# Surjective Pseudo-Invertible Neural Networks

**Yamit Ehrlich**[1] **Amit Arad**[1] **Nimrod Berman**[2] **Assaf Shocher**[1]
[1]Technion – Israel Institute of Technology [2]Ben-Gurion University

## Abstract

The Moore-Penrose Pseudo-inverse (PInv) is the fundamental tool for inverting linear operators. We propose a natural generalization to the non-linear regime and introduce Surjective Pseudo-invertible Neural Networks (SPNN): architectures that admit a tractable non-linear PInv by construction and satisfy the corresponding geometric properties. Building on this, we formalize Non-Linear Back-Projection (NLBP), the non-linear analogue of $x' = x + A^\dagger(y - Ax)$: an update that projects any sample to its closest state consistent with $f(x) = y$. Diffusion-based null-space projection revolutionized zero-shot solving of linear inverse problems via closed-form back-projection; NLBP extends this paradigm to non-linear learned "degradations" in the broad sense, spanning semantic mappings such as classification and object detection. The result is zero-shot inversion of complex degradations and precise semantic control over generative outputs without retraining the prior.

## 1 Introduction

The generalized inverse is foundational to signal processing and data analysis. In the linear regime, the Moore-Penrose pseudo-inverse (PInv) is the canonical solution for inverting singular operators, defined by the four Penrose Identities Penrose [1955]. Uniquely, it guarantees the **minimal-norm solution** among all valid inverses while providing the **optimal least-squares** approximation when no exact solution exists, and it enables exact consistency enforcement via null-space back-projection. In contrast, "inversion" in modern deep learning is typically approached as a regression task (autoencoders) or a probabilistic generation task (conditional diffusion). These approaches lack the structural guarantees of their linear counterparts, failing to satisfy reflexive consistency ($f(f^{-1}(f(x))) \neq f(x)$). Architectures that do enforce rigorous invertibility, such as Invertible Neural Networks Ardizzone et al. [2019] and Normalizing Flows Rezende and Mohamed [2015], Dinh et al. [2017], are constrained by strict bijectivity: they must preserve dimensionality, rendering them unsuitable for tasks where information is discarded, such as classification or detection.

We propose a natural generalization of PInv to the non-linear regime by strictly maintaining the first two Penrose identities ($gg^\dagger g = g$, $g^\dagger g g^\dagger = g^\dagger$), ensuring reflexive consistency. The remaining two require a linear adjoint with no canonical non-linear extension; we relax them rather than substituting them. This leaves the question of which generalized inverse to single out as the PInv. Unlike Gofer and Gilboa [2023], who pick the minimum-norm right-inverse $\|g^\dagger(y)\|$, we pick the solution induced by a Bijective Completion of the operator. Both coincide in the linear regime but diverge non-linearly; we argue in §4 that ours has natural properties (coordinate consistency, orthogonal back-projection).

We introduce **Surjective Pseudo-invertible Neural Networks (SPNN)**: surjective architectures where dimensionality is reduced and a **built-in Pseudo-Inverse** is provided by construction via a coupling mechanism whose residual degrees of freedom are resolved by a learned auxiliary network. We leverage SPNNs to introduce **Non-Linear Back-Projection (NLBP)**, enabling semantic control over generative outputs without retraining. Our main contributions are:

- **Non-linear Pseudo-inverse:** a natural definition justified by coordinate consistency.
- **SPNN:** to the best of our knowledge, the first deep architecture admitting a tractable, unique PInv satisfying this definition and the Penrose identities by construction.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

- **Non-Linear Back-Projection (NLBP):** we generalize Back-Projection to the non-linear regime, enabling solution of complex inverse problems that is zero-shot with respect to the generative prior.
- **SPNN-Linearizer:** we relax the Linearizer [Berman et al., 2025] to surjective $g$, yielding a dimension-reducing one-step flow-matching generator with improved FID.

## 2 Related work

**Deep invertible architectures.** Modern invertible networks trace back to the Feistel cipher Feistel [1973], revitalized by NICE Dinh et al. [2014] (additive coupling), RealNVP Dinh et al. [2017] (affine coupling), and Glow Kingma and Dhariwal [2018] (invertible $1\times1$ convs, often Cayley-parameterized Trockman and Kolter [2021]). Normalizing flows Rezende and Mohamed [2015] use these for density estimation; invertible nets have also been applied to inverse problems Ardizzone et al. [2019].

**Surjective and pseudo-invertible architectures.** Several lines relax strict bijectivity while retaining structured reverse mappings. SurVAE flows Nielsen et al. [2020] introduce surjective layers under a stochastic ELBO; PIE Beitler et al. [2021] proposes a pseudo-bijective likelihood-based autoencoder for dimensionality-reducing compression; PsI-NN Bolluyt and Comaniciu [2024] learns an approximate inverse via auxiliary regression. SPNN differs by singling out the *natural* PInv via a bijective completion (the non-linear analogue of the last two Penrose identities) and by deploying it for zero-shot inverse problems.

**Injective flows.** A parallel line makes flows dimension-reducing while keeping a tractable decoder: M-flows Brehmer and Cranmer [2020] and rectangular flows Caterini et al. [2021] learn a lower-dimensional manifold together with a mapping onto it. Both reduce dimension in one step, at the end or in the middle of the network.

**Generalized inversion of non-linear operators.** Gofer and Gilboa [2023] define non-linear pseudo-inversion via the *Best Approximate Solution* property (minimal-norm pre-image). Their definition and ours both reduce to the linear PInv but encode different non-linear geometric choices.

**Zero-shot inverse problems.** Pretrained diffusion priors are SOTA for $y = \mathcal{D}(x)$. *Null-space methods (linear):* DDRM Kawar et al. [2022], DDNM Wang et al. [2023], SNIPS Kawar et al. [2021] exploit the SVD of the degradation, projecting onto the measurement subspace; these are stable but limited to linear operations. *Gradient guidance (non-linear):* DPS Chung et al. [2023], PnP-Diffusion Chung et al. [2022] backpropagate $\nabla_x \|y - \mathcal{D}(x)\|^2$; requires differentiable $\mathcal{D}$ and is often unstable. ΠGDM Song et al. [2023] guides a frozen prior with a pseudo-inverse of the measurement model; there $\mathcal{D}^\dagger$ must be supplied, analytically for linear operators or by hand for particular non-linear ones such as JPEG decoding. NLBP unifies these by generalizing Iterative Back-Projection Irani and Peleg [1991], replacing the heuristic kernel with a learned, structural PInv.

## 3 Preliminaries

For a linear $A \in \mathbb{C}^{m\times n}$, the Moore-Penrose PInv $A^\dagger$ is the unique matrix satisfying the four Penrose identities:

$$AA^\dagger A = A, \quad A^\dagger AA^\dagger = A^\dagger, \quad (AA^\dagger)^* = AA^\dagger, \quad (A^\dagger A)^* = A^\dagger A. \tag{1}$$

The first two yield *reflexive consistency*; the last two make $AA^\dagger$ and $A^\dagger A$ orthogonal projections, so $x^* = A^\dagger y$ is the minimal-norm least-squares solution. From these one derives the linear back-projection update

$$x' = x + A^\dagger(y - Ax) \tag{2}$$

the cornerstone of null-space methods. Its iterative version Irani and Peleg [1991] reads $x_{k+1} = x_k + \lambda H(y - Ax_k)$, where $H \approx A^\dagger$ and $\lambda$ is a step size.

## 4 Non-linear pseudo-inverse

In the linear regime the Moore-Penrose PInv is the unique operator satisfying the four Penrose identities. Extending this non-linearly is non-trivial: the last two identities require a linear adjoint

with no canonical non-linear extension. We propose a definition based on *bijective completion*. Throughout, we work with $\mathcal{X} = \mathbb{R}^D$ and $\mathcal{Y} = \mathbb{R}^d$ equipped with the standard Euclidean structure, and with $\mathcal{Z} = \mathbb{R}^{D-d}$ as the latent coordinate space. We assume $g$ is a **smooth surjective submersion**. When $g$ is a learned surrogate for a target degradation $\mathcal{D}$ that is not itself a submersion (e.g., a real classifier with saturating logits), the guarantees apply to the surrogate $g \approx \mathcal{D}$, not to $\mathcal{D}$ itself.

**Definition 4.1** (Bijective completion)**.** Let $g : \mathcal{X} \to \mathcal{Y}$ be a smooth surjective *submersion* (i.e., $Dg(x)$ has full rank $d$ for every $x \in \mathcal{X}$). A **bijective completion** of $g$ is a diffeomorphism $G : \mathcal{X} \to \mathcal{Y} \times \mathcal{Z}$ of the form

$$G(x) = \begin{bmatrix} g(x) \\ q(x) \end{bmatrix}, \tag{3}$$

where $q : \mathcal{X} \to \mathcal{Z}$ is an auxiliary smooth map. We fix the convention $q(0) = 0$, which can always be arranged by translating $q$ alone (leaving $g$ untouched). The choice of origin as reference point in Definition 4.3 is therefore a convention rather than a substantive assumption: minimizing $\|G(x) - G(x_0)\|$ for any fixed $x_0$ defines an equivalent framework, recovered by translating $q$.

*Remark* 4.2 (Existence and non-uniqueness of $G$). Our SPNN provides an explicit elementary completion (Section 5); no topological machinery is needed for the construction we use. More generally, when $g$ is a smooth surjective submersion, the level sets $g^{-1}(y)$ are smooth $(D-d)$-manifolds (submersion theorem) and form a locally trivial fiber bundle over $\mathcal{Y}$; for contractible $\mathcal{Y}$ the bundle is globally trivializable, guaranteeing existence of $G$. $G$ is *not* unique: many choices of $q$ yield valid completions, each inducing its own pseudo-inverse, and this freedom is the non-linear analogue of choosing a basis for the null-space complement in the linear case.

**Definition 4.3** (Natural non-linear pseudo-inverse)**.** Given a surjective submersion $g$ and a bijective completion $G$, the **natural non-linear pseudo-inverse induced by** $G$, denoted $g^\dagger$, selects, for each $y$, the pre-image of $y$ whose $G$-image has minimal norm:

$$g^\dagger(y) = \operatorname*{argmin}_{x \in g^{-1}(y)} \|G(x)\|_2. \tag{4}$$

We assume the minimum in Eq. (4) is attained on every fiber. In the linear case $g = A$ with $G = [A; A^\perp]$, this reduces exactly to $A^\dagger$.

### 4.1 Theoretical justification

We justify this definition by demonstrating two properties shared with the linear pseudo-inverse, plus a structural property (orthogonal projection) that genuinely distinguishes our definition.

**1. Reflexive consistency.** Identity $gg^\dagger g = g$ follows from the pre-image constraint: for any $x$, $g^\dagger(g(x)) \in g^{-1}(g(x))$, hence $g(g^\dagger(g(x))) = g(x)$. Identity $g^\dagger g g^\dagger = g^\dagger$ follows immediately from the right-inverse property $gg^\dagger = \mathrm{Id}$. This aligns with Gofer and Gilboa [2023].

**2. Coordinate consistency.** Any reasonable extension of the Moore–Penrose inverse should commute with diffeomorphic re-parameterization: if $g = A \circ \phi$ for a linear surjection $A$ and a diffeomorphism $\phi$ with $\phi(0) = 0$, the inverse should reduce to the linear case in the $\phi$-coordinates, namely $g^\dagger(y) = \phi^{-1}(A^\dagger y)$. Our definition satisfies this criterion: the natural completion is $G(x) = \phi(x)$, so minimizing $\|G(x)\|$ over the pre-image yields exactly $\phi^{-1}(A^\dagger y)$. The minimum-$x$-norm definition of Gofer and Gilboa [2023] encodes a different geometric choice (minimization in the ambient coordinates rather than in the coordinates in which $g$ is linear) and consequently does *not* reduce to the linear case under non-trivial $\phi$.

