PixelDense: Dense Prediction as Representation Alignment for Pixel Diffusion

Lehan Yang[1] **Daiqing Qi**[1] **Wenhao Zhang**[1] **Avery Li**[1] **Yiqing Yang**[1] **Yifan Li**[2]
Yu Kong[2] **Haitian Zheng**[3] **Zhifei Zhang**[3] **Zhe Lin**[3] **Varun Jampani**[4] **Sheng Li**[1]

[1]University of Virginia [2]Michigan State University [3]Adobe [4]Arcade AI
lehan.yang@virginia.edu



Figure 1: **Qualitative text-to-image results of PixelDense.** PixelDense generates diverse 512×512 samples with strong compositional fidelity, including object count, attribute binding, spatial relations, and geometry-aware scene layout.

Abstract

Representation alignment (REPA) accelerates diffusion transformer training, but its alignment targets are almost exclusively semantic encoders such as DINOv2 and CLIP. Recent analysis points to spatial structure, not global semantics, as the carrier of the alignment effect, yet dense-prediction foundation models trained to predict that structure remain overlooked as REPA targets. In pixel-space diffusion, SAM2, Depth Anything v2, and Metric3D v2 each outperform the DINOv2-only GenEval baseline, with the two geometric teachers leading the segmentation teacher. A flat sum of all four teachers, however, lands below the best single geometric teacher, as semantic and geometric gradients compete for one denoiser projection. We introduce *PixelDense*, which routes DINOv2 and SAM2 through a semantic projection stream, routes Depth Anything v2 and Metric3D v2 through a geometric projection stream, and adds a weight-space orthogonality penalty that keeps the two streams in disjoint subspaces. All four teachers are frozen during training and dropped at inference. Applied to PixelGen and DeCo with a single recipe, PixelDense improves GenEval, DPG-Bench, and HPS v2.1, raises PixelGen-XXL's GenEval Overall from 0.7927 to 0.8093, and beats every single-teacher and unfactored multi-teacher variant. In partial-noise reconstruction, independent panoptic, depth, and surface-normal probes show up to 53.1% PQ gain and 36.0% depth AbsRel reduction at $\tau = 0.5$ across COCO and Flickr30K. From random initialization, PixelDense also reaches the baseline's peak GenEval $1.23\times$ faster. In SDEdit editing on PIE-Bench, PixelDense keeps more of the source background and layout at every edit strength, raising background PSNR by up to 2.2 dB.

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

1 Introduction

Dense-prediction foundation models can now extract increasingly rich spatial structure from a single RGB image: object-level semantic features from DINOv2 [37], instance and region boundaries from SAM2 [22, 43], monocular depth from Depth Anything v2 [59, 60], and metric surface normals from Metric3D v2 [16]. Together, they capture object semantics, boundaries, depth, and surface orientation, jointly providing semantic and geometric constraints on what a realistic image should look like. We therefore ask whether such semantic and geometric knowledge can directly supervise image generation during training, without requiring dense-prediction outputs at inference.

The recent REPresentation Alignment (REPA) [61] provides a direct way to use a frozen encoder to supervise diffusion training. By matching intermediate denoiser features to frozen encoder features, REPA accelerates DiT training by an order of magnitude without changing inference. However, REPA-style alignment has primarily relied on semantic encoders such as DINOv2 and CLIP [42]. iREPA [46] shows that REPA-style alignment benefits primarily from the spatial structure rather than the global semantics of the teacher encoder. This suggests that dense-prediction foundation models may provide stronger structural guidance for diffusion models, as they are explicitly trained to capture spatial structures. Although video diffusion has explored geometric [53] and multi-encoder [23] alignment for temporal consistency, it remains unclear how to use semantic and geometric encoders as structural constraints for realistic image synthesis.

We instantiate this idea in the setting of pixel-space diffusion, where the denoiser tokens lie on the RGB patch grid that dense teachers consume, so each token aligns directly with the teacher feature. On a PixelGen [35] text-to-image backbone fine-tuned at 512×512, we show that each of SAM2, Depth Anything v2, and Metric3D v2 improves GenEval Overall over the DINOv2-only baseline when used as the added alignment target, and the two geometric teachers outperform the segmentation teacher. This matches iREPA's finding that spatial structure carries the alignment signal.

