} h_x &= x - S_8 x, & h_y &= y - S_8 y, \\ \beta(x, y) &= \text{clip}_{[0,1]} \frac{\mathcal{B}_9[\sum_c h_{x,c} h_{y,c}]}{\max(\mathcal{B}_9[\sum_c h_{y,c}^2], 10^{-8})}, \\ e_y &= \sqrt{\|\nabla S_8 Y(y)\|_2^2 + 10^{-12}}, & \omega_y &= \exp\left(-\frac{2e_y}{\max(\langle e_y \rangle, 10^{-5})}\right), \\ \mathcal{P}_{\text{noise}}^m(x, y) &= \big[x - S_8(x - y) - \omega_y \beta(x, y) h_y\big]_{01}. \end{aligned} \tag{24}$$

Here the forward differences used for $e_y$ are zero-padded at the final row and column. The local coefficient is recomputed from the current aligned prediction; it is not a cached clean-image estimate. This proposal does not replace the score in Eq. 23.

**Deblurring.** The late proxy is $\mathcal{D}_{\mathrm{blur}}(x) = K_b * x$, with a normalized $11 \times 11$ Gaussian kernel of standard deviation 2 and reflection padding. No blur kernel is estimated for the early/middle appearance criterion:

$$A_t(x; y) = \langle \rho(S_8 x - S_8 y) \rangle. \tag{25}$$

The later task proposal starts at $p^{(0)} = [S_8 y + (I - S_8)x]_{01}$ and applies four updates $p^{(j+1)} = [p^{(j)} + S_8 y - S_8 p^{(j)}]_{01}$. This specifies $\mathcal{P}^t_{\mathrm{blur}}$; the initial proposal uses the eight-update construction above.

**Super-resolution.** The late map $\mathcal{D}_{\mathrm{SR}}$ is area reduction to the observed LR size. For early/middle proposals and appearance scores, let $A_{\mathrm{SR}}$ be antialiased bicubic reduction to that size and $B_{\mathrm{SR}}$ bicubic interpolation to the output size. The measurement is the LR image itself:

$$\begin{aligned} A_h(x; y) = A_t(x; y) &= \langle \rho(A_{\mathrm{SR}} x - y) \rangle, \\ p^{(0)} &= [B_{\mathrm{SR}} y + (I - B_{\mathrm{SR}} A_{\mathrm{SR}})x]_{01}, \\ p^{(j+1)} &= [p^{(j)} + B_{\mathrm{SR}}(y - A_{\mathrm{SR}} p^{(j)})]_{01}. \end{aligned} \tag{26}$$

Both initial and later task proposals use eight updates. No full-resolution anchor $b_q(y)$ is needed. Bicubic interpolation is used without antialiasing for the upward map. These proposal operators and the late area-reduction proxy have different roles; neither is asserted to reproduce the full RealSR camera pipeline.

**Colorization.** (Levin et al., 2004) The late map is $\mathcal{D}_{\mathrm{color}}(x) = \bar{Y}(x)\mathbf{1}_3$, where $\mathbf{1}_3$ replicates the scalar image across RGB channels. Set

$$\begin{aligned} A_h(x; y) = A_t(x; y) &= \langle \rho(\bar{Y}(x) - \bar{Y}(y)) \rangle, \\ p^{(0)} &= x, \\ p^{(j+1)} &= [p^{(j)} + (\bar{Y}(y) - \bar{Y}(p^{(j)}))\mathbf{1}_3]_{01}. \end{aligned} \tag{27}$$

The initial and later task proposals use eight and four updates, respectively. Each unclipped update changes only the shared grayscale component; clipping handles out-of-range channels. This task does not use the spatial pyramid operators $A_q, B_q$ or an RGB restoration anchor.

**Dehazing: input-derived anchor and late model.** Let $\mathrm{dark}_{15}(u)_p$ be the minimum over RGB channels and the $15 \times 15$ neighborhood of $p$, restricted to valid image pixels (He et al., 2011). For the early anchor, take the pixels with the largest dark-channel values (the top $\max(1, \lceil 0.001HW \rceil)$), select the one with the largest $Y$, and clip its RGB values from below at 0.10 to obtain airlight $\mathbf{a}_h$. Compute

$$\begin{aligned} t_h^{\mathrm{raw}} &= [1 - 0.95 \,\mathrm{dark}_{15}(y/\mathbf{a}_h)]_{01}, \\ t_h &= \mathrm{clip}_{[0.15,1]} \,\mathrm{GF}_{20,10^{-3}}(Y(y), t_h^{\mathrm{raw}}), \\ b_{\mathrm{haze}}(y) &= [(y - \mathbf{a}_h)/t_h + \mathbf{a}_h]_{01}. \end{aligned} \tag{28}$$

Here the guided filter is explicitly (He et al., 2013)

$$\begin{aligned} a &= \frac{\mathcal{B}_{41}(Ip) - \mathcal{B}_{41} I \, \mathcal{B}_{41} p}{\mathcal{B}_{41}(I^2) - (\mathcal{B}_{41} I)^2 + 10^{-3}}, \\ b &= \mathcal{B}_{41} p - a \mathcal{B}_{41} I, \qquad \mathrm{GF}_{20,10^{-3}}(I, p) = (\mathcal{B}_{41} a) I + \mathcal{B}_{41} b. \end{aligned} \tag{29}$$

The late model uses a separate, input-only transmission estimate. Among the top $\max(1, \mathrm{round}(0.001HW))$ dark-channel pixels, select the largest $\bar{Y}$ and clip its RGB values from below at 0.05 to obtain $\mathbf{a}_\ell$. Set

$$\begin{aligned} t_\ell &= \mathrm{clip}_{[0.20,1]} \, \mathcal{B}_{15}[1 - 0.95 \,\mathrm{dark}_{15}([y/\mathbf{a}_\ell]_{01})] \,, \\ \widehat{\eta}_{\mathrm{haze}}(y) &= (t_\ell, \mathbf{a}_\ell), \qquad \mathcal{D}_{\mathrm{haze}}(x; \widehat{\eta}_{\mathrm{haze}}) = [t_\ell x + (1 - t_\ell)\mathbf{a}_\ell]_{01}. \end{aligned} \tag{30}$$

Both estimates are fixed after construction. The early estimate supplies an image anchor, whereas the late estimate only predicts the observation.

**Dehazing: cached middle appearance model.** Let $x_{\text{init}}$ be the decoded reference at the first intervention attempt, before its correction. Define $\mathcal{S}_{16}$ as area reduction to an aspect-preserving grid of long side 16 (at least two pixels per dimension), followed by bilinear interpolation back to the image grid. The middle airlight $\mathbf{a}_m$ is the mean RGB of the top $\max(1, \lceil 0.001HW \rceil)$ pixels ranked by input luminance $Y(y)$. Cache

$$t_m = \mathcal{S}_{16}\left[\text{clip}_{[0.05,1]} \frac{\sum_c (y_c - a_{m,c})(x_{\text{init},c} - a_{m,c})}{\max(\sum_c (x_{\text{init},c} - a_{m,c})^2, 10^{-4})}\right]. \tag{31}$$

With $F_m(x) = t_m x + (1 - t_m)\mathbf{a}_m$, use

$$A_t(x; y) = \langle \rho(F_m(x) - y) \rangle, \qquad p^{(j+1)} = \left[p^{(j)} + \frac{4t_m(y - F_m(p^{(j)}))}{1 + 4t_m^2}\right]_{01}. \tag{32}$$

Starting at $p^{(0)} = x$, four updates give the later task proposal $\mathcal{P}^t_{\text{haze}}$. The cached $(t_m, \mathbf{a}_m)$ are not re-estimated for trial candidates and are distinct from the late input-only parameters.

**Low light: input-derived anchor and late model.** For the early anchor, approximate linear-light RGB by $y^{2.2}$ and set

$$\begin{aligned} \gamma_y &= \text{clip}_{[1,8]} \frac{0.80}{\max(Q_{0.90}(Y(y^{2.2})), 10^{-4})}, \\ b_{\text{low}}(y) &= [\gamma_y y^{2.2}]_{01}^{1/2.2}. \end{aligned} \tag{33}$$

$Q_{0.90}$ is the spatial 90th percentile. For the late model, estimate illumination directly from the input:

$$\widehat{\eta}_{\text{low}}(y) = \ell_y = \text{clip}_{[0.12,1]}\, \mathcal{B}_{31}(\max_c y_c), \qquad \mathcal{D}_{\text{low}}(x; \ell_y) = [\ell_y x]_{01}. \tag{34}$$

This illumination field is fixed throughout late optimization.