**3. Orthogonal back-projection.** The linear back-projection update $x' = x + A^\dagger(y - Ax)$ provides a closed form for the closest vector that satisfies a linear consistency constraint. A non-linear PInv should admit an equivalent closed form. We propose such a form, *Non-Linear Back-Projection (NLBP)*, and show that it is doubly robust: it converges to a valid pre-image for *any* reflexive right-inverse $g^\dagger$ (Proposition 4.5), but it acts as an *orthogonal projection onto the nearest pre-image*, in the $G$-metric, only for the specific choice of $g^\dagger$ in Definition 4.3 (Proposition 4.6). The latter is the genuine extra structure our definition buys over Gofer & Gilboa.

**Definition 4.4** (Non-Linear Back-Projection)**.** Let $g$ be a surjective submersion with bijective completion $G$. The NLBP of $x$ towards $y$ is

$$x' = G^{-1}\big(G(x) - G(g^\dagger(g(x))) + G(g^\dagger(y))\big). \tag{5}$$

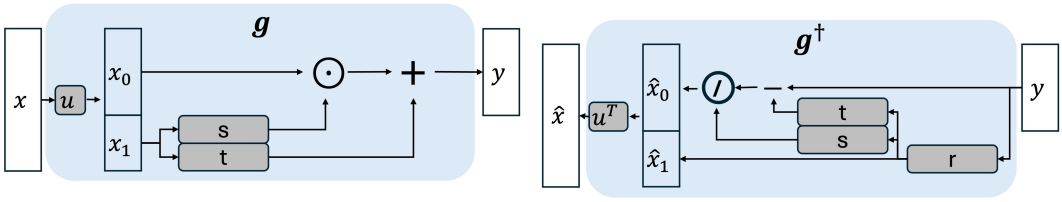



Figure 1: **The SPNN block.** *Left:* the forward map $g$ rotates the input by $U$, splits it into $(x_0, x_1)$, and computes $y = x_0 \odot s(x_1) + t(x_1)$. *Right:* the inverse predicts the discarded coordinates with $r$, reverses the coupling, and undoes the rotation to recover a consistent pre-image.

**Proposition 4.5** (Consistency)**.** *For any right-inverse $g^\dagger$ satisfying the reflexive Penrose identity $gg^\dagger g = g$, the update $x'$ in Eq.* (5) *lies on the pre-image of $y$: $g(x') = y$.*

*Proof.* By construction $G(x) = [g(x), q(x)]^\top$. The $y$-component of Eq. (5) is $g(x') = g(x) - g(g^\dagger(g(x))) + g(g^\dagger(y))$. By the first identity in (1) applied to $x$, the middle term equals $g(x)$. By surjectivity (right-inverse property of $g^\dagger$) the third term equals $y$. Hence $g(x') = y$. □

**Proposition 4.6** (Orthogonal projection)**.** *Assume $g^\dagger$ is the natural pseudo-inverse of Definition 4.3, and that $q \circ g^\dagger$ is constant on $\mathcal{Y}$ (so the minimizer's $q$-coordinate is independent of $y$; this holds for SPNN by construction, see Proposition 4.7). Then the NLBP update $x'$ is the orthogonal projection (in the $G$-metric) of $x$ onto the solution manifold $g^{-1}(y)$.*

*Proof.* Let $\Delta G = G(x') - G(x)$. From Eq. (5), the range component is $g(x') - g(x) = y - g(x)$ and the null-space component is $q(x') - q(x) = q(g^\dagger(y)) - q(g^\dagger(g(x)))$. By the constancy assumption, $q(g^\dagger(y)) = q(g^\dagger(g(x)))$, so the null-space component vanishes: $\Delta G = [y - g(x), 0]^\top$. Any tangent vector $v$ to the solution manifold satisfies $dG(v) = [0, dq]^\top$, hence $\langle \Delta G, dG(v) \rangle = 0$. □

**Proposition 4.7** (Exact constancy for SPNN)**.** *Let $g$ be an SPNN block with bijective completion $G(x) = [g(x); x_1]$ (i.e., $q = x_1$). Then the natural pseudo-inverse of Definition 4.3 satisfies $q \circ g^\dagger \equiv 0$. Consequently, the constancy hypothesis of Proposition 4.6 holds, and NLBP acts as exact orthogonal projection (in the $G$-metric) onto the solution manifold $g^{-1}(y)$.*

*Proof.* On each fiber $g^{-1}(y)$ we have $\|G(x)\|^2 = \|y\|^2 + \|x_1\|^2$, which separates into a fiber-fixed term $\|y\|^2$ and a term $\|x_1\|^2$ that is strictly convex in the unconstrained parameterization of the fiber by $x_1$. The minimum is at $x_1 = 0$ for every $y$, so $q \circ g^\dagger \equiv 0$. □

*Remark* 4.8 (Multi-block case). For a single block, $x_1 = 0$ shows any constant $r$ would suffice. In a multi-block SPNN, channels are mixed between blocks by the learned orthogonal $U$, so that each block introduces new discarded coordinates. The joint minimizer across the composition is then $y$-dependent, making $r$ non-trivial: §7.2 verifies that $r \equiv 0$ retains $gg^\dagger = \mathrm{Id}$ but breaks NLBP. More generally, Proposition 4.6's constancy assumption holds whenever $\|G\|^2$ splits into a fiber-fixed term and a $y$-independent fiber-varying term.

## 5 Surjective Pseudo-invertible Neural Network (SPNN)

SPNN maps $\mathbb{R}^D$ to $\mathbb{R}^d$, with $d < D$, while retaining an exact right-inverse. Its architecture guarantees consistency; training determines which of the possible pre-images the inverse selects.

### 5.1 Surjective pseudo-invertible block

Figure 1 illustrates the forward and inverse operations of an SPNN block. We first apply a learnable orthogonal transformation $U$ and split the result into retained and discarded coordinates. An affine coupling then defines the forward map:

$$[x_0; x_1] = Ux, \qquad x_0 \in \mathbb{R}^d, \quad x_1 \in \mathbb{R}^{D-d}, \qquad g(x) = x_0 \odot s(x_1) + t(x_1), \tag{6}$$

where $s, t : \mathbb{R}^{D-d} \to \mathbb{R}^d$ are neural networks and $s > 0$ elementwise. For any output $y$ and any $x_1$, this equation has a unique solution for $x_0$, making $g$ surjective. An auxiliary network $r : \mathbb{R}^d \to \mathbb{R}^{D-d}$ selects the discarded coordinates for the inverse:

$$\hat{x}_1 = r(y), \qquad \hat{x}_0 = \frac{y - t(\hat{x}_1)}{s(\hat{x}_1)}, \qquad g^\dagger(y) = U^\top[\hat{x}_0; \hat{x}_1]. \tag{7}$$

The corresponding bijective completion is $G(x) = [g(x); q(x)]$ with $q(x) = x_1$. It retains the coordinates discarded by $g$, so its inverse follows from the same coupling formula. With smooth $s, t$, it is a global diffeomorphism with $|\det DG(x)| = \prod_i s_i(x_1)$.

**Proposition 5.1** (Exact right-inverse and reflexive consistency)**.** *For any $r$ and any $s, t$ with $s > 0$, Eq.* (7) *satisfies $gg^\dagger = \mathrm{Id}$, and hence $gg^\dagger g = g$ and $g^\dagger gg^\dagger = g^\dagger$.*

*Proof.* Substitution gives $g(g^\dagger(y)) = (y - t(r(y))) + t(r(y)) = y$. The two identities follow by composition. □

Stacking these blocks preserves the right-inverse property by applying their inverses in reverse order. Thus $r$ is needed to select a useful pre-image, while consistency holds for every selection, including a constant one.

### 5.2 Learning the pseudo-inverse

Consistency alone does not distinguish among the points on a fiber $g^{-1}(y)$. For a fixed completion $G$, the natural pseudo-inverse selects the point with the smallest $G$-norm (Definition 4.3). This motivates

$$\mathcal{L}_{\text{natural}} = \mathbb{E}_y\big[\|G(g^\dagger(y))\|_2^2\big]. \tag{8}$$

However, $G$ is itself learned: a point with small $G$-norm need not be representative of the inputs that produce $y$. To give this selection a Euclidean reference, we relate its discarded coordinates to those of the data. The following result shows how reconstruction error controls this relation.

**Proposition 5.2** (Euclidean anchoring)**.** *Fix an SPNN block and its completion $q(x) = x_1$. For a conditional distribution with finite second moments, let $R(y) = q(g^\dagger(y))$, $m(y) = \mathbb{E}[q(x) \mid g(x) = y]$, and $\varepsilon(y) = \|R(y) - m(y)\|_2$. Writing $\mathrm{Var}(q \mid y) = \mathbb{E}[\|q(x) - m(y)\|_2^2 \mid g(x) = y]$, we have*

$$\begin{aligned} \varepsilon(y)^2 + \mathrm{Var}(q \mid y) &= \mathbb{E}[\|q(x) - R(y)\|_2^2 \mid g(x) = y] \\ &\le \mathbb{E}[\|x - g^\dagger(y)\|_2^2 \mid g(x) = y], \end{aligned} \tag{9}$$

*and*

$$\big|\|G(g^\dagger(y))\|_2^2 - \big(\|y\|_2^2 + \|m(y)\|_2^2\big)\big| \le 2\varepsilon(y)\|m(y)\|_2 + \varepsilon(y)^2. \tag{10}$$

This motivates the reconstruction objective

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_x\big[\|x - g^\dagger(g(x))\|_2^2\big]. \tag{11}$$

Its discarded-coordinate component controls $\varepsilon(y)$ and thus anchors the selected inverse to a Euclidean statistic of the data. For fixed $g$, the ideal fiberwise optimum of this loss is the point on the fiber closest to the conditional mean, which need not itself lie on the fiber; the derivation and the proof of Proposition 5.2 are in Appendix I.1.

Finally, we include $\mathcal{L}_{\text{surj}} = \mathbb{E}_y\|y - g(g^\dagger(y))\|_2^2$, which is zero in exact arithmetic and serves only as a guard against accumulated numerical error.

We optimize these objectives in two phases. In **Phase I**, the forward parameters $\theta_g = \{s, t, U\}$ and the inverse networks $r$ are trained jointly with

$$\mathcal{L}_{\text{I}} = \mathcal{L}_{\text{task}} + \lambda_{\text{rec}}^{\text{I}}\mathcal{L}_{\text{rec}} + \lambda_{\text{surj}}\mathcal{L}_{\text{surj}}. \tag{12}$$

The task loss learns the desired forward mapping. In **Phase II**, we freeze the forward map and its completion and optimize $r$ alone using

$$\mathcal{L}_{\text{II}} = \mathcal{L}_{\text{natural}} + \lambda_{\text{rec}}^{\text{II}}\mathcal{L}_{\text{rec}}. \tag{13}$$

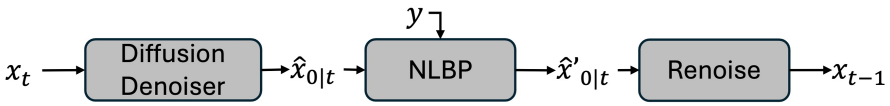



Figure 2: **NLBP-guided diffusion sampling step.** At each timestep, the denoiser produces $\hat{x}_{0|t}$, NLBP corrects it toward $y$, and the corrected $\hat{x}'_{0|t}$ is renoised to $x_{t-1}$.

Consistency remains exact, but balancing the two losses need not yield the exact natural pseudo-inverse; the resulting deviation from orthogonal back-projection is analyzed in Appendix I.2 and measured in Section 7.2.

The weights and task-specific settings are given in Appendix A. The SPNN-Linearizer uses a different training objective without a reconstruction term (Section 6.2); its denoising MSE directly measures Euclidean prediction error. Section 7.2 evaluates the contribution of the natural objective against reconstruction-only training.

### 5.3 Implementation details

We interleave `PixelUnshuffle` operations (factor $4$) with `ConvSPNN` blocks, with task-dependent depth. Each block splits its channels after a learned orthogonal transformation $U$ (Cayley transform for CelebA-HQ, a product of Householder reflections for ImageNet and detection). The $s, t, r$ networks use a U-Net-style `ConvMLP` architecture, with $s > 0$ enforced by $\exp(2\tanh(\cdot))$. Full specifications are in Appendix A.

