INTERVENABLE PARTICLE REPRESENTATIONS FOR CONDITIONAL IMAGE RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce Particle-U-Net (PUN), a generalization of neural particle automata (NPA). PUN carries particle self-organization from pixel space into the spatial latent space of a frozen non-compressing autoencoder. It injects target-derived features into the particle update via feature-wise linear modulation and reads the resulting particle render into an image with a residual U-Net. NPA learns one rule per target and does not reach high fidelity on natural images. PUN removes both limitations: it handles natural images, generalizes to unseen faces without per-image optimization, and recovers image fidelity that pixel-space fitting does not reach. Beyond reconstruction, we study the explicit particle representation through recovery, coordinate interventions, and component comparisons. Position updates improve recovery from localized damage. Localization after imposed coordinate transformations depends on the spatial FiLM condition during subsequent updates. We also apply directional CLIP guidance to the pixel-domain particle update rule, asking whether the learned dynamics can be steered toward attribute-level edits.

1 INTRODUCTION

Conditional image reconstruction underlies restoration, super-resolution, and editing: a model trained once maps an encoding of a target image x to a reconstruction $y \approx x$, with no optimization at inference. Nearly every system performs it in a single dense forward pass, and the computation that produces the image survives only as a stack of feature maps. Nothing inside that stack corresponds to a named piece of the image, so the process that formed it can be inspected only in aggregate and manipulated not at all. A representation with explicit dynamics would make image formation something to study, not merely to use. Neural cellular automata (NCA) take a step in that direction by treating an image as the endpoint of a process (Mordvintsev et al., 2020): cells on a fixed pixel lattice repeatedly apply a local update rule, and the image emerges from the collective state. NPA replaces that lattice with particles: each carries a continuous position and an internal state, and one shared learnable rule updates both (Pajouheshgar et al., 2026a).

A trained NPA, however, is tied to its training target: its rule is learned per target, so every new target requires a separate training run. Nor does it reach high fidelity on natural images: the particles must carry the image's detail themselves, and with their number fixed, the detail they can hold is bounded.

We introduce PUN, a generalization of NPA that keeps the explicit particle representation while removing the per-target fitting (Fig. 1). PUN moves self-organization into the spatial latent of a frozen non-compressing autoencoder, injects target-derived features into the particle update through FiLM (Perez et al., 2018), and reads the particle render into an image with a residual U-Net. Placing the rollout in that latent removes both limitations at once: the latent preserves the spatial resolution at which particle neighbourhoods are defined, and the particles no longer carry pixel-level detail themselves. Because a shared rule alone cannot tell two targets apart, the update is also conditioned on a feature sampled at each particle's own position.

We validate PUN in two ways. The first is reconstruction: trained once on CelebA, PUN reconstructs disjoint held-out faces without per-image optimization across three seeds. The second asks what the explicit particle representation makes possible beyond reconstruction: what the trained readout depends on, whether position updates improve recovery from localized damage, and how spatial

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.

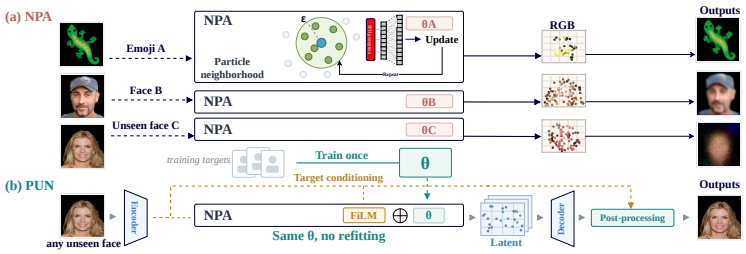

Figure 1: (a) NPA learns the rule per target: each new image needs its own training run. (b) PUN trains it once and conditions the update on the latent of a frozen encoder, so an unseen face is reconstructed without refitting.

conditioning affects imposed coordinate transformations during subsequent updates. An additional exploration asks whether directional guidance can steer the pixel-domain update rule.

Our contributions are as follows.

1. **A shared conditional particle reconstruction framework.** PUN organizes particles in the latent of a frozen non-compressing autoencoder, injects target-derived features into the particle update through FiLM, and reads the particle render back into an image with a residual U-Net. Particle dynamics and readout are trained jointly, and particle positions and states remain explicit throughout the rollout.
2. **Held-out reconstruction without per-image optimization.** Trained once on CelebA, one shared model reconstructs disjoint held-out faces, and component comparisons characterize what each part of the pipeline contributes.
3. **Intervenable particle states.** Path scaling and state substitution establish the particle pathway's contribution to the trained readout. Position updates improve recovery from localized damage. Coordinate interventions on the particle render reveal how spatial FiLM conditioning affects localization during subsequent updates (Secs. 4.2–4.4).

2 Related Work

Cellular automata and particle systems. Neural cellular automata replace hand-crafted rules with a learned rule: cells on a fixed lattice carry an internal state and repeatedly update it using local perception. Applications include texture synthesis (Niklasson et al., 2021), growth and morphogenesis (Mordvintsev et al., 2020), distributed classification (Randazzo et al., 2020), and segmentation (Kalkhof et al., 2023). NPA removes the lattice itself (Pajouheshgar et al., 2026a): each particle carries a continuous position and an internal state, both updated by one shared learnable rule, and perception becomes a mesh-free smoothed particle hydrodynamics (SPH) operator over particle neighbourhoods. That rule is still learned for the target it is applied to. We keep the particle dynamics explicit and remove the per-target element: one rule is trained once over an image distribution. Related particle-based methods include learned morphogenesis (Pahng et al., 2025) and physics-based simulation of Gaussian representations (Xie et al., 2024).

Conditional reconstruction and the readout stage. Dense image models include denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020; Saharia et al., 2023), flow and diffusion models based on interpolants (Ma et al., 2024), adversarial super-resolution (Ledig et al., 2017), and residual diffusion (Liu et al., 2024). Latent variants map the image to a learned code and model the distribution there (Kingma & Welling, 2014; Rombach et al., 2022), and recent work tunes the autoencoder and the generator together to reduce the cost of that split (Leng et al., 2025; Lin et al., 2026). A second line renders an intermediate representation and refines it, whether the render is a point cloud (Aliev et al., 2020), neural textures (Thies et al., 2019), a radiance field (Wang et al., 2022; Zhou et al., 2023; Bösiger et al., 2024), or a degraded image restored by a residual network (Kim et al., 2016; Zhang et al., 2017; Zamir et al., 2022) or a diffusion model (Sun et al., 2025; Wang et al., 2025; 2024; Selikhanovych et al., 2026). The automata family approaches the same

shape from the other side, pairing a coarse grid with an implicit decoder that renders cell states at arbitrary resolution (Pajouheshgar et al., 2026b). Our first stage is neither a decimated image nor a code: a particle rollout produces the render, and it runs in the latent of a frozen, non-compressing autoencoder, so the spatial resolution at which particle neighbourhoods are defined survives the encoding. A residual U-Net (Ronneberger et al., 2015) then reads out that render. Distortion and perceptual quality are distinct evaluation criteria (Blau & Michaeli, 2018); we report both when comparing these stages against dense baselines.

Conditioning, guidance, and stability. Feature-wise linear modulation conditions a network by scaling and shifting its hidden activations (Perez et al., 2018), and it is what lets one particle rule respond to different targets and to different regions of the same target. Guidance toward a text- or attribute-level direction has been studied in GAN latents (Patashnik et al., 2021; Gal et al., 2022), in pixel and feature optimization (Kwon & Ye, 2022; Bar-Tal et al., 2022), and as a set of editing operations on diffusion trajectories (Hertz et al., 2023; Mokady et al., 2023; Kawar et al., 2023; Couairon et al., 2023; Kim et al., 2022). Conflicting objectives motivate gradient surgery (Yu et al., 2020) and conflict-averse gradient descent (Liu et al., 2021), while GradNorm balances gradient magnitudes (Chen et al., 2018). Exponential moving averages of model weights are also used for inference (Karras et al., 2020). We also apply directional guidance to the particle update rule. It asks whether the learned dynamics can be steered and what stabilization that steering requires.

3 METHOD

In this section, we first review the particle rule and the frozen autoencoder whose latent hosts the rollout. Sec. 3.3 then describes the conditioned rollout and its rendering, Sec. 3.4 the residual readout, and Sec. 3.5 the objective that trains both stages jointly. Fig. 2 summarizes the pipeline.