**Low light: cached middle appearance model.** Using the same first reference $x_{\text{init}}$ and $\mathcal{S}_{16}$ as above, define

$$\begin{aligned} \ell_m &= \exp\Big\{\mathcal{S}_{16}\Big[\text{clip}_{[-6,1]}(\log(Y(y) + 0.01) - \log(Y(x_{\text{init}}) + 0.01))\Big]\Big\}, \\ c_m &= \text{clip}_{[0.5,2]}\, \text{median}_p \frac{y}{\max(\ell_m x_{\text{init}}, 0.01)}, \qquad g_m = \text{clip}_{[0.01,2]}(\ell_m c_m). \end{aligned} \tag{35}$$

The median is computed separately per RGB channel. Cache $g_m$ and use

$$A_t(x; y) = \langle \rho([g_m x]_{01} - y) \rangle, \qquad p^{(j+1)} = \left[p^{(j)} + \frac{4g_m(y - [g_m p^{(j)}]_{01})}{1 + 4g_m^2}\right]_{01}. \tag{36}$$

Four updates from $p^{(0)} = x$ define the re-exposure proposal $\mathcal{P}^t_{\text{low}}$. This cached model supplies the later appearance score; the actual middle proposal additionally uses the reflectance construction below.

**History protection and the selected proposal.** The preceding task paragraphs define the raw scores $A_h$ and $A_t$. The existing history-dependent rule combines their normalized values into $A_{q,k}$. In the selected configuration, the same weight $w_h$ also forms the base later proposal

$$\mathcal{P}^{\text{base}}_q(x, y) = (1 - w_h)\mathcal{P}^t_q(x, y) + w_h \mathcal{P}^h_q(x, y). \tag{37}$$

Before an initial intervention has been accepted, the base proposal and appearance criterion remain $\mathcal{P}^h_q$ and $A_h/s_h$. RouteCal uses this initial proposal. In StructAlign, blur, haze, SR, and colorization use $\mathcal{P}^{\text{base}}_q$; noise uses Eq. 24. These proposal choices leave the history-protected appearance score unchanged.

**Low-light middle proposal: reflectance and positive gain.** For an image $x$, decompose log-luminance as (Land & McCann, 1971; Wei et al., 2018)

$$\begin{aligned} l(x) &= \log(Y(x) + 0.02), \qquad i(x) = S_{16} l(x), \\ r(x) &= S_4(l(x) - i(x)), \qquad R(x) = r(x) - \langle r(x) \rangle, \\ I(x) &= i(x) + \langle r(x) \rangle, \qquad d(x) = l(x) - i(x) - r(x). \end{aligned} \tag{38}$$

Here $S_{16}$ reduces each spatial dimension by a factor of 16; it is different from the fixed-long-side smoother $\mathcal{S}_{16}$. Starting at $u^{(0)} = x$, let $\mu = \langle Y(x) \rangle$ and apply four reflectance updates:

$$\begin{aligned} L^{(j)} &= [\exp(I(u^{(j)}) + R(y) + d(u^{(j)})) - 0.02]_{01}, \\ \widetilde{L}^{(j)} &= [\mu L^{(j)} / \max(\langle L^{(j)} \rangle, 10^{-4})]_{01}, \\ u^{(j+1)} &= \left[ u^{(j)} \frac{\widetilde{L}^{(j)}}{\max(Y(u^{(j)}), 10^{-3})} \right]_{01}. \end{aligned} \tag{39}$$

Normalize once more to obtain $u = [\mu u^{(4)} / \max(\langle \max(Y(u^{(4)}), 10^{-4}) \rangle, 10^{-4})]_{01}$. This transfers the input's zero-mean coarse reflectance while retaining the prediction's illumination and fine residual. Let $p = \mathcal{P}_{\text{low}}^{\text{base}}(x, y)$. The positive gain is

$$\begin{aligned} a &= \text{clip}_{[0, \log 4]} \, S_8[\log(Y(p) + 0.01) - \log(Y(u) + 0.01)], \\ \mathcal{P}_{\text{low}}^{m}(x, y) &= \left[ u \frac{[Y(u) \exp(a)]_{01}}{\max(Y(u), 10^{-3})} \right]_{01}. \end{aligned} \tag{40}$$

The gain is recomputed for the current proposal and does not update the cached $g_m$ used to score candidates.

**Parameter sources.** The noise radii, early haze/exposure anchors, and all late nuisance parameters are estimated from $y$ alone. Only the middle haze transmission and low-light gain use the first reference endpoint $x_{\text{init}}$ in addition to $y$; they are then cached. The denoising regression and low-light positive-gain proposal depend on the current aligned image. Blur, SR, and colorization have fixed late forward maps, so $\widehat{\eta}_q$ is empty for those maps (as it is for the identity denoising map). None of these estimates uses a ground-truth reference.

### A.4 RouteCal proposals and latent refinement

**Shared structural optimization.** The sampling map composes a similarity transform with the exponential of a coarse stationary velocity field. Translation is bounded to 32 pixels in each direction, rotation to $3^\circ$, and log-scale to $0.04$. The local field has grid long side 12 and displacement scale 8 pixels, and is integrated by five scaling-and-squaring steps. The map specifies where each output pixel samples the input image; it is not a forward pixel-displacement operation. We use 16 Adam steps with learning rate $0.08$. The regularizer combines squared normalized global parameters, squared normalized field magnitude, and a squared first-difference smoothness statistic. Its weight is $0.02$. The initial and 16 updated parameter states are screened at amplitudes $\{0.125, 0.25, 0.5, 0.75, 1\}$. Among candidates passing discrete deformation checks and attaining at least 80% of the best structural gain, a smaller deformation is retained. Discrete Jacobian checks are safeguards, not a proof of globally invertible image resampling.

**Coarse-content anchoring.** For the pyramid proposal, let $A_q$ be area analysis downsampling and $B_q$ bilinear interpolation. The input-derived anchor is $b_q(y)$. Starting from aligned image $x_k^w$, we compute

$$\begin{aligned} x^{(0)} &= \text{clip}_{[0,1]} \left[ B_q A_q b_q(y) + (I - B_q A_q) x_k^w \right], \\ x^{(j+1)} &= \text{clip}_{[0,1]} \left[ x^{(j)} + B_q (A_q b_q(y) - A_q x^{(j)}) \right]. \end{aligned} \tag{41}$$

Clipping is applied at initialization and after each of the eight back-projection updates. Appendix A.3 specifies the task-dependent $A_q$, $B_q$, and $b_q(y)$, together with the separate SR and colorization proposals. At initialization, the anchor supplies the coarse measurements $A_q b_q(y)$, while the aligned prediction supplies $(I - B_q A_q) x_k^w$. $B_q A_q$ is not generally an idempotent projector, so this is a coarse/fine construction, not an exact range/null-space decomposition. Clipping can also change the fine-scale component.

**Latent candidate selection and refinement.** We test fractions $\{0.25, 0.5, 0.75, 1\}$ of the encoding difference in Eq. 7, bounding each candidate to RMS 0.5. Each candidate is added to the reference clean latent and decoded for evaluation with $J_{q,k}$. The candidate with the lowest joint score is denoted by $d_0$.

Starting from a zero residual, we perform three Adam iterations with learning rate 0.02 to optimize an additional correction:

$$\begin{aligned}\min_r\ & J_{q,k}(\mathcal{G}(h_k^- + d_0 + r); y) + 0.05\langle r^2\rangle \\ & + 0.05\left\langle |\mathcal{G}(h_k^- + d_0 + r) - \mathcal{G}(h_k^- + d_0)|\right\rangle .\end{aligned} \tag{42}$$

After each iteration, the residual is spatially smoothed and bounded to RMS 0.04; the total correction remains within RMS 0.5. We evaluate each resulting candidate with $J_{q,k}$ and retain the lowest-scoring correction across the initial candidates and all refinement iterates as $d_k$. Thus, refinement does not automatically replace a better initial candidate. Only the residual is optimized; the editor and VAE parameters remain frozen.

