DYNAMICS HESSIAN PENALTY FOR DISENTANGLED REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Disentangled representation learning seeks to represent distinct factors of variation in separate latent dimensions. Some methods encourage disentanglement by restricting interactions between latent dimensions during generation. When imposed on the complete generative mapping, these restrictions can favor additive structure and limit the nonlinear composition of latent effects in observation space. We introduce the Dynamics Hessian Penalty (DHP) for iterative generators, which applies this interaction prior to the individual conditional predictions, penalizing their mixed second-order derivatives with respect to the latent representation. Our analysis shows that the complete generative mapping need not collapse to an additive form, even when all such local mixed derivatives vanish. We implement DHP using stochastic finite differences and apply it to diffusion and rectified-flow models, without unrolling the sampler during training or adding inference cost. Experiments show strong disentanglement on synthetic benchmarks and improvements in both disentanglement and generation quality on real-world images. Qualitative results further illustrate controllable generation through latent manipulation.

1 INTRODUCTION

Observed data are generated by multiple underlying factors of variation, such as object identity, position, scale, lighting or color. Disentangled representation learning aims to align these factors with distinct dimensions of a learned latent representation, making individual factors easier to identify, control, and recombine (Bengio et al., 2013; Higgins et al., 2017; Kim & Mnih, 2018). In fully unsupervised settings, however, observations alone do not uniquely identify such an alignment; successful methods must therefore rely on explicit inductive biases (Locatello et al., 2019).

One class of inductive biases promotes disentanglement by limiting interactions between latent dimensions in a generative response. The Hessian Penalty (HP) penalizes mixed second-order derivatives of the selected response (Peebles et al., 2020). Each mixed derivative measures how the sensitivity to one latent dimension changes with another. However, when all such derivatives vanish throughout the latent domain, the response decomposes into a sum of functions of individual latent dimensions. HP encourages additive structure through a soft penalty, while additive decoders explicitly enforce a sum of functions of disjoint latent groups (Lachapelle et al., 2023).

When imposed on the complete generative mapping, this preference for additivity can restrict how latent effects combine in observation space. Yet separately controllable factors of variation need not combine additively. For example, an object's shape and color can be manipulated separately, while the pixelwise effect of changing its color depends on the region occupied by its current shape (Figure 1A). Suppressing this interaction can hinder nonlinear composition of latent effects and limit the model's expressive capacity. We therefore seek to encourage disentanglement while allowing latent effects to compose nonlinearly in the complete generative mapping.

Iterative generators, such as diffusion models, construct a sample through successive updates. When generation is conditioned on a learned latent representation, each prediction uses both this representation and the current noisy sample x_t. This suggests discouraging interactions between latent dimensions within each prediction, while allowing their effects to depend on x_t. During sampling, these effects are carried forward through the updated sample and can influence subsequent predictions, allowing nonlinear composition across updates.

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.

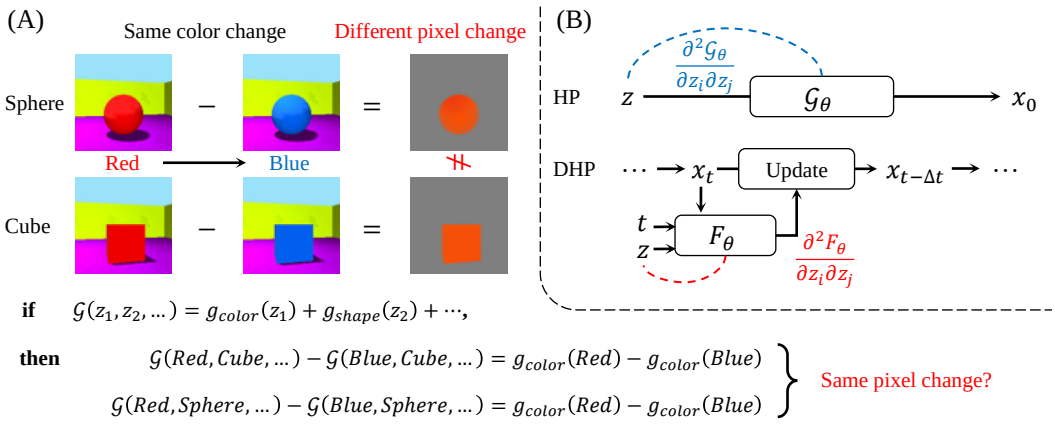


Figure 1: (A) The same color change produces shape-dependent pixelwise differences, incompatible with additivity in shape and color. (B) HP on the complete generative mapping; DHP on local conditional predictions, allowing nonlinear composition through successive updates.

Motivated by this observation, we introduce the *Dynamics Hessian Penalty* (DHP), which penalizes mixed latent interactions within each conditional prediction (Figure 1B). Our analysis shows that the complete generative mapping need not collapse to an additive form, even when all local mixed latent derivatives vanish.

We implement DHP using stochastic finite-difference residuals, with forward and central variants that integrate into standard diffusion and rectified-flow training. Neither variant requires factor annotations or sampler unrolling during training.

Compared with the baselines, both DHP variants achieve higher mean DCI disentanglement scores on Shapes3D, MPI3D-toy, and Cars3D, as well as higher mean Total AUROC Difference (TAD) and lower mean Fréchet Inception Distance (FID) on CelebA. Experiments with rectified flow also show gains over the corresponding baseline. Qualitative results illustrate controllable generation through latent traversal and swapping.

Our contributions are:

- We introduce DHP and show that the complete generative mapping can remain non-additive even when all local mixed latent derivatives vanish.
- We provide stochastic finite-difference implementations compatible with standard diffusion and rectified-flow training, without sampler unrolling.
- We demonstrate gains in disentanglement and generation quality, with further evaluations of downstream factor prediction and latent control.

2 Related Work

Early approaches promote unsupervised disentanglement by strengthening the variational bottleneck, penalizing statistical dependence between latent variables, or maximizing mutual information between latent codes and generated observations (Higgins et al., 2017; Kim & Mnih, 2018; Chen et al., 2018; 2016). Recovering the underlying factors from observations alone generally requires additional assumptions or inductive biases (Locatello et al., 2019). Recent approaches also explore latent quantization (Hsu et al., 2023), geometric regularization (Uscidda et al., 2025), and hierarchical probabilistic priors (Patil et al., 2026). Our work promotes disentanglement by constraining how latent dimensions jointly influence generation.

2.1 Interaction-Based Disentanglement Priors

The Hessian Penalty (HP) penalizes mixed second-order derivatives of generator outputs or selected hidden activations with respect to latent inputs (Peebles et al., 2020). Tripod incorporates a scale-

normalized version into an autoencoding framework (Hsu et al., 2024). Additive decoders parameterize the generative mapping as a sum of functions of disjoint latent groups (Lachapelle et al., 2023). Related first-order methods encourage sparse or orthogonal Jacobian effects (Rhodes & Lee, 2021; Wei et al., 2021). Timilsina et al. (2026) establish conditions for content–style identifiability in multi-domain settings using orthogonal generator Jacobian subspaces.

Brady et al. (2025) establish conditions for disentanglement and compositional generalization by requiring interactions across latent groups to be less complex than those within groups. Matthes et al. (2026) connect structural constraints on generative mechanisms, including mixed-derivative conditions, to latent-subspace identifiability. In a complementary study, Liang et al. (2025) show that disentangled inputs alone do not ensure compositional generalization, highlighting the role of the learned generative mapping. Our approach regularizes mixed latent interactions within individual conditional predictions in iterative generators.

2.2 Disentangled Representations in Iterative Generative Models

Diffusion Autoencoders jointly learn a representation encoder and a conditional diffusion decoder (Preechakul et al., 2022), while PDAE uses a pretrained diffusion model for representation learning (Zhang et al., 2022). To promote disentanglement, InfoDiffusion encourages informative latent representations through mutual-information regularization (Wang et al., 2023). DisDiff decomposes a pretrained diffusion model's gradient field into factor-conditioned subfields (Yang et al., 2023), while FDAE uses content–mask factorization with a conditional diffusion decoder (Wu & Zheng, 2024). CL-Dis combines β-VAE distillation with diffusion feedback (Jin et al., 2024).

DiTi organizes representations according to attributes that disappear at different noise levels (Yue et al., 2024). Isometric Diffusion regularizes the geometry of the mapping from noisy samples to intermediate denoiser features, improving latent-space interpolation and manipulation (Hahm et al., 2024). EncDiff maps individual latent dimensions to concept tokens and conditions the denoiser through cross-attention (Yang et al., 2024). Jun et al. (2025) combine dynamic Gaussian anchoring to organize latent units with skip dropout to encourage the denoiser to use latent conditions. Jung et al. (2025) promote disentanglement through latent recombination, a pretrained diffusion prior, and compositional consistency. Theoretical work also establishes conditions for diffusion-based identifiability under weak supervision (Wang et al., 2025).

Flow matching has also been explored for representation learning. Structured Flow Autoencoders combine latent-variable models with conditional continuous normalizing flows through a latent-variable-aware flow-matching objective (Xu et al., 2026). SCFlow studies style–content disentanglement by learning an invertible mapping between style–content reference representations and their mixtures using structured training triplets (Ma et al., 2025). Our approach applies the same local interaction regularizer to both diffusion and rectified-flow models.