3.1 NEURAL PARTICLE AUTOMATA

In NPA (Pajouheshgar et al., 2026a), particle i carries a continuous position $\boldsymbol{x}_i \in \mathbb{R}^D$ and an internal state $\boldsymbol{s}_i \in \mathbb{R}^C$, and one rule updates both. Perception is mesh-free: a differentiable SPH operator aggregates kernel-weighted contributions from the particles inside a support radius ϵ, which gives each particle the perception vector

$$\boldsymbol{p}_i = \left[\, \boldsymbol{s}_i,\ \tilde{\boldsymbol{s}}_i,\ \nabla \boldsymbol{s}_i,\ \nabla \rho_i \,\right], \tag{1}$$

stacking its own state, the neighbourhood-smoothed state $\tilde{\boldsymbol{s}}_i$, the local state gradient $\nabla \boldsymbol{s}_i$, and the density gradient $\nabla \rho_i$. A small multilayer perceptron (MLP) f_θ, shared across particles, maps this vector to position and state increments:

$$(\Delta \boldsymbol{x}_i, \Delta \boldsymbol{s}_i) = f_\theta(\boldsymbol{p}_i), \tag{2}$$

which are applied additively to a random subset of particles. Each particle is selected independently at each step with probability 0.5, so a rollout of T steps is the repeated application of one shared rule. The states are then decoded into renderable attributes and rasterized by Gaussian splatting.

NPA learns this rule for the target it is applied to, so each new image needs its own training run, and the particles must carry the image's detail themselves.

3.2 FROZEN NON-COMPRESSING AUTOENCODER

PUN performs self-organization in the spatial latent of an autoencoder rather than in pixel space. The encoder maps an image $x \in \mathbb{R}^{3\times 64\times 64}$ to a latent $c = \mathrm{Enc}(x) \in \mathbb{R}^{8\times 64\times 64}$, and the decoder inverts that map. The architecture is non-compressing, so the latent keeps the spatial resolution at which particle neighbourhoods are defined. This is what makes the latent usable as a space for the rollout. A particle neighbourhood is a region of the image, and a latent that had already pooled that region would remove the locality the update rule acts on.

We train the autoencoder once on 46 computer-generated faces and freeze it before the particle model is trained. The target encoding c is therefore fixed throughout, and the particles never carry pixel-level detail on their own: the update organizes the latent, and the readout turns that organization into image fidelity. Channel count, kernel size, and depth were selected independently of the test set (App. K); the reference configuration uses 8 channels, 3×3 kernels, and depth 2.

3.3 CONDITIONED PARTICLE ROLLOUT IN THE LATENT

The rollout reuses NPA's particles and runs them on the latent grid. We use 4,096 particles with 16-dimensional states, positions confined to $[-1, 1]^2$ and lifted by a fixed Fourier positional encoding, and an update MLP with one 128-unit hidden layer and 18 outputs, two for the position increment and sixteen for the state increment.

Conditioning by feature-wise linear modulation. A shared rule alone cannot tell two targets apart, and it must also behave differently in different regions of a single image. We therefore modulate the hidden activations of the update MLP using FiLM (Perez et al., 2018) rather than modulating its input. For each particle,

$$\boldsymbol{h} \leftarrow \boldsymbol{\gamma}_i \odot \boldsymbol{h} + \boldsymbol{\beta}_i, \tag{3}$$

where $(\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i)$ is predicted from the 8-dimensional target feature sampled from c at $\boldsymbol{x}_i$. Sampling at the particle's own position is what ties conditioning to geometry, so two particles with the same state in different regions still receive different modulation.

Rendering the latent. After T shared updates, a learned readout $h_\theta : \mathbb{R}^{16} \to \mathbb{R}^8$ maps each particle state to latent features, and normalized Gaussian splatting forms the latent grid:

$$\boldsymbol{z}[k, \boldsymbol{u}] = \frac{\sum_i h_\theta(\boldsymbol{s}_i)[k]\, K_{\sigma,i}(\boldsymbol{u} - \boldsymbol{q}_i)}{\sum_i K_{\sigma,i}(\boldsymbol{u} - \boldsymbol{q}_i) + \epsilon}, \tag{4}$$

where k indexes latent channels, $\boldsymbol{u}$ is a grid location, and $\boldsymbol{q}_i$ is the particle position in pixel coordinates. Each particle's Gaussian weights are normalized over its local stencil, with $\sigma = 0.5$ and $\epsilon = 10^{-8}$. The frozen decoder then produces the particle render $\boldsymbol{I} = \mathrm{Dec}(\boldsymbol{z})$.

3.4 CONDITIONED RESIDUAL READOUT

The frozen decoder converts the splatted latent grid into a particle-rendered image, which the residual U-Net subsequently refines. A residual U-Net U_ϕ (Ronneberger et al., 2015) takes the render as its only spatial input and predicts a correction:

$$\widehat{\boldsymbol{I}} = \mathrm{clamp}\big(\boldsymbol{I} + \mathrm{U}_\phi(\boldsymbol{I}) + g_\psi(c),\ 0, 1\big). \tag{5}$$

The target latent reaches this stage through the output-side conditional head $g_\psi(c)$, which adds a target-dependent residual; it is not passed to the U-Net itself, so the particle render remains the only spatial input to the refinement, and conditioning is left to the FiLM field and to g_ψ. The U-Net's output convolution is initialized to zero, so the readout begins as the identity map and the residual branch has to earn its contribution (Zhang et al., 2019; 2023). The U-Net has 1,753,187 parameters.

3.5 JOINT TRAINING OBJECTIVE

The two stages are trained together, which supervises the rollout at the level of the representation it builds and the readout at the level of the image it produces. The rollout matches the target latent under a mean Huber objective:

$$\mathcal{L}_{\text{latent}} = \mathrm{mean}[\mathrm{Huber}_{\delta=1}(\boldsymbol{z} - c)], \tag{6}$$

and the readout is supervised on foreground pixels $\mathcal{M} = \{\boldsymbol{u} : \alpha(\boldsymbol{u}) > 0.05\}$:

$$\mathcal{L}_{\text{readout}} = \frac{1}{|\mathcal{M}|} \sum_{\boldsymbol{u} \in \mathcal{M}} \frac{1}{3} \big\| \widehat{\boldsymbol{I}}[\boldsymbol{u}] - \boldsymbol{I}^{\text{tgt}}[\boldsymbol{u}] \big\|_2^2. \tag{7}$$

The joint objective adds range regularizers on the particle positions and states:

$$\mathcal{L} = \mathcal{L}_{\text{latent}} + 5\mathcal{L}_{\text{readout}} + 100\mathcal{R}_{\text{pos}} + 100\mathcal{R}_{\text{state}}, \tag{8}$$

where $\mathcal{R}_{\text{pos}}$ and $\mathcal{R}_{\text{state}}$ are the mean absolute deviations of positions and states from their respective values clipped to $[-1, 1]$. The main model is trained for 30,000 iterations with a batch size of 4, and inference starts from a fixed particle initialization $\boldsymbol{S}_0$ with no per-image fine-tuning. App. D gives the optimizer schedule, the parameter split, and the inference cost, and Sec. 4.2 compares frozen and joint training from a shared particle warm start.

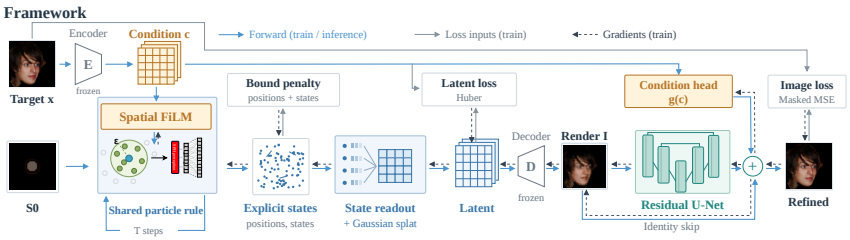

Figure 2: The PUN pipeline. A frozen encoder provides the condition that modulates the shared particle rule, which updates a fixed particle set for T steps. The states are splatted into a latent grid, decoded by a frozen decoder, and refined by a residual U-Net. Training supervises the latent, the image, the particle ranges, and the condition head.

4 Experiments

Our experiments follow the two axes along which we validate PUN: whether a shared conditional rule reconstructs held-out faces, and what the explicit particle states make possible once it does. Sec. 4.1 covers the first; the remaining experiments examine particle-path participation, component contributions, recovery from damage, and the dependence of coordinate interventions on spatial conditioning. Sec. 4.5 presents an exploratory study of the pixel-domain update rule.