**Early write cap.** With $n_k$ the RMS native scheduler displacement and $N_k$ the previous intervention count, the maximum coefficient is

$$c_{\max} = \min\left\{4(1-\sigma_k)(0.75)^{N_k}, \frac{2n_k}{\max(\|d_k\|_{\mathrm{rms}}, \varepsilon)}\right\}. \tag{43}$$

The implementation screens fractions $\{1, 0.75, 0.5, 0.25\}c_{\max}$ and $\min(1-\sigma_k, c_{\max})$. Let $v_k^-$ be the cached velocity from the preceding scheduler update, and let $\sigma_k$ denote the noise level of the state actually being modified. The reference is $h_k^- = z_k - \sigma_k v_k^-$. For $\sigma_k < 1$, the implementation screens $\mathcal{G}(h_k^- + [c/(1-\sigma_k)]d_k)$ and writes $z_k + cd_k$. This decoded endpoint preview is not a fresh same-time model prediction, nor is it simply $\mathcal{G}(z_k + cd_k - \sigma_k v_k^-)$. A minimum absolute decrease of $\epsilon_R = 0.002$ in the normalized joint score is required. The cap bounds the live displacement but does not imply $c \le 1 - \sigma_k$.

### A.5 StructAlign safeguards

**Middle proposals and primary selection.** Alignment precedes the middle task proposal. Appendix A.3 defines the denoising regression proposal and low-light reflectance/positive-gain proposal, as well as the history-protected proposals for the other tasks. These proposal operations leave the appearance criterion $A_{q,k}$ unchanged. Primary endpoint strengths $\{0.5, 1\}$ are evaluated through fresh same-time velocity queries. Among actions retaining at least 80% of the best joint-score reduction, selection minimizes new high-frequency flat-region chroma. The statistic ranks full candidate actions and does not filter the retained image.

**Primary target following.** After an accepted primary action, an additional bounded direction toward the encoded full image proposal can be tested within the same callback. It must improve image-target distance, not worsen the joint score, and pass directional/orthogonal-response checks. The RMS bound is 0.08. This image-target correction is disabled for the second alignment; it must not be conflated with the subsequent internal latent-error correction.

**Second-alignment gate and selection.** Let $\Omega$ be the analysis grid, $\Omega_e$ the upper-quartile reliable joint-gradient support, and $\chi(p)$ the cosine of the task-filtered, locally standardized luminance gradients. The gate is

$$G_k = \frac{1}{|\Omega|}\sum_{p\in\Omega_e} \mathbf{1}[\chi(p) \le 0.90], \qquad G_k \ge 0.010. \tag{44}$$

The denominator is the entire analysis grid, not $|\Omega_e|$. For the second action, the actual RGB target-distance reduction must be at least 5%; signed edge agreement must improve by 0.01 or reach 0.90. Directional cosine must be at least 0.2, with orthogonal-response fraction at most 0.95. The smallest passing tested strength is selected.

**Subsequent latent-error cancellation.** Let $a_j$ include the accepted endpoint action and any accepted immediate target-following residual. The requested endpoint and stored discrepancy are

$$\hat{z}_{0,j}^{\text{target}} = \hat{z}_{0,j}^{\text{ref}} + a_j, \qquad e_j = \hat{z}_{0,j}^{\text{real}} - \hat{z}_{0,j}^{\text{target}}. \tag{45}$$

Using the fresh endpoint map $F_k$ from Eq. 2, define $T_k(z, a) = z + (1 - \sigma_k)a$ as a compact notation local to this cancellation test. The bounded request and measured response are

$$d_c = -e_j \min\left(1, \frac{0.08}{\|e_j\|_{\text{rms}} + \varepsilon}\right), \qquad r_c = F_k(T_k(z_k, d_c)) - F_k(z_k). \tag{46}$$

The response must satisfy

$$\begin{aligned} \|d_c\|_{\text{rms}} - \|r_c - d_c\|_{\text{rms}} &\geq 10^{-5}, \\ \cos(r_c, d_c) &\geq 0.2, \\ \frac{\|r_c - \text{proj}_{d_c} r_c\|_{\text{rms}}}{\max(\|r_c\|_{\text{rms}}, \varepsilon)} &\leq 0.95. \end{aligned} \tag{47}$$

No observation score enters this test. A valid stored structural transaction is required; the request is applied relative to the current endpoint.

### A.6 ObsRefine screening

Let $Y$ be luminance, $B_s$ a box filter, and tRMS the RMS after discarding the lowest and highest 10% of absolute responses. On the detail-analysis canvas, define

$$\begin{aligned} \nu(x) &= \frac{Y(x) - B_{15}Y(x)}{\sqrt{B_{15}[(Y(x) - B_{15}Y(x))^2] + 10^{-4}}}, \\ H(x) &= 0.65\,\text{tRMS}(B_3\nu(x) - B_7\nu(x)) + 0.35\,\text{tRMS}(B_7\nu(x) - B_{15}\nu(x)). \end{aligned} \tag{48}$$

For a decoded proposal or actual same-time candidate $x'$, require

$$\begin{aligned} E_q(x'; y) &\leq 0.99 E_q(\hat{x}_{0,k}; y), \\ L_{\text{stru}}(x'; y) &\leq L_{\text{stru}}(\hat{x}_{0,k}; y) + 0.002 + 0.01|L_{\text{stru}}(\hat{x}_{0,k}; y)|, \\ H(x') &\geq 0.95 H(\hat{x}_{0,k}). \end{aligned} \tag{49}$$

These are local, score-based conditions, not guarantees about ground truth or the final output. The image preservation regularizer in Eq. 8 uses native decoder values in $[-1, 1]$; the observation term separately converts to $[0, 1]$.

Adam iterates must improve observation error by at least 1% to remain eligible. If any reach the 50% reduction target, selection is restricted to those iterates. Among iterates within 1% of the best remaining error, the smallest RMS residual is retained. This iterate selection precedes the strength screening and same-time tests; it is not an additional requirement that every accepted live action achieve a 50% reduction.

### A.7 Execution order and accounting

All sampling-step indices are one-based. An intervention at step $k$ acts on the state after the $k$th scheduler update and uses the noise level associated with that state. Ordinary sampling resumes after the checks and any accepted update at that intervention step.

**Qwen-Image-Edit (40 steps).** The execution order is

1. At step 5, construct the early alignment/appearance proposal, screen latent corrections and cached-velocity previews, and apply a passing bounded early write.
2. Starting at step 15, attempt the primary structural transaction. If it fails, retry within the window ending at step 22.
3. Following a successful primary transaction, evaluate the observation-based second-alignment gate. If a second alignment is unnecessary, attempt the stored latent-error correction and close the transaction. Otherwise, test the second structural update and perform its subsequent correction when a valid accepted update is available. The transaction must finish before late control.
4. At steps $\{25, 27, 29, 31, 33, 35, 37, 39\}$, independently compute the current endpoint, optimize a late residual, screen strengths, and verify eligible realized responses.

**FLUX.2 (50 steps).** The controller uses the same stage-specific proposal and acceptance rules with the following execution schedule:

1. At step 10, construct and screen the early alignment/appearance proposal and apply a passing bounded RouteCal update.
2. At step 19, attempt the primary structural transaction. Unlike Qwen's adaptive retry window, the FLUX.2 configuration uses fixed middle-stage opportunities at steps 19, 20, and 21.
3. At step 20, evaluate the second-alignment gate when a valid primary transaction is available. If a second alignment is skipped or rejected, attempt the stored latent-error correction at this opportunity. If the second structural update is accepted, attempt its subsequent latent-error correction at step 21. A correction requires a valid stored accepted transaction; the middle-stage opportunities do not extend beyond step 21.
4. At steps $\{35, 37, 39, 41, 43, 45, 47, 49\}$, independently compute the current endpoint, optimize a late residual, screen strengths, and verify eligible realized responses.

**Shared late-stage settings and query accounting.** For both backbones, the multipliers $(m_1, \dots, m_8) = (1, 0.75, 0.5, 0.375, 0.25, 0.1875, 0.125, 0.0625)$ correspond one-to-one to the eight late interventions in chronological order. At the $i$th late intervention, the tested endpoint strengths are $\alpha \in \{0.25m_i, m_i\}$. Adam uses learning rate $0.03$ and RMS bound $0.30$, with four to eight iterations. After at least four iterations, it can stop upon a 50% observation improvement or two consecutive relative gains below $0.5\%$. A reference error at or below $10^{-10}$ skips refinement. The realization search needs at most two candidate velocity queries, in addition to its reference query; these are not necessarily individual network forward passes when guidance is used.