3 Dynamics Hessian Penalty

3.1 Model Setup

Let $y \sim p_{\mathrm{data}}$ denote an observed image, $x_0 \in \mathbb{R}^d$ its corresponding clean sample in the generator's modeling space, and $z = \tau_\phi(y) \in \mathbb{R}^N$ its learned representation. Each coordinate z_i is a scalar. We construct noisy samples as

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I_d), \quad t \in [0, 1], \tag{1}$$

where t denotes normalized time and α_t and σ_t specify the signal and noise scales, respectively. Larger t corresponds to higher noise levels.

Given the noisy sample x_t, time t, and a condition $c(z)$ constructed from the representation, the model produces the prediction $F_\theta(x_t, t, c(z))$. The baseline training objective is

$$\mathcal{L}_{\mathrm{base}} = \mathbb{E}_{y,t,\epsilon}\left[\|F_\theta(x_t, t, c(z)) - q_t\|\right], \tag{2}$$

where q_t is the prediction target.

For variance-preserving diffusion, we instantiate F_θ with the noise predictor ϵ_θ and set $q_t = \epsilon$. The schedule coefficients are $(\alpha_t, \sigma_t) = (\sqrt{\bar{\alpha}_t}, \sqrt{1 - \bar{\alpha}_t})$, with t sampled uniformly over the discrete

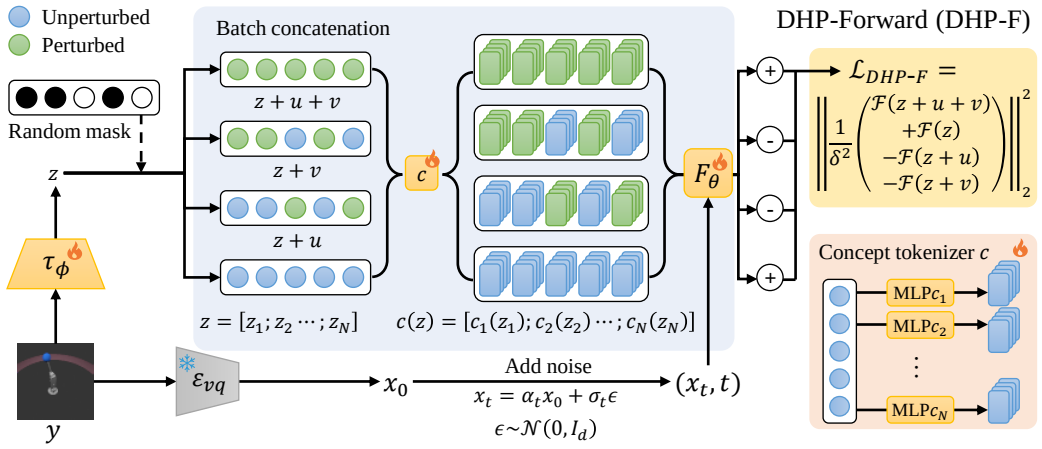


Figure 2: **Overview of DHP-Forward (DHP-F).** For each sample, four latent branches share the same (x_t, t) and are evaluated in a single batch through the concept tokenizer c and predictor F_θ. Their predictions form the normalized mixed finite-difference residual R_{F}, whose squared norm is penalized. Flames and snowflakes indicate trainable and frozen modules, respectively.

training time grid (Ho et al., 2020; Yang et al., 2024). For rectified flow (RF), we instantiate F_θ with the velocity predictor v_θ and use linear interpolation, $(\alpha_t, \sigma_t) = (1-t, t)$. Under this data-to-noise convention, the target is $q_t = \epsilon - x_0$, with $t \sim \mathcal{U}[0,1]$ (Lipman et al., 2023; Liu et al., 2023; Ma et al., 2024). In both cases, the prediction parameterizes the current generative update.

Following EncDiff (Yang et al., 2024), we use a frozen VQGAN autoencoder (Esser et al., 2021; Rombach et al., 2022) to obtain $x_0 = \mathcal{E}_{\mathrm{vq}}(y)$ and a trainable representation encoder to obtain $z = \tau_\phi(y)$. A *concept tokenizer* c independently maps each scalar latent dimension to a token, forming the conditioning sequence

$$c(z) := [c_1(z_1); \ldots; c_N(z_N)], \qquad c_i : \mathbb{R} \to \mathbb{R}^{d_{\mathrm{tok}}}. \tag{3}$$

The tokens condition the predictor through cross-attention. To place the latent dimensions on a comparable scale, we apply BatchNorm1d (BN1d) at the output of τ_ϕ before perturbation and tokenization; z denotes the normalized encoder output throughout.

3.2 Regularizing Conditional Generative Dynamics

For a twice continuously differentiable response $h = (h_1, \ldots, h_D) : \mathbb{R}^N \to \mathbb{R}^D$, each mixed derivative $\partial^2 h_\ell / \partial z_i \partial z_j$, with $i \neq j$, measures how the sensitivity to one latent dimension changes with another. The Hessian Penalty (HP) discourages this dependence as an inductive bias toward disentanglement (Peebles et al., 2020). We quantify these interactions using the total off-diagonal Hessian energy, summed over output dimensions:

$$\mathcal{H}(h; z) := \sum_{\ell=1}^{D} \sum_{i \neq j} \left(\frac{\partial^2 h_\ell(z)}{\partial z_i \, \partial z_j} \right)^2 . \tag{4}$$

The effect of this prior depends on the response being regularized. With the sampling randomness held fixed, let $\mathcal{G}_\theta(z)$ denote the complete generative mapping. If $\mathcal{G}_\theta$ is twice continuously differentiable on a rectangular latent domain and $\mathcal{H}(\mathcal{G}_\theta; z) = 0$ throughout that domain, then

$$\mathcal{G}_\theta(z) = g_0 + \sum_{i=1}^{N} g_i(z_i). \tag{5}$$

Applied to the complete mapping, this prior thus favors additive structure in observation space, potentially restricting nonlinear composition of latent effects.

We instead encourage disentanglement through more separable latent contributions to individual generative updates, while allowing these contributions to depend on the current sample. For a given noisy sample x_t and time t, define

$$\mathcal{F}(z) := F_\theta(x_t, t, c(z)), \tag{6}$$

with the dependence on θ, x_t, and t left implicit. We penalize $\mathcal{H}(\mathcal{F}; z)$ during joint training of the encoder and predictor, favoring representations whose coordinates make more separable contributions to these updates.

Under the same smoothness and domain conditions, $\mathcal{H}(\mathcal{F}; z) = 0$ throughout the latent domain for each fixed (x_t, t) implies

$$F_\theta(x_t, t, c(z)) = f_0(x_t, t) + \sum_{i=1}^{N} f_i(x_t, t, z_i). \tag{7}$$

We call this structure *additivity in the conditional dynamics*, or *local additivity*. Each contribution f_i can still depend on x_t; for example, a color-related effect may vary with the object region present in the current sample. During sampling, latent effects can compose nonlinearly through the evolving state, so local additivity does not force the complete generative mapping to be additive. Appendices A.2 and A.3 provide the sensitivity recursion and a counterexample.

3.3 Forward and Central Estimators

We approximate the local interaction prior using mixed finite differences of $\mathcal{F}$, without explicitly computing Hessians. For a dimension-wise perturbation scale $\delta > 0$, we sample a mask $m \in \{0,1\}^N$ with independent $m_i \sim \mathrm{Bernoulli}(1/2)$ and an independent Rademacher vector $\xi \in \{-1,+1\}^N$, whose entries independently take either sign with equal probability. We construct two perturbations with disjoint support:

$$u = \delta\, m \odot \xi, \qquad v = \delta\, (1 - m) \odot \xi, \tag{8}$$

where $\odot$ denotes element-wise multiplication. The mask randomly partitions the latent dimensions into two groups, allowing us to probe interactions between them.

DHP-Forward (DHP-F). The forward mixed finite-difference residual is

$$R_{\mathrm{F}}(z; u, v) := \frac{\mathcal{F}(z+u+v) + \mathcal{F}(z) - \mathcal{F}(z+u) - \mathcal{F}(z+v)}{\delta^2}. \tag{9}$$

Its numerator compares the change induced by the joint perturbation with the sum of the changes induced by the two separate perturbations. Figure 2 illustrates this construction.

DHP-Central (DHP-C). Using symmetric perturbations around z, we define the central mixed finite-difference residual as

$$R_{\mathrm{C}}(z; u, v) := \frac{\mathcal{F}(z+u+v) + \mathcal{F}(z-u-v) - \mathcal{F}(z+u-v) - \mathcal{F}(z-u+v)}{4\delta^2}. \tag{10}$$

Here, "forward" and "central" refer to the finite-difference constructions in latent space. For both variants, all branches share the same (x_t, t); only z changes.

Because u and v have disjoint support, the leading-order term of either residual involves only off-diagonal Hessian entries. For a twice continuously differentiable response, averaging their squared norms over the random masks and signs gives

$$\lim_{\delta \to 0} \mathbb{E}_{m,\xi} \left[\|R_s(z; u, v)\|_2^2 \right] = \frac{1}{4} \mathcal{H}(\mathcal{F}; z), \qquad s \in \{\mathrm{F}, \mathrm{C}\}. \tag{11}$$

Both residuals also vanish exactly for an additive response, for any perturbation scale whose evaluation points remain within the domain.