However, dense teachers do not compose under a flat sum. A four-way summed REPA loss across DINOv2, SAM2, Depth Anything v2, and Metric3D v2 through a single denoiser projection lands below the best single geometric teacher, diluting the fine-grained structure that motivated adding more teachers in the first place. The cause is that semantic and geometric teachers reward incompatible invariances. Semantic encoders push toward viewpoint invariance and instance identity, while geometric encoders push toward metric layout and surface orientation. When both groups of gradients share a single feature basis, the denoiser cannot allocate distinct subspaces to distinct kinds of structure, and the teachers compete instead of compose. The core challenge is therefore representation routing. Semantic and geometric teachers must supervise the same denoiser through separate subspaces while the RGB sampler stays unchanged.

Dense perception offers richer alignment supervision than semantic encoders alone, but only if the alignment objective respects the semantic and geometric structure of the teacher bank. We propose *PixelDense*, a factored dense-alignment framework that provides this structure. It decomposes the multi-teacher objective into two disjoint streams: a semantic stream that carries DINOv2 and SAM2 through a shared semantic projection bottleneck, and a geometric stream that carries Depth Anything v2 and Metric3D through a separate geometric projection bottleneck. A weight-space orthogonality regularizer on the two bottlenecks' input projections keeps the denoiser from collapsing them into a common subspace, so the two streams specialize rather than average. The teachers remain frozen and are dropped at inference, so PixelDense adds no cost to sampling.

PixelDense is not tied to a particular pixel-space backbone. We instantiate it on PixelGen and on DeCo [34], two pixel diffusion transformers with different loss recipes, and the same alignment recipe transfers without modification. On PixelGen at 512×512, full PixelDense raises GenEval Overall over the DINOv2-only baseline from 0.7927 to 0.8093, a 2.09% relative gain, and beats every single-teacher and unfactored multi-teacher variant we compare against. It also preserves source-image structure more faithfully than PixelGen in partial-noise reconstruction. Under standalone segmentation, depth, and surface-normal probes, PixelDense has better scores in every COCO and Flickr30K setting in Table 2. At $\tau = 0.5$, panoptic PQ rises by 35.3% against COCO panoptic annotations and by 53.1% against Flickr30K pseudo labels. Depth AbsRel falls by up to 36.0%. On DeCo, PixelDense also improves over the baseline. The same structure preservation carries over to image editing: with SDEdit on PIE-Bench, PixelDense keeps more of the source background and layout at every edit strength.

The main contributions of this work are summarized as follows:

- We introduce *PixelDense*, a dense-perception representation-alignment framework for pixel-space diffusion. PixelDense uses semantic, segmentation, depth, and surface-normal models as training-time teachers while retaining the original RGB generator for inference.
- We propose a semantic–geometric factorization for dense teacher banks. PixelDense routes DINOv2 and SAM2 through a semantic projection stream, routes Depth Anything v2 and Metric3D v2 through a geometric projection stream, and uses weight space orthogonality to keep the two forms of supervision from collapsing into the same denoiser subspace.
- We identify a negative-transfer effect in dense multi-teacher alignment. Although each dense teacher improves over the DINOv2-only baseline, simply summing semantic and geometric REPA losses forces incompatible invariances through a shared projection and degrades the combined signal. This reframes teacher composition as a representation-routing problem.
- Extensive experiments demonstrate the effectiveness and transferability of PixelDense. On PixelGen and DeCo, PixelDense improves compositional generation over matched fine-tunes, single-teacher alignment, and unfactored multi-teacher baselines. In partial-noise reconstruction, standalone segmentation, depth, and surface-normal estimators show stronger structure preservation on COCO and Flickr30K, and this carries over to SDEdit image editing on PIE-Bench.

2 Related Work

Diffusion models for image generation. Diffusion models [14, 47] and flow matching [29, 31] now form a common image synthesis framework, and classifier-free guidance [13] has become a standard conditioning tool. Latent diffusion [44] made compressed VAE space the default for efficient generation, while transformer denoisers including DiT [38], U-ViT [2], SiT [33], MDT [8], and DDT [50] improved the backbone. Pixel-space methods remove the VAE bottleneck and its reconstruction artifacts. Simple Diffusion [15], PixelFlow [5], and self-supervised pretraining [24] show that direct raw-pixel training is viable, and JiT [26], DeCo [34], PixelDiT [62], PixelGen [35], and Latent Forcing [1] further improve prediction targets, frequency treatment, architectures, perceptual losses, or trajectory design. These methods strengthen the denoisers, but their supervision remains an RGB reconstruction target, which leaves the semantic and geometric structure of the image implicit. We build on this pixel-space line and ask what additional training signal can supervise the denoiser.