Setup. We use the 5,000-face CelebA setting (Liu et al., 2015): aligned 64×64 crops with black-background composites and cutout-derived masks, partitioned into 4,000 training and 1,000 held-out images. The primary metric is foreground-masked PSNR ($\alpha > 0.05$). We also report full-image PSNR, full-image SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) with the AlexNet backbone, in a face-cropped and a full-image variant. The full pipeline and the conditional DDPM (cDDPM) baseline are trained with seeds $\{42, 99, 2024\}$; the remaining baselines and the component configurations use seed 42. Every reconstruction comparison in Secs. 4.1 and 4.2 is evaluated on the same 1,000 held-out images, which supports the paired analysis of App. A. App. B gives the evaluation details behind each table, and App. D the parameter counts and inference cost.

4.1 Reconstruction on held-out faces

PUN reconstructs the 1,000 disjoint held-out faces (Table 1). Its nearest conditional baseline, cDDPM, shares its conditioning yet remains far behind. PUN achieves masked PSNR 35.03 ± 0.08 dB, exceeding cDDPM's 20.21 ± 2.31 dB by 14.82 dB in the mean. The small spread across PUN's three training seeds shows that this reconstruction performance is consistent across the evaluated runs. Each run uses one shared rule for all test images.

The perceptual comparison is wider. PUN reaches LPIPS 0.0112 ± 0.0003 on the face crop and 0.0100 ± 0.0002 on the full image, against 0.1311 ± 0.0474 and 0.1380 ± 0.0524 for cDDPM. The dense baselines we compare against fall between the two on perceptual metrics and below PUN on masked PSNR, and they spread widely within that group: SiT and SRGAN lead it at LPIPS 0.0474 and 0.0801, while DDPM (low-res.) and VNCA trail at 0.1694 and 0.2025. Original NPA, scored under its own evaluation, is the weakest row at 12.13 dB.

One shared rule, trained once, reconstructs unseen faces across three seeds, while leading every baseline on masked PSNR. Fig. 3 shows the rollout itself, with particle positions and the pre-readout render forming on a held-out face, and Fig. 4 complements the aggregate scores with qualitative outputs on five held-out faces, annotated with their per-face PSNR. The two views connect representation formation to the final output: the rollout visualization exposes the evolving positions and particle render, while the qualitative reconstructions include the complete residual readout.

4.2 Particle and conditional-readout contributions

Particle-path participation. Two probes act on the readout: scaling the particle-dependent path by a scalar $\lambda \in [0, 1]$, and substituting another image's mature particle state. Sweeping λ from 0 to 1 raises full-image PSNR monotonically from 15.42 to 36.78 dB on 32 training images at seed 42;

Table 1: Reconstruction on the 1,000 held-out CelebA faces. PUN and cDDPM average three training seeds $\{42, 99, 2024\}$ (population SD); the remaining rows use seed 42. LPIPS uses the AlexNet backbone. Full-image PSNR and SSIM are in Table 6.

Model	Masked PSNR (dB)	LPIPS (face)	LPIPS (full)
PUN	$\mathbf{35.03 \pm 0.08}$	$\mathbf{0.0112 \pm 0.0003}$	$\mathbf{0.0100 \pm 0.0002}$
cDDPM	20.21 ± 2.31	0.1311 ± 0.0474	0.1380 ± 0.0524
VAE (β-anneal)	24.76	0.1085	0.1281
SRGAN	21.59	0.0801	0.0812
RDDM	19.00	0.1504	0.1561
DDPM (low-res.)	17.42	0.1694	0.1751
SiT	19.93	0.0474	0.0450
VNCA	17.11	0.2025	0.1992
Original NPA	12.13	0.5536	0.5914

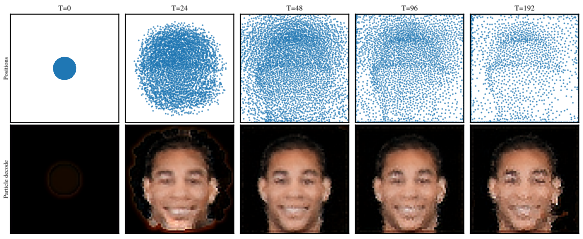

Figure 3: Formation on a held-out face from the headline checkpoint, at rollout steps $T = 0/24/48/96/192$. Top: particle positions. Bottom: the particle render, before the residual U-Net reads it into an image.



Figure 4: Qualitative outputs on five held-out faces.

substitution lowers mean PSNR by 3.48, 3.37, and 2.91 dB across the three particle seeds, on $32/32$, $31/32$, and $32/32$ images. The first measures how much the readout uses the particle pathway, the second whether what it uses is specific to the target; together they establish that the particle state carries image-specific information the jointly trained readout consumes (App. G).

Table 2: Particle-path sensitivity on 32 training images. Top: full-image PSNR as the particle-dependent readout is scaled by λ (seed 42). Bottom: mean PSNR decrease and number of images with increased error after substituting another image's mature particle state.

Setting	Mean full-image PSNR (dB)				
λ	0	0.25	0.5	0.75	1
Particle (42)	15.42	17.55	20.65	26.02	36.78
Donor	Mean PSNR drop (dB)		Worse images / 32		
Particle (42)	3.48		32/32		
Particle (99)	3.37		31/32		
Particle (2024)	2.91		32/32		

Matched comparison and component configurations. The matched comparison isolates the particle branch: both arms enable the conditional head, disable FiLM, and share the training budget, data split, checkpoint rule, and seed 42, differing only in whether particles are present (Table 13). Removing the particles removes the render, so that arm takes the target latent directly and its readout input widens from three channels to eight. With that widening it leads by 1.50 dB on masked PSNR, while the two arms differ by at most 0.0009 in LPIPS. Freezing the particle MLP and training only the readout reaches 35.75 dB: it recovers part of the masked-PSNR gap but performs worse on LPIPS, so the leading arm depends on the metric. The broader grid points the same way, with its strongest dense configuration G reaching 36.38 dB against 34.92 dB for PUN at seed 42 and the output-side head alone accounting for 6.24 dB. G is not the no-particle arm above: it keeps the frozen decoder in its readout path, whereas that arm removes the decoder and feeds the eight-channel latent to its U-Net, so the two scores are not two measurements of one configuration. Together the comparisons locate PUN's gain in the interaction of the conditional head, the readout, and particle self-organization, rather than in the particle path carrying the reconstruction on its own (App. I).

Freezing the particle rule. A warm-start comparison asks whether freezing the rule helps consistently. From a shared pretrained checkpoint and 20K further iterations, the preferred regime reverses with the conditioning mode: freezing leads by 0.82 dB with FiLM and the output head enabled, joint training by 1.36 dB with both disabled. This schedule differs from the headline run, so the levels are not comparable to Table 6 (App. B).

4.3 Recovery from localized damage

To isolate recovery in the particle render, we bypass the residual U-Net and output conditioning head at evaluation time. We clone a mature state at step 192, delete or displace the particles in a radius-8 px region, and let the rollout continue for 192 steps while a time-matched clean branch shares its conditioning and random stream. Three branches of the same trained model differ only in what they update: *mobile* updates positions and states, *fixed-tail* states only, and *hold* neither. The experiment uses 100 training images and models trained with seeds $\{42, 99, 2024\}$, reporting the time-normalized AUC of the excess ROI MSE and the final absolute ROI MSE (App. F).

Mobile yields lower excess-error AUC than fixed-tail under full FiLM conditioning in three training seeds, with all image-paired 95% bootstrap intervals for the mean difference below zero (App. G). The final absolute ROI MSE follows the same pattern (Table 3). At seed 42, mobile updates increase coverage of the damaged region from 25.79% to 99.97% by step 32, while fixed-tail and hold remain at 25.79%. This is consistent with surviving particles moving into the damaged region. Because the clean branch also continues to evolve, excess ROI MSE measures the additional error associated with damage against a time-matched reference. Both mobile and fixed-tail update states, so their comparison tests the contribution of position updates; hold supplies a reference without further updates. Coverage at step 32 describes spatial filling, whereas the step-192 metrics describe the final reconstruction. At seed 42, mobile reaches final ROI MSE 0.0046695, compared with 0.1197542 for fixed-tail. These measurements link early spatial filling to lower final reconstruction error.

The mobile advantage in excess-error AUC persists when FiLM is blocked near the damage or disabled during subsequent updates (App. G).

Table 3: Particle-render recovery on 100 training images at seed 42 with full FiLM conditioning. Results are averaged over deletion and displacement within each image and then over images. The residual U-Net and output conditioning head are bypassed. Metrics are measured after 192 recovery steps.

Branch	Final ROI MSE ↓	Full PSNR (dB) ↑	Final coverage ↑
mobile	**0.0046695**	**24.698**	**99.96%**
fixed-tail	0.1197542	21.035	25.79%
hold	0.1210342	20.559	25.79%

Table 4: Particle-render recovery on the shared 1,000 held-out faces over three seeds, measured on the pre-U-Net render. A score of 1.0 is the clean level. The two lowest random-deletion doses are omitted: they barely move the clean-damaged error and leave the ratio unstable.