The two constructions differ in their finite-difference accuracy. For fixed m and ξ, the normalized forward residual has $O(\delta)$ approximation error under C^3 smoothness, while the central residual has $O(\delta^2)$ error under C^4 smoothness. Derivations of these properties and Eq. (11) are provided in Appendix B.

3.4 Training Objective and Computation

Using the normalized residuals defined above, we write the regularization loss as

$$\mathcal{L}^{(s)}_{\mathrm{DHP}} := \mathbb{E}_{y,t,\epsilon,m,\xi}\left[\|R_s(z;u,v)\|_2^2\right], \qquad s \in \{\mathrm{F},\mathrm{C}\}. \tag{12}$$

Here, (y, t, ϵ) follow the sampling distributions specified in Section 3.1. The mask m and signs ξ are independent of each other. We train each variant using

$$\mathcal{L}^{(s)} = \mathcal{L}_{\mathrm{base}} + \lambda \mathcal{L}^{(s)}_{\mathrm{DHP}}, \tag{13}$$

where $\lambda \geq 0$ controls the regularization strength.

During training, x_t is constructed directly from x_0 and sampled noise using Eq. (1). We jointly optimize the representation encoder τ_ϕ, the concept tokenizer c, and the predictor F_θ, while the image autoencoder remains frozen. The finite-difference formulation requires only predictor evaluations and standard backpropagation, without explicit Hessians or higher-order automatic differentiation.

For each training image, DHP-F uses four prediction branches: the unperturbed prediction $\mathcal{F}(z)$, shared with the base loss, and three perturbed predictions. Training with DHP-C uses five branches: the unperturbed prediction for the base loss and four perturbed predictions for the regularizer. The branches can be concatenated along the batch dimension and evaluated in a single batched forward pass.

The latent perturbations and finite-difference residuals are used only during training. At inference, we use the same sampling procedure as the corresponding baseline; DHP introduces no additional sampling steps or predictor evaluations. Algorithm 1 summarizes a training update.

4 Experiments

We evaluate DHP on three synthetic benchmarks for disentanglement and on CelebA for both disentanglement and image generation quality. Further experiments examine its applicability to rectified-flow velocity prediction and its sensitivity to key hyperparameters. Finally, we illustrate controllability using latent traversals and swapping, and measure downstream sample efficiency with limited factor labels.

4.1 Experimental Setup

Datasets and metrics. We evaluate disentanglement on Shapes3D (Burgess & Kim, 2018), MPI3D-toy (Gondal et al., 2019), and Cars3D (Reed et al., 2015), with all images resized to 64×64. We report the FactorVAE score (Kim & Mnih, 2018) and DCI disentanglement (Eastwood & Williams, 2018); higher values indicate better disentanglement for both metrics. Our main synthetic-benchmark results are summarized as the mean and sample standard deviation over ten independent training runs. Ground-truth factor labels are used for evaluation only.

On CelebA (Liu et al., 2015), we report TAD (Yeats et al., 2022) for disentanglement and FID (Heusel et al., 2017) for image generation quality. TAD uses the first $10{,}000$ CelebA images in filename order and aggregates scores over five evaluation subsets, following InfoDiffusion's implementation (Wang et al., 2023). FID uses five sets of $10{,}000$ images generated with latent conditions $z \sim \mathcal{N}(0, I_N)$. For each model, standard deviations reflect evaluation variability within a single checkpoint. Appendix D provides implementation and evaluation details.

Model configurations. Our primary models build on the EncDiff architecture (Yang et al., 2024) with diffusion noise prediction. We train DHP-F and DHP-C separately using Eq. (13); both apply BN1d before latent perturbation and tokenization. We train each model for $150{,}000$ steps and use the final checkpoint for evaluation. Unless otherwise specified, we use $N = 20$ latent dimensions to match EncDiff, a regularization weight of $\lambda = 0.08$, and a perturbation scale of $\delta = 0.1$. For each training image, we independently sample a Bernoulli mask m and a Rademacher vector ξ. Diffusion sampling uses DDIM (Song et al., 2021) with 200 steps and $\eta = 0$.

We also evaluate DHP under rectified-flow velocity prediction using the linear path and prediction target in Section 3.1, while retaining the architecture and other training settings of the diffusion implementation.

Table 1: Disentanglement on synthetic benchmarks. DHP results are reported as mean ± sample standard deviation over ten independent training runs. Baseline results are taken from prior publications; – denotes an unreported result. The best and second-best means in each column are boldfaced and underlined, respectively; ties share a rank.

	Shapes3D		MPI3D-toy		Cars3D	
Method	FactorVAE ↑	DCI ↑	FactorVAE ↑	DCI ↑	FactorVAE ↑	DCI ↑
FactorVAE	0.840 ± 0.066	0.611 ± 0.082	0.152 ± 0.025	0.240 ± 0.051	0.906 ± 0.052	0.161 ± 0.019
β-TCVAE	0.873 ± 0.074	0.613 ± 0.114	0.179 ± 0.017	0.237 ± 0.056	0.855 ± 0.082	0.140 ± 0.019
DisCo-GAN	0.877 ± 0.031	0.708 ± 0.048	0.371 ± 0.030	0.292 ± 0.024	0.855 ± 0.074	0.271 ± 0.037
DisDiff-VQ	0.902 ± 0.043	0.723 ± 0.013	0.617 ± 0.070	0.337 ± 0.057	0.976 ± 0.018	0.232 ± 0.019
FDAE	0.987 ± 0.023	0.917 ± 0.038	–	–	0.918 ± 0.027	0.418 ± 0.036
EncDiff	$\underline{0.999} \pm 0.000$	0.969 ± 0.030	0.872 ± 0.049	0.685 ± 0.044	0.773 ± 0.060	0.279 ± 0.022
DyGA	$\mathbf{1.000} \pm 0.000$	0.938 ± 0.001	$\mathbf{0.930} \pm 0.004$	0.627 ± 0.002	0.941 ± 0.002	0.414 ± 0.013
DHP-F	$\mathbf{1.000} \pm 0.000$	$\mathbf{0.975} \pm 0.022$	0.914 ± 0.043	$\underline{0.740} \pm 0.014$	$\underline{0.994} \pm 0.011$	$\underline{0.752} \pm 0.060$
DHP-C	$\mathbf{1.000} \pm 0.000$	$\underline{0.972} \pm 0.031$	$\underline{0.924} \pm 0.036$	$\mathbf{0.747} \pm 0.021$	$\mathbf{0.997} \pm 0.008$	$\mathbf{0.780} \pm 0.063$

Table 2: Disentanglement and generation on CelebA. DHP results are means and standard deviations from one checkpoint per model. TAD uses five evaluation subsets; FID uses five sets of 10,000 generated images. Baseline results are taken from prior publications.

Method	TAD ↑	FID ↓
β-VAE	0.088 ± 0.043	99.8 ± 2.4
DiffAE	0.155 ± 0.010	22.7 ± 2.1
InfoDiffusion	0.299 ± 0.006	23.6 ± 1.3
DisDiff	0.305 ± 0.010	18.2 ± 2.1
EncDiff	0.638 ± 0.008	14.8 ± 2.3
DyGA	0.954 ± 0.024	12.0 ± 1.2
DHP-F	$\underline{1.228} \pm 0.025$	$\underline{7.24} \pm 0.08$
DHP-C	$\mathbf{1.233} \pm 0.080$	$\mathbf{7.00} \pm 0.13$

Table 3: Effect of BN1d and DHP on MPI3D-toy under noise and velocity prediction. All models use the same ten training seeds. Results are means and sample standard deviations. Rankings are computed within each prediction setting; ties share a rank.

Method	FactorVAE ↑	DCI ↑
EncDiff	0.901 ± 0.023	0.705 ± 0.010
+ BN1d	$\underline{0.914} \pm 0.029$	0.716 ± 0.030
+ BN1d + DHP-F	$\underline{0.914} \pm 0.043$	$\underline{0.740} \pm 0.014$
+ BN1d + DHP-C	$\mathbf{0.924} \pm 0.036$	$\mathbf{0.747} \pm 0.021$
RF baseline	0.888 ± 0.027	0.692 ± 0.021
+ BN1d	$\mathbf{0.938} \pm 0.032$	0.686 ± 0.044
+ BN1d + DHP-F	$\underline{0.922} \pm 0.057$	$\mathbf{0.731} \pm 0.035$
+ BN1d + DHP-C	$\mathbf{0.938} \pm 0.024$	$\underline{0.726} \pm 0.030$

Baselines. On synthetic datasets, we compare with VAE-based FactorVAE (Kim & Mnih, 2018) and β-TCVAE (Chen et al., 2018), GAN-based DisCo-GAN (Ren et al., 2022), and diffusion-based DisDiff-VQ (Yang et al., 2023), FDAE (Wu & Zheng, 2024), EncDiff (Yang et al., 2024), and DyGA (Jun et al., 2025). On CelebA, we compare with β-VAE (Higgins et al., 2017), DiffAE (Preechakul et al., 2022), InfoDiffusion (Wang et al., 2023), DisDiff, EncDiff, and DyGA. Baseline results in Tables 1 and 2 are taken from prior publications with their original evaluation protocols and run counts. For a fair comparison, all configurations in Table 3 are trained using the same ten training seeds.

4.2 Main Results

Disentanglement on synthetic benchmarks. Table 1 reports results on Shapes3D, MPI3D-toy, and Cars3D. Both DHP variants achieve higher mean DCI scores than all compared baselines across the three datasets. The largest DCI gains occur on Cars3D, where DHP-F and DHP-C achieve 0.752 and 0.780, respectively, compared with 0.418 for FDAE, the best-performing baseline on this metric.