## B Derivations and scope of local guarantees

The following statements explain the update and verification rules. They do not establish an optimal partition into three stages or a ground-truth restoration guarantee.

**Fixed-noise transport.** Suppose a state is represented as $z_\sigma = (1-\sigma)z_0 + \sigma\epsilon$. Replacing $z_0$ by $z_0 + a$ and holding $\epsilon$ fixed gives

$$z'_\sigma = (1-\sigma)(z_0 + a) + \sigma\epsilon = z_\sigma + (1-\sigma)a. \tag{50}$$

This is the transport in Eq. 9. It is an algebraic construction under the chosen interpolation, not a claim that the frozen vector field preserves the chosen decomposition after intervention.

**Requested versus realized endpoint.** At fixed time and condition, abbreviate the vector field by $v(z)$ and write $F(z) = z - \sigma v(z)$. With $z' = z + (1-\sigma)a$, the realized response is

$$F(z') - F(z) = (1-\sigma)a - \sigma\big(v(z') - v(z)\big). \tag{51}$$

Consequently, its discrepancy from the requested endpoint is exactly

$$F(z') - \big(F(z) + a\big) = -\sigma\big[a + v(z') - v(z)\big]. \tag{52}$$

This follows by substitution and collecting terms. For $\sigma > 0$, exact endpoint realization would require $v(z') - v(z) = -a$, which is not enforced by the frozen editor. The identity therefore motivates same-time evaluation; it does not imply that residual cancellation must succeed. It applies to middle/late fresh-velocity evaluation, not to RouteCal's decoded preview rule.

**Intervention-local late non-increase.** Let $x_{\mathrm{ref}} = \mathcal{G}(F_k(z_k))$ and let $x_{\mathrm{out}}$ be the same-time endpoint after the late decision. Assume the reference and candidates are evaluated with the same fixed observation model, uncertainty floor, and score definitions. If all candidates fail, $x_{\mathrm{out}} = x_{\mathrm{ref}}$. Otherwise, the retained candidate passes the three tests in Eq. 49. To match the numerical acceptance criteria in that equation, define

$$\gamma_E = 0.01, \qquad \epsilon_{L,k} = 0.002 + 0.01\,|L_{\mathrm{stru}}(x_{\mathrm{ref}}; y)|\,, \qquad \gamma_H = 0.95. \tag{53}$$

In either case,

$$\begin{aligned} E_q(x_{\mathrm{out}}; y) &\le E_q(x_{\mathrm{ref}}; y), \\ L_{\mathrm{stru}}(x_{\mathrm{out}}; y) &\le L_{\mathrm{stru}}(x_{\mathrm{ref}}; y) + \epsilon_{L,k}, \\ H(x_{\mathrm{out}}) &\ge \gamma_H H(x_{\mathrm{ref}}). \end{aligned} \tag{54}$$

For a retained nonzero action with positive reference error, the first inequality strengthens to

$$E_q(x_{\text{out}}; y) \leq (1 - \gamma_E) E_q(x_{\text{ref}}; y). \tag{55}$$

The proof is the two-case argument above: rejection is identity, and acceptance explicitly enforces the inequalities. This is a property of the acceptance rule, not convergence of the Adam subproblem.

The inequalities apply only at the evaluated intervention step. Subsequent scheduler steps can change all three scores. A mismatched forward model can favor a restoration that is less faithful to the unknown clean image, and preserved mid-band energy does not imply preserved semantic detail. Thus, neither monotone error across the whole trajectory nor improved PSNR/LPIPS follows from Eq. 54.

**Scope of minimum control and stage decomposition.** The late rule prefers the smallest proposal-feasible strength on a finite tested direction, with a distinct observation-best fallback. It is not a minimization over all latent directions, all strengths, or all realized-feasible candidates. Likewise, the three-stage design is motivated by phase-dependent evidence and editability and tested empirically; these derivations do not prove that all three stages help every task. This distinction is particularly important when an ablation improves a task-specific metric.

# C Experimental details

## C.1 Dataset sources and benchmark construction

Table 3 specifies the evaluation sources and the pairing used for each task. The benchmark uses fixed subsets, not the full test sets of these datasets. Source-pair selection uses seed 42. Haze is balanced between indoor and outdoor images, and REED fixes the input exposure to N3 before selecting scene pairs. FlowSteer100 uses uniform sampling without replacement from its 100 source images.

Table 3: Evaluation data. Each task contains 50 pairs. Noise and haze are synthetic degradations; GoPro blur is constructed by frame averaging. REED and RealSR provide paired captures.

| Task | Source | Input–reference pairing |
|---|---|---|
| Denoising | CBSD68 | Released `noisy25` image and same-name `original_png` image. |
| Dehazing | RESIDE SOTS | 25 indoor and 25 outdoor hazy/clean pairs. |
| Low light | REED test | Same scene, N3 ($-3$EV) input and N0 reference. |
| Super-resolution | RealSR V3 test | Canon/Nikon $\times 4$ LR/HR pairs from the released test directories. |
| Deblurring | GoPro test | Same sequence and frame name in the blur/sharp directories. |
| Colorization | FlowSteer100 | Arithmetic RGB mean, rounded down to 8-bit values and replicated across channels; original RGB reference. |

**Public sources.** CBSD68 derives from the Berkeley segmentation benchmark (Martin et al., 2001); the paired clean/noisy files used here are distributed in the CBSD68 dataset archive. The other sources are the RESIDE SOTS release, REED release, RealSR V3 release, GoPro test release, and FlowSteer100 archive. The manifest records source filenames, scene identifiers, input/reference paths, and the preprocessing applied to each pair.

**Cropping and resolution.** The crop construction uses seed 42, a maximum reference area of $1{,}048{,}576$ pixels, and a spatial alignment of 16 pixels. Input and reference use the same crop coordinates for equal-resolution tasks. For RealSR, reference coordinates are four times the low-resolution coordinates, preserving the released scale relationship. No resampling is applied when constructing these crops, and the derived pairs are saved as PNG files. The selected FlowSteer100 images remain at their native resolution without cropping. These procedures are separate from any internal resizing or canvas handling performed by an editor.

**Metrics.** PSNR and SSIM are computed on RGB arrays in $[0, 1]$ using a data range of one. SSIM uses the channel-aware implementation in scikit-image. LPIPS uses AlexNet with images mapped to $[-1, 1]$. Each metric is computed per image and then averaged over the 50-image task subset. No additional evaluation border crop or luminance-only conversion is applied. RealSR follows this same RGB evaluation protocol. References are used only for evaluation.

**Restoration instructions.** Table 4 lists the exact task-specific restoration instructions supplied with the degraded input. The same instruction is used for all 50 images of a task, for both Qwen-Image-Edit and FLUX.2, and across all compared methods.

Table 4: Exact restoration instructions used for the six tasks.

| Task | Restoration instruction |
|---|---|
| Denoising | Remove image noise and restore clean, natural details while preserving the original content, geometry, identity, colors, and text. |
| Dehazing | Remove haze and restore clear visibility and natural contrast while preserving the original content, geometry, identity, colors, and text. |
| Low light | Correct low-light degradation and restore natural brightness and contrast while preserving the original content, geometry, identity, colors, and text. |
| Super-resolution | Upscale the image by 4x and restore natural high-resolution details while preserving the original content, geometry, identity, colors, and text. |
| Deblurring | Remove motion blur and restore a sharp, natural image while preserving the original content, geometry, identity, colors, and text. |
| Colorization | Colorize this grayscale image with natural, realistic, and semantically plausible colors while preserving the original content, geometry, identity, lighting, text, and fine details. |

### C.2 Baseline reproduction and adaptation

Our reproduction principle is to preserve the published update sequence and official settings where applicable, while making the modifications required by a common input benchmark and conditional latent backbones explicit. The reference implementations are PnP-Flow and FlowSteer. Neither baseline uses RestoreSteer's three-stage controller or its accept/reject checks.