Representation alignment for diffusion models. REPA [61] answers this question by matching intermediate denoiser features to a frozen encoder such as DINOv2 [37], accelerating DiT training with no inference cost. iREPA [46] further identified spatial structure as the active ingredient of the alignment signal, which reframes alignment as a question about which encoder exposes that structure most directly. Follow-up work extends the recipe through VAE tuning (REPA-E [25]), hierarchical alignment (SARA [3]), time-limited objectives (HASTE [51], ERW [30], curriculum decay [49]), and entanglement or self-alignment variants (REG [52], LayerSync [10]), while PixelREPA [45] and V-Co [27] study pixel-space collapse and co-denoising. Closer to us, Align4Gen [23] and Geometry Forcing [53] bring multi-encoder geometry-aware alignment to video diffusion, but they target temporal consistency and sum teacher signals without addressing the conflict between semantic and geometric supervision. Image-generation alignment still defaults to single semantic teachers such as DINOv2, CLIP [42], and SigLIP [63], leaving open which encoders actually carry the spatial structure iREPA points to.

Dense prediction meets generation. Dense prediction foundation models, including SAM [22] and SAM2 [43] for segmentation, Depth Anything [59] and Depth Anything V2 [60] for depth, and Metric3D v2 [16] for depth and surface normals, are trained explicitly to expose this spatial structure from a single RGB image. Most prior work explores the reverse direction. Marigold [20], GeoWizard [7], Lotus [12], GenPercept [55], VPD [64], and ODISE [56] repurpose diffusion features for dense perception, while UniGS [41], SemFlow [48], UDPDiff [58], and Diff-2-in-1 [65] couple image and dense outputs in unified generators. These models still emit dense predictions or target perception gains. We take the opposite direction and use these frozen discriminative encoders as alignment teachers for a pixel-space generator, routing semantic and geometric supervision through separate streams so the two forms of structure specialize rather than average, and the dense teachers are dropped at inference.

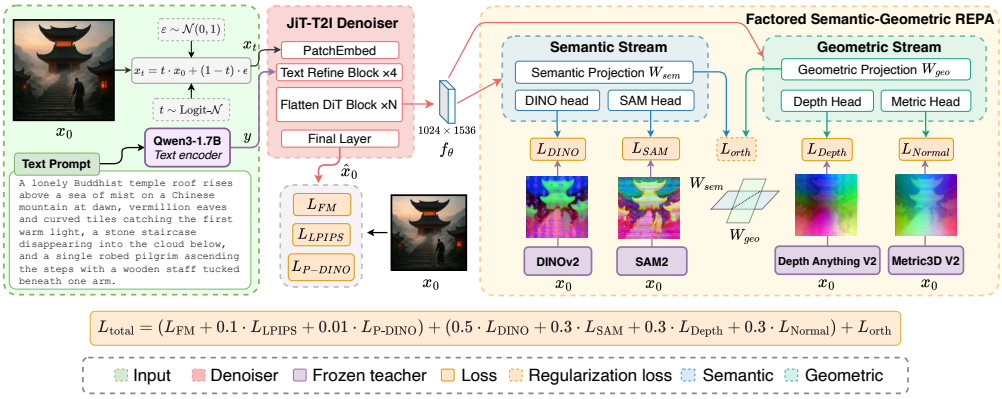

Figure 2: **PixelDense factors dense-prediction supervision into a semantic stream and a geometric stream.** The pixel diffusion denoiser f_θ takes the noisy image $\mathbf{x}_t$ and a Qwen3 text embedding y, and reads its block-8 token state into two parallel projections, a $D_s = 768$ semantic projection W_{sem} aligned to frozen DINOv2 and SAM2 features, and a $D_g = 1024$ geometric projection W_{geo} aligned to frozen Depth Anything v2 and Metric3D v2 features. A weight-space orthogonality penalty $\mathcal{L}_{\text{orth}}$ on the rows of W_{sem} and W_{geo} keeps the two streams from collapsing into a shared subspace of the denoiser feature. Only the denoiser, the projections, and the per-teacher heads receive gradients. The four dense teachers are frozen during training and dropped at inference, so the deployed model is the original three-channel RGB generator with its original sampler.