Disentanglement and generation on CelebA. Table 2 evaluates disentanglement and image generation quality on CelebA using TAD and FID, respectively. Both DHP variants achieve higher mean TAD and lower mean FID than all compared baselines. DHP-F and DHP-C obtain TAD scores of 1.228 and 1.233, with FID scores of 7.24 and 7.00, respectively. DyGA, the strongest baseline on both metrics, obtains a TAD of 0.954 and an FID of 12.0.

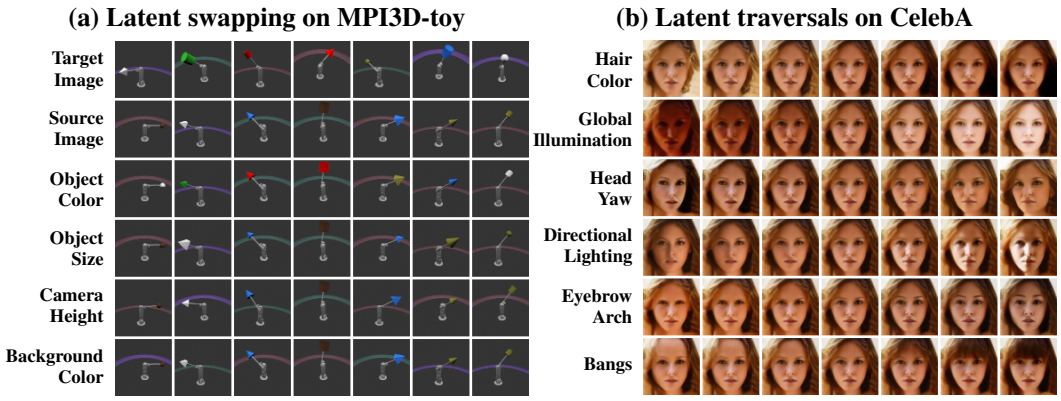


Figure 3: **Latent swapping and traversals with DHP-C.** (a) MPI3D-toy: the first two rows show target and source images, respectively. Each subsequent row replaces one selected dimension of the source representation with its value from the target representation, using the same dimension across columns. (b) CelebA: each row varies one latent dimension by adding offsets $\{-3, -2, -1, 0, 1, 2, 3\}$ from left to right. All other latent dimensions remain unchanged. The initial noise is fixed within each column in (a) and each row in (b). Row labels indicate the main visible attribute changes.

4.3 Qualitative Analysis

Figure 3 illustrates controllability through latent swapping and traversals with DHP-C. On MPI3D-toy, single-dimension latent swapping transfers object color, object size, camera height, or background color from the target image to the source image. To transfer a given attribute, we swap the same latent dimension across different image pairs. The examples suggest that these dimensions retain a consistent semantic role across different source images. On CelebA, traversing selected dimensions produces interpretable changes in attributes such as hair color, illumination, and head pose. The changes develop gradually along each traversal while preserving much of the remaining facial appearance. Appendix E provides the evaluation details.

4.4 Extension to Rectified Flow

We evaluate DHP under diffusion noise prediction and rectified-flow velocity prediction on MPI3D-toy (Table 3). To control for latent normalization, we compare each DHP variant with the corresponding BN1d-only baseline.

DHP-F and DHP-C improve mean DCI by 0.024 and 0.031 under noise prediction, and by 0.045 and 0.040 under velocity prediction, respectively. These improvements persist with the same latent normalization, supporting the contribution of DHP and its applicability to rectified flow.

4.5 Hyperparameter Sensitivity

We examine the sensitivity of DHP-C to the regularization weight λ and the number of latent dimensions N on MPI3D-toy. Results are reported as mean $\pm$ sample standard deviation over ten independent training runs for each setting.

Regularization weight. With $N = 20$, we vary $\lambda \in \{0.016, 0.08, 0.4\}$ (Table 4). Mean DCI remains stable across the tested weights, ranging from 0.745 to 0.747. The mean FactorVAE score varies from 0.924 to 0.956, with the highest value obtained at $\lambda = 0.016$.

Number of latent dimensions. With $\lambda = 0.08$, we vary $N \in \{10, 20, 30\}$ (Table 5). Both mean disentanglement scores increase over this range, reaching 0.940 for FactorVAE and 0.778 for DCI at $N = 30$. We retain $N = 20$ in the main comparison to match EncDiff.

Table 4: Sensitivity of DHP-C to λ on MPI3D-toy, with $N = 20$.

λ	FactorVAE ↑	DCI ↑
0.4	$\underline{0.930} \pm 0.042$	$\underline{0.745} \pm 0.015$
0.08	0.924 ± 0.036	$\mathbf{0.747} \pm 0.021$
0.016	$\mathbf{0.956} \pm 0.011$	$\underline{0.745} \pm 0.013$

Table 5: Sensitivity of DHP-C to N on MPI3D-toy, with $\lambda = 0.08$.

N	FactorVAE ↑	DCI ↑
10	0.817 ± 0.052	0.673 ± 0.041
20	$\underline{0.924} \pm 0.036$	$\underline{0.747} \pm 0.021$
30	$\mathbf{0.940} \pm 0.021$	$\mathbf{0.778} \pm 0.007$

4.6 Downstream Tasks

Following Jun et al. (2025), we evaluate the sample efficiency of downstream classification using the learned representations on Shapes3D and MPI3D-toy. We train gradient boosted tree (GBT) classifiers to predict ground-truth factors from frozen representations. Let Acc_n denote test accuracy averaged over factors when the classifiers are trained with n labeled samples. We compare training with 100 or 1,000 samples against a 10,000-sample reference through the ratios $\mathrm{Acc}_{100}/\mathrm{Acc}_{10000}$ and $\mathrm{Acc}_{1000}/\mathrm{Acc}_{10000}$. Higher ratios indicate better sample efficiency under this evaluation protocol. Further details are provided in Appendix D.5.

Table 6: Sample efficiency of GBT classifiers on Shapes3D and MPI3D-toy. DHP results are reported as mean ± standard deviation over ten checkpoints. Baseline results are taken from Jun et al. (2025); – denotes an unreported result.

	Shapes3D		MPI3D-toy	
Method	$\mathrm{Acc}_{1000}/\mathrm{Acc}_{10000}$ ↑	$\mathrm{Acc}_{100}/\mathrm{Acc}_{10000}$ ↑	$\mathrm{Acc}_{1000}/\mathrm{Acc}_{10000}$ ↑	$\mathrm{Acc}_{100}/\mathrm{Acc}_{10000}$ ↑
DisDiff	0.928 ± 0.001	0.732 ± 0.002	0.862 ± 0.001	0.700 ± 0.002
FDAE	0.979 ± 0.002	0.751 ± 0.019	–	–
EncDiff	0.975 ± 0.000	0.772 ± 0.002	0.853 ± 0.005	0.701 ± 0.001
DyGA	$\underline{0.990} \pm 0.000$	0.863 ± 0.001	0.888 ± 0.002	$\mathbf{0.757} \pm 0.001$
DHP-F	$\mathbf{0.992} \pm 0.001$	$\mathbf{0.868} \pm 0.020$	$\underline{0.919} \pm 0.011$	0.754 ± 0.013
DHP-C	$\mathbf{0.992} \pm 0.002$	$\underline{0.867} \pm 0.029$	$\mathbf{0.924} \pm 0.007$	$\underline{0.756} \pm 0.010$

Table 6 shows that both DHP variants achieve competitive sample efficiency on Shapes3D and MPI3D-toy, with higher mean ratios than the compared baselines in most evaluated settings. At the 1,000-sample budget, both variants retain most of their accuracy under the 10,000-sample reference. These results complement the disentanglement metrics by showing that factor information in the frozen representations remains accessible with limited supervision.

Comparison of DHP variants. DHP-F and DHP-C offer a trade-off between training computation and finite-difference accuracy. DHP-F reuses the unperturbed prediction from the base loss, requiring four prediction branches per training image, compared with five for DHP-C. The symmetric construction of DHP-C provides a higher-order residual approximation under the smoothness conditions in Section 3.3. Empirically, DHP-C achieves better mean performance in most evaluated settings, while DHP-F also outperforms most baselines.

5 Conclusion

We introduced DHP, a disentanglement prior that builds on the iterative structure of conditional generative models. DHP discourages interactions between latent dimensions within each prediction, while allowing their effects to depend on the current sample and compose nonlinearly through successive updates. Our analysis shows that the complete generative mapping need not collapse to an additive form even when all local mixed latent derivatives vanish.

Experiments demonstrate gains in disentanglement with both diffusion and rectified-flow models. Evaluations of image generation, downstream factor prediction, and latent control further support the utility of the learned representations. Implemented through stochastic finite differences, DHP integrates into standard training without sampler unrolling or additional inference cost.

AI USE STATEMENT

In this work, we used generative AI tools to aid writing and polish the language of the manuscript. We have reviewed all AI-assisted text and revised it for accuracy and clarity. We take responsibility for the final content of this work, including the AI-assisted text.

REFERENCES

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

Jack Brady, Julius von Kügelgen, Sébastien Lachapelle, Simon Buchholz, Thomas Kipf, and Wieland Brendel. Interaction asymmetry: A general principle for learning composable abstractions. In *International Conference on Learning Representations*, 2025.