**PnP-Flow.** Our implementation follows the update sequence of the original PnP-Flow solver (Martin et al., 2025): a data-fidelity gradient step, reprojection onto the flow path by interpolation with fresh noise, time-dependent flow denoising, and averaging over noise samples. Following the official test script, we use 100 iterations, five noise samples per iteration, a base learning rate of 1.0, and the $(1-t)^{\alpha}$ learning-rate schedule. The schedule exponent $\alpha$ is set to 0.8 for denoising, 0.01 for deblurring, and 0.3 for super-resolution. Because the original implementation does not provide settings for dehazing, low-light restoration, or colorization, we use $\alpha = 1.0$ for these task extensions.

The original solver operates on image-space flow priors. Here its state is a clean latent of the chosen editor, and the data gradient is computed through the frozen VAE decoder. The flow prior is conditioned on the benchmark observation and task instruction. The adjoint-based image initialization is encoded into the latent space; haze and low-light extensions use identity-adjoint initialization. This implements the published update sequence with the chosen conditional latent prior.

All methods use the same benchmark observations. The forward maps are identity for noise, Gaussian convolution for blur, area reduction for SR, arithmetic grayscale for colorization, and estimated scattering or illumination for haze and low light. Gaussian blur and area reduction are approximations for GoPro motion blur and RealSR camera degradation.

**FlowSteer.** The adaptation retains the published operator-aware correction along the flow trajectory and the task-dependent correction schedule. The Qwen denoising run uses the official-schedule identity configuration; deblurring uses the official-schedule Gaussian/Wiener configuration. SR and colorization use task-specific observation operators. FLUX.2 uses a 50-step adaptation with the correction schedule expressed relative to the sampling length. Haze and low-light restoration extend the original task set using atmospheric and illumination operators. We adapt the time convention, latent

interface, and task conditioning to Qwen and FLUX.2 while retaining FlowSteer's operator-based correction procedure.

**Comparison budget.** The native editors, FlowSteer, and RestoreSteer use 40 Qwen or 50 FLUX.2 scheduler steps, whereas PnP-Flow has its own iteration and resampling budget. RestoreSteer also adds VAE optimization and same-time velocity queries. The comparison uses each method's stated inference schedule, with computational costs determined by its optimization and resampling operations.

### C.3 RestoreSteer settings

**Intervention schedules.** On Qwen, RouteCal acts after step 5. StructAlign starts at step 15, with primary retries through step 22; accepted structural actions determine the subsequent alignment and response-correction opportunities. ObsRefine acts at steps $\{25, 27, 29, 31, 33, 35, 37, 39\}$. On FLUX.2, the corresponding early step is 10, middle opportunities are around steps 19–21, and the late intervention steps are $\{35, 37, 39, 41, 43, 45, 47, 49\}$. For both backbones, the ordered multipliers

$$(m_1, \ldots, m_8) = (1, 0.75, 0.5, 0.375, 0.25, 0.1875, 0.125, 0.0625)$$

correspond one-to-one to the eight late interventions. At the $i$th late intervention, we test endpoint strengths $\alpha \in \{0.25m_i, m_i\}$.

**Early and middle optimization.** Structural alignment uses 16 Adam steps at learning rate 0.08, deformation regularization weight 0.02, and retains a smaller tested deformation achieving at least 80% of the best structural gain. Encoding differences are tested at fractions $\{0.25, 0.5, 0.75, 1\}$ with total RMS at most 0.5. Decoder-based residual refinement uses three Adam steps at learning rate 0.02, latent and image penalty weights of 0.05, and residual RMS at most 0.04. RouteCal's cached-velocity preview requires an absolute decrease of at least $0.002$ in the normalized joint score before a bounded state write. Middle-stage primary endpoint strengths are $\{0.5, 1\}$. The second-alignment gate uses a gradient-direction cosine threshold of 0.90 and a support fraction threshold of 0.010. Response checks require directional cosine at least 0.2 and orthogonal-response fraction at most 0.95. The internal latent-error correction is bounded to RMS 0.08.

**Late optimization and acceptance.** ObsRefine uses Adam (Kingma & Ba, 2015) with learning rate 0.03, four to eight iterations, correction RMS at most 0.30, and regularizer weights $\lambda_z = 0.02$ and $\lambda_x = 0.05$. After at least four iterations, optimization stops on reaching a 50% observation-error reduction or after two consecutive relative gains below 0.5%. Candidate eligibility requires at least 1% observation-error reduction, a structure-score increase of at most $0.002 + 0.01|L_{\mathrm{stru}}(\hat{x}_{0,k}; y)|$, and retention of at least 95% of the reference detail statistic. These tests are applied to decoded proposals and re-evaluated endpoints. The smallest passing proposal strength is tested in the live state first, followed, if necessary, by the observation-best screened strength when distinct. This is a bounded search, not an exhaustive minimum over all live actions.

**Observers.** The observation objective uses average-pooling scales $\{1, 2, 4\}$, robust-penalty smoothing constant $10^{-3}$, and an input-estimated uncertainty floor clipped to $[0.01, 0.20]$. Haze and illumination parameters are estimated from the input and fixed during late optimization. Regional structure uses a $9 \times 16$ grid, combining the overall mean with the mean over the highest-error 20% of regions; edge-magnitude and direction weights are 0.35 and 0.1. The detail statistic combines locally contrast-normalized mid-band responses with box-filter sizes 3, 7, and 15 and weights 0.65 and 0.35. Its normalization constant is $10^{-4}$, and trimmed RMS excludes the lowest and highest 10% of absolute responses. Full mathematical definitions remain in the controller-details appendix.

### C.4 Inference time

Table 5 reports measured per-image wall-clock times recorded during the benchmark runs used in Table 1, rather than estimates based on sampling steps or model-query counts. We report four tasks, with 50 images per task and 200 images per method. Model loading and evaluation metrics are excluded. Native sampling, FlowSteer, and RestoreSteer use 40 steps on Qwen and 50 on FLUX.2; PnP-Flow uses 100 iterations with five noise samples per iteration.

RestoreSteer adds computation beyond native sampling through spatial-alignment optimization, VAE encoding and decoding of proposals, decoder-based latent refinement, and fresh same-time model queries for candidate verification. These operations account for additional computational work even though the backbone remains frozen and the number of native scheduler steps is unchanged. The recorded mean times are higher than those of Native and FlowSteer on both backbones, but lower than those of PnP-Flow under the evaluated schedules.

The runs were not all collected on the same physical GPU. These results therefore document the actual recorded costs of the benchmark configurations, rather than a controlled latency comparison.

Table 5: Mean recorded inference time on four restoration tasks. Each task averages 50 inputs; Average pools the 200 inputs from these four tasks per method. Methods use their respective inference schedules.

| Method | Inference time (s/image) ↓ | | | | |
|---|---|---|---|---|---|
| | Dehazing | Low light | SR | Deblurring | Average |
| **Qwen-Image-Edit** | | | | | |
| Native | 51.46 | 91.02 | 86.91 | 83.76 | 78.29 |
| PnP-Flow | 666.02 | 1179.58 | 1142.73 | 1094.10 | 1020.61 |
| FlowSteer | 56.62 | 106.21 | 88.99 | 92.56 | 86.10 |
| RestoreSteer | 99.04 | 200.04 | 178.73 | 168.69 | 161.63 |
| **FLUX.2** | | | | | |
| Native | 73.35 | 138.03 | 94.65 | 134.55 | 110.15 |
| PnP-Flow | 265.68 | 1034.84 | 527.21 | 909.34 | 684.27 |
| FlowSteer | 75.43 | 167.15 | 100.43 | 141.75 | 121.19 |
| RestoreSteer | 104.77 | 272.41 | 278.74 | 261.31 | 229.31 |

### C.5 Random seeds and evaluation protocol

**Random seeds.** Qwen native sampling uses seed 12345 for each image. RestoreSteer, the component ablations, PnP-Flow, and the FLUX.2 comparisons use per-image seeds determined by the benchmark manifest index. Each task score averages one sampling run per image. In the component ablations, images and per-image seeds are held fixed across variants.

**Evaluation procedure.** We evaluate saved final outputs against the paired references specified by the benchmark manifests. Sample identifiers establish the correspondence between inputs, references, and outputs. All methods use the same task-specific output canvas and the RGB metric protocol described in Appendix C.1.

**Qualitative presentation.** The qualitative figure uses final outputs from the quantitative experiments. We assemble corresponding inputs, restored images, and references into a single comparison grid, preserving image content without additional restoration or contrast adjustment.