## 6 Applications

### 6.1 Non-Linear Back-Projection for inverse problems

SPNNs enable inversion of **learned non-linear** "degradations", zero-shot with respect to the generative prior. In this sense, degradations include semantic mappings such as classification and object detection. We treat $\mathcal{D}$ as a black box, train an SPNN $g \approx \mathcal{D}$ via Section 5, and use NLBP to drive a pre-trained diffusion prior to samples consistent with a target measurement. SPNN supplies the forward operator and its right-inverse; the pre-trained diffusion model supplies the generative prior. Inspired by DDNM Wang et al. [2023], we integrate NLBP into a pre-trained sampling loop (Fig. 2). At each timestep $t$, given a clean estimate $\hat{x}_{0|t}$ and target $y$:

$$\hat{x}'_{0|t} = G^{-1}\big(G(\hat{x}_{0|t}) + \lambda\left[G(g^\dagger(y)) - G(g^\dagger(g(\hat{x}_{0|t})))\right]\big)\,, \tag{14}$$

where $\lambda \in [0, 1]$ is a guidance scale. In $G$-coordinates this becomes $g(\hat{x}'_{0|t}) = (1-\lambda)g(\hat{x}_{0|t}) + \lambda y$ and (under Proposition 4.6's constancy) $q(\hat{x}'_{0|t}) = q(\hat{x}_{0|t})$: semantic content is steered towards $y$ via linear interpolation, while null-space details from the prior remain invariant. This enables reconstruction from highly compressed inputs (e.g., class labels) without disrupting texture coherence. At $\lambda = 1$, the update enforces $g(\hat{x}'_{0|t}) = y$ for any $r$; null-coordinate preservation requires the constancy condition above. Full sampling details in App. C.

### 6.2 SPNN based Linearizer

The Linearizer framework Berman et al. [2025] composes a linear core between two invertible networks $g_x, g_y$ to obtain a map $f(x) := g_y^{-1}(A\, g_x(x))$ that, while non-linear over standard Euclidean vector spaces, is linear w.r.t. *lifted* vector-space structures on $\mathcal{X}, \mathcal{Y}$ defined by pulling vector addition and scalar multiplication back through $g_x, g_y$. E.g., on $\mathcal{X}$:

$$a \odot_x x := g_x^{-1}(a\, g_x(x))\,, \quad x_1 \oplus_x x_2 := g_x^{-1}(g_x(x_1) + g_x(x_2))\,. \tag{15}$$

We observe that this construction depends only on $g\, g^\dagger = \mathrm{Id}$, not on full invertibility, so it extends to surjective $g$ by replacing $g^{-1}$ with $g^\dagger$ above. The resulting operations make the $d$-dimensional submanifold $\mathrm{im}(g_x^\dagger) \subset \mathcal{X}$ a vector space canonically isomorphic to $(\mathcal{Y}, +, \cdot)$ via $g_x$. SPNNs satisfy the right-inverse property by construction (Proposition 5.1), with the additional benefit that $A$ acts on a low-dimensional latent rather than on the full input space.

**Proposition 6.1** (SPNN-Linearizer is linear in the lifted operations)**.** *Let $g_x, g_y$ be SPNNs and let $A$ be a linear operator. $f(x) := g_y^\dagger(A\, g_x(x))$ is linear over the lifted vector-spaces.*

The proof, together with the alternative lifting through the completion $G$, is in App. I.3.

**Iteration collapse.** When $g_x = g_y = g$ and we iterate $f$ with linear cores $A_1, A_2, \ldots, A_N$ (which may differ per step, e.g., a time-varying flow operator $A_t$), every intermediate $g\, g^\dagger$ telescopes to the identity, so the whole composition reduces to one matrix product wrapped by a single $g/g^\dagger$ pair:

$$f_{A_N} \circ \cdots \circ f_{A_1}(x) = g^\dagger(B\, g(x)), \qquad B := A_N \cdots A_1. \tag{16}$$

The $N$-step flow can therefore be precomputed once as a single matrix $B$ and applied with one forward/inverse SPNN pass. This is the basis of one-step flow-matching distillation, the main application demonstrated by Berman et al. [2025], now extended to dimension-reducing backbones. We instantiate this in §7.6.

# 7 Experiments

We evaluate the proposed framework on three settings: (i) **semantic restoration** from 40-dim attribute logits on CelebA-HQ Karras et al. [2018], with the 40 attribute annotations of Liu et al. [2015] and from 1000-dim class logits on ImageNet Russakovsky et al. [2015] (both at $256\times256$, classifier as the forward operator); (ii) **compositional generation** from object detection on Pascal VOC 2007 Everingham et al. [2010] (§7.5, CenterNet-style detector Zhou et al. [2019] as the forward operator); and (iii) **one-step flow-matching generation** on CelebA $64\times64$ via the SPNN-Linearizer (§7.6). The diffusion prior is an unconditional DDPM Dhariwal and Nichol [2021] for the classification settings and a class-conditional ADM Dhariwal and Nichol [2021] for the detection setting; architecture details in Appendix A.

SPNN is trained for classification, detection, and the Linearizer task independently of a diffusion prior. The CelebA-HQ classifier achieves $89.1\%$ test accuracy (App. J); the Linearizer is evaluated independently in Table 3, and Section 7.2 examines the inverse selection and training objectives.

## 7.1 Semantic reconstruction

We reconstruct an image given *only* its classification logit vector. We extract $y_{\text{GT}} = g(x_{\text{orig}})$ from an unseen test image and run NLBP-guided diffusion (Eq. 14) with $y_{\text{GT}}$ fixed. Fig. 3 shows CelebA-HQ reconstructions matching 40-dim attribute logits (left) and ImageNet matching 1000-dim class logits (right). The reconstruction is one sample from the classifier's pre-image manifold. To quantify fidelity on CelebA-HQ, we computed reconstruction accuracy over 100 test samples (Fig. 5). Mean binary agreement is $92.3\%$ across all 40 attributes. Structural attributes (*Eyeglasses*, *Hats*, *Neckties*) reconstruct near-perfectly ($> 97\%$). Error concentrates in ambiguous features (*High Cheekbones*, *Arched Eyebrows*), where the classifier's own uncertainty is also higher.

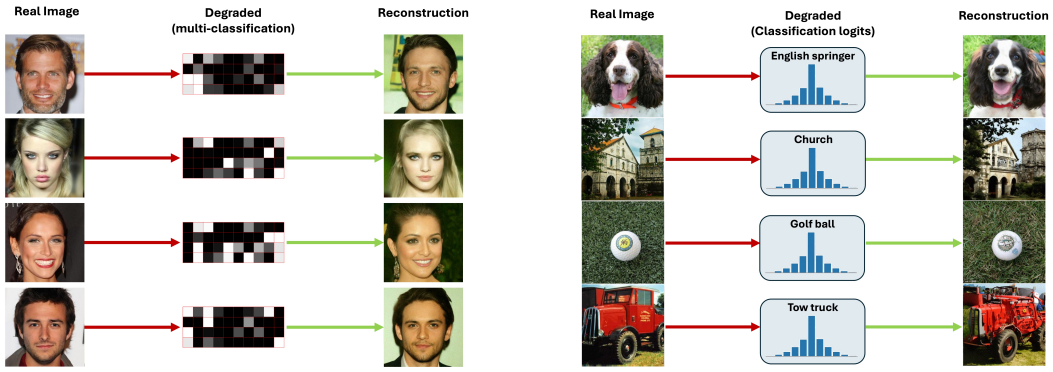



Figure 3: **Reconstruction from semantics.** Each row: clean image / measurement / NLBP-guided reconstruction. *Left:* CelebA-HQ, where the measurement is a 40-dim attribute logit vector (shown as a binary grid for illustration). *Right:* ImageNet, where the measurement is the full 1000-dim class logit vector (illustrated by the top-1 class and a stylized peaked distribution). The reconstruction is one sample from the classifier's pre-image manifold; it need not match the original pixel-wise, only in the logit measurement. More examples in App. F.

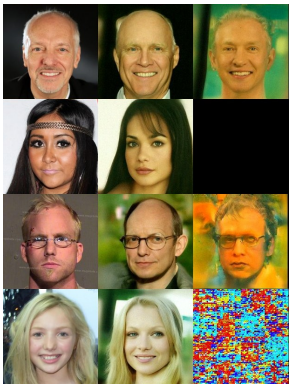


Figure 4: **Ablation of $r$.** With $r \equiv 0$ the NLBP-guided sampler diverges; with trained $r$ it does not.

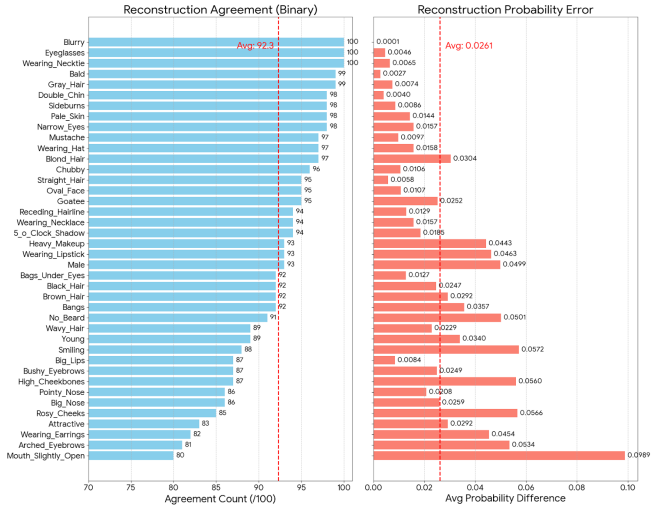



Figure 5: **Per-attribute reconstruction fidelity on CelebA-HQ.** Per-attribute binary agreement (left bars) and mean absolute error in probability space (right bars) on 100 test samples. Mean binary agreement is 92.3% across all 40 attributes. Numerical values in Appendix E.



Figure 6: **DPS outputs for ImageNet class 0** (tench, *Tinca tinca*). These samples receive confident target-class predictions despite lacking a photorealistic fish. Further classes are shown in App. K.

### 7.2 Ablations

We replace $r$ with the constant predictor $r \equiv 0$, preserving the right-inverse identity $gg^\dagger = \mathrm{Id}$. All other components ($g$, NLBP sampler, diffusion prior) are kept identical. Fig. 4 shows the result on four test inputs. With $r \equiv 0$, the sampler diverges in different ways across inputs (color drift, black-out, high-frequency noise). Thus, right-inverse consistency alone does not ensure useful guidance.

Retraining without Phase II, so that $r$ is learned from reconstruction alone, worsens CelebA-HQ FID from 45.1 to 50.6. Removing $\mathcal{L}_{\mathrm{surj}}$ leaves performance unchanged, consistent with its numerical role and typical value of $10^{-6}$. The SPNN-Linearizer uses no reconstruction loss; its denoising MSE directly measures prediction error in image space (Section 7.6). Measurements of the trained NLBP update's deviation from orthogonality are given in App. I.2.

### 7.3 Comparison to zero-shot baselines

We compare against DPS Chung et al. [2023], as a zero-shot baseline with pre-trained diffusion. For DPS we train a ResNet classifier matched to our SPNN; both use the same pre-trained DDPM Dhariwal and Nichol [2021]. To avoid adversarial self-evaluation, agreement is reported against three classifiers: the method's own ("self"), the other's ("cross"), and an independent EfficientNet ("third-party"). The CelebA-HQ comparison uses 6,000 samples. We additionally include a (non-comparable)

Table 1: **Zero-shot semantic reconstruction on CelebA-HQ** (6,000 **samples).** Lower is better for FID/KID/Err; higher for Agree. "self"/"cross": method's own / other's classifier (SPNN, ResNet); "third-party": independently-trained EfficientNet. $z$: DPS step size. Comparison to text-to-image priors (SD 1.5, SDXL) is in App. G.

| Method | Guidance | FID↓ | KID↓ | Agree self↑ | Agree cross↑ | Agree 3rd↑ | Err self↓ | Err cross↓ | s/img |
|---|---|---|---|---|---|---|---|---|---|
| NLBP+SPNN (ours) | NLBP | **45.1** | **.033** | **92.5%** | **88.7%** | **87.4%** | **0.073** | **0.123** | 5.7 |
| DPS+ResNet | DPS ($z$=1.0) | 185.4 | .166 | 92.2% | 73.8% | 73.2% | 0.077 | 0.273 | 1.7 |
| DPS+ResNet | DPS ($z$=0.5) | 101.4 | .071 | 91.7% | 79.8% | 80.2% | 0.083 | 0.200 | 1.7 |
| DPS+ResNet | DPS ($z$=0.1) | 120.0 | .085 | 90.4% | 81.3% | 84.1% | 0.098 | 0.186 | 1.7 |

| Method | FID↓ | KID↓ | Agree↑ |
|---|---|---|---|
| NLBP (ours) | **30.65** | **.021** | 49% |
| DPS ($z$=1.0) | 54.00 | .034 | 71.2% |
| DPS ($z$=0.5) | 50.42 | .033 | **71.6%** |
| DPS ($z$=0.1) | 81.78 | .068 | 38.9% |

Table 2: **Zero-shot reconstruction on ImageNet** (1000-dim logits, 256×256; protocol as Table 1, over 10,000 samples). Agree: overlap with target's top-5 under an independent ConvNeXt. Examples in Fig. 6 illustrate that high classifier agreement need not imply visual fidelity to the target class.