Intervention	Dose	Recovery	Damage at $T/2$
state shuffle	all	1.000 ± 0.000	0.117
state zero	50%	0.998 ± 0.001	0.052
state zero	25%	0.996 ± 0.003	0.021
position noise	$\sigma = 0.10$	1.001 ± 0.001	0.030
position noise	$\sigma = 0.05$	1.003 ± 0.002	0.016
position noise	$\sigma = 0.02$	1.001 ± 0.005	0.005
local delete	25%	0.888 ± 0.054	0.047
local delete	50%	0.843 ± 0.038	0.130
random delete	50%	0.072 ± 0.171	0.005
random delete	75%	0.115 ± 0.058	0.011

A separate experiment probes particle-render recovery on the full 1,000-image held-out set across the same three training seeds (App. H). The branch comparison tests whether position updates contribute to recovery, while the held-out experiment examines how recovery varies with the type and extent of damage. Together, they connect the role of particle motion to the response of the representation under different perturbations.

State-level damage is essentially free to repair: state shuffling, state zeroing, and position noise all recover to between 0.996 and 1.003 (Table 4). Contiguous deletion is where the ratio separates from random deletion of the same size. Contiguous deletion yields recovery scores of 0.888 and 0.843 at 25% and 50% deletion, respectively. At the matched 50% dose, random deletion yields only 0.072; increasing random deletion to 75% yields 0.115. A shared conditional rule, applied to images it was never trained on, still repairs local damage to its own representation.

4.4 Coordinate interventions and spatial conditioning

We evaluate the particle render with the residual U-Net and output conditioning head bypassed. After a 192-step rollout, we apply a smooth local displacement to the particle coordinates. During subsequent updates, we retain the original FiLM condition c, spatially warp it by the same mapping, or disable FiLM, with either mobile positions or fixed post-intervention positions. The warped source image serves only as the evaluation target. We evaluate 2- and 4-pixel rightward displacements in upper and lower image regions on the same 100 training images as the recovery probe for seeds $\{42, 99, 2024\}$, at $t \in \{0, 8, 16, 32\}$ after intervention (App. F).

Warping the spatial FiLM condition together with the imposed coordinate transformation improves localization during subsequent updates at every operation point (Table 10).

Fig. 5 illustrates the response under original and warped FiLM conditions. The initial coordinate displacement is the same in both branches; their subsequent updates differ in the spatial condition they sample. Original FiLM retains the target features at their original locations, whereas warped FiLM resamples those features under the imposed mapping. Since each particle samples conditioning at its current position (Sec. 3.3), this comparison connects the localization response to the spatial alignment between the displaced particles and the conditioning field. These results characterize condition-dependent geometric behavior of the particle render on training images (App. G).

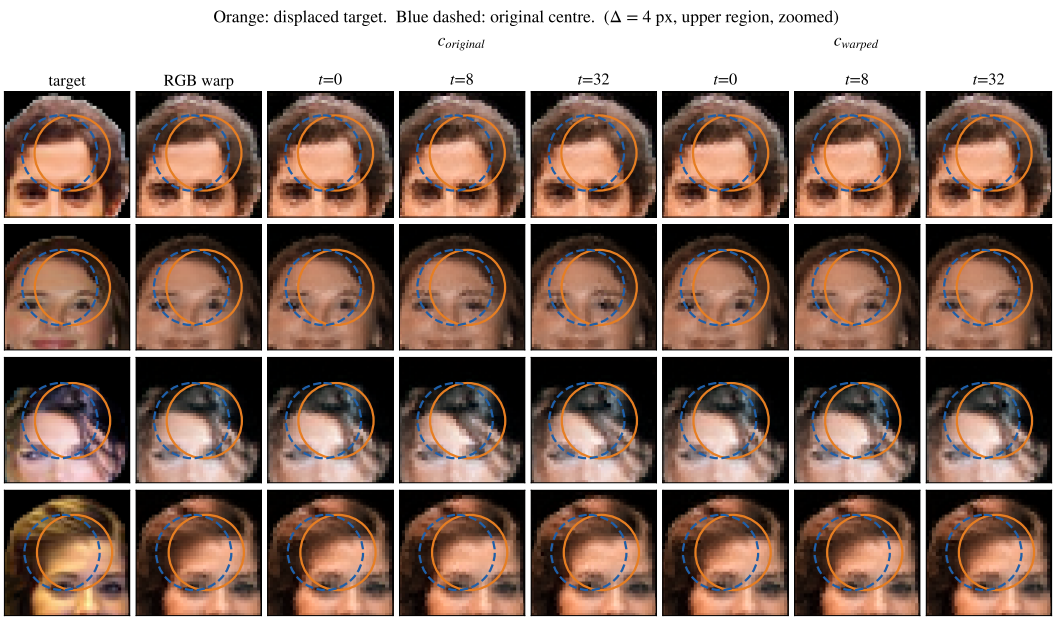

Figure 5: Particle-render responses to a 4-pixel rightward displacement in the upper image region, with the residual U-Net and output conditioning head bypassed. Rows show four examples; columns compare the target, direct RGB warping, and original or warped FiLM conditions at $t = 0, 8, 32$. Orange circles mark displaced target centers; blue dashed circles mark original centers. Images show zoomed regions.

4.5 Directional Guidance

We also ask whether the update rule itself can be steered: we fine-tune the MLP of a pixel-domain NPA toward a directional CLIP objective (Radford et al., 2021). Without stabilization only $6/10$ runs complete; with a state clamp, gradient surgery, and an exponential moving average, all 30 runs across ten directions and three seeds do. The guards address training-time stability; editing quality remains unestablished (App. L). Unlike the preceding probes, which intervene on particle states or coordinates while keeping the learned rule fixed, directional guidance changes the update rule itself. It therefore explores a complementary intervention: adapting the dynamics that produce the image.

5 Conclusion

We introduced PUN, which removes some of NPA's limitations: it handles natural images, generalizes to unseen faces without per-image optimization, and reaches a fidelity pixel-space fitting cannot. Trained once on 4,000 CelebA faces and evaluated on 1,000 disjoint held-out faces across three training seeds, PUN achieves a clear perceptual lead without per-image optimization, with LPIPS roughly an order of magnitude lower than cDDPM.

Beyond reconstruction, the explicit particle representation supports two kinds of intervention. Position updates improve recovery from localized damage. Coordinate interventions reveal that localization during subsequent updates depends on the spatial FiLM condition. A matched particle-branch ablation locates the gain in the FiLM, readout, and particle self-organization interaction, and directional CLIP guidance steers the learned dynamics toward attribute-level edits.

Limitations. Headline results use aligned 64×64 CelebA images; cross-domain and higher-resolution transfer are untested. The component and warm-start training comparisons use a single seed. The CLIP-guided study evaluates training-time stability rather than editing quality.

AI USE STATEMENT

In this work, we used generative AI tools to draft the schematic diagram (Fig. 2), to brainstorm figure layouts, to assist with experiment design, and to support narrative restructuring and edits to the code and text. We reviewed all AI-assisted output and manually verified it against the experimental artifacts (the evaluation details and experimental results in the appendix). No other disclosure categories apply (dataset generation, theory development, or mathematical claims and proofs). We take full responsibility for the final content, including AI-aided text, claims, and artifacts.

ETHICS STATEMENT

CelebA crops are used for non-commercial academic benchmarking under the dataset's terms. The reproducibility package will provide preprocessing instructions without redistributing CelebA images. The frozen autoencoder was trained on 46 computer-generated faces, not on photographs of real individuals (App. M).

REPRODUCIBILITY STATEMENT

The method and evaluation procedure are described in Secs. 3–4. Per-seed scores, component settings, and intervention details appear in the supplementary materials. Complete per-model configurations and an anonymous code/checkpoint package will be released for independent reproduction.

REFERENCES

K. Aliev, A. Sevastopolsky, M. Kolos, D. Ulyanov, and V. Lempitsky. Neural point-based graphics. In *ECCV*, 2020.

O. Bar-Tal, D. Ofri-Amar, R. Fridman, Y. Kasten, and T. Dekel. Text2LIVE: Text-driven layered image and video editing. In *ECCV*, 2022.

Y. Blau and T. Michaeli. The perception-distortion tradeoff. In *CVPR*, 2018.

L. Bösiger, M. Dusmanu, M. Pollefeys, and Z. Bauer. MaRINeR: Enhancing novel views by matching rendered images with nearby references. In *ECCV*, 2024.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, 2018.