3 Method

Based on a pixel-space diffusion transformer [35, 34], the proposed PixelDense framework distills semantic and geometric perception into the denoiser through factored representation alignment. Figure 2 gives an overview of PixelDense. An intermediate denoiser token state is projected into two parallel streams: a semantic stream aligned to DINOv2 and SAM2 features, and a geometric stream aligned to Depth Anything v2 and Metric3D v2 features. A weight-space orthogonality penalty on the streams' input projections keeps the two streams from collapsing into a shared subspace of the denoiser feature. All four teachers are frozen during training and dropped at inference.

3.1 Pixel Diffusion with Single-Teacher REPA

We work with a pixel-space diffusion transformer f_θ that maps a noised image $\mathbf{x}_t \in \mathbb{R}^{H\times W\times 3}$ and a text embedding y to a clean pixel prediction $\hat{\mathbf{x}}_0 = f_\theta(\mathbf{x}_t, t, y)$. The denoiser is trained with a flow-matching velocity loss $\mathcal{L}_{\text{FM}}$ on $\hat{\mathbf{x}}_0$. The clean image, scaled to $[0, 1]$, denoted $\mathbf{r}$, is the input domain of the frozen teachers.

Denoiser features. At an intermediate transformer block ℓ, we read the visual-stream token state $\mathbf{h}_\ell \in \mathbb{R}^{N\times d_h}$, where N is the number of patch tokens and d_h is the hidden width. Each token $\mathbf{h}_{\ell,n}$ is anchored to a patch on the same image grid that the dense teachers consume, so its alignment target is a single teacher-feature vector at the matching spatial location. We set $\ell = 8$ in a 16-block denoiser.

Single-teacher REPA. For any teacher $k \in \mathcal{K}$, a frozen teacher encoder T_k maps the clean image to a token-aligned feature grid $T_k(\mathbf{r}) \in \mathbb{R}^{N\times d_k}$. With a tokenwise projector $P_k : \mathbb{R}^{d_h} \to \mathbb{R}^{d_k}$, REPA aligns the projected denoiser feature to the teacher feature with a cosine loss,

$$\mathcal{L}_k(\theta, P_k) = \mathbb{E}_{\mathbf{x}_0,\varepsilon,t}\left[\frac{1}{N}\sum_{n=1}^{N} \ell_{\cos}(P_k(\mathbf{h}_{\ell,n}),\, T_k(\mathbf{r})_n)\right], \quad \ell_{\cos}(a, b) = 1 - \frac{\langle a, b\rangle}{\|a\|\,\|b\|}, \tag{1}$$

where the teacher receives no gradient. This is the alignment loss used by REPA [61], iREPA [46] and related follow-up works, which typically align to a single semantic teacher such as DINOv2.

3.2 Why a Single Alignment Stream Cannot Carry Dense Perception

We use a teacher bank $\mathcal{K} = \{\text{DINO}, \text{SAM}, \text{DA2}, \text{M3D}\}$ that combines semantic and geometric supervision. DINOv2 [37] and SAM2 [43] capture object identity, regions, and boundaries, whereas

Depth Anything v2 [60] and Metric3D v2 [16] capture 3D layout, depth, and surface orientation. As illustrated in Figure 6 of Section C.1, the two groups provide complementary spatial cues for the same image, motivating an alignment objective that preserves this semantic–geometric split.

Unfactored multi-teacher REPA. The natural extension of Equation (1) sums the per-teacher losses, $\mathcal{L}_{\text{unfactored}} = \sum_{k\in\mathcal{K}} \lambda_k \mathcal{L}_k(\theta, P_k)$, with independent projector P_k for each teacher. Although these heads decouple the teachers' target spaces, all losses still update through the same denoiser features $\mathbf{h}_{\ell,n}$. Thus, the denoiser must satisfy four supervision signals through a single input feature space.