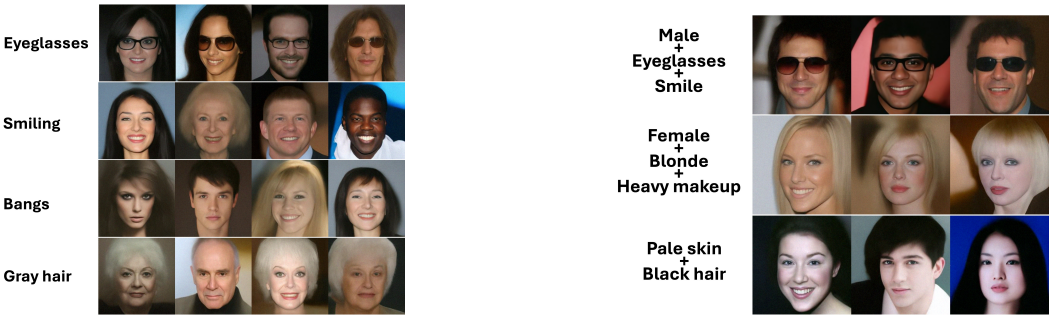



Figure 7: **Attribute-controlled generation.** *Left:* enforcing one attribute (e.g., "Eyeglasses") produces diverse samples adhering to that constraint while freely hallucinating the rest. *Right:* simultaneously enforcing several bits of $y$ steers generation to the intersection of semantic manifolds.

reference comparison against large text-to-image priors (Stable Diffusion 1.5 Rombach et al. [2022], SDXL Podell et al. [2023]) in App. G. Results are in Tables 1 (CelebA-HQ) and 2 (ImageNet). Our method beats DPS by a large FID/KID margin on both, and on CelebA-HQ in agreement under all three classifiers. DPS does win on ImageNet top-5 agreement, but Fig. 6 illustrates that its outputs can classify into the target class without depicting it. On CelebA-HQ, DPS also exhibits a self-vs-independent agreement gap ($\sim$91% vs. $\sim$73–84%). These observations motivate reporting classifier agreement alongside distributional fidelity. In the separate text-to-image comparison (App. G), our method also achieves better FID/KID and independent attribute agreement. App. J reports per-image variability, bootstrap confidence intervals, paired significance tests, PSNR, and DPS results for detection.

### 7.4 Attribute-controlled generation

We also control specific attributes without fixing the full semantic vector. Instead of a static $y$, at each step we compute $y_{\text{cur}} = g(\hat{x}_{0|t})$ and use a dynamic target that agrees with $y_{\text{cur}}$ except at the controlled index $n$, where $y_{\text{target}}[n] = \mu_n + 2\sigma_n$ from the empirical statistics. The latent-space update is then sparse, so the prior freely hallucinates the remaining attributes (Fig. 7, left); fixing several bits steers generation to the intersection of semantic manifolds (right). Here, attributes constrain diffusion sampling directly, rather than edits to a pre-trained generator's latent code Shen et al. [2020], Härkönen et al. [2020] (App. C).

### 7.5 Compositional generation from object detection

We use object detection as the forward operator: given a real image with a known object, generate a new image whose detector output matches at the original bounding box (same class, same location) while the rest of the scene is freely sampled. The operator is a CenterNet-style head Zhou et al. [2019] on top of our ImageNet SPNN backbone, producing per-class center heatmaps, sub-pixel offsets, and box dimensions ($24 \times 64 \times 64$ at stride 4). The prior is the class-conditional 256×256 ADM of Dhariwal and Nichol [2021]; per image we condition on the ImageNet class closest to the VOC class. NLBP is applied only for $t > 650$ (the noisier $\sim$35% of the schedule), and after each NLBP step we Gaussian-mask the corrected $\hat{x}_{0|t}$ around the bounding box, so the constraint is enforced where the box is and the prior renders the rest. Fig. 8 shows that the object class and location are preserved while the rest of the scene is freshly sampled. Constraining the full detector output without masking instead defines a semantic reconstruction task. On all 4,952 VOC2007 test images, its FID is 12.81, versus 70.17 for compositional generation; these settings impose different constraints (App. J). Full pipeline in App. D.



Figure 8: **Compositional generation from object detection.** *Top:* VOC2007 test images with detection bounding boxes and class labels (person, horse, car, bus, chair, aeroplane). *Bottom:* NLBP-guided samples projected onto the set of images that the detector maps to the same output (top): the object class and location are preserved while the rest of the scene is freshly sampled.

### 7.6 One-step generation as an SPNN-Linearizer

We instantiate the SPNN-Linearizer of Proposition 6.1 as a one-step flow-matching generator on CelebA Berman et al. [2025] (Linearizer). The forward $g$ is an SPNN; $A_t$ is a low-rank matrix from an MLP. After training, the $N$-step Euler trajectory collapses to $B = \prod_i (I + \Delta t\, A_{t_i})$, and generation is $\hat{x}_1 = g^\dagger(B\, g(x_0))$. We outperform the bijective baseline at every operating point (Table 3); qualitative samples in Fig. 11, with the largest gap at one-step (325 vs. 405). Dimension reduction makes the linear core smaller: $A_t$ acts on a 128-dim latent rather than 12,288-dim, shrinking from 101.5M to 1.7M parameters (59×). Total parameters drop from 102M to 74M, $B$-build from 6.21 to 0.14 s, and inference from 0.55 to 0.34 ms. The construction also applies to other surjective architectures with a deterministic right-inverse; this comparison evaluates SPNN against the bijective baseline.

Table 3: **SPNN-Linearizer on CelebA** $64\times64$**.** Generation quality (FID, left block) and efficiency of the linear core (right block) vs. the bijective baseline Berman et al. [2025]. "$N\to1$" collapses an $N$-step Euler trajectory $B = \prod_i (I + \Delta t\, A_{t_i})$ into a single matrix and runs one forward/inverse SPNN pass at inference.

| | **FID ↓** | | | | | **Efficiency ↓** | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 step | 10 steps | 100 steps | 100→1 | 1000→1 | Dim | $A_t$ params | Total params | Build $B$ | Inf./img |
| Ours | **325** | **120.9** | **110.4** | **124.5** | **113.2** | **128** | **1.7M** | **74.0M** | **0.14** s | **0.345** ms |
| Linearizer | 405 | 152 | 127 | 131 | 123 | 12,288 | 101.5M | 102.4M | 6.21 s | 0.552 ms |

## 8 Limitations

Two caveats accompany our framework. First, Definition 4.1 requires $g$ to be a smooth surjective submersion. With smooth $s, t$, the SPNN architecture satisfies this by construction ($s > 0$), but real-world degradations $\mathcal{D}$ that we model with $g$ (e.g., classifiers with saturating logits, hard quantization) are not themselves submersions; the guarantees apply to the surrogate $g \approx \mathcal{D}$, not to $\mathcal{D}$. Second, exact orthogonal back-projection in the $G$-metric requires $q \circ g^\dagger$ to be constant (Proposition 4.6). Training need not achieve this condition, although $gg^\dagger = \mathrm{Id}$ remains exact; App. I.2 analyzes and measures the resulting deviation.

## Acknowledgments and Disclosure of Funding

A.S. is supported by the Chaya Career Advancement Chair. We thank an anonymous reviewer for pointing out that the reconstruction optimum is constrained to the fiber.

## References

Lynton Ardizzone, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. In *ICLR*, 2019.

Jan Jetze Beitler, Ivan Sosnovik, and Arnold Smeulders. Pie: Pseudo-invertible encoder. *arXiv preprint arXiv:2111.00619*, 2021.

Nimrod Berman, Assaf Hallak, and Assaf Shocher. Who said neural networks aren't linear? *arXiv preprint arXiv:2510.08570*, 2025.

Elijah D. Bolluyt and Cristina Comaniciu. Pseudoinvertible neural networks. *IEEE Transactions on Artificial Intelligence*, 5(2):602–612, 2024.

Johann Brehmer and Kyle Cranmer. Flows for simultaneous manifold learning and density estimation. In *NeurIPS*, 2020.

Anthony L Caterini, Gabriel Loaiza-Ganem, Geoff Pleiss, and John P Cunningham. Rectangular flows for manifold learning. In *NeurIPS*, 2021.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In *NeurIPS*, 2022.

Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *ICLR*, 2023.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. In *NeurIPS*, 2021.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *ICLR*, 2017.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88 (2):303–338, 2010.

Horst Feistel. Cryptography and computer privacy. *Scientific American*, 228(5):15–23, 1973.

Eyal Gofer and Guy Gilboa. Generalized inversion of nonlinear operators. *arXiv preprint arXiv:2111.10755*, 2023.

Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. GANSpace: Discovering interpretable GAN controls. In *NeurIPS*, 2020.

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

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018.

Bahjat Kawar, Gregory Vaksman, and Michael Elad. SNIPS: Solving noisy inverse problems stochastically. In *NeurIPS*, 2021.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *NeurIPS*, 2022.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018.

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

Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. SurVAE flows: Surjections to bridge the gap between VAEs and flows. In *NeurIPS*, 2020.

Roger Penrose. A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society*, 51(3):406–413, 1955.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3): 211–252, 2015.

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of GANs for semantic face editing. In *CVPR*, 2020.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *ICLR*, 2023.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the Cayley transform. In *ICLR*, 2021.

Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *ICLR*, 2023.

Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the contributions: a natural non-linear pseudo-inverse via bijective completion, an architecture (SPNN) that implements it, and a Non-Linear Back-Projection used for zero-shot semantic inverse problems. See Sections 4, 5, 6, 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.

- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 8.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All theoretical results (Definitions 4.1, 4.3; Propositions 4.5, 4.6, 5.1; Remarks 4.2, 4.8) are stated with explicit assumptions and proved in the main text.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Full architecture, training setup, hyper-parameters, and sampling schedule are provided in Section 5 and Appendices C–B.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We use the public CelebA-HQ dataset and a standard pre-trained unconditional DDPM; training and evaluation protocols are fully documented. Code is being prepared for release on the project page.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Optimizer, learning rate, batch size, number of epochs, loss weights and sampling parameters are listed in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Quantitative comparisons in Table 1 are averaged over 6,000 generated samples of Celeb-HQ and 10,000 of ImageNet. Per-attribute reconstruction is reported over 100 test samples (Fig. 5, Appendix E).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Per-image sampling time for each method is reported in Table 1; training configuration is summarized in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts - including risk of misuse of attribute-editing methods for synthetic media and the inheritance of dataset biases - are discussed in Appendix L.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release high-risk pretrained models; the method relies on a standard pre-trained DDPM and a classifier, applied to the public CelebA-HQ dataset.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite all external assets used (CelebA-HQ Karras et al. [2018], ImageNet Russakovsky et al. [2015], Pascal VOC 2007 Everingham et al. [2010], DDPM backbone Dhariwal and Nichol [2021], Stable Diffusion 1.5 Rombach et al. [2022], SDXL Podell et al. [2023], DPS Chung et al. [2023]) and use them under their respective public licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

Justification: No new dataset is released; model code will be made available along with a README documenting training and evaluation.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: No crowdsourcing or human-subject experiments were conducted.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No human-subjects research was conducted; all imagery is from the publicly released CelebA-HQ dataset.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs were used only as a standard writing aid (proof-reading and rephrasing) and did not contribute to any core methodological or scientific content.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.