G. Couairon, J. Verbeek, H. Schwenk, and M. Cord. DiffEdit: Diffusion-based semantic image editing with mask guidance. In *ICLR*, 2023.

R. Gal, O. Patashnik, H. Maron, A. H. Bermano, G. Chechik, and D. Cohen-Or. StyleGAN-NADA: CLIP-guided domain adaptation of image generators. *ACM Transactions on Graphics*, 41(4): 141:1–141:13, 2022. doi: 10.1145/3528223.3530164.

A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *ICLR*, 2023.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

John Kalkhof, Camila González, and Anirban Mukhopadhyay. Med-NCA: Robust and lightweight segmentation with neural cellular automata. In *IPMI*, pp. 705–716, 2023. doi: 10.1007/978-3-031-34048-2_54.

T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020.

B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani. Imagic: Text-based real image editing with diffusion models. In *CVPR*, 2023.

G. Kim, T. Kwon, and J. C. Ye. DiffusionCLIP: Text-guided diffusion models for robust image manipulation. In *CVPR*, 2022.

J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, 2016.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

G. Kwon and J. C. Ye. CLIPstyler: Image style transfer with a single text condition. In *CVPR*, 2022.

C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017.

X. Leng, J. Singh, Y. Hou, Z. Xing, S. Xie, and L. Zheng. REPA-E: Unlocking VAE for end-to-end tuning of latent diffusion transformers. In *ICCV*, 2025.

B. Lin, Z. Liu, C. Lin, S. Chen, Y. Ge, Y. Lin, J. Zhang, M. Yang, Z. Zhong, L. Bo, and L. Yuan. GEAR: Guided end-to-end autoregression for image synthesis. *arXiv preprint arXiv:2606.32039*, 2026.

B. Liu, X. Liu, X. Jin, P. Stone, and Q. Liu. Conflict-averse gradient descent for multi-task learning. In *NeurIPS*, 2021.

Jiawei Liu, Qiang Wang, Huijie Fan, Yinong Wang, Yandong Tang, and Liangqiong Qu. Residual denoising diffusion models. In *CVPR*, 2024.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

N. Ma, M. Goldstein, M. S. Albergo, N. M. Boffi, E. Vanden-Eijnden, and S. Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *ECCV*, 2024.

R. Mokady, A. Hertz, K. Aberman, Y. Pritch, and D. Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *CVPR*, 2023.

Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, and Michael Levin. Growing neural cellular automata. *Distill*, 2020. doi: 10.23915/distill.00023.

Eyvind Niklasson, Alexander Mordvintsev, Ettore Randazzo, and Michael Levin. Self-organising textures. *Distill*, 2021. doi: 10.23915/distill.00027.003.

S. H. Pahng, G. Guan, B. Fefferman, and S. Hormoz. DiffeoMorph: Learning to morph 3D shapes using differentiable agent-based simulations. *arXiv preprint arXiv:2512.17129*, 2025.

E. Pajouheshgar, H. Kim, S. Süsstrunk, W. Jakob, and J. Park. Neural particle automata: Learning self-organizing particle dynamics. In *ACM SIGGRAPH*, 2026a. doi: 10.1145/3799902.3811052.

E. Pajouheshgar, Y. Xu, A. Abbasi, A. Mordvintsev, W. Jakob, and S. Süsstrunk. Neural cellular automata: From cells to pixels. In *ACM SIGGRAPH*, 2026b. doi: 10.1145/3799902.3811053.

R. B. Palm, M. González-Duque, S. Sudhakaran, and S. Risi. Variational neural cellular automata. In *ICLR*, 2022.

O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski. StyleCLIP: Text-driven manipulation of StyleGAN imagery. In *ICCV*, 2021.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Ettore Randazzo, Alexander Mordvintsev, Eyvind Niklasson, Michael Levin, and Sam Greydanus. Self-classifying MNIST digits. *Distill*, 2020. doi: 10.23915/distill.00027.002.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4): 4713–4726, 2023. doi: 10.1109/TPAMI.2022.3204461.

D. Selikhanovych, D. Li, A. Leonov, N. Gushchin, S. Kushneriuk, A. Filippov, E. Burnaev, I. Koshelev, and A. Korotin. One-step residual shifting diffusion for image super-resolution via distillation. In *ICML*, 2026.

Lingchen Sun, Rongyuan Wu, Jie Liang, Zhengqiang Zhang, Hongwei Yong, and Lei Zhang. Improving the stability and efficiency of diffusion models for content consistent super-resolution. *IEEE Transactions on Image Processing*, 34:8421–8434, 2025. doi: 10.1109/TIP.2025.3640863.

J. Thies, M. Zollhöfer, and M. Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics*, 38(4), 2019. doi: 10.1145/3306346.3323035.

C. Wang, X. Wu, Y. Guo, S. Zhang, Y. Tai, and S. Hu. NeRF-SR: High-quality neural radiance fields using supersampling. In *ACM MM*, 2022.

C. Wang, L. Guo, Z. Fu, S. Yang, H. Cheng, A. C. Kot, and B. Wen. Reconciling stochastic and deterministic strategies for zero-shot image restoration using diffusion model in dual. In *CVPR*, 2025.

Y. Wang, W. Yang, X. Chen, Y. Wang, L. Guo, L. Chau, Z. Liu, Y. Qiao, A. C. Kot, and B. Wen. SinSR: Diffusion-based image super-resolution in a single step. In *CVPR*, 2024.

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. doi: 10.1109/TIP.2003.819861.

T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang. PhysGaussian: Physics-integrated 3D gaussians for generative dynamics. In *CVPR*, 2024.

T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn. Gradient surgery for multi-task learning. In *NeurIPS*, 2020.

S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, and M. Yang. Restormer: Efficient transformer for high-resolution image restoration. In *CVPR*, 2022.

H. Zhang, Y. N. Dauphin, and T. Ma. Fixup initialization: Residual learning without normalization. In *ICLR*, 2019.

K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

K. Zhou, W. Li, Y. Wang, T. Hu, N. Jiang, X. Han, and J. Lu. NeRFLix: High-quality neural view synthesis by learning a degradation-driven inter-viewpoint mixer. In *CVPR*, 2023.

SUPPLEMENTARY MATERIAL

This appendix provides evaluation details, statistical analyses, and additional results for the reconstruction and intervention experiments. It also describes the training configuration, autoencoder ablations, and directional CLIP objective.

A SCOPE OF THE EXPERIMENTAL EVIDENCE

We distinguish comparisons on held-out images from exploratory interventions on training images and report uncertainty for the paired reconstruction comparisons.

Cross-metric consistency. In the three-seed primary comparison, the LPIPS gap between PUN (0.0112 ± 0.0003 face, 0.0100 ± 0.0002 full) and cDDPM (0.1311 ± 0.0474 face, 0.1380 ± 0.0524 full) is roughly $12\times$ on face and $14\times$ on full, while the masked-PSNR difference is 14.82 dB. Across the rows of Table 6 the face and full LPIPS columns agree on the qualitative ranking. This ordering holds across a large fidelity separation; the metric disagreement in Table 5 concerns the closer component configurations rather than the matched no-particle arm.

Paired bootstrap over the shared test images. Every model in Table 6, Table 12, and the matched comparison of Sec. 4.2 is evaluated on the $1{,}000$ held-out images, which makes a paired comparison valid. We resample those images with replacement $2{,}000$ times and report the 95% percentile interval of each mean difference against the matched particle arm of Sec. 4.2. An interval that excludes zero indicates a statistically significant difference.

Table 5: Paired bootstrap against the matched particle arm of Sec. 4.2 on the shared $1{,}000$ images, $2{,}000$ resamples. A positive masked-PSNR or SSIM difference favours the comparison arm, and a negative LPIPS difference does.