Gradient interference. The four signals need not point in compatible directions. Semantic encoders reward viewpoint invariance and instance identity, whereas geometric encoders reward metric layout and surface orientation. With a shared denoiser feature, the gradients pulling $\mathbf{h}_{\ell,n}$ toward semantic identity interfere with those pulling it toward depth and normal layout. Dense perception offers richer alignment supervision than a single semantic teacher, but only if the alignment objective respects the semantic and geometric structure of the teacher bank.

3.3 PixelDense: Factored Streams with Orthogonal Subspaces

As show in Figure 2, PixelDense replaces the single denoiser-side projector with two parallel projection streams, one for each REPA teacher group, and constrains their input maps to occupy orthogonal subspaces of $\mathbb{R}^{d_h}$.

Two streams. A semantic stream π_s and a geometric stream π_g read the same denoiser token,

$$\pi_s(\mathbf{h}_{\ell,n}) = W^{(2)}_{\text{sem}}\,\sigma(\text{LN}(W_{\text{sem}}\,\mathbf{h}_{\ell,n}))\,, \qquad \pi_g(\mathbf{h}_{\ell,n}) = W^{(2)}_{\text{geo}}\,\sigma(\text{LN}(W_{\text{geo}}\,\mathbf{h}_{\ell,n}))\,, \tag{2}$$

with $W_{\text{sem}} \in \mathbb{R}^{D_s\times d_h}$, $W_{\text{geo}} \in \mathbb{R}^{D_g\times d_h}$, SiLU nonlinearity σ, and channel-wise LayerNorm. We set $D_s = 768$ and $D_g = 1024$ to match the native widths of the semantic and geometric teacher groups. Four light teacher heads then map each stream output to its teachers' feature widths,

$$\begin{aligned} \mathbf{p}_{\text{DINO},n} &= g_{\text{DINO}}(\pi_s(\mathbf{h}_{\ell,n})), \quad \mathbf{p}_{\text{SAM},n} = g_{\text{SAM}}(\pi_s(\mathbf{h}_{\ell,n})), \\ \mathbf{p}_{\text{DA2},n} &= g_{\text{DA2}}(\pi_g(\mathbf{h}_{\ell,n})), \quad \mathbf{p}_{\text{M3D},n} = g_{\text{M3D}}(\pi_g(\mathbf{h}_{\ell,n})). \end{aligned} \tag{3}$$

Semantic teachers therefore share the semantic bottleneck π_s, geometric teachers share the geometric bottleneck π_g, avoiding a single shared input projection for all four signals.

Orthogonality between streams. Two streams alone do not guarantee specialization, because π_s and π_g both read $\mathbf{h}_\ell$. If W_{sem} and W_{geo} learn the same row subspace of $\mathbb{R}^{d_h}$, the streams compete for the same denoiser coordinates. PixelDense adds a weight-space penalty on the input-facing linear maps,

$$\mathcal{L}_{\text{orth}}(W_{\text{sem}}, W_{\text{geo}}) = \frac{1}{D_s D_g}\left\|\bar{W}_{\text{sem}}\,\bar{W}_{\text{geo}}^{\top}\right\|_1, \qquad \bar{W}_{i,:} = \frac{W_{i,:}}{\|W_{i,:}\|_2}, \tag{4}$$

the mean absolute cosine between every semantic row and geometric row, where each row of W is unit-length normalized to form $\bar{W}$. The penalty uses only projector weights. Therefore it is data-free and it adds no inference costs. When $\mathcal{L}_{\text{orth}} = 0$, every row of W_{sem} is orthogonal to every row of W_{geo}, so semantic and geometric gradients enter $\mathbf{h}_\ell$ through orthogonal subspaces at the first projection layer. The full nonlinear streams are not globally constrained to be orthogonal, but the penalty discourages the two streams from reading the same input directions at their most direct projection.