## D Additional qualitative comparisons

We present additional examples selected to illustrate improvements in scene correspondence, appearance, and detail. Qwen-Image-Edit and FLUX.2 are shown in separate groups; corresponding sets use the same inputs. Rows cover denoising, dehazing, low light, super-resolution, deblurring, and colorization. The comparisons use saved final outputs, with full image content and aspect ratios preserved and no additional restoration or contrast adjustment.

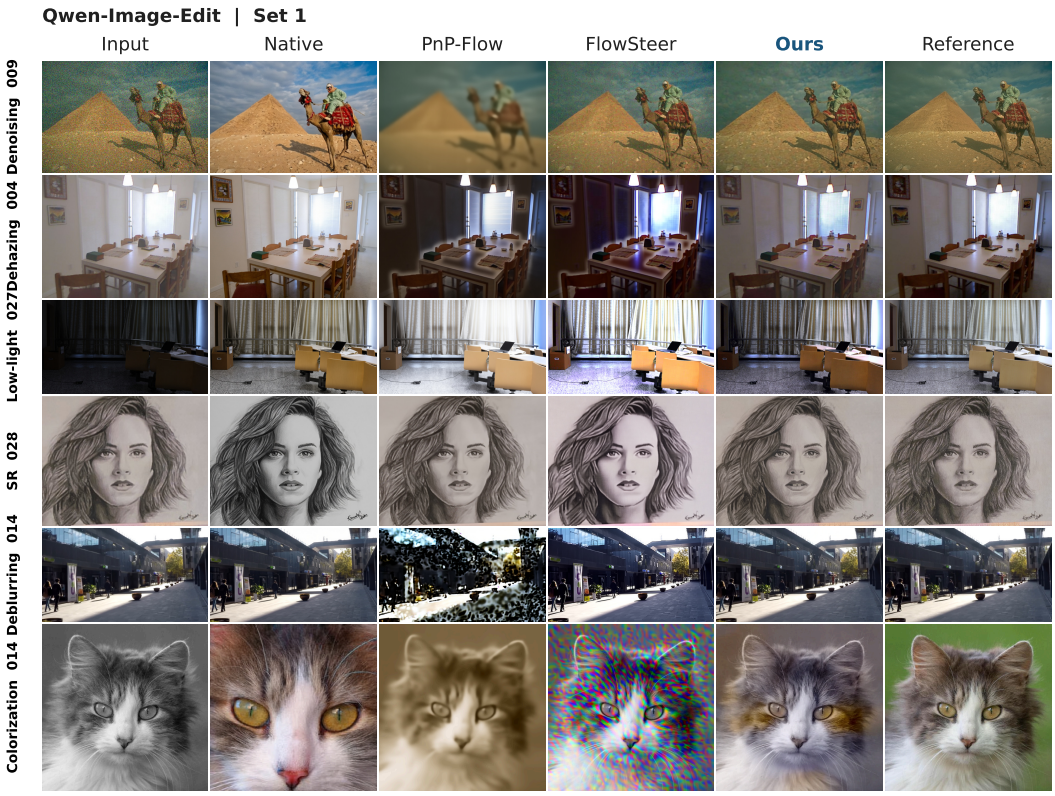



Figure 3: **Additional comparisons on Qwen-Image-Edit (set 1).** Columns show the input, native editor, PnP-Flow, FlowSteer, RestoreSteer, and reference. Row labels identify the task and sample.

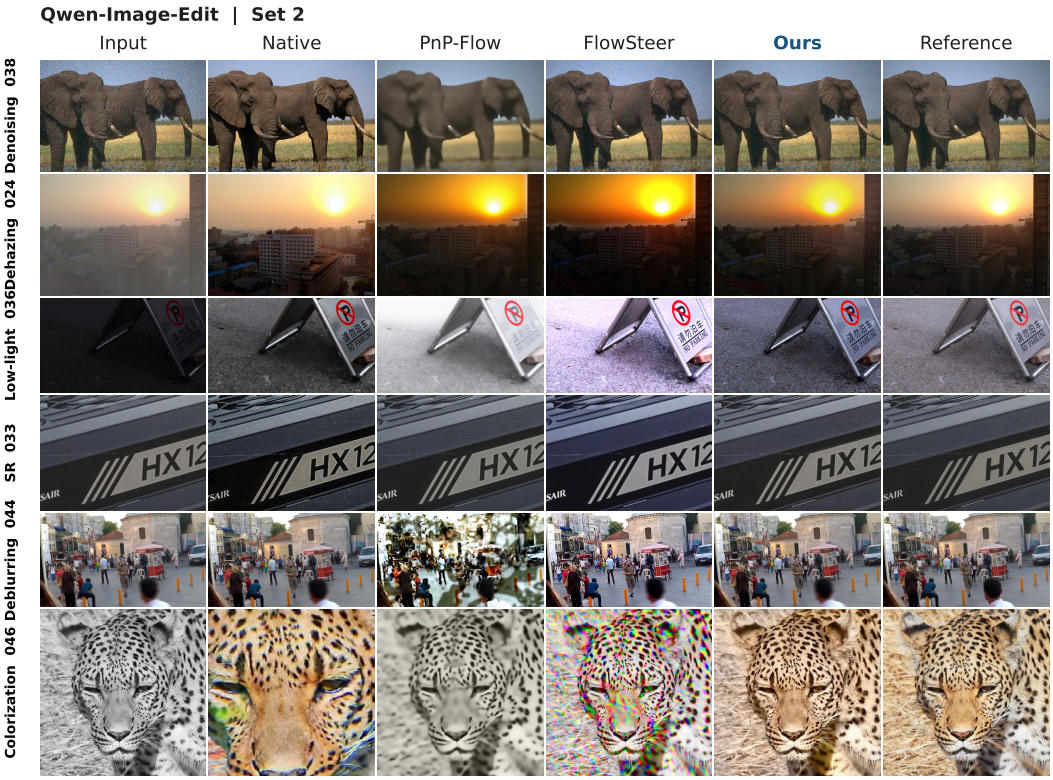



Figure 4: **Additional comparisons on Qwen-Image-Edit (set 2).** Columns show the input, native editor, PnP-Flow, FlowSteer, RestoreSteer, and reference. Row labels identify the task and sample.

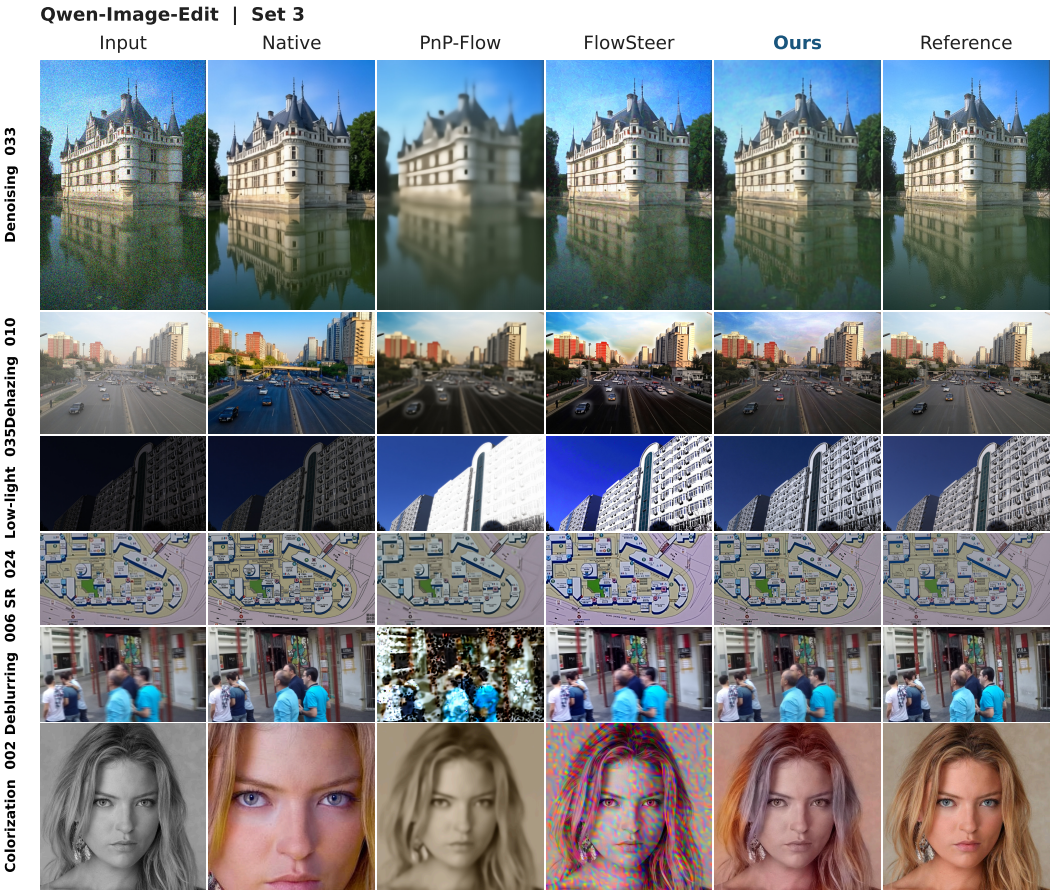



Figure 5: **Additional comparisons on Qwen-Image-Edit (set 3).** Columns show the input, native editor, PnP-Flow, FlowSteer, RestoreSteer, and reference. Row labels identify the task and sample.