# A Detailed SPNN architecture

This appendix gives the full per-stage channel configuration of our $256\times256$ SPNN classifiers. The network alternates `PixelUnshuffle` (factor $4$) stages with `ConvSPNN` coupling blocks. Table 4 shows the configuration for the CelebA-HQ classifier ($40$ attribute logits): a single `ConvSPNN` block at each spatial scale. The ImageNet classifier (Table 5) uses the same building blocks but stacks two `ConvSPNN` blocks at each of the early scales, gradually reducing the channel count in two halving steps rather than a single drop. This deeper schedule is needed to accommodate the much higher-dimensional output ($1000$ class logits vs. $40$ attribute logits) without overly aggressive per-block compression.

Table 4: Channel layout of our CelebA-HQ SPNN (input $3 \times 256 \times 256$). Each `PixelUnshuffle`($r$) reduces spatial resolution by $r$ and multiplies channels by $r^2$. Each `ConvSPNN`($c_{\text{in}} \to c_{\text{out}}$) retains $c_{\text{out}}$ "content" channels and discards $c_{\text{in}} - c_{\text{out}}$ "null-space" channels. All blocks use orthogonal mixing ($U$), hidden width $128$ for the internal `ConvMLP`, and $s$-activation $\exp(2\tanh(\cdot))$.

| Stage | Block | Output shape | Notes |
|---|---|---|---|
| Input | – | $[3, 256, 256]$ | RGB image |
| Unshuffle (1) | `PixelUnshuffle`(4) | $[48, 64, 64]$ | spatial $\times \frac{1}{4}$ |
| Block 1 | `ConvSPNN`($48 \to 12$) | $[12, 64, 64]$ | U-Net ConvMLP ($128\to256$) |
| Unshuffle (2) | `PixelUnshuffle`(4) | $[192, 16, 16]$ | |
| Block 2 | `ConvSPNN`($192 \to 48$) | $[48, 16, 16]$ | U-Net ConvMLP ($256\to512$) |
| Unshuffle (3) | `PixelUnshuffle`(4) | $[768, 4, 4]$ | |
| Block 3 | `ConvSPNN`($768 \to 192$) | $[192, 4, 4]$ | U-Net ConvMLP ($512\to1024$) |
| Unshuffle (4) | `PixelUnshuffle`(4) | $[3072, 1, 1]$ | collapse spatial |
| Block 4 | `ConvSPNN`($3072 \to 1024$) | $[1024, 1, 1]$ | stack of $1\times1$ convs |
| Block 5 | `ConvSPNN`($1024 \to 40$) | $[40, 1, 1]$ | output logits (40 attrs) |

Table 5: Channel layout of our ImageNet SPNN (input $3 \times 256 \times 256$). The same building blocks as the CelebA-HQ configuration (Table 4), but with two `ConvSPNN` blocks at each of the early spatial scales (gradual channel reduction by halves) rather than a single block per scale.

| Stage | Block | Output shape | Notes |
|---|---|---|---|
| Input | – | $[3, 256, 256]$ | RGB image |
| Unshuffle (1) | `PixelUnshuffle`(4) | $[48, 64, 64]$ | spatial $\times \frac{1}{4}$ |
| Block 1 | `ConvSPNN`($48 \to 24$) | $[24, 64, 64]$ | U-Net ConvMLP |
| Block 2 | `ConvSPNN`($24 \to 12$) | $[12, 64, 64]$ | U-Net ConvMLP |
| Unshuffle (2) | `PixelUnshuffle`(4) | $[192, 16, 16]$ | |
| Block 3 | `ConvSPNN`($192 \to 96$) | $[96, 16, 16]$ | U-Net ConvMLP |
| Block 4 | `ConvSPNN`($96 \to 48$) | $[48, 16, 16]$ | U-Net ConvMLP |
| Unshuffle (3) | `PixelUnshuffle`(4) | $[768, 4, 4]$ | |
| Block 5 | `ConvSPNN`($768 \to 192$) | $[192, 4, 4]$ | U-Net ConvMLP |
| Unshuffle (4) | `PixelUnshuffle`(4) | $[3072, 1, 1]$ | collapse spatial |
| Block 6 | `ConvSPNN`($3072 \to 1024$) | $[1024, 1, 1]$ | stack of $1\times1$ convs |
| Block 7 | `ConvSPNN`($1024 \to 1000$) | $[1000, 1, 1]$ | output logits (1000 classes) |

`ConvMLP` **(sub-network for $s$, $t$, $r$).** For blocks where the feature map is larger than $1\times1$, $s$, $t$, and $r$ each use a U-Net-style `ConvMLP`: two $3\times3$ convolutions at the feature resolution, a stride-2 down-step, two $3\times3$ convolutions at the halved resolution, a stride-2 transposed-conv up-step, and a final $3\times3$ projection to the output channels. All activations are ReLU. The final projection is initialized to zero so the layer starts as identity. For the terminal $1\times1$ blocks we replace the U-Net with a 3-layer $1\times1$ convolution stack ($c_{\text{in}} \to 2c_{\text{in}} \to 1.5c_{\text{in}} \to c_{\text{out}}$, ReLU between layers, last layer zero-initialized).

**Orthogonal mixing.** Each `ConvSPNN` block is preceded by a learnable orthogonal $U \in \mathbb{R}^{C\times C}$ applied as a $1\times1$ convolution across channels. $U$ is parameterized either via the Cayley transform $U = (I - A)(I + A)^{-1}$ for a learned skew-symmetric $A = B - B^\top$ (CelebA-HQ), or as a product of

$k$=8 Householder reflections $U = H_k \cdots H_1$, $H_i = I - 2v_i v_i^\top / \|v_i\|^2$ (ImageNet, detection). Both guarantee $U^\top U = I$ by construction.

**Training-loss weights.** For CelebA-HQ, Phase I uses BCE classification loss ($\lambda_{\text{task}} = 1$), right-inverse loss ($\lambda_{\text{surj}} = 40$), and image-reconstruction loss ($\lambda_{\text{rec}} = 40$). These weights keep the task and reconstruction terms balanced: in the measured Phase I windows the scaled task gradient is $2.4$–$3.7\times$ the reconstruction gradient, while the surjectivity term remains a small numerical guard (Table 16). Phase II freezes the forward path and optimizes only $r$ with $\lambda_{\text{natural}} = 0.3$, $\lambda_{r_\text{surj}} = 1$, $\lambda_{r_\text{rec}} = 1$. Full training hyper-parameters appear in the next appendix (Table 6). For ImageNet, the task loss is softmax cross-entropy over the $1000$ classes (training details in Table 7).

# B Architecture and hyper-parameters

Table 6: SPNN-Celeb-HQ architecture and training configuration.

| Category | Setting |
|---|---|
| **Model architecture** | |
| Input resolution | $256 \times 256$ RGB |
| Number of SPNN blocks | 5 |
| Orthogonal mixing | Cayley $1\times1$ convolutions |
| Latent partitioning | Channel-wise |
| Activation (scale) | $\tanh(\cdot) \times 2.0$, then $\exp$ |
| $s$, $t$, $r$ networks | `ConvMLP`, hidden width 128 (see Appendix A) |
| Number of attributes | 40 (CelebA-HQ) |
| **Training setup** | |
| Dataset | CelebA-HQ |
| Batch size | 256 |
| Optimizer | Adam |
| Learning rate | $2 \times 10^{-4}$ |
| Adam $(\beta_1, \beta_2)$ | $(0.9, 0.999)$ |
| Epochs (Phase I) | 15 |
| Gradient clipping | 1.0 |
| Warmup iterations | 200 |
| Precision | FP32 |
| Random seed | 556 |
| **Loss weights** | |
| $\lambda_{\text{task}}$ | 1.0 |
| $\lambda_{\text{surj}}$ (Phase I) | 40.0 |
| $\lambda_{\text{rec}}$ (Phase I) | 40.0 |
| $\lambda_{\text{natural}}$ (Phase II) | 0.3 |
| $\lambda_{r_\text{surj}}$ (Phase II) | 1.0 |
| $\lambda_{r_\text{rec}}$ (Phase II) | 1.0 |
| $r$-optimization epochs | 50 |
| $r$-optimization learning rate | $1 \times 10^{-4}$ |
| $r$-optimization batch size | 256 |

Table 7: SPNN-ImageNet architecture and training configuration.

| Category | Setting |
|---|---|
| **Model architecture** | |
| Input resolution | $256 \times 256$ RGB |
| Number of `ConvSPNN` blocks | 7 |
| Orthogonal mixing | Householder $1\times1$ convolutions |
| Latent partitioning | Channel-wise |
| Activation (scale) | $\tanh(\cdot) \times 1.0$, then $\exp$ |
| $s$, $t$, $r$ networks | `ConvMLP`, hidden width 256 (see Appendix A) |
| Number of classes | 1000 (ImageNet ILSVRC-2012) |
| **Training setup** | |
| Dataset | ImageNet ILSVRC-2012 Russakovsky et al. [2015] (1.28M train, 50K val) |
| Hardware | 8 GPUs (DistributedDataParallel, NCCL) |
| Batch size | 512 total (64 per GPU) |
| Data-loader workers | 64 |
| Optimizer | SGD |
| Learning rate (initial) | 0.1 |
| Momentum | 0.9 |
| Weight decay | $1 \times 10^{-4}$ |
| LR schedule | Cosine annealing with linear warmup |
| Warmup epochs | 5 |
| Epochs (Phase I) | 100 |
| Gradient clipping (max-norm) | 1.0 |
| Precision | FP32 |
| Train augmentation | RandomResizedCrop(256), RandomHorizontalFlip |
| Test augmentation | Resize(292), CenterCrop(256) |
| Normalization | ImageNet stats ($\mu$=[0.485, 0.456, 0.406], $\sigma$=[0.229, 0.224, 0.225]) |
| **Loss weights** | |
| $\lambda_{\text{task}}$ | 1.0 |
| $\lambda_{\text{surj}}$ (Phase I) | 5.0 |
| $\lambda_{\text{rec}}$ (Phase I) | 5.0 |
| $\lambda_{\text{natural}}$ (Phase II) | 0.3 |
| $\lambda_{r_\text{surj}}$ (Phase II) | 1.0 |
| $\lambda_{r_\text{rec}}$ (Phase II) | 1.0 |
| $r$-optimization epochs | 50 |
| $r$-optimization learning rate | $1 \times 10^{-4}$ |
| $r$-optimization batch size | 512 |

Table 8: **SPNN-CenterNet (object detection) architecture and training configuration.** We reuse the early blocks of our SPNN-ImageNet classifier (Table 7) as the detection backbone and add a single deeper detection-head block; the full chain $A$ is then fine-tuned end-to-end on Pascal VOC 2007 (Phase I), after which the $r$-nets are optimized via the natural-PInv objective with the forward path frozen (Phase II).

| Category | Setting |
|---|---|
| **Model architecture** | |
| Input resolution | $256 \times 256$ RGB |
| Output | $24 \times 64 \times 64$ at stride 4 (20-class hmap + 2 offset + 2 box-size) |
| Backbone | 2 `ConvSPNN` blocks ($12 \to 8$ at $128 \times 128$, $32 \to 28$ at $64 \times 64$) interleaved with `PixelUnshuffle`($r{=}2$); init from SPNN-ImageNet (Table 7) |
| Detection head | 1 deep `ConvSPNN` block ($28 \to 24$ at $64 \times 64$), 3-level U-net $s/t/r$ ConvMLPs, hidden 128 |
| Orthogonal mixing | Householder $1 \times 1$ convolutions |
| $s$ activation | $\tanh(\cdot) \times 1.0$, then $\exp$ |
| **Phase I (forward training: backbone + head + adapters jointly)** | |
| Dataset | Pascal VOC 2007 Everingham et al. [2010] (trainval for training, test for evaluation) |
| Hardware | 8 GPUs (DistributedDataParallel, NCCL) |
| Batch size | 64 total (8 per GPU) |
| Data-loader workers | 4 per GPU |
| Optimizer | Adam |
| Learning rate (initial) | $5 \times 10^{-4}$ |
| LR schedule | MultiStepLR, $\gamma = 0.1$ at epochs $\{90, 120\}$ |
| Epochs | 250 |
| Validation interval | every 5 epochs |
| Augmentation | Standard CenterNet (random crop, flip, color jitter, photometric distortion) |
| Normalization | ImageNet stats, BGR channel order (CenterNet convention) |
| **Phase I loss weights** | |
| $\lambda_{\text{hmap}}$ (focal loss) | 1.0 |
| $\lambda_{\text{regs}}$ (L1, sub-pixel offset) | 1.0 |
| $\lambda_{\text{wh}}$ (L1, box dimensions) | 0.1 |
| $\lambda_{\text{img_rec}}$ ($\|g^{\dagger}(Z(g(x))) - x\|^2$, $Z$ = peak sparsifier, App. D) | 10.0 |
| **Phase II ($r$-net only, forward path frozen)** | |
| Trainable parameters | per-block $r$ networks (all of $\{s, t, U\}$ and adapters frozen) |
| Batch size | 64 total (8 per GPU, DDP) |
| Optimizer | Adam |
| Learning rate | $1 \times 10^{-3}$ |
| Epochs | 200 |
| Gradient clipping (max-norm) | 1.0 |
| $\lambda_{r_\text{norm}}$ ($\|z(g^{\dagger}(Z(g(x)))) - z(0)\|^2$, latent regularizer) | 1.0 |
| $\lambda_{r_\text{rec}}$ ($\|g^{\dagger}(Z(g(x))) - x\|^2$, image reconstruction) | 10.0 |

## C NLBP implementation details

We implement NLBP within a standard diffusion sampling framework, adopting the structure of DDNM Wang et al. [2023].