Alignment loss. For each teacher, the factored cosine loss is

$$\mathcal{L}^{\text{F}}_k = \mathbb{E}_{\mathbf{x}_0,\varepsilon,t}\left[\frac{1}{N}\sum_{n=1}^{N}\ell_{\cos}(\mathbf{p}_{k,n},\, T_k(\mathbf{r})_n)\right], \tag{5}$$

which replaces $P_k(\mathbf{h}_{\ell,n})$ in Equation (1) with the corresponding factored projection $\mathbf{p}_{k,n}$.

$$\mathcal{L}_{\text{align}} = \big(0.5\,\mathcal{L}^{\text{F}}_{\text{DINO}} + 0.3\,\mathcal{L}^{\text{F}}_{\text{SAM}} + 0.3\,\mathcal{L}^{\text{F}}_{\text{DA2}} + 0.3\,\mathcal{L}^{\text{F}}_{\text{M3D}}\big), \tag{6}$$

Full objective. PixelDense trains the denoiser parameters θ and the alignment-side parameters $\Phi = \{\pi_s, \pi_g, g_{\text{DINO}}, g_{\text{SAM}}, g_{\text{DA2}}, g_{\text{M3D}}\}$ under

$$\mathcal{L}(\theta,\Phi) = \mathcal{L}_{\text{FM}}(\theta) + \mathcal{L}_{\text{align}}(\theta,\Phi) + \lambda_{\text{orth}}\,\mathcal{L}_{\text{orth}}(W_{\text{sem}}, W_{\text{geo}}) + \mathcal{L}_{\text{aux}}(\theta), \tag{7}$$

with $\lambda_{\text{orth}} = 0.01$ and $\mathcal{L}_{\text{aux}}$ inherited from LPIPS and Perceptual-DINO loss [35]. The teachers and the parameters in Φ are dropped after training. The deployed model is the original three-channel RGB generator with its original sampler.

Table 1: **PixelDense lifts compositional alignment over the matched PixelGen fine-tune at no inference-time cost.** GenEval [9], DPG-Bench [17], and HPS v2.1 [54] on the released PixelGen-XXL [35] backbone at 512×512. Our rows use the $10{,}000$-step, effective-batch-256, $2\times$H200 recipe of Section 4.1. Higher is better for every metric. Bold marks PixelDense entries that improve over the matched PixelGen fine-tune.

Model	Type	#Params	GenEval ↑							DPG ↑	HPS ↑
			Sing. Obj.	Two Obj.	Count.	Colors	Attr.	Pos.	Overall		
SDXL [40]	latent	2.6B	98.00	74.00	39.00	85.00	23.00	15.00	0.55	74.7	–
SD3 [6]	latent	8B	98.00	84.00	66.00	74.00	43.00	40.00	0.68	–	–
PixelDiT (1024 res) [62]	pixel	1.3B	100.00	94.00	70.00	90.00	65.00	53.00	0.78	83.7	0.250
PixelFlow [5]	pixel	0.9B	–	–	–	–	–	–	0.60	77.9	–
PixelGen [35]	pixel	1.1B	99.00	88.00	59.00	90.00	70.00	70.00	0.79	-	0.281
PixelGen + fine-tune	pixel	1.1B	**99.38**	88.89	57.50	89.36	69.00	71.50	0.7927	78.7	0.280
PixelDense	pixel	1.1B	**99.38**	**89.39**	**58.75**	**93.09**	**70.50**	**74.50**	**0.8093**	**78.9**	**0.282**

4 Experiments

We first introduce the experimental setting: pixel-space backbones, dense teachers, fine-tuning recipe, datasets, metrics, and baselines. We then evaluate PixelDense on PixelGen-XXL [35] and test whether partial-noise reconstructions preserve scene geometry under independent panoptic, depth, and surface-normal predictors. Next, we analyze how each dense teacher contributes to generation quality, study multi-teacher composition, and ablate the semantic–geometric factorization, projection width, alignment block, alignment-loss weights, and orthogonality regularizer. Finally, we examine training from scratch, transfer the same recipe to DeCo [34] without retuning, test the trained model on image editing, and provide qualitative visualization under matched prompts and samplers.

4.1 Setup

Our main backbone is the 1.1B PixelGen-XXL [35]. We also finetune DeCo [34] for testing the generality. Both models are pretrained with DINOv2 REPA [61]. During fine-tuning, this DINOv2 signal remains active. Because both backbones already retain the original DINOv2 REPA signal during fine-tuning, our primary comparison baseline is therefore a single teacher DINOv2 baseline. We conduct all experiments on $2\times$ NVIDIA H200 GPUs. Each GPU uses a batch size of 16 with gradient accumulation of 8, giving an effective total batch size of 256. We train for $10{,}000$ optimizer steps with AdamW and a learning rate of 1e-5. At inference, we use classifier-free guidance of 4.0.