Christopher Burgess and Hyunjik Kim. 3D Shapes Dataset. `https://github.com/google-deepmind/3d-shapes`, 2018.

Tian Qi Chen, Xuechen Li, Roger B. Grosse, and David K. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12873–12883, 2021.

Muhammad Waleed Gondal, Manuel Wuthrich, Djordje Miladinovic, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: A new disentanglement dataset. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Jaehoon Hahm, Junho Lee, Sunghyun Kim, and Joonseok Lee. Isometric representation learning for disentangled latent space of diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17224–17245, 2024.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Kyle Hsu, Will Dorrell, James C. R. Whittington, Jiajun Wu, and Chelsea Finn. Disentanglement via latent quantization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 45463–45488, 2023.

Kyle Hsu, Jubayer Ibn Hamid, Kaylee Burns, Chelsea Finn, and Jiajun Wu. Tripod: Three complementary inductive biases for disentangled representation learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 19101–19122, 2024.

Xin Jin, Bohan Li, Baao Xie, Wenyao Zhang, Jinming Liu, Ziqiang Li, Tao Yang, and Wenjun Zeng. Closed-loop unsupervised representation disentanglement with β-VAE distillation and diffusion probabilistic feedback. In *European Conference on Computer Vision*, Lecture Notes in Computer Science, pp. 270–289, 2024.

Youngjun Jun, Jiwoo Park, Kyobin Choo, Tae Eun Choi, and Seong Jae Hwang. Disentangling disentangled representations: Towards improved latent units via diffusion models. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pp. 3559–3569, 2025.

Whie Jung, Dong Hoon Lee, and Seunghoon Hong. Disentangled representation learning via modular compositional bias. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2649–2658, 2018.

Sébastien Lachapelle, Divyat Mahajan, Ioannis Mitliagkas, and Simon Lacoste-Julien. Additive decoders for latent variables identification and cartesian-product extrapolation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Qiyao Liang, Daoyuan Qian, Liu Ziyin, and Ila R. Fiete. Compositional generalization via forced rendering of disentangled latents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 37370–37395, 2025.

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 Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3730–3738, 2015.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4114–4124, 2019.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, Lecture Notes in Computer Science, 2024.

Pingchuan Ma, Xiaopei Yang, Yusong Li, Ming Gui, Felix Krause, Johannes Schusterbauer, and Björn Ommer. SCFlow: Implicitly learning style and content disentanglement with flow models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14919–14929, 2025.

Stefan Matthes, Zhiwei Han, and Hao Shen. Mechanistic independence: A principle for identifiable disentangled representations. In *International Conference on Learning Representations*, 2026.

Vaishnavi S. Patil, Siddhi Patil, Matthew Evanusa, Amit Kumar Kundu, Cornelia Fermüller, and Joseph JáJá. A Bayesian nonparametric framework for learning disentangled representations. In *International Conference on Learning Representations*, 2026.

William Peebles, John Peebles, Jun-Yan Zhu, Alexei A. Efros, and Antonio Torralba. The hessian penalty: A weak prior for unsupervised disentanglement. In *European Conference on Computer Vision*, volume 12351 of *Lecture Notes in Computer Science*, pp. 581–597, 2020. doi: 10.1007/978-3-030-58539-6_35.

Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10619–10629, 2022.

Scott E. Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Xuanchi Ren, Tao Yang, Yuwang Wang, and Wenjun Zeng. Learning disentangled representation by exploiting pretrained generative models: A contrastive learning view. In *International Conference on Learning Representations*, 2022.

Travers Rhodes and Daniel D. Lee. Local disentanglement in variational auto-encoders using jacobian L_1 regularization. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22708–22719, 2021.

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

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Subash Timilsina, Hoang-Son Nguyen, Sagar Shrestha, and Xiao Fu. Content-style identification via differential independence. In *International Conference on Machine Learning*, 2026.

Théo Uscidda, Luca Eyring, Karsten Roth, Fabian Theis, Zeynep Akata, and Marco Cuturi. Disentangled representation learning with the Gromov–Monge gap. In *International Conference on Learning Representations*, 2025.

Liming Wang, Muhammad Jehanzeb Mirza, Yishu Gong, Yuan Gong, Jiaqi Zhang, Brian H. Tracey, Katerina Placek, Marco Vilela, and Jim Glass. Can diffusion models disentangle? A theoretical perspective. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Yingheng Wang, Yair Schiff, Aaron Gokaslan, Weishen Pan, Fei Wang, Christopher De Sa, and Volodymyr Kuleshov. InfoDiffusion: Representation learning using information maximizing diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36336–36354, 2023.

Yuxiang Wei, Yupeng Shi, Xiao Liu, Zhilong Ji, Yuan Gao, Zhongqin Wu, and Wangmeng Zuo. Orthogonal jacobian regularization for unsupervised disentanglement in image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6721–6730, 2021.

Ancong Wu and Wei-Shi Zheng. Factorized diffusion autoencoder for unsupervised disentangled representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 5930–5939, 2024.

Yidan Xu, Yixin Wang, and XuanLong Nguyen. Structured flow autoencoders: Learning structured probabilistic representations with flow matching. In *International Conference on Learning Representations*, 2026.

Tao Yang, Yuwang Wang, Yan Lu, and Nanning Zheng. DisDiff: Unsupervised disentanglement of diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 69130–69156, 2023.

Tao Yang, Cuiling Lan, Yan Lu, and Nanning Zheng. Diffusion model with cross attention as an inductive bias for disentanglement. In *Advances in Neural Information Processing Systems*, volume 37, pp. 82465–82492, 2024.

Eric Yeats, Frank Liu, David Womble, and Hai Li. NashAE: Disentangling representations through adversarial covariance minimization. In *European Conference on Computer Vision*, pp. 36–51, 2022.

Zhongqi Yue, Jiankun Wang, Qianru Sun, Lei Ji, Eric I-Chao Chang, and Hanwang Zhang. Exploring diffusion time-steps for unsupervised representation learning. In *International Conference on Learning Representations*, 2024.

Zijian Zhang, Zhou Zhao, and Zhijie Lin. Unsupervised representation learning from pre-trained diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22117–22130, 2022.

A Local Conditioning and Nonlinear Generative Composition

This appendix formalizes the distinction in Section 3.2: local additivity does not imply additivity of the complete generative mapping. Model parameters and sampling randomness are held fixed throughout.

A.1 Vanishing mixed derivatives and additive responses

Proposition. Let $h : \Omega \to \mathbb{R}^D$ be C^2 on a rectangular domain $\Omega = I_1 \times \cdots \times I_N$, where each I_i is an open interval. Then $\mathcal{H}(h; z) = 0$ throughout Ω if and only if

$$h(z) = h_0 + \sum_{i=1}^{N} g_i(z_i), \qquad g_i : I_i \to \mathbb{R}^D. \tag{A.1}$$

Proof. Since $\mathcal{H}$ is a sum of squares, it vanishes exactly when $\partial_{z_j}\partial_{z_i} h = 0$ for all $i \neq j$. On the rectangular domain, this means that $\partial_{z_i} h$ depends only on z_i; write it as $v_i(z_i)$. For any anchor $\bar{z} \in \Omega$, integration along a coordinate path gives

$$h(z) = h(\bar{z}) + \sum_{i=1}^{N} \int_{\bar{z}_i}^{z_i} v_i(s)\,\mathrm{d}s. \tag{A.2}$$

This is the required decomposition. Conversely, each mixed derivative of an additive C^2 map is zero.

Under the stated smoothness assumptions, applying the proposition to the complete mapping $\mathcal{G}_\theta$ gives Eq. (5). Applying it to the fixed-state response $\mathcal{F}(z) = F_\theta(x, t, c(z))$ gives

$$F_\theta(x, t, c(z)) = f_0(x, t) + \sum_i f_i(x, t, z_i). \tag{A.3}$$

The contributions f_i can still depend on the current state x, which may already carry the effects of other latent dimensions from earlier updates.

A.2 Sensitivities along a sampling trajectory

A latent dimension can affect an update both directly and through the state produced by earlier updates. To separate these paths, write the sampling trajectory in generation order as $x^{(k)} := x_{t_k}$, with $t_0 > \cdots > t_K$, $x^{(0)} = \omega$, and $x^{(k+1)} = U_k(x^{(k)}, z)$. The superscript indexes sampling steps; the initial state $x^{(0)}$ is noise, whereas x_0 denotes a clean sample. The initial noise ω is independent of z, and $S_\theta(z) = x^{(K)}$ denotes the final sampling state.

Assume each update is C^2 near the trajectory. Let $P_{k,i} = \partial_{z_i} x^{(k)}$ measure the state's sensitivity to z_i, and let $Q_{k,ij} = \partial_{z_i}\partial_{z_j} x^{(k)}$, for $i \neq j$, measure the interaction between z_i and z_j. Both are vectors in $\mathbb{R}^d$ and are initially zero. The chain rule gives

$$P_{k+1,i} = D_x U_k\, P_{k,i} + \partial_{z_i} U_k, \tag{A.4}$$

$$\begin{aligned} Q_{k+1,ij} = {} & D_x U_k\, Q_{k,ij} + D_x^2 U_k[P_{k,i}, P_{k,j}] \\ & + D_x \partial_{z_i} U_k[P_{k,j}] + D_x \partial_{z_j} U_k[P_{k,i}] + \partial_{z_i}\partial_{z_j} U_k. \end{aligned} \tag{A.5}$$

All update derivatives are evaluated at $(x^{(k)}, z)$, and explicit z derivatives hold the state argument fixed. For vector-valued U_k, $D_x^2 U_k[a, b]$ denotes the vector of coordinatewise Hessian bilinear forms. The second equation follows by differentiating the first with respect to z_j. In the first equation, the two terms describe influence through the existing state and direct influence on the current update. In the second equation, the first term propagates existing interactions, the middle three terms capture interactions mediated by the state, and the last term is the direct mixed latent derivative at a fixed state.