**Diffusion backbone.** We use a pre-trained unconditional DDPM Dhariwal and Nichol [2021] with $T = 1000$ timesteps. The reverse process follows the standard noise schedule $\beta_t$. At each step $t$, we estimate $\hat{x}_{0|t}$ via the denoising network and apply the NLBP correction before re-sampling $x_{t-1}$.

**Reconstruction strategy.** We do not apply NLBP from the very first sampling step: in the early steps the predicted image is too blurry and noisy and is out of distribution for the degradation network. Over the diffusion schedule (T=1000 timesteps, sampled at 100 strides), we start NLBP at step $800$ for semantic reconstruction and at step $500$ for attribute-controlled generation.

**Adaptive $\lambda$.** For single and multi-attribute editing (Fig. 7), we scale $\lambda$ dynamically with the discrepancy between current and target. Let $\delta_n = |y_{\text{cur}}[n] - y_{\text{target}}[n]|$. We set $\lambda_t = \alpha \cdot \tanh(\gamma \cdot \delta_n)$, yielding stronger guidance when the controlled attribute is far from the target and gentler updates as it converges.

**Back-projection guidance scheduling.** *Semantic reconstruction:* BP guidance is applied throughout the entire reverse process ($t \in [1000, 0]$) to preserve global semantic structure. *Single-attribute editing:* BP guidance is restricted to $t \in [500, 0]$; at earlier timesteps ($t > 500$) the BP step size is set to zero, allowing the diffusion prior to establish global structure before attribute control kicks in. *Multi-attribute editing:* we widen the guidance window to $t \in [800, 0]$ to account for the increased semantic coupling between attributes.

**Relation to latent-space editing.** Latent-editing methods manipulate a pre-trained generator's code along attribute directions Shen et al. [2020], Härkönen et al. [2020]. In our attribute-controlled generation task, no source image is given: NLBP constrains the requested attributes during diffusion sampling, while the prior samples identity and the remaining content (Section 7.4).

**Covariance adjustment.** Naive modification of a single attribute (e.g., forcing "Beard" to 1) can create an adversarial target if correlated attributes (e.g., "Male") are not adjusted accordingly. To maintain a plausible $y_{\text{target}}$, we utilize the empirical covariance matrix $\Sigma$ of the training attributes. When modifying index $n$ by $\Delta y_n$, we adjust all other indices $j \neq n$ according to

$$\Delta y_j = \frac{\Sigma_{jn}}{\Sigma_{nn}} \Delta y_n. \tag{17}$$

This keeps $y_{\text{target}}$ on the natural manifold of attribute correlations, avoiding out-of-distribution artifacts. In practice, we use two complementary strategies: the covariance-based adjustment above for highly correlated attributes, and sparse single-attribute modification for weakly correlated ones (e.g., *Eyeglasses*).

## D Object-detection NLBP: pipeline and hyperparameters

This appendix details the compositional generation experiment of §7.5. The setup combines (i) an SPNN-CenterNet detector as the forward operator $A$ and its built-in pseudo-inverse $A^\dagger$, (ii) a class-conditional ImageNet ADM as the diffusion prior, (iii) a time-travel DDIM sampler with NLBP enabled only for noisy timesteps, and (iv) a pixel-space Gaussian smooth-paste that localizes the constraint at the bounding box.

**Detector ($A$, $A^\dagger$).** We attach a CenterNet-style head Zhou et al. [2019] to the ImageNet SPNN backbone of Appendix A. The head outputs (i) a per-class center heatmap $\mathrm{hmap} \in \mathbb{R}^{20\times64\times64}$, (ii) sub-pixel center offsets $\mathrm{regs} \in \mathbb{R}^{2\times64\times64}$, and (iii) box width/height in feature-map units $\mathrm{wh} \in \mathbb{R}^{2\times64\times64}$, concatenated as $y = [\mathrm{hmap}, \mathrm{regs}, \mathrm{wh}] \in \mathbb{R}^{24\times64\times64}$ at stride 4 ($256/4 = 64$). The head is built with Householder-mixed invertible affine layers and a depth-2 `ConvMLP` of hidden width 128; the backbone is unfrozen during detector training. The pseudo-inverse $A^\dagger$ is provided by the SPNN's auxiliary network $r$, trained via the natural-PInv objective of Section 5.2.

**Diffusion prior.** We use the public class-conditional $256\times256$ ADM checkpoint of Dhariwal and Nichol Dhariwal and Nichol [2021]: 256 channels, 4 heads, attention at resolutions $\{32, 16, 8\}$, 2 residual blocks per level, learned $\sigma$, scale-shift normalization, FP16. Each VOC class is mapped to its closest ImageNet-1k class for conditioning the prior; the mapping table is given in Table 9.

**Sampling schedule.** Time-travel DDIM Wang et al. [2023] with $T_{\text{sampling}} = 100$, travel length 10, travel repeat 3 ($\sim 430$ network evaluations per image). NLBP is enabled only for $t > 650$ (the noisier $\sim$35% of the schedule), with guidance scale $\lambda = 1.0$; for $t \le 650$ we let the prior sample freely.

**Smooth paste in pixel space.** After each NLBP step we replace $\hat{x}'_{0|t}$ by a Gaussian-weighted mix of the corrected estimate $\hat{x}^{\text{NLBP}}_{0|t}$ and the un-corrected prior estimate $\hat{x}_{0|t}$:

$$\hat{x}'_{0|t}(p) = w(p)\,\hat{x}^{\text{NLBP}}_{0|t}(p) + (1 - w(p))\,\hat{x}_{0|t}(p), \qquad w(p) = \alpha \exp\!\Big(-\tfrac{1}{2}\,\big\|(p - c_B)/(s\,d_B)\big\|^2\Big), \tag{18}$$

where $c_B$ is the bounding-box center, $d_B$ is the box (height, width), $s = 0.5$ is the spatial smoothness in BB-units, and $\alpha = 1.0$. The Gaussian mask localizes the constraint at the bounding box and lets the prior render the rest of the scene without spatial conflicts.

**Hyperparameters.** Best configuration: `min_bp_step` $= 650$, $\lambda_2 = 1.0$, $\lambda_1 = 0.0$, `bp_every` $= 1$, `post_bp_noise` $= 0$, `ref_paste_smoothness` $= 0.5$, `ref_paste_alpha` $= 1.0$, per-image VOC$\to$ImageNet class. A sweep over `min_bp_step` $\in \{300, 500, 550, 650, 750, 800\}$ confirmed 650 as the sweet spot: lower values over-constrain (output too close to original); higher values fail to localize. Per-image generation takes $\sim 22$ s on an RTX 5090.

Table 9: VOC class $\to$ ImageNet-1k class index used for conditioning the ADM prior (§7.5).

| VOC | idx | ImageNet | VOC | idx | ImageNet |
|---|---|---|---|---|---|
| aeroplane | 404 | airliner | dog | 207 | golden retriever |
| bicycle | 671 | mountain bike | horse | 339 | sorrel |
| bird | 14 | indigo bunting | motorbike | 670 | motor scooter |
| boat | 814 | speedboat | person | 981 | ballplayer |
| bottle | 898 | water bottle | pottedplant | 738 | pot |
| bus | 779 | school bus | sheep | 348 | ram |
| car | 817 | sports car | sofa | 831 | studio couch |
| cat | 281 | tabby | train | 466 | bullet train |
| chair | 765 | rocking chair | tvmonitor | 851 | television |
| cow | 345 | ox | diningtable | 532 | dining table |

## E Full quantitative results on attribute reconstruction

Table 10 reports per-attribute reconstruction fidelity for the semantic-reconstruction experiment of Section 7.1, sorted by binary agreement. The mean over all 40 attributes is 92.3% binary agreement,

with the highest-agreement attributes corresponding to clear visual structure (*Eyeglasses*, *Wearing_Necktie*, *Bald*) and the lowest-agreement ones corresponding to subjective or ambiguous features (*Mouth_Slightly_Open*, *Arched_Eyebrows*, *Wearing_Earrings*), where the classifier itself is least confident.

Table 10: Attribute reconstruction fidelity on CelebA-HQ, sorted by binary agreement (descending). "Prob diff" is the mean absolute difference in probability space between the reconstructed and original attribute.

| # | Attribute | Prob diff | Agreement (%) |
|---|---|---|---|
| 0 | Blurry | 0.0001 | 100 |
| 1 | Eyeglasses | 0.0046 | 100 |
| 2 | Wearing Necktie | 0.0065 | 100 |
| 3 | Bald | 0.0027 | 99 |
| 4 | Gray Hair | 0.0074 | 99 |
| 5 | Sideburns | 0.0086 | 98 |
| 6 | Pale Skin | 0.0144 | 98 |
| 7 | Narrow Eyes | 0.0157 | 98 |
| 8 | Double Chin | 0.0040 | 98 |
| 9 | Mustache | 0.0097 | 97 |
| 10 | Wearing Hat | 0.0158 | 97 |
| 11 | Blond Hair | 0.0304 | 97 |
| 12 | Chubby | 0.0106 | 96 |
| 13 | Straight Hair | 0.0058 | 95 |
| 14 | Oval Face | 0.0107 | 95 |
| 15 | Goatee | 0.0252 | 95 |
| 16 | 5 o'Clock Shadow | 0.0185 | 94 |
| 17 | Receding Hairline | 0.0129 | 94 |
| 18 | Wearing Necklace | 0.0157 | 94 |
| 19 | Heavy Makeup | 0.0443 | 93 |
| 20 | Wearing Lipstick | 0.0463 | 93 |
| 21 | Male | 0.0499 | 93 |
| 22 | Bags Under Eyes | 0.0127 | 92 |
| 23 | Black Hair | 0.0247 | 92 |
| 24 | Brown Hair | 0.0292 | 92 |
| 25 | Bangs | 0.0357 | 92 |
| 26 | No Beard | 0.0501 | 91 |
| 27 | Young | 0.0340 | 89 |
| 28 | Wavy Hair | 0.0229 | 89 |
| 29 | Smiling | 0.0572 | 88 |
| 30 | Big Lips | 0.0084 | 87 |
| 31 | Bushy Eyebrows | 0.0249 | 87 |
| 32 | High Cheekbones | 0.0560 | 87 |
| 33 | Pointy Nose | 0.0208 | 86 |
| 34 | Big Nose | 0.0259 | 86 |
| 35 | Rosy Cheeks | 0.0566 | 85 |
| 36 | Attractive | 0.0292 | 83 |
| 37 | Wearing Earrings | 0.0454 | 82 |
| 38 | Arched Eyebrows | 0.0534 | 81 |
| 39 | Mouth Slightly Open | 0.0989 | 80 |

# F Additional semantic reconstruction results

Figures 9 and 10 show further NLBP reconstructions under the protocol of Section 7.1.