Comparison arm	Δ masked PSNR (dB)	Δ SSIM	Δ LPIPS (face)
matched component comparisons of Sec. 4.2			
No particles	$+1.505$ $[+1.405, +1.600]$	$+0.0396$ $[+0.0385, +0.0407]$	-0.0009 $[-0.0013, -0.0004]$
Particles frozen	$+0.583$ $[+0.534, +0.633]$	$+0.0319$ $[+0.0311, +0.0328]$	$+0.0010$ $[+0.0008, +0.0012]$
component configurations of Table 12			
G	$+1.209$ $[+1.130, +1.285]$	$+0.0350$ $[+0.0340, +0.0361]$	-0.0005 $[-0.0010, -0.0001]$
E	$+0.480$ $[+0.435, +0.528]$	$+0.0273$ $[+0.0266, +0.0280]$	$+0.0011$ $[+0.0009, +0.0014]$
F	$+0.514$ $[+0.464, +0.563]$	$+0.0292$ $[+0.0283, +0.0301]$	$+0.0011$ $[+0.0009, +0.0013]$
H	-0.751 $[-0.826, -0.681]$	$+0.0113$ $[+0.0108, +0.0118]$	-0.0003 $[-0.0007, +0.0001]$
J	$+0.099$ $[+0.045, +0.158]$	$+0.0255$ $[+0.0245, +0.0264]$	$+0.0030$ $[+0.0027, +0.0033]$

The intervals separate the two families. The two no-particle entries come from different models: *no particles* is the matched arm of Sec. 4.2, whose readout consumes the eight-channel latent directly, whereas G is the component row of Table 12, whose readout refines a decoded image through a three-channel U-Net. The two matched arms behave differently: *no particles* leads on all three metrics, though its LPIPS margin is only 0.0009, while *particles frozen* leads on masked PSNR and SSIM but trails on LPIPS. Against the matched particle arm, configurations E, F, and J have a masked-PSNR advantage and a significant face-cropped LPIPS disadvantage, so the two metrics rank those arms on opposite sides of the particle arm. Against H the masked-PSNR difference reverses sign while the SSIM difference keeps the control ahead. The one interval that includes zero is H on LPIPS.

Joint and frozen training. The warm-start comparisons (Sec. 4.2) are single-seed, and both modes are evaluated on the same $1{,}000$-image CelebA test split as the main result, but under a

Table 6: Reconstruction on 1,000 held-out CelebA faces. PUN and cDDPM average three training seeds $\{42, 99, 2024\}$ (population SD); the remaining rows use seed 42. cDDPM is conditioned on the target AE latent and DDPM (low-res.) on a bicubic low-resolution observation. LPIPS uses the AlexNet backbone, cropped to the foreground bounding box ("face") or to the whole output ("full"). VNCA follows Palm et al. (2022). The main text reports this table without the full-image PSNR and SSIM columns.

Model	Masked PSNR (dB)	Full PSNR (dB)	Full SSIM	LPIPS (face)	LPIPS (full)
PUN	$\mathbf{35.03 \pm 0.08}$	**36.30**	**0.9610**	$\mathbf{0.0112 \pm 0.0003}$	$\mathbf{0.0100 \pm 0.0002}$
cDDPM	20.21 ± 2.31	18.39	0.5313	0.1311 ± 0.0474	0.1380 ± 0.0524
VAE (β-anneal)	24.76	17.57	0.5741	0.1085	0.1281
SRGAN	21.59	22.99	0.5414	0.0801	0.0812
RDDM	19.00	21.51	0.4521	0.1504	0.1561
DDPM (low-res.)	17.42	20.15	0.5066	0.1694	0.1751
SiT	19.93	22.90	0.7811	0.0474	0.0450
VNCA	17.11	20.15	0.6463	0.2025	0.1992
Original NPA	12.13	14.39	0.2284	0.5536	0.5914

warm-start regime rather than the main-run schedule, so their level is not directly comparable to the Table 6 rows.

cDDPM seed dispersion. The cDDPM masked PSNR is 20.21 ± 2.31 dB (mean and population standard deviation). The individual scores are 17.99, 23.39, and 19.26 dB; all three are included in the aggregate.

B Evaluation details by table

The reconstruction, component, and training-regime comparisons all use the same 1,000 held-out images.

Main result (Table 6). We partition 5,000 CelebA images into 4,000 training and 1,000 test images using seed 42. Foreground-masked PSNR uses $\alpha > 0.05$. PUN and cDDPM are trained with seeds $\{42, 99, 2024\}$; the remaining baselines use seed 42. The dispersions are the population SDs of the reported per-seed values. Computed from the rounded scores they are 0.0826 and 2.3053 dB, reported as 0.08 and 2.31 dB.

Baseline parameter and training-budget alignment. Dense baselines in Table 6 are trained with parameter counts within 5% of 1.75 M and a training budget of 30,000 iterations, matching the residual U-Net size and the headline training schedule of PUN. cDDPM satisfies this budget and is averaged over three seeds $\{42, 99, 2024\}$; the remaining baselines are evaluated at seed 42.

Component ablation (Table 12). All variants use the same CelebA split, seed 42, and 30,000 training iterations. Configurations E, F, and G enable the same components (P, U, g) but differ in their readout inputs (App. I); their masked-PSNR range is 0.73 dB.

Frozen-vs-joint 2×2 (Sec. 4.2). All four configurations use the same CelebA split and seed 42. Training begins from a shared pretrained particle model and continues for 20K iterations. The frozen arm updates only the readout, while the joint arm also updates particle weights. Mode A enables FiLM and the output head, and mode B disables both. Within mode A, freezing reaches 35.52 dB against 34.70 dB for joint training, a lead of 0.82 dB; within mode B the direction reverses and joint training leads by 1.36 dB (26.40 against 25.04 dB).

CLIP-guided editing (Table 14). We evaluate ten editing directions using the pixel-domain particle representation. The stabilized configuration uses seeds $\{42, 99, 2024\}$; the unstabilized comparison uses a single seed.

C Dataset configuration

The particle reconstruction model uses a fixed split of 4,000 training and 1,000 held-out CelebA images. The frozen autoencoder is trained separately on 46 computer-generated faces. Its training data and reproducibility materials are described in App. M.

D Training configuration and computational cost

The main model is trained for 30,000 iterations with a batch size of 4 using AdamW (Loshchilov & Hutter, 2019) and an initial learning rate of 5×10^{-4}. The learning rate is multiplied by 0.3 at 12K, 21K, and 27K iterations. The joint loss is defined in Eq. 8. The model has 1,800,947 trainable parameters: 47,760 in the particle network and 1,753,187 in the residual U-Net.

Single-image inference with $T = 192$ takes 635.0 ms on a V100 GPU, corresponding to 1.57 images per second and 78.8 GFLOPs. Across 20 measurements, the median is 635.1 ms and the 90th percentile is 639.5 ms. Reported encoding, rollout, and rendering times are 0.49, 632.19, and 3.92 ms, respectively.

E Training dynamics

Fig. 6 connects optimization progress to held-out reconstruction and illustrates the rollout underlying the main results.

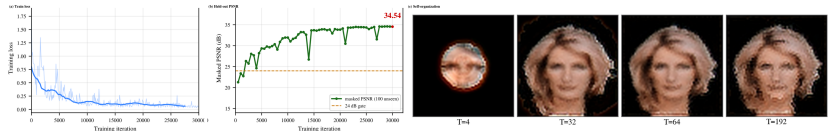

Figure 6: Training dynamics on the 5,000-face CelebA setting for the FiLM-conditioned pipeline. *(a)* Per-100-iteration training loss. *(b)* Masked PSNR on held-out faces at fixed checkpoints. *(c)* Inference-time self-organization on one held-out face at rollout steps $T = 4/32/64/192$.

F Particle intervention details

Images and trained models. Particle-path sensitivity is evaluated on 32 training images using three particle models trained with seeds $\{42, 99, 2024\}$ and a dense control trained with seed 42. Localized-damage recovery is evaluated on 100 training images with the three particle models and one perturbation repeat. The coordinate-intervention probe uses the same 100 training images. For both probes, the residual U-Net and output conditioning head are bypassed at evaluation time; the frozen decoder and particle-update weights are unchanged. The held-out particle-render recovery experiment is described separately in App. H.

For the coordinate-intervention probe, the three conditions are $c = \mathrm{Enc}(x)$, $W_F(c)$, and disabled FiLM, where W_F denotes spatial resampling under the imposed mapping. The transformed target $W_F(x)$ is used only for evaluation and is not re-encoded as model input.

Statistical analysis. For localized-damage recovery, deletion and displacement results are averaged within each image for the primary comparison. Image-paired 95% bootstrap intervals use 2,000 resamples; condition-blocked and condition-disabled comparisons are reported separately for each damage type. For coordinate interventions, descriptive tables report the median over images within each seed and then the median across seeds, separately for each region, displacement, condition, update mode, and time. Paired comparisons use per-image differences and 2,000 bootstrap resamples with seed 20260916 to estimate 95% percentile intervals for the mean difference.