For updates of the form

$$U_k(x, z) = a_k x + b_k F_\theta(x, t_k, c(z)), \tag{A.6}$$

with coefficients independent of z, the direct mixed derivative is $\partial_{z_i}\partial_{z_j}U_k = b_k\partial_{z_i}\partial_{z_j}F_\theta$. Thus zero local mixed derivatives remove only the last term of Eq. (A.5). State curvature and mixed state–condition derivatives can still create interactions, which subsequent updates propagate.

Deterministic DDIM without clipping or quantization of the predicted clean state has this form, with

$$a_k = \frac{\alpha_{t_{k+1}}}{\alpha_{t_k}}, \qquad b_k = \sigma_{t_{k+1}} - a_k\sigma_{t_k}, \quad \alpha_{t_k} > 0. \tag{A.7}$$

Euler integration of rectified flow also has this form, with $a_k = 1$ and $b_k = t_{k+1} - t_k < 0$ under our data-to-noise convention (Song et al., 2021; Liu et al., 2023).

A.3 A TWO-STEP COUNTEREXAMPLE AND SCOPE

For $z = (z_1, z_2)$ and a scalar state, take $U_0(x, z) = z_1$ and $U_1(x, z) = xz_2$. Both satisfy $\partial_{z_1}\partial_{z_2}U_k(x, z) = 0$ at every fixed x, but

$$x^{(1)} = z_1, \qquad x^{(2)} = z_1 z_2, \qquad \partial_{z_1}\partial_{z_2}x^{(2)} = 1. \tag{A.8}$$

The second update combines z_2 with a state determined by z_1. For any a_k and $b_k \neq 0$, these updates have the form in Eq. (A.6) with local prediction $(U_k - a_k x)/b_k$, whose direct mixed latent derivatives also vanish. Thus, vanishing mixed derivatives of the local prediction do not imply an additive generative mapping.

For latent-space generation, the complete image mapping is $\mathcal{G}_\theta = \mathcal{D}_{\mathrm{vq}} \circ S_\theta$. The example uses an identity observation decoder, showing that the iterative updates alone suffice to produce a non-additive mapping.

The result concerns the model's capacity to represent interactions; it does not establish identifiability or guarantee semantic disentanglement. The additive characterization requires zero mixed derivatives throughout the domain. A small penalty on sampled inputs does not establish this condition. The derivative arguments apply on regions satisfying their stated smoothness assumptions.

B PROPERTIES OF THE FINITE-DIFFERENCE ESTIMATORS

Fix (x_t, t) and write $\mathcal{F}(z) = F_\theta(x_t, t, c(z)) \in \mathbb{R}^D$. For noise and velocity prediction, the output dimension is $D = d$. Let $a = m \odot \xi$ and $b = (1 - m) \odot \xi$, so $u = \delta a$, $v = \delta b$, and $a_i b_i = 0$. Assume that all finite-difference evaluation points lie in the rectangular domain Ω. The masks and signs are sampled as in Section 3.3.

The smoothness assumptions below concern $\mathcal{F}$ as a function of z. This map includes the concept tokenizer and conditional predictor; the representation encoder precedes z, and the observation decoder is not part of this map. In particular, VQ decoding is not involved in the residuals. The tokenizer uses ELU activations, which are smooth within each activation region but need not satisfy the required higher-order differentiability at region boundaries. The derivative limits and error bounds below are therefore conditional on the stated local smoothness assumptions. The finite-difference residuals themselves remain defined across such boundaries.

B.1 INTEGRAL REPRESENTATIONS, CANCELLATION, AND ERROR

For $\mathcal{F} \in C^2$, define $\psi(r_1, r_2) = \mathcal{F}(z + \delta r_1 a + \delta r_2 b)$. The chain rule gives

$$\partial_{r_1}\partial_{r_2}\psi(r_1, r_2) = \delta^2 D^2\mathcal{F}(z + \delta r_1 a + \delta r_2 b)[a, b].$$

Here $D^2\mathcal{F}(w)[a, b]$ is the vector whose ℓth entry is $a^\top\nabla^2\mathcal{F}_\ell(w)b$. Integrating the mixed derivative over $[0, 1]^2$ gives

$$\int_0^1\int_0^1 \partial_{r_1}\partial_{r_2}\psi(r_1, r_2)\,\mathrm{d}r_2\,\mathrm{d}r_1 = \psi(1, 1) - \psi(1, 0) - \psi(0, 1) + \psi(0, 0).$$

These are exactly the four terms in the forward residual numerator. Integrating over $[-1,1]^2$ similarly gives the central residual numerator. Dividing by δ^2 and $4\delta^2$, respectively, yields

$$R_{\mathrm{F}}(z;u,v) = \int_0^1 \int_0^1 D^2\mathcal{F}(z + r_1 u + r_2 v)[a,b]\,\mathrm{d}r_2\,\mathrm{d}r_1, \tag{B.1}$$

$$R_{\mathrm{C}}(z;u,v) = \frac{1}{4}\int_{-1}^1 \int_{-1}^1 D^2\mathcal{F}(z + r_1 u + r_2 v)[a,b]\,\mathrm{d}r_2\,\mathrm{d}r_1. \tag{B.2}$$

Thus, each residual averages a mixed directional derivative over its perturbation region. The factor $1/4$ normalizes the area of $[-1,1]^2$. Since $a_i b_i = 0$, the diagonal Hessian terms vanish from the bilinear form:

$$D^2\mathcal{F}_\ell(w)[a,b] = \sum_{i\neq j} a_i b_j\, \partial_{z_i}\partial_{z_j}\mathcal{F}_\ell(w). \tag{B.3}$$

Hence only mixed derivatives contribute. If $\mathcal{F}(z) = g_0 + \sum_i g_i(z_i)$, each univariate term cancels directly between the four branches, so both residuals vanish for every admissible δ, even without differentiability of the g_i.

Hessian continuity implies, for fixed N, m, ξ,

$$R_s(z;\delta a,\delta b) \xrightarrow{\delta\to 0} D^2\mathcal{F}(z)[a,b], \qquad s\in\{\mathrm{F},\mathrm{C}\}. \tag{B.4}$$

If $\mathcal{F}\in C^3$ near z, the Hessian is locally Lipschitz, giving an $O(\delta)$ forward error. If $\mathcal{F}\in C^4$, expanding the central integrand gives

$$D^2\mathcal{F}(z+\delta(r_1a+r_2b))[a,b] = D^2\mathcal{F}(z)[a,b] + \delta D^3\mathcal{F}(z)[r_1a+r_2b,a,b] + O(\delta^2). \tag{B.5}$$

For fixed a, b, the remainder is uniform over $(r_1, r_2) \in [-1,1]^2$. The two contributions to the linear term are proportional to r_1 and r_2, respectively, and both integrate to zero on this symmetric rectangle. Under the respective smoothness assumptions above,

$$R_{\mathrm{F}} = D^2\mathcal{F}(z)[a,b] + O(\delta), \tag{B.6}$$

$$R_{\mathrm{C}} = D^2\mathcal{F}(z)[a,b] + O(\delta^2). \tag{B.7}$$

These are errors of the normalized residuals; the constants depend on the directions and local derivative bounds. At finite δ, the residuals average mixed curvature over a neighborhood of z.

B.2 Expected off-diagonal Hessian energy

Let $H_\ell = \nabla^2\mathcal{F}_\ell(z)$. Combining the (i,j) and (j,i) terms using Hessian symmetry gives

$$a^\top H_\ell b = \sum_{i<j} (H_\ell)_{ij}\xi_i\xi_j \mathbf{1}\{m_i \neq m_j\}. \tag{B.8}$$

The indicator retains a pair exactly when its dimensions belong to different mask groups. To compute the second moment, expand the square of this sum. Conditioned on m, every cross term between distinct unordered pairs contains at least one independent Rademacher sign to the first power, so its expectation over ξ is zero. For each squared term, $(\xi_i\xi_j)^2 = 1$ and $\Pr(m_i \neq m_j) = 1/2$. Consequently,

$$\mathbb{E}_{m,\xi}[(a^\top H_\ell b)^2] = \frac{1}{2}\sum_{i<j}(H_\ell)_{ij}^2 = \frac{1}{4}\sum_{i\neq j}(H_\ell)_{ij}^2. \tag{B.9}$$

The last equality counts each unordered pair twice in the sum over $i \neq j$. Summing over outputs and applying Eq. (B.4) yields

$$\lim_{\delta\to 0}\mathbb{E}_{m,\xi}[\|R_s(z;u,v)\|_2^2] = \frac{1}{4}\mathcal{H}(\mathcal{F};z), \qquad s\in\{\mathrm{F},\mathrm{C}\}. \tag{B.10}$$

For fixed N, the masks and signs have finite support, which justifies exchanging the limit and expectation. Thus, as $\delta \to 0$, the expected squared residual converges to one quarter of the off-diagonal Hessian energy of the local conditional prediction with respect to z.