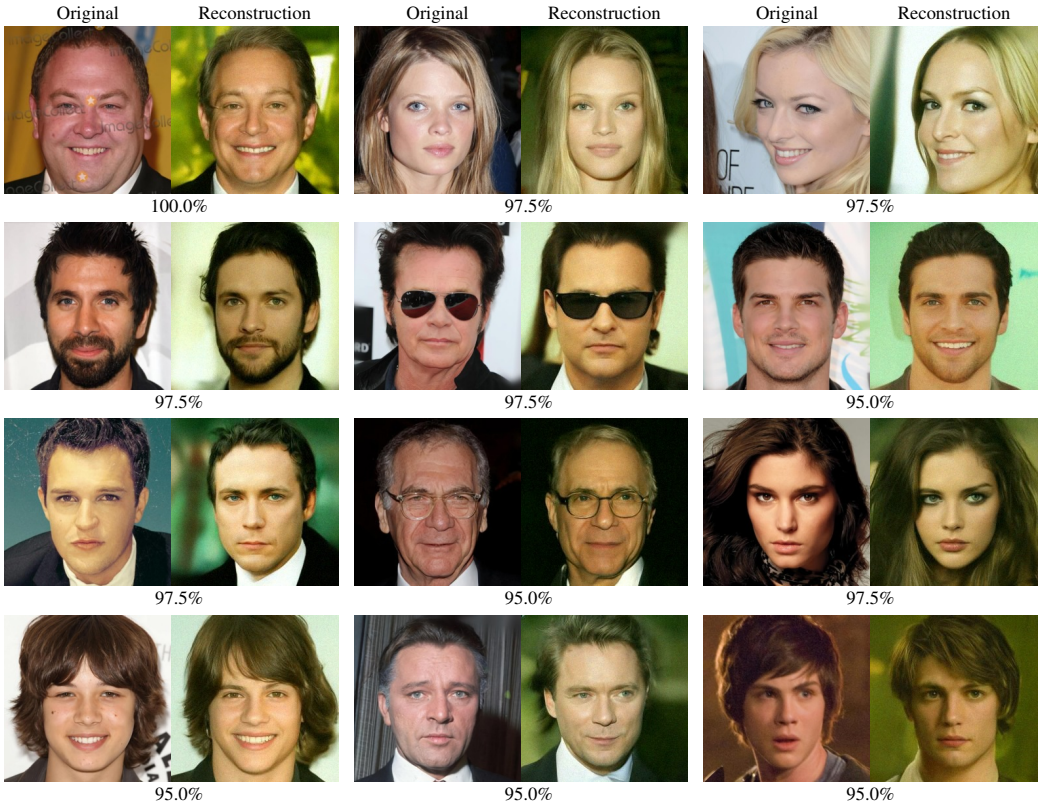



Figure 9: **More CelebA-HQ reconstructions from the $40$-dim attribute logits.** The number below each pair is the fraction of the $40$ attributes (e.g., *Smiling*, *Eyeglasses*) that an independent EfficientNet, not used by our method, predicts identically for the original and the reconstruction. It is the per-image version of "Agree 3rd" in Table 1 (test-set mean $87.4\%$). Full test-set results are in Table 1.

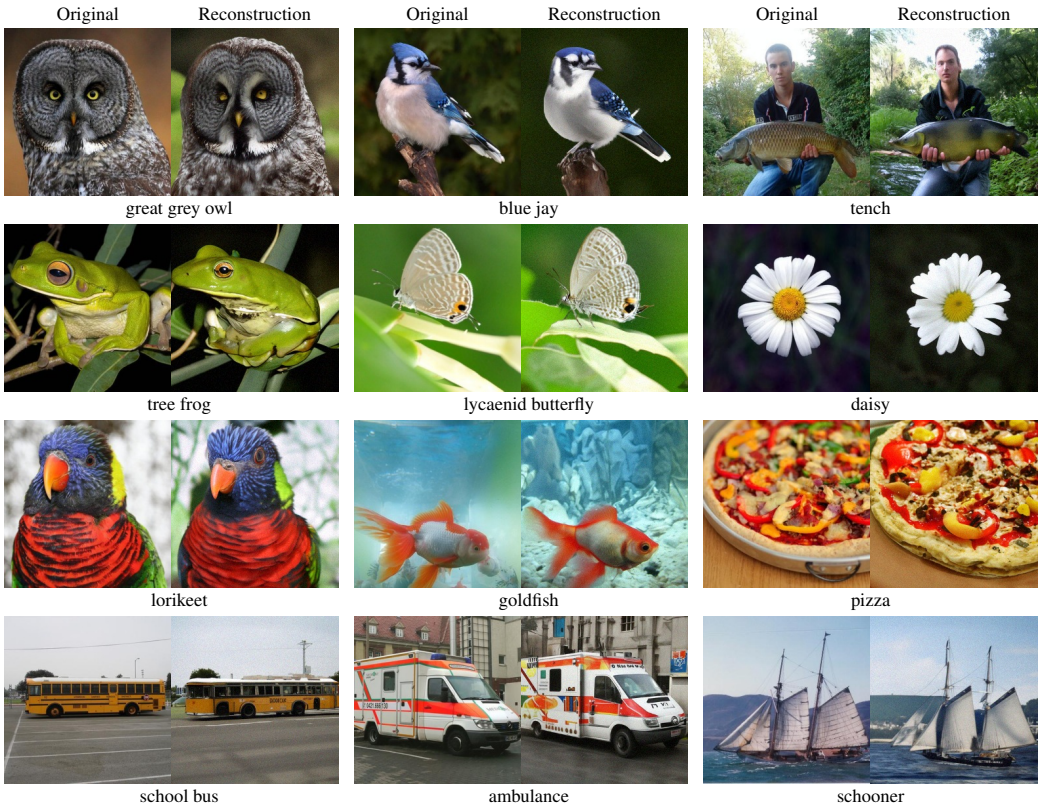



Figure 10: **More ImageNet reconstructions from the full** 1000**-dim class logits.** Below each pair: the ground-truth ImageNet class of the original. Quantitative results over 10,000 samples are in Table 2.

## G Comparison to text-to-image priors

As a non-comparable reference, we additionally evaluate against large pre-trained text-to-image priors – Stable Diffusion 1.5 Rombach et al. [2022] and SDXL Podell et al. [2023], prompted with a natural-language description constructed from the target attribute vector (Table 11). These models receive a different conditioning channel from a logit-vector measurement and are not trained for the 40-way attribute task, so the comparison is not direct; we report it for reference. Despite the looser conditioning, our method beats every T2I configuration in both FID and third-party attribute agreement.

Table 11: Reference comparison against text-to-image priors on CelebA-HQ (6,000 samples, 40-dim attribute prompts). gs: classifier-free guidance scale.

| Method | Guidance | FID↓ | KID↓ | Agree 3rd↑ | s/img |
|---|---|---|---|---|---|
| NLBP+SPNN (ours) | NLBP | **45.1** | **.033** | **87.4%** | 5.7 |
| SD 1.5 | prompt (gs=7.5) | 78.08 | .057 | 81.5% | 0.9 |
| SD 1.5 | prompt (gs=9.5) | 82.58 | .065 | 81.0% | 0.9 |
| SD 1.5 | prompt (gs=11.5) | 86.53 | .072 | 80.9% | 0.9 |
| SDXL | prompt (gs=7.5) | 82.32 | .058 | 79.5% | 3.7 |
| SDXL | prompt (gs=9.5) | 84.95 | .060 | 78.3% | 3.7 |
| SDXL | prompt (gs=11.5) | 88.59 | .061 | 77.9% | 3.7 |

## H Qualitative Samples from the SPNN-Linearizer

Figure 11 shows random samples generated by the SPNN-Linearizer trained on CelebA-64, using one-step generation (100→1 steps collapsed via $B$, FID: 124.5).

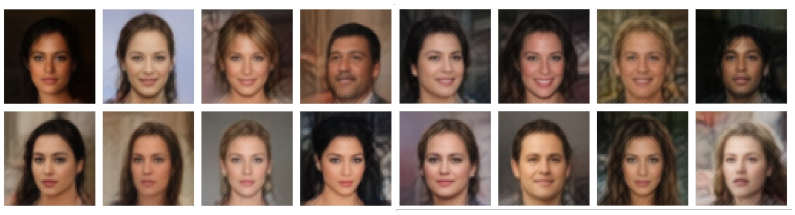


Figure 11: **Generated faces.** Random samples from the SPNN-Linearizer on CelebA 64×64, one-step generation (100→1), FID = 124.5.

# I Proofs and analysis

## I.1 Reconstruction and Euclidean anchoring

Fix a continuous $g$ and a nonempty fiber $F_y = g^{-1}(y)$. Assume that $x$ conditioned on $g(x) = y$ has finite second moment, and write $\mu(y) = \mathbb{E}[x \mid g(x) = y]$. For any deterministic $c \in F_y$, the bias–variance decomposition gives

$$\mathbb{E}[\|x - c\|_2^2 \mid g(x) = y] = \mathrm{Var}(x \mid y) + \|c - \mu(y)\|_2^2, \tag{19}$$

where $\mathrm{Var}(x \mid y) = \mathbb{E}[\|x - \mu(y)\|_2^2 \mid g(x) = y]$. The cross term vanishes because $\mathbb{E}[x - \mu(y) \mid g(x) = y] = 0$. Thus the ideal fiberwise reconstruction optimum is

$$c_{\mathrm{rec}}(y) \in \operatorname*{argmin}_{c \in F_y} \|c - \mu(y)\|_2^2, \tag{20}$$

and every deterministic right-inverse satisfies

$$\mathcal{L}_{\mathrm{rec}} \geq \mathbb{E}_y \left[ \mathrm{Var}(x \mid y) + \mathrm{dist}(\mu(y), F_y)^2 \right]. \tag{21}$$

The minimum exists because $F_y$ is closed and nonempty in finite-dimensional Euclidean space; it need not be unique. A parameterized inverse attains this bound only if it realizes a minimizing choice on almost every fiber. The notation $c_{\mathrm{rec}}$ describes this optimum, not an extra quantity computed during training. Squared error selects a mean-based representative; other objectives can target a median or mode instead.

For a closed convex fiber, $\mu(y) \in F_y$ and the distance term vanishes. Nonconvexity permits a positive gap: two fiber points can have a midpoint outside the fiber. For example, two equally weighted points at $(\pm\sqrt{1 - a^2}, a)$ on the upper unit semicircle, with $0 < a < 1$, have mean $(0, a)$; its closest point on the arc is $(0, 1)$, between the two modes. This illustrates constrained estimation, rather than the topology of a complete SPNN fiber. We thank the reviewer for identifying the fiber constraint and suggesting the semicircle example.

**Proof of Proposition 5.2.** For a fixed block, write $R = R(y)$ and $m = m(y)$. Applying the same decomposition to the discarded coordinates gives

$$\mathbb{E}[\|q(x) - R\|_2^2 \mid g(x) = y] = \|R - m\|_2^2 + \mathrm{Var}(q \mid y) = \varepsilon(y)^2 + \mathrm{Var}(q \mid y). \tag{22}$$

Since $q$ is a coordinate projection after the orthogonal transformation $U$, $\|q(x) - q(g^\dagger(y))\|_2 \leq \|x - g^\dagger(y)\|_2$, proving Eq. (9). Exact consistency also gives $\|G(g^\dagger(y))\|_2^2 = \|y\|_2^2 + \|R\|_2^2$. Setting $e = R - m$,

$$\left| \|R\|_2^2 - \|m\|_2^2 \right| = \left| 2\langle m, e \rangle + \|e\|_2^2 \right| \leq 2\|m\|_2 \varepsilon(y) + \varepsilon(y)^2,$$

which proves Eq. (10). The bound anchors the selected inverse's $G$-norm to a conditional Euclidean statistic; it does not imply that $G$ preserves Euclidean distances. For a nonlinear full-stack completion, the decomposition and norm bound still hold when $q(x)$ has finite second moment, but relating its coordinate error to input-space reconstruction requires an additional regularity bound on $q$.