Outputs and measurements. Particle-path sensitivity is measured by full-image PSNR of the final output; numerical consistency is assessed for both the particle latent and the final image. Localized-damage recovery measures excess error as the damaged branch's ROI MSE relative to the target minus that of a time-matched clean branch with the same update mode, conditioning, and

random stream. Both branches are evaluated on $\mathrm{clamp}(\mathrm{Dec}(\boldsymbol{z}), 0, 1)$ within the intersection of the damage region and the foreground mask. The normalized AUC is the time average of $\delta(t)$ over the 192-step recovery window. The recovery criterion requires the excess error to remain below 10% of its initial value for three consecutive sampled times. Coordinate interventions are evaluated on $\mathrm{clamp}(\mathrm{Dec}(\boldsymbol{z}), 0, 1)$, excluding both residual U-Net refinement and output-head injection. We report NCC-based localization relative to the transformed target, with reconstruction metrics as secondary diagnostics. After a 192-step rollout and an externally imposed coordinate transformation, we evaluate at $t = 0, 8, 16, 32$ additional update steps. Mobile updates positions and states; fixed retains the transformed positions and updates states only. Static forward–inverse cycles contain no intervening updates. Outputs are clamped to $[0, 1]$. The localized-damage and coordinate probes both bypass the residual U-Net and output conditioning head. The separate held-out particle-render experiment also excludes the residual U-Net, but uses a different image set, intervention protocol, and evaluation schedule.

Spatial measurements. ROI coverage is the fraction of pixels in the damaged region whose splatted density exceeds 0.01. Reconstruction errors are measured on the intersection with the foreground mask. For coordinate editing, $w(\boldsymbol{q}) = \max(0, 1-\|\boldsymbol{q}-\boldsymbol{q}_0\|^2/R^2)^3$, with center $\boldsymbol{q}_0$ and radius $R = 12$ pixels. NCC locates a 7×7 source patch at integer positions within ± 8 pixels of its original center; localization error is the distance from the detected center to the requested target center.

Runtime measurements. Inference from the initial particle state is timed three times per image after computational warmup, and the median across images is reported. Recovery time includes only updates to the damaged branch, excluding the initial rollout, clean reference branch, and readout. For coordinate interventions, the reported cumulative update time includes only updates to the transformed particle branch. It excludes the initial 192-step rollout, the clean reference branch, rendering, and metric evaluation. RGB-warp time measures only image resampling.

G Additional particle intervention results

The following results use the interfaces in Secs. 4.2–4.4; App. F specifies the samples and measurements. Table 2 reports path sensitivity in the main text; Tables 7–11 provide the additional comparisons.

Numerical consistency and particle-path sensitivity. We assess numerical consistency on 32 training images across the four trained models. With absolute and relative tolerances of 10^{-5} and 10^{-4}, the largest final-image deviation is 5.365×10^{-7} and the largest particle-latent deviation is 1.907×10^{-5}; both satisfy the combined tolerance criterion. Scaling the particle-dependent path increases full-image PSNR monotonically, and substituting another image's particle state lowers it across all three training seeds (Table 2). These results establish that the trained readout uses the particle pathway. In contrast, the retrained dense control is more accurate and faster on the same images (Table 7). These measurements concern training images, and the reported latency is specific to the evaluated implementation and hardware.

Table 7: Reconstruction accuracy and inference time on the 32 training images used for the particle-path analysis. Parentheses identify training seeds. Inference starts from the initial particle state; time is the median across images, while PSNR and MSE are image averages.

Model	Full PSNR (dB) ↑	Full MSE ↓	Inference time (ms) ↓
Particle (42)	36.779	2.381×10^{-4}	721.6
Particle (99)	37.241	2.183×10^{-4}	743.8
Particle (2024)	37.335	2.134×10^{-4}	743.4
Dense (42)	**40.975**	$\mathbf{9.510 \times 10^{-5}}$	**2.71**

Recovery under preserved conditioning. With full FiLM conditioning, mobile updates lower the excess-error AUC relative to fixed-tail in all three training seeds (Table 9). Final absolute ROI MSE and full-image PSNR also improve over fixed-tail and hold (Table 8). Full-image SSIM is lower than fixed-tail for seeds 42 and 99, but higher for seed 2024. When FiLM is blocked near

the damage or disabled during subsequent updates, the excess-error gap can shrink while both clean and damaged reconstructions deteriorate. These relative recovery measures therefore need to be interpreted alongside absolute reconstruction quality.

Variation across training seeds. The mean mobile-minus-fixed-tail AUC differences under full conditioning are -0.112899, -0.113342, and -0.115094 for seeds 42, 99, and 2024, respectively, with negative differences on all 100 images per seed. With FiLM blocked near the damage, the differences range from -0.050521 to -0.018217 across the two damage types and three seeds; with FiLM disabled, they range from -0.050460 to -0.018745. All corresponding image-paired 95% bootstrap intervals lie below zero.

Table 8: Particle-render recovery on 100 training images with full FiLM conditioning and the residual U-Net and output conditioning head bypassed. Each row reports the mean over deletion and displacement within each image and then over images, for one branch under one training seed, after 192 recovery steps.

Seed	Branch	ROI MSE ↓	Full PSNR (dB) ↑	Full SSIM ↑	Coverage ↑
42	mobile	0.0046695	24.698	0.789227	99.96%
42	fixed	0.1197542	21.035	0.807952	25.79%
42	hold	0.1210342	20.559	0.771340	25.79%
99	mobile	0.0045600	25.338	0.804556	99.51%
99	fixed	0.1198709	21.258	0.810709	25.81%
99	hold	0.1213614	20.800	0.791183	25.81%
2024	mobile	0.0026485	26.990	0.868999	100.00%
2024	fixed	0.1207379	21.424	0.839901	27.57%
2024	hold	0.1185207	21.149	0.812286	27.57%

Table 9: Particle-render recovery across three training seeds on 100 training images under full FiLM conditioning. Each row reports the mean excess-error AUC for the mobile and fixed-tail branches, the mean mobile-minus-fixed-tail difference, and the image-paired 95% bootstrap interval (2,000 resamples) for the mean difference. The mobile-minus-fixed-tail difference is negative on all 100 images per seed.

Seed	Mobile AUC	Fixed-tail AUC	Difference	95% CI
42	0.004063	0.116962	-0.112899	$[-0.121208, -0.104455]$
99	0.004137	0.117479	-0.113342	$[-0.121607, -0.104818]$
2024	0.003302	0.118396	-0.115094	$[-0.123593, -0.106915]$

Coordinate and conditioning transformations. We evaluate rightward displacements of 2 and 4 px in upper and lower image regions, using a compact-support displacement field of radius 12 px. These regions are not semantic landmark annotations. Table 10 reports localization at $t = 32$ for both update modes and the original and warped FiLM conditions.

Warped FiLM conditions yield lower localization error than original conditions for all 72 nonzero-time comparisons, with image-paired 95% bootstrap intervals for the mean difference below zero. Both mobile and fixed-position updates show this dependence. Relative to direct RGB warping, 65 of 96 intervals include zero across all four recorded times (47 of 72 at nonzero times).

Without intervening updates, the forward–inverse coordinate cycle yields a mean particle-render MSE of 3.6203×10^{-13} and a maximum coordinate deviation of 7.3016×10^{-7} world units across all 1,200 image–seed–operation records. The corresponding mean RGB-warp cycle MSE is 3.0498×10^{-5}.

H Recovery of the pre-readout particle render

Table 4 complements Sec. 4.3. Damage at $T/2$ is the induced masked MSE at intervention time.

Table 10: NCC localization error (pixels) on 100 training images with the residual U-Net and output conditioning head bypassed. Entries are cross-seed medians of per-seed image medians. Particle branches are measured at $t = 32$; RGB warping is measured at $t = 0$. Upper/lower indicate regions, and all displacements are rightward.

Setting	Upper/2 px	Upper/4 px	Lower/2 px	Lower/4 px
Mobile, warped FiLM	0.564	1.070	0.486	1.107
Fixed, warped FiLM	0.481	0.918	0.438	0.966
Mobile, original FiLM	1.934	3.896	2.120	4.108
Fixed, original FiLM	1.923	3.874	2.112	4.102
RGB warp	0.502	0.949	0.464	1.090

Table 11: Static forward–inverse cycles with the U-Net and output head bypassed. Cycle MSE is the mean over 100 images, three seeds, and four operations (1,200 records); coordinate deviation is the maximum over all records. No dynamics updates occur during the cycle. RGB warping resamples the same particle render.

Cycle	Mean cycle MSE	Maximum coordinate deviation (world units)
Particle coordinates	3.6203×10^{-13}	7.3016×10^{-7}
Particle RGB warp	3.0498×10^{-5}	n/a

Two further random-deletion doses (10% and 25%) are excluded. They damage the readout by only 6×10^{-4} and 1.8×10^{-3} masked MSE, so the denominator of the recovery ratio approaches zero and the per-image scores become unstable. We therefore treat both as below the noise floor.