C Training Algorithm and Computation

C.1 A batched training update

Algorithm 1 implements the objective in Section 3.4 using one independently sampled perturbation pair per image. Uppercase versions of sample-level variables denote stacked minibatches. The operation cat concatenates tensors along the batch dimension, and $\mathrm{repeat}_{K_{\mathrm{br}}}$ stacks K_{br} copies of a minibatch in the same order. Here K_{br} is the number of prediction branches per image, and P_j denotes the jth block of B predictions. The representation encoder τ_ϕ includes the final BN1d normalization.

For our experiments, the prediction norm in Eq. (2) is the L_1 norm. The base loss is the mean absolute prediction error, and the DHP loss is the mean squared normalized finite-difference residual. In the algorithm, mean averages over both the minibatch and all scalar predictor outputs. If each prediction has D entries, this applies a common factor $1/D$ to both terms relative to the norm notation in Eq. (13). The relative weight remains λ.

Algorithm 1 One training update with DHP-F or DHP-C

Require: Minibatch $Y = \{y_b\}_{b=1}^{B}$, variant $s \in \{\mathrm{F}, \mathrm{C}\}$, scale $\delta > 0$, weight λ.

Encode the images and sample training inputs.

$X_0 \leftarrow \mathrm{stopgrad}(\mathcal{E}_{\mathrm{vq}}(Y))$.

$Z \leftarrow \tau_\phi(Y)$, evaluated once on the original minibatch.

Sample each t_b uniformly from the diffusion training grid or from $[0, 1]$ for RF.

Independently sample $\epsilon_b \sim \mathcal{N}(0, I_d)$ for each image.

$x_{t_b} \leftarrow \alpha_{t_b} x_{0,b} + \sigma_{t_b} \epsilon_b$; collect $X_t = \{x_{t_b}\}_{b=1}^{B}$ and $T = \{t_b\}_{b=1}^{B}$.

$q_b \leftarrow \epsilon_b$ for diffusion or $q_b \leftarrow \epsilon_b - x_{0,b}$ for RF; collect $Q = \{q_b\}_{b=1}^{B}$.

Construct perturbations in normalized representation space.

Sample $M \in \{0, 1\}^{B \times N}$ and $\Xi \in \{-1, +1\}^{B \times N}$ with independent equiprobable entries.

$U \leftarrow \delta M \odot \Xi$; $V \leftarrow \delta(1 - M) \odot \Xi$.

if $s = \mathrm{F}$ **then**

  $K_{\mathrm{br}} \leftarrow 4$; $Z_{\mathrm{all}} \leftarrow \mathrm{cat}(Z, Z + U, Z + V, Z + U + V)$.

else

  $K_{\mathrm{br}} \leftarrow 5$; $Z_{\mathrm{all}} \leftarrow \mathrm{cat}(Z, Z + U + V, Z - U - V, Z + U - V, Z - U + V)$.

end if

Evaluate all branches in one batched forward pass.

$C_{\mathrm{all}} \leftarrow c(Z_{\mathrm{all}})$.

$P_{\mathrm{all}} \leftarrow F_\theta\big(\mathrm{repeat}_{K_{\mathrm{br}}}(X_t), \mathrm{repeat}_{K_{\mathrm{br}}}(T), C_{\mathrm{all}}\big)$.

Split P_{all} into K_{br} blocks $(P_0, \ldots, P_{K_{\mathrm{br}}-1})$ of B predictions each, in concatenation order.

Compute the losses and update the trainable parameters.

if $s = \mathrm{F}$ **then**

  $R \leftarrow (P_3 - P_1 - P_2 + P_0)/\delta^2$.

else

  $R \leftarrow (P_1 + P_2 - P_3 - P_4)/(4\delta^2)$.

end if

$\widehat{\mathcal{L}}_{\mathrm{base}} \leftarrow \mathrm{mean}(|P_0 - Q|)$; $\widehat{\mathcal{L}}_{\mathrm{DHP}} \leftarrow \mathrm{mean}(R \odot R)$.

$\widehat{\mathcal{L}} \leftarrow \widehat{\mathcal{L}}_{\mathrm{base}} + \lambda \widehat{\mathcal{L}}_{\mathrm{DHP}}$.

Backpropagate $\widehat{\mathcal{L}}$ and take one AdamW step for τ_ϕ, c, and F_θ; keep the VQGAN autoencoder frozen.

C.2 Gradient paths and computational cost

The frozen VQGAN encoder produces X_0 without gradient tracking. The noisy samples X_t and prediction targets Q are constructed directly from X_0, the sampled timesteps, and the sampled noise. They remain fixed across the branches for each image. Training does not simulate a denoising trajectory or backpropagate through a sampler.

The normalized representations Z are computed once on the original B images. The perturbations U and V are then added to Z, and the resulting representations are passed directly to the concept

tokenizer. BN1d is not applied again to the perturbed branches. Neither Z nor its perturbed copies are detached, so the regularizer backpropagates through every residual branch to the representation encoder, concept tokenizer, and predictor. The masks and signs are sampled independently of the trainable parameters. In evaluation mode, BN1d uses the running mean and variance accumulated during training.

DHP-F uses four predictions per image. Its unperturbed prediction P_0 is shared by the base loss and the regularizer. DHP-C uses five predictions per image: P_0 for the base loss and four symmetrically perturbed predictions for the regularizer. Both variants evaluate the representation encoder once on B images; the tokenizer and predictor process $4B$ or $5B$ branches in one batched forward pass. The single call therefore still performs four or five conditional predictions per image, with the associated computation and activation memory. These branch counts are independent of N, although the cost of each prediction can depend on the number of concept tokens. Ordinary backpropagation suffices; no explicit Hessian or higher-order automatic differentiation is required. At inference, the regularizer introduces no extra prediction branches or sampling steps.

D Experimental Details

D.1 Architecture and training configuration

We build on the publicly released code for EncDiff (Yang et al., 2024). We retain its scalar-to-token conditioning framework and apply BN1d to the scalar encoder outputs before latent perturbation and concept tokenization. The representation encoder, concept tokenizer, and predictor are jointly trained with the DHP objective.

We load pretrained weights for the VQGAN image autoencoder and keep its encoder and decoder frozen throughout representation learning. The encoder maps 64×64 images y to clean VQGAN encodings x_0 of size $3 \times 16 \times 16$ in the generator's modeling space. The decoder maps generated samples in this space back to images.

The trainable representation encoder maps each image y to N scalar latent dimensions. BN1d without affine parameters normalizes these outputs to obtain z. Table 7 lists the encoder layers in order. Each normalized latent dimension is mapped to a 16-dimensional concept token by a separate MLP with widths $1 \to 64 \to 128 \to 16$ and ELU activations in the hidden layers. The resulting concept tokens condition the U-Net predictor through cross-attention. Table 8 summarizes the remaining architecture settings.

We train with AdamW using minibatches of 128 images on a single GPU. The configured base learning rate is 2×10^{-6}. With the default learning-rate scaling and no gradient accumulation, this corresponds to a peak learning rate of 2.56×10^{-4}. The learning-rate multiplier increases linearly from 10^{-6} to 1 over the first 10,000 optimization steps and remains at 1 thereafter. AdamW uses $(\beta_1^{\mathrm{opt}}, \beta_2^{\mathrm{opt}}) = (0.9, 0.999)$, $\epsilon_{\mathrm{opt}} = 10^{-8}$, and weight decay 0.01.

Each training run lasts 150,000 steps, and all evaluations use the final checkpoint at step 150,000.

The diffusion model uses 1,000 training timesteps and the noise schedule

$$\beta_j = \left(\sqrt{0.0015} + \frac{j}{999} \left(\sqrt{0.0155} - \sqrt{0.0015} \right) \right)^2, \quad j = 0, \ldots, 999. \tag{D.1}$$

Thus, $\sqrt{\beta_j}$ varies linearly across the diffusion timesteps. During training, timesteps are sampled uniformly. The RF experiments use the path and velocity target defined in Section 3.1, with the same network architecture, optimizer settings, batch size, and training duration as the diffusion experiments.

DHP-F and DHP-C are trained separately using an L_1 prediction loss and a squared finite-difference regularizer. Both variants use a dimension-wise perturbation scale of $\delta = 0.1$ and one independently sampled perturbation pair per image per update. Appendix C describes the training update, gradient paths, and computational cost.

The default number of latent dimensions is $N = 20$, matching EncDiff. Unless otherwise specified, both DHP variants use $\lambda = 0.08$ on all datasets, as in Eq. (13). Both sensitivity studies use DHP-C

Table 7: Representation encoder architecture. Convolutions are specified by kernel size and input/output channels. All four downsampling convolutions use stride 2 and padding 1. Output shapes omit the batch dimension; the default is $N = 20$.