## I.2 Deviation from orthogonal back-projection

The natural and reconstruction objectives can prefer different points on a fiber, so their combination need not attain the exact natural pseudo-inverse. For the full NLBP update in Eq. (5), let $u = g(x)$ and $R(v) = q(g^\dagger(v))$. Its change in completion coordinates is

$$G(x') - G(x) = \begin{bmatrix} y - u \\ \Delta_{\mathrm{null}} \end{bmatrix}, \qquad \Delta_{\mathrm{null}} = R(y) - R(u). \tag{23}$$

Consistency is exact for every $R$; the update is orthogonal when $\Delta_{\mathrm{null}} = 0$. Writing $d_G(a, b) = \|G(a) - G(b)\|_2$, the exact projection is $P_y(x) = G^{-1}([y; q(x)])$, and

$$d_G(x', P_y(x)) = \|\Delta_{\mathrm{null}}\|_2, \qquad d_G(x, x')^2 = \|y - u\|_2^2 + \|\Delta_{\mathrm{null}}\|_2^2. \tag{24}$$

Thus the residual measures both the displacement from the exact projection and the excess squared update length in the $G$-metric.

To relate this to reconstruction, choose a constrained optimum $c_{\mathrm{rec}}$ from Eq. (20) and write $R_{\mathrm{rec}}(v) = q(c_{\mathrm{rec}}(v))$, $e(v) = R(v) - R_{\mathrm{rec}}(v)$. Then

$$\|\Delta_{\mathrm{null}}\|_2 \le \|R_{\mathrm{rec}}(y) - R_{\mathrm{rec}}(u)\|_2 + \|e(y)\|_2 + \|e(u)\|_2. \tag{25}$$

For a single block, $R = r$, so $e$ is the error of $r$ relative to the reconstruction-optimal discarded coordinates. Even a small $e$ need not imply orthogonality: $R_{\mathrm{rec}}$ itself may vary with the measurement. This separates the reconstruction objective from the canonical selection targeted by $\mathcal{L}_{\mathrm{natural}}$.

On the trained CelebA-HQ model, evaluation over 100 unrelated $(x, y)$ pairs gave a mean null-residual norm of approximately 2.7 and an angular deviation of approximately $0.16^\circ$ from the pure-range update. For $y \neq u$, this angle is $\arctan(\|\Delta_{\mathrm{null}}\|_2 / \|y - u\|_2)$; zero denotes an exactly orthogonal update. These measurements characterize the tested model and pairs, rather than a uniform guarantee over all inputs.

### I.3 SPNN-Linearizer

The lifted operations of Proposition 6.1 define vector spaces on $\mathrm{im}(g_x^\dagger)$ and $\mathrm{im}(g_y^\dagger)$. On these sets, $g_x$ and $g_y$ are bijections onto their latent spaces, with inverses $g_x^\dagger$ and $g_y^\dagger$.

*Proof.*

$$\begin{aligned} f(a_1 \odot_x x_1 \oplus_x a_2 \odot_x x_2) &= g_y^\dagger\big(A\, g_x\big(g_x^\dagger(a_1 g_x(x_1) + a_2 g_x(x_2))\big)\big) \\ &= g_y^\dagger(A\,(a_1 g_x(x_1) + a_2 g_x(x_2))) \\ &= g_y^\dagger\big(g_y g_y^\dagger(a_1 A g_x(x_1) + a_2 A g_x(x_2))\big) \\ &= a_1 \odot_y f(x_1) \oplus_y a_2 \odot_y f(x_2). \end{aligned}$$

□

Alternatively, the completion defines vector-space operations on the full input space:

$$x_1 \oplus_x^G x_2 = G_x^{-1}(G_x(x_1) + G_x(x_2)), \qquad a \odot_x^G x = G_x^{-1}(a G_x(x)).$$

The construction above uses the lower-dimensional inverse images instead; it requires only $g_x g_x^\dagger = \mathrm{Id}$ and $g_y g_y^\dagger = \mathrm{Id}$ and keeps $A$ in the reduced latent space.

## J Additional measurements

**Agreement and uncertainty.** We use bootstrap 95% confidence intervals with 2,000 resamples and Wilcoxon signed-rank tests paired by target image. Table 12 reports mean per-image attribute agreement, standard deviation, and confidence intervals under the independent EfficientNet evaluator on 6,000 CelebA-HQ images. Table 13 reports independent ConvNeXt top-5 agreement and confidence intervals on 10,000 ImageNet images.

Table 12: Per-image attribute agreement on CelebA-HQ, third-party evaluator (6,000 images).

| Method | Agree mean ± std (%) | 95% CI |
|---|---|---|
| NLBP+SPNN | $87.35 \pm 5.54$ | $[87.21, 87.48]$ |
| DPS $z = 1.0$ | $73.23 \pm 10.93$ | $[72.96, 73.50]$ |
| DPS $z = 0.5$ | $80.20 \pm 8.12$ | $[79.99, 80.41]$ |
| DPS $z = 0.1$ | $84.06 \pm 7.36$ | $[83.88, 84.23]$ |

The CelebA-HQ third-party comparisons yield $p < 10^{-45}$ for each NLBP–DPS pair; the self-evaluated agreement differences are also significant. The ImageNet comparisons are significant as well, including the higher DPS agreement at $z = 1.0$ and $z = 0.5$.

**PSNR.** Table 14 reports PSNR against the source images over the same evaluation sets. Because many images can satisfy a semantic measurement, PSNR measures pixel correspondence to that particular source, complementing agreement and distributional fidelity.

Table 13: ImageNet top-5 agreement with bootstrap 95% confidence intervals (10,000 images).

| Method | Agree (%) | 95% CI |
|---|---|---|
| NLBP+SPNN | 49.0 | [48.0, 49.9] |
| DPS $z = 1.0$ | 71.2 | [70.3, 72.1] |
| DPS $z = 0.5$ | 71.6 | [70.7, 72.5] |
| DPS $z = 0.1$ | 38.9 | [37.9, 39.9] |

Table 14: PSNR (dB, higher is better) over the evaluation sets of Tables 1 and 2.

| Method | CelebA-HQ | ImageNet |
|---|---|---|
| NLBP+SPNN | 14.46 | 15.92 |
| DPS $z = 1.0$ | 7.35 | 8.69 |
| DPS $z = 0.5$ | 9.02 | 9.36 |
| DPS $z = 0.1$ | 9.90 | 9.55 |

**Detection.** Table 15 reports results on all 4,952 VOC2007 test images. NLBP uses the SPNN detector; DPS uses an off-the-shelf ResDCN-18 CenterNet. The two NLBP rows distinguish compositional generation, which constrains one object and its location, from unmasked reconstruction, which constrains the full detector output.

Table 15: FID on VOC2007 (4,952 test images) with the detector as forward operator.

| Method | FID ↓ |
|---|---|
| NLBP + SPNN detector, compositional generation (§7.5) | 70.17 |
| NLBP + SPNN detector, unmasked reconstruction | 12.81 |
| DPS + off-the-shelf ResDCN-18 CenterNet ($z$=1.0) | 301 |
| DPS + off-the-shelf ResDCN-18 CenterNet ($z$=0.5) | 235 |
| DPS + off-the-shelf ResDCN-18 CenterNet ($z$=0.1) | 199 |

Table 16: Scaled gradient magnitudes $\|\nabla(\lambda_i \mathcal{L}_i)\|_2$ over the CelebA-HQ Phase I run (94 steps per epoch, 15 epochs), with respect to all trained parameters $\{s, t, U, r\}$.

| Slice | Task (BCE) | Reconstruction | Surjectivity | BCE / Reconstruction |
|---|---|---|---|---|
| First 100 steps | 51.40 | 21.81 | 0.005 | 2.36 |
| Last 100 steps | 32.00 | 9.64 | 0.411 | 3.32 |
| Last 50 steps | 32.07 | 8.73 | 0.498 | 3.67 |

**Gradient magnitudes.** Table 16 summarizes the scaled gradient norms recorded during Phase I. In the reported windows, the task gradient is 2.4–3.7× the reconstruction gradient, although it has no component on $r$; the classifier reaches 89.1% test accuracy. Reconstruction therefore contributes substantially alongside the task loss. The smaller surjectivity gradients are consistent with its numerical-guard role (Section 5.2).

Table 17: Per-step cost breakdown of NLBP-guided sampling on CelebA-HQ.

| Component | Calls/step | ms/step |
|---|---|---|
| Diffusion U-Net (CelebA-HQ DDPM) | 1 | 15.0 |
| SPNN $g$ (forward) | 4 | 56.5 |
| SPNN $g^\dagger$ (inverse) | 3 | 44.9 |
| Other (DDIM math, I/O) | – | 9.0 |
| NLBP step total | | 125.4 |

**Per-step timing.** Evaluations of $g$ and $g^\dagger$ account for about 81% of the cost of a step with NLBP (Table 17). Total runtime also depends on how many sampling steps apply the correction (App. C).

**Completion choice.** Channel splitting and learned orthogonal mixing determine each block's completion. The completion $q(x) = x_1$ depends on how each block splits $Ux$ into retained and discarded coordinates (Eq. (6)). Exactness (Propositions 4.5 and 5.1) holds for any split, but the selected pseudo-inverse, and hence generation, may depend on it (Remark 4.2). Table 18 compares the split of Eq. (6) (*original*) with a *permuted* split, which randomly permutes the channels of $Ux$ in every block before splitting. This changes which coordinates are discarded, and hence the completion $q$. Each split is retrained with three seeds.

Table 18: **Sensitivity of the completion choice (CelebA-HQ).** Spread over random seeds and channel splittings; the metrics are those of Table 1.

| Seed | Split | FID↓ | Agree self↑ | Agree cross↑ | Agree 3rd↑ | Err self↓ | Err cross↓ |
|---|---|---|---|---|---|---|---|
| Table 1 model | original | 45.1 | 92.5% | 88.7% | 87.4% | 0.073 | 0.123 |
| 556 | original | 43.5 | 91.7% | 86.7% | 87.4% | 0.080 | 0.131 |
| 557 | original | 43.5 | 91.4% | 86.6% | 87.1% | 0.083 | 0.132 |
| 558 | original | 46.4 | 91.5% | 86.5% | 87.5% | 0.082 | 0.132 |
| 556 | permuted | 44.0 | 91.0% | 86.5% | 87.0% | 0.086 | 0.133 |
| 557 | permuted | 44.3 | 90.9% | 86.5% | 86.7% | 0.089 | 0.133 |
| 558 | permuted | 43.3 | 91.2% | 86.6% | 87.5% | 0.084 | 0.132 |
| Mean ± std (7 runs) | | 44.3 ± 1.1 | 91.5 ± 0.5% | 86.9 ± 0.8% | 87.2 ± 0.3% | 0.082 ± 0.005 | 0.131 ± 0.003 |

## K Additional DPS outputs




Figure 12: **DPS outputs for further ImageNet classes** (class index in parentheses).

Figure 12 shows DPS outputs for five further ImageNet classes under the setting of Fig. 6.

## L Broader impact

This paper presents a theoretical framework for defining and computing non-linear pseudo-inverses in deep neural networks. The primary goal of this work is to advance the mathematical foundations of inverse problems, offering rigorous tools for ill-posed challenges in scientific computing, medical imaging, and physics where non-linear measurement operators are prevalent. We demonstrate the method through semantic restoration and attribute editing on facial images. We acknowledge that

techniques for controlling generative models, such as the one proposed here, carry potential risks related to the creation of synthetic media and deepfakes. While our method is designed to enforce consistency with semantic constraints, the ability to modify attributes or reconstruct plausible identities from abstract descriptors could in principle be misused for image manipulation. Furthermore, our definition of the natural pseudo-inverse relies on learning the geometry of the training data. Consequently, any biases present in the training dataset (e.g., correlations between demographic attributes and visual features) will be reflected in the reconstructed solutions. Practitioners applying this framework should be aware that the "canonical" solution provided by the model is only as neutral as the data distribution it was trained on.