I Component-ablation details

Component configurations. Table 12 compares the particle, residual U-Net, and conditional-head components. E uses the initial-state render $I_0 + U(I_0) + g(c)$; F uses $U(0) + g(c)$; G uses $D + U(D) + g(c)$ with $D = \mathrm{Dec}(c)$. Thus, E omits particle evolution, F uses a zero input, and G directly decodes the target latent.

Table 12: Masked PSNR, full-image PSNR, full-image SSIM, and LPIPS for each component configuration on the shared 1,000 held-out CelebA faces (seed 42). P, U, and g denote the particle path, the residual U-Net, and the output-side conditional head. For E, the particle switch denotes evolution: its readout uses the unevolved initial-state render. G is not the no-particle arm of Sec. 4.2: it retains the frozen decoder in the readout path, whereas that arm removes the decoder and widens the readout input to eight channels.

Variant	P	U	g	Masked PSNR (dB)	Full PSNR (dB)	Full SSIM	LPIPS (face)	LPIPS (full)
B	✓	✓	×	28.68	30.93	0.8636	0.0206	0.0180
C	✓	×	✓	26.08	27.67	0.8680	0.0389	0.0357
D	✓	×	×	26.20	28.55	0.8857	0.0383	0.0365
E	×	✓	✓	35.65	37.28	0.9752	0.0122	0.0109
F	×	✓	✓	35.68	36.92	0.9770	0.0121	0.0110
G	×	✓	✓	**36.38**	**38.62**	**0.9829**	**0.0105**	**0.0094**
H	×	✓	×	34.42	36.85	0.9592	0.0107	0.0096
J	×	×	✓	35.27	36.47	0.9733	0.0140	0.0125

Configuration comparisons. The strongest dense configuration G reaches 36.38 dB, compared with 34.92 dB for PUN at seed 42. The head-only configuration J reaches 35.27 dB, while H removes the head from G and reaches 34.42 dB. On the particle path, the converged particle-plus-U-Net configuration B, which disables the head, reaches 28.68 dB, above the U-Net-free rows C and D. The head thus contributes 6.24 dB to the full pipeline, but does not improve the U-Net-free particle

path (C versus D, 0.12 dB in favour of the head-free row). The matched comparison in Sec. 4.2 controls the conditioning and training settings separately (Table 13).

Table 13: Matched particle-branch comparison on the shared 1,000 held-out faces (seed 42). The particles-on and no-particles arms both enable the output head and disable FiLM, and their readout input differs by necessity: three rendered channels with particles, eight latent channels without. The particles-frozen arm instead freezes the particle MLP and trains only the readout. Paired intervals for the first two arms are in Table 5.

Arm	Masked PSNR (dB)	Full PSNR (dB)	Full SSIM	LPIPS (face)	LPIPS (full)
Particles on	35.17	36.39	0.9478	0.0110	0.0098
No particles	**36.67**	**38.81**	**0.9874**	**0.0102**	**0.0092**
Particles frozen	35.75	37.09	0.9798	0.0121	0.0109

J SINGLE-TARGET-FACE SETTING

This setting contains 21 computer-generated images of one target face, produced using the same procedure as the autoencoder training data. We use 17 images for training and four for validation, with held-out indices $\{0, 5, 10, 15\}$. The primary metric is masked PSNR on these four images. A separate single-image fitting experiment uses the same image for training and evaluation to assess memorization. This setup differs from the CelebA reconstruction evaluation, and its scores are not pooled with the CelebA results.

K AUTOENCODER ABLATION DETAILS

The autoencoder ablation varies channel count $(4, 8, 16)$, kernel size (3×3 vs. 5×5), and depth (2 vs. 3), using the 46-face training set and seed 42, reporting $\mathrm{Dec} \circ \mathrm{Enc}$ reconstruction on 1,000 held-out test faces. These test images were not used to select the reference architecture. The reconstruction scores vary little with channel count (30.33 / 30.30 / 30.05 dB for 4 / 8 / 16 channels), while a 5×5 kernel and an extra layer yield lower reconstruction scores (26.26 / 27.80 dB). The reference architecture uses 8 channels, 3×3 kernels, and depth 2.

L CLIP-GUIDED EDITING: OBJECTIVE, STABILIZATION MECHANISMS, AND RESULTS

Following Gal et al. (2022), we define the directional CLIP loss as

$$\mathcal{L}_{\mathrm{dir}} = 1 - \cos\left(\Delta \boldsymbol{R},\ \Delta \boldsymbol{T}\right), \tag{9}$$

where $\Delta \boldsymbol{R} = E_I(\boldsymbol{I}_{\mathrm{edit}}) - E_I(\boldsymbol{I}_{\mathrm{base}})$ and $\Delta \boldsymbol{T} = E_T(T^+) - E_T(T^-)$ are the image and text embedding differences of a CLIP model (Radford et al., 2021). We fine-tune the particle update rule using this objective and an absolute anchor term to preserve identity. Initial states are produced by rolling out a frozen pixel-domain NPA checkpoint (trained on a single photograph) on the base image. The same initial states are then rolled forward by a trainable MLP and, in parallel, by a frozen baseline copy, and the directional CLIP loss between the two paired renders is back-propagated to the trainable weights.

We diagnose three failure modes of the unstabilized pipeline. (i) *Failure to improve directional alignment*: the directional cosine does not increase, while the regularizer grows to 10^8–10^{13}. (ii) *Late-stage optimization collapse*: the directional cosine exceeds 0.2 and then abruptly declines. (iii) *High inter-seed variance*: identical configurations produce directional cosine scores in $[0.08, 0.45]$ across seeds. We attribute this instability to unbounded state updates, large back-propagated CLIP gradients, stochastic perturbations to particle states, and positive feedback. We use three stabilization mechanisms. **State clamping**: the update is passed through $\tau \tanh(\Delta \boldsymbol{s}/\tau)$ for clipping scale $\tau > 0$, which approximates the identity for small updates and saturates large ones, bounding each state increment. **Gradient surgery** (PCGrad (Yu et al., 2020)): when the CLIP gradient makes an obtuse angle with the anchor/regularizer gradients, its component along the latter is projected away, preventing the mutual cancellation between objectives. **Exponential moving average** (Karras et al., 2020): inference uses an exponential moving average of the training weights to reduce fluctuations across optimization steps.

With all three mechanisms, optimization completes in all 30 runs across ten directions and three seeds, compared with $6/10$ unstabilized runs (Table 14). Fig. 7 shows outputs for seed 42. Completion measures training stability, while directional cosine measures alignment with the requested edit; neither alone establishes editing quality.

Table 14: CLIP-guided editing on ten directions (pixel-domain particle representation). "Without stabilization" is single-seed; the stabilized columns report mean $\pm$ sample standard deviation over three seeds and converged runs per direction. Parenthesized step counts mark divergence.

Direction	Without stabilization	Seed 42	Seed 99	Seed 2024	mean $\pm$ std	Converged runs
boy	fail (762)	0.328	0.341	0.319	0.329 ± 0.011	3/3
beard	fail (2122)	0.351	0.311	0.351	0.338 ± 0.023	3/3
bald	fail (1338)	0.279	0.283	0.135	0.232 ± 0.084	3/3
gold	fail (2541)	0.503	0.657	0.396	0.519 ± 0.131	3/3
dark skin	0.453	0.533	0.521	0.347	0.467 ± 0.104	3/3
younger woman	0.456	0.553	0.498	0.347	0.466 ± 0.107	3/3
smile	0.409	0.456	0.333	0.325	0.371 ± 0.074	3/3
stained glass	0.420	0.579	0.554	0.450	0.528 ± 0.068	3/3
black hair	0.159	0.458	0.459	0.337	0.418 ± 0.071	3/3
older	0.375	0.423	0.445	0.249	0.373 ± 0.107	3/3
Convergence rate	6/10	10/10	10/10	10/10	–	**10/10**



Figure 7: CLIP-guided editing of the pixel-domain particle update rule. Left: the unedited base render. Remaining panels show individual editing directions with stabilized training at seed 42. The directional cosine scores correspond to the seed-42 column of Table 14.

M Data availability and ethical considerations

Aligned CelebA crops are used under the dataset's terms solely for non-commercial academic reconstruction benchmarking. We will provide preprocessing instructions; users must obtain CelebA images from the original source under its terms, as our package will not redistribute them. The frozen autoencoder is trained on 46 computer-generated faces rather than photographs of real individuals; the source images came from thispersondoesnotexist.com.

To support reproducibility, we will release the implementation, trained models, evaluation results, software dependencies, and random seeds upon publication.