Layer	Output shape
Input image y	$3 \times 64 \times 64$
Conv 4×4, $3 \to 128$	$128 \times 32 \times 32$
BatchNorm2d; ReLU	$128 \times 32 \times 32$
Conv 4×4, $128 \to 128$	$128 \times 16 \times 16$
BatchNorm2d; ReLU	$128 \times 16 \times 16$
Conv 4×4, $128 \to 128$	$128 \times 8 \times 8$
BatchNorm2d	$128 \times 8 \times 8$
Conv 4×4, $128 \to 128$	$128 \times 4 \times 4$
BatchNorm2d; ReLU	$128 \times 4 \times 4$
Residual block, 128 channels	$128 \times 4 \times 4$
BatchNorm2d; ReLU	$128 \times 4 \times 4$
Residual block, 128 channels	$128 \times 4 \times 4$
Flatten	2048
FC $2048 \to N$	N
BatchNorm1d, without affine parameters	N

Each residual block adds its input to a branch consisting of ReLU, a 3×3 convolution, BatchNorm2d, ReLU, and a 1×1 convolution. Both convolutions have 128 input and output channels and stride 1, with padding 1 and 0, respectively.

Table 8: Additional architecture settings for DHP-F and DHP-C. The default is $N = 20$ scalar latent dimensions.

Component / parameter	Setting
Concept tokenizer	N independent MLPs, $1 \to 64 \to 128 \to 16$
U-Net input / output	$3 \times 16 \times 16$
U-Net model channels / channel multipliers	64 / $[1, 2, 4, 4]$
U-Net residual blocks per level	2
Attention downsampling factors / heads	$[1, 2, 4]$ / 8
Cross-attention context dimension	16
VQGAN code dimension / codebook size	3 / 2048
VQGAN base channels / multipliers	32 / $[1, 2, 4]$
VQGAN residual blocks per level	2

on MPI3D-toy: the weight sweep varies $\lambda \in \{0.016, 0.08, 0.4\}$ at $N = 20$, and the dimension sweep varies $N \in \{10, 20, 30\}$ at $\lambda = 0.08$.

D.2 Sampling

For diffusion models, we use DDIM (Song et al., 2021) with 200 sampling steps, uniformly spaced timesteps, and $\eta = 0$. The normalized representation z remains fixed throughout each sampling trajectory, and the predictor is conditioned on its concept tokens $c(z)$ at every step. Sampling is deterministic given the initial noise and z. DHP uses the same sampling procedure as the corresponding baseline and introduces no additional sampling steps or predictor evaluations at inference.

D.3 Disentanglement on synthetic datasets

Images from Shapes3D, MPI3D-toy, and Cars3D are resized to 64×64. We evaluate the normalized encoder representation z. Ground-truth factor labels are used only for evaluation and are not used to train the representation model. We report the FactorVAE score (Kim & Mnih, 2018) and DCI disentanglement (Eastwood & Williams, 2018), using the same metric implementations and evaluation settings as EncDiff (Yang et al., 2024).

For the FactorVAE score, we estimate global latent variances using $10{,}000$ observations. The majority-vote classifier uses $10{,}000$ training votes and $5{,}000$ evaluation votes. Each vote is obtained from a group of 64 images with one ground-truth factor held fixed. Latent dimensions with an estimated standard deviation below 0.05 are excluded.

For DCI, we use $10{,}000$ labeled representations to train gradient boosted tree classifiers and $5{,}000$ for evaluation. The disentanglement score is computed from the classifiers' feature-importance matrix using EncDiff's implementation.

All DHP experiments on the synthetic datasets use the same set of ten training seeds. This seed set is also used for all comparison configurations in Table 3. Each configuration is trained in ten separate runs, one per seed, and we report the mean and sample standard deviation across runs. FactorVAE and DCI are evaluated on the same checkpoint within each run, so both metrics assess the same learned representation. Baseline results taken from prior publications retain the evaluation and aggregation protocols of their original sources, as specified in Section 4.1.

D.4 Disentanglement and generation on CelebA

We evaluate disentanglement using TAD (Yeats et al., 2022), following InfoDiffusion's attribute-selection and evaluation procedures (Wang et al., 2023), and assess image generation quality using FID (Heusel et al., 2017).

For FID evaluation, we sample latent representations $z \sim \mathcal{N}(0, I_N)$ and condition the diffusion model on $c(z)$. The initial diffusion noise is sampled independently of z. For each evaluated checkpoint, we generate five sets of $10{,}000$ images and compute FID separately for each set. We report the mean and standard deviation of the five FID scores.

FID reference statistics are computed from all CelebA images. We apply the D2C face crop and resize the reference images to 64×64, then compute the empirical mean and covariance of their $2{,}048$-dimensional Inception features. The same reference statistics are used for all five generated image sets.

For TAD evaluation, we encode the first $10{,}000$ CelebA images in ascending filename order. We randomly partition these images into five equally sized folds. Following InfoDiffusion, each evaluation subset consists of four folds, yielding five overlapping subsets of $8{,}000$ images. Each latent dimension is standardized separately within each subset before computing TAD.

For each of CelebA's 40 attributes, we compute the AUROC using each scalar latent dimension as a prediction score. For each dimension, we take the higher AUROC obtained from its value and its negation. We retain an attribute if its best AUROC across latent dimensions is at least 0.75 and its maximum pairwise mutual information with any other attribute, normalized by the entropy of the attribute being evaluated, is at most 0.2. TAD sums the differences between the highest and second-highest AUROCs across latent dimensions over all retained attributes. We report the mean and population standard deviation (`ddof=0`) of the five TAD scores, following InfoDiffusion's implementation.

For each evaluated model, all repetitions use a single trained checkpoint. The reported FID standard deviation therefore reflects variability across generated image sets, while the TAD standard deviation reflects variability across the five overlapping evaluation subsets. These standard deviations describe evaluation variability within a checkpoint, rather than variability across independent training runs.

D.5 Downstream factor prediction

Following Jun et al. (2025), we evaluate the sample efficiency of downstream factor prediction on Shapes3D and MPI3D-toy using gradient boosted tree (GBT) classifiers. The representation encoder is frozen throughout evaluation. Ground-truth factor labels are used only to train and evaluate the downstream classifiers and do not update the representation model.

We use scikit-learn's `GradientBoostingClassifier` with its default hyperparameters. At each label budget $n \in \{100, 1000, 10000\}$, we train a separate classifier for each ground-truth factor using n labeled training samples. For each budget, we draw a new set of $5{,}000$ test samples.

Let Acc_n denote test accuracy averaged over factors when the classifiers are trained with n labeled samples. We report the sample-efficiency ratios $\mathrm{Acc}_{1000}/\mathrm{Acc}_{10000}$ and $\mathrm{Acc}_{100}/\mathrm{Acc}_{10000}$. We first average test accuracy over factors at each budget and then compute the ratios. Each ratio compares accuracies obtained using the same frozen encoder, measuring how much of the $10{,}000$-sample reference accuracy is retained with fewer labeled training samples.

For each DHP variant, we evaluate the final checkpoints from ten independent training runs. We compute the ratios separately for each checkpoint and report their means and standard deviations across runs. Baseline results in Table 6 are taken from Table 5 of Jun et al. (2025).

Table 9 reports the absolute test accuracies at the $10{,}000$-sample budget used as the denominators of these ratios. Both DHP variants achieve approximately 99.95% accuracy on Shapes3D. On MPI3D-toy, DHP-F and DHP-C achieve approximately 87.8% and 88.7%, respectively. These absolute accuracies provide context for interpreting the sample-efficiency ratios in Table 6.

Table 9: Absolute GBT test accuracy with $10{,}000$ labeled training examples. Accuracy is averaged over ground-truth factors and reported as mean $\pm$ standard deviation across ten independently trained models.

Method	Shapes3D	MPI3D-toy
DHP-F	0.999470 ± 0.000177	0.877709 ± 0.012803
DHP-C	0.999547 ± 0.000177	0.887457 ± 0.010228

E Qualitative Evaluation Details

Figure 3 illustrates latent swapping and traversals with DHP-C. Both operations modify the normalized representation before concept tokenization. Let $\widehat{y}(z;\omega)$ denote an image generated with representation z, where ω collects the sampling randomness, including the initial noise. We use deterministic DDIM sampling and fix the initial noise across all generated outputs within each MPI3D-toy column and each CelebA row. Thus, differences within each group arise from changes to the latent condition.

Latent swapping on MPI3D-toy. Let z^{src} and z^{tgt} denote the normalized representations of the source and target images, respectively. For attribute a, we select a latent dimension with index j_a and replace its value in the source representation with the corresponding value from the target representation:

$$z_i^{\mathrm{mix}} = \begin{cases} z_i^{\mathrm{tgt}}, & i = j_a, \\ z_i^{\mathrm{src}}, & i \neq j_a. \end{cases} \tag{E.1}$$

The generated output is $\widehat{y}(z^{\mathrm{mix}};\omega)$. Each swap starts from the original source representation and changes only the selected latent dimension. In Figure 3(a), the first two rows show the target and source images, respectively. The following four rows show transfers of object color, object size, camera height, and background color, respectively. For a given attribute, the same latent dimension j_a is used across all source–target pairs shown in the figure.

Latent traversals on CelebA. For a reference image with normalized representation z, we vary the value of a selected latent dimension i by constructing $z^{(i,a)} = z + ae_i$, where e_i is the ith standard basis vector. From left to right, the seven columns in Figure 3(b) use offsets $a \in \{-3, -2, -1, 0, 1, 2, 3\}$. Each offset is added to the encoded value z_i, so the central column ($a = 0$) uses the unmodified representation. All other latent dimensions retain their encoded values throughout each traversal. The generated outputs are $\widehat{y}(z^{(i,a)};\omega)$. Row labels indicate the main visible attribute changes in the selected examples.