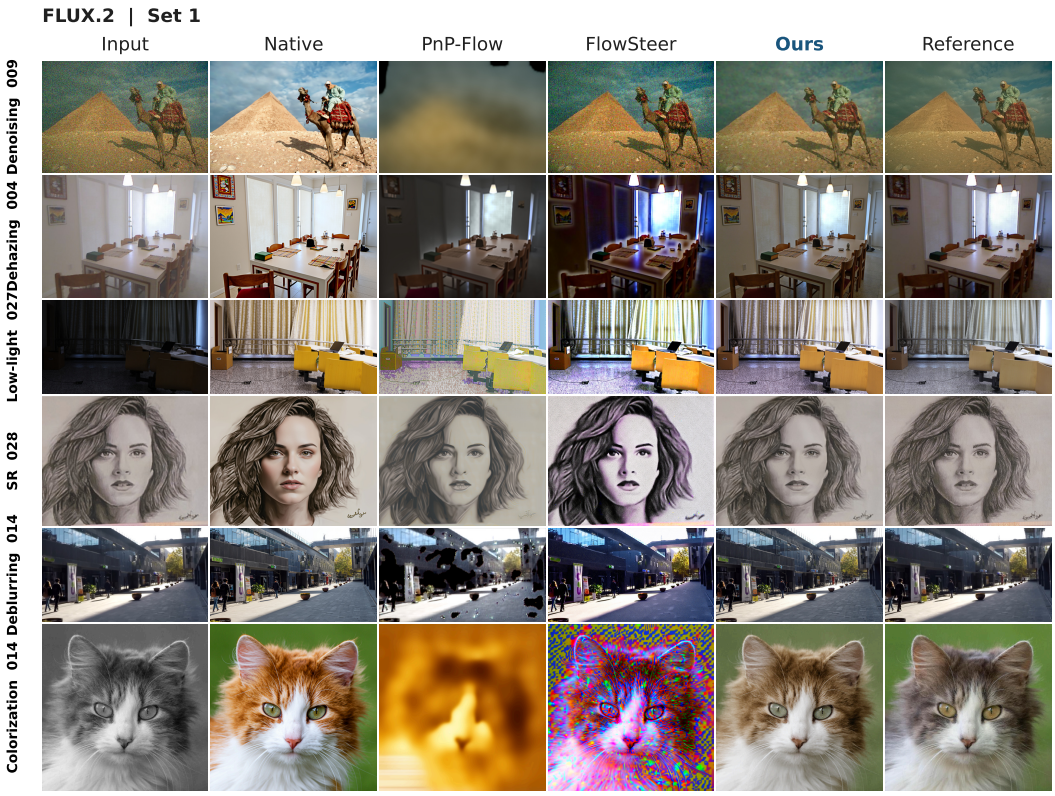



Figure 6: **Additional comparisons on FLUX.2 (set 1).** Columns show the input, native editor, PnP-Flow, FlowSteer, RestoreSteer, and reference. Row labels identify the task and sample.

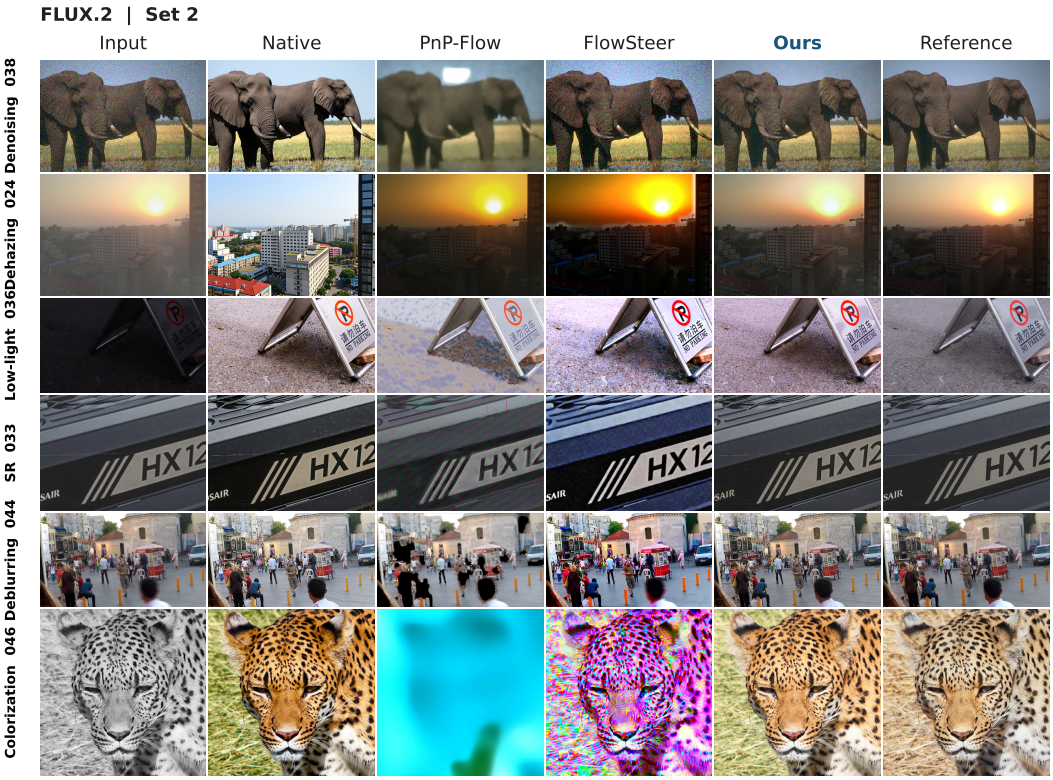



Figure 7: **Additional comparisons on FLUX.2 (set 2).** Columns show the input, native editor, PnP-Flow, FlowSteer, RestoreSteer, and reference. Row labels identify the task and sample.

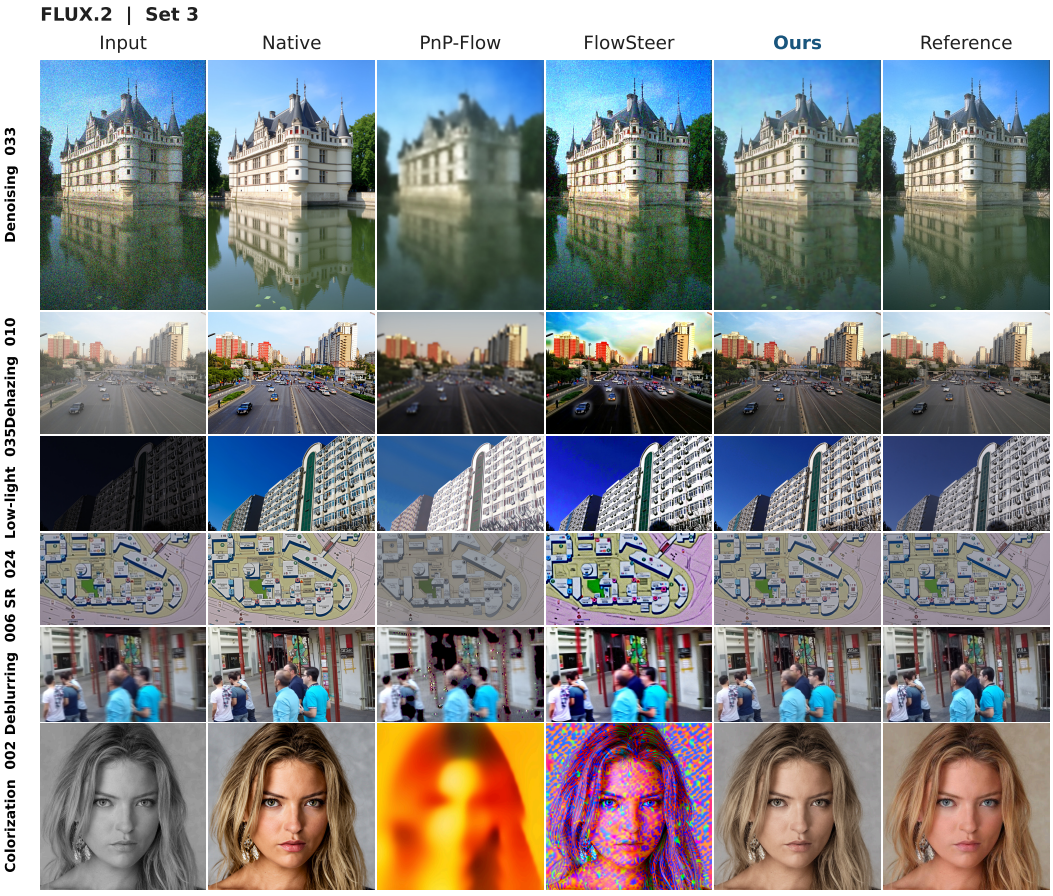
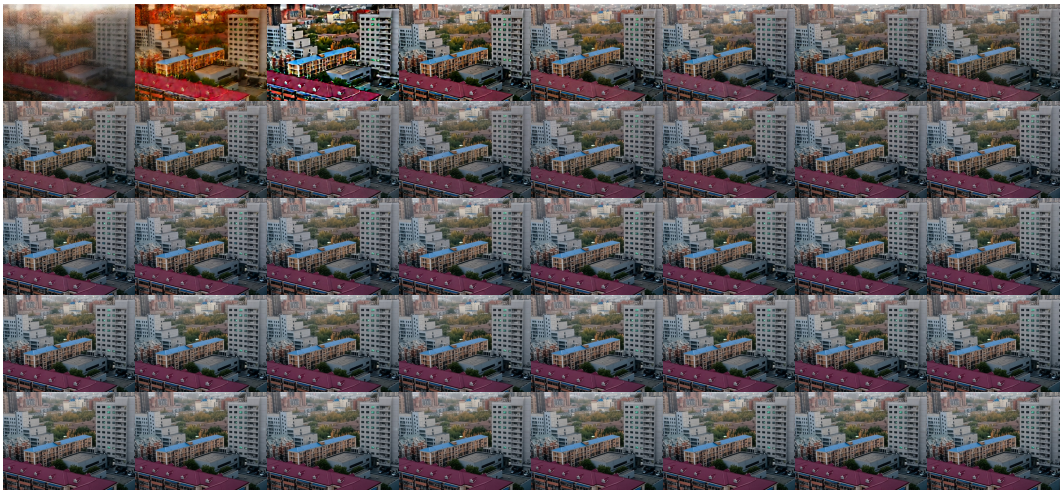



Figure 8: **Additional comparisons on FLUX.2 (set 3).** Columns show the input, native editor, PnP-Flow, FlowSteer, RestoreSteer, and reference. Row labels identify the task and sample.

## E VISUALIZATION OF THE SAMPLING PROCESS

Figures 9 and 10 visualize the sampling process of native Qwen-Image-Edit and RestoreSteer on the same dehazing example. For each method, the upper grid shows the current clean endpoint predictions, while the lower grid shows decoded live latent states. Each grid contains all 40 sampling steps arranged in five rows and eight columns, with sampling progressing from left to right and top to bottom. These visualizations distinguish the evolving clean-image prediction from the noisy state through which sampling proceeds.

**(a) Clean endpoint predictions**

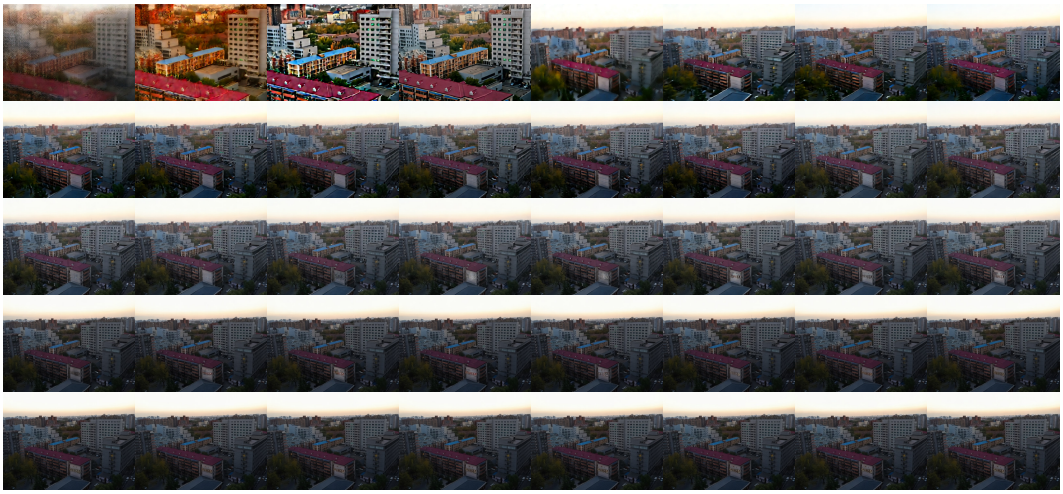


**(b) Decoded sampling states**



Figure 9: **Sampling process of native Qwen-Image-Edit on HAZE-014.** (a) Current clean endpoint predictions at the recorded sampling steps. (b) Decoded live latent states at the corresponding sampling steps. Each grid shows all 40 sampling steps in chronological order, Each grid shows all 40 sampling steps in chronological order, from left to right and top to bottom.

**(a) Clean endpoint predictions**



**(b) Decoded sampling states**

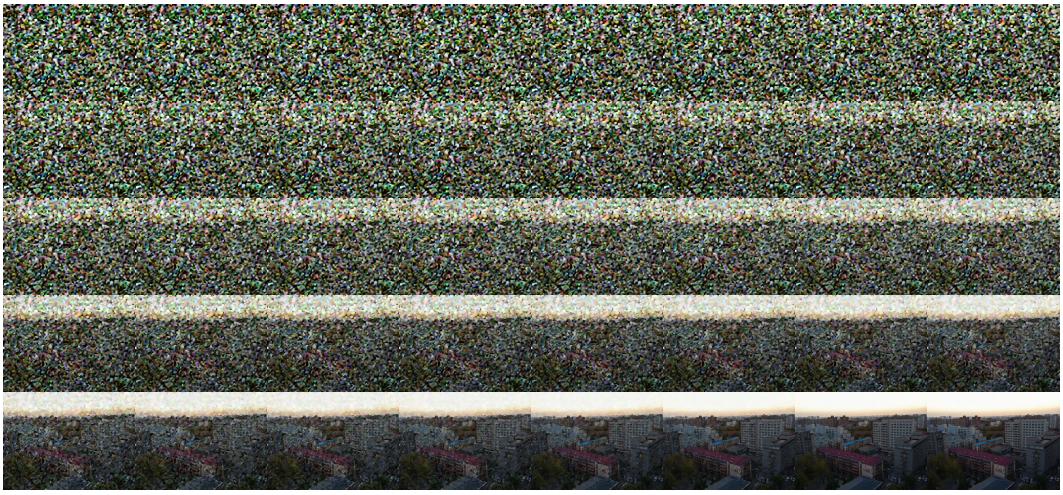


Figure 10: **Sampling process of RestoreSteer on Qwen-Image-Edit for HAZE-014.** (a) Current clean endpoint predictions along the controlled trajectory. (b) Decoded live latent states at the corresponding sampling steps. Each grid shows all 40 sampling steps in chronological order, from left to right and top to bottom.