We use four frozen dense prediction perception models as alignment targets. DINOv2-B/14 [37] provides object-level semantic features. SAM2 Hiera-L [43] provides segmentation aware boundary features from the last image-encoder layer. Depth Anything v2 ViT-L [60] provides monocular depth features from layer 17. Metric3D v2 ViT-L [16] provides surface normal features from layer 17. We align all teacher signals at block 8 of the diffusion denoiser. All four teachers remain frozen during training and are dropped at inference.

The finetuning dataset follows the third stage of PixelGen training and uses BLIP3-o-60K [4]. We report results on GenEval [9], DPG-Bench [17], and HPS v2.1 [54]. We compare with pixel-space methods, including PixelDiT [62], PixelFlow [5], PixelGen [35], and DeCo [34]. We also include latent-space reference methods, SDXL [40] and SD3 [6].

4.2 Results

As shown in Table 1, PixelDense improves over the matched finetune of PixelGen across GenEval Overall, DPG-Bench, and HPS v2.1. It raises GenEval Overall from 0.7927 to 0.8093, a 2.09% relative gain over the baseline. The gains land on the compositional axes that touch object boundaries, depth layout, and surface geometry, which are the axes that dense-prediction supervision is designed to lift. Compared to the listed pixel-space backbones at this scale, PixelDense reaches the strongest GenEval Overall. The dense teachers are removed at test time, so inference speed is not changed.

Geometry Preservation under Partial-Noise Reconstruction. Text-to-image metrics score prompt alignment, compositional correctness, and human aesthetic preference, but they are agnostic to whether a generation preserves the geometry of a specific input scene. We therefore add a partial-noise

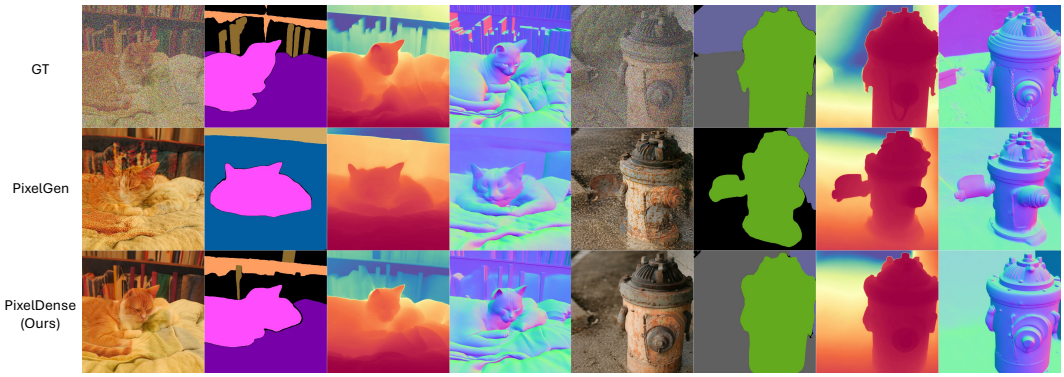

Figure 3: **Qualitative geometry preservation comparison between PixelGen and PixelDense.** We show noised/reconstructed images and pseudo labels for segmentation, depth, and normals. PixelGen shows more ghosting and structural distortions, while PixelDense better preserves GT geometry.

Table 2: **Partial-noise reconstruction shows stronger geometry preservation for PixelDense.** PixelGen and PixelDense reconstruct $\hat{\mathbf{x}}_0$ from $\mathbf{x}_t = (1-\tau)\mathbf{x}_0 + \tau\varepsilon$ at three noise levels. OneFormer PQ/mIoU are averaged over Swin-L and DiNAT-L COCO panoptic models; Depth cells report mean/median AbsRel after per-image affine fitting; normal cells report mean angular error in degrees. Pseudo references are obtained by applying the predictor to the original image after the shared resize and center crop. Higher is better for PQ and mIoU; lower is better for depth and normals.

Dataset	Reference	τ	OneFormer PQ ↑		OneFormer mIoU ↑		Depth AbsRel ↓		Normal err. (°) ↓	
			PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense
COCO val	Real panoptic	0.5	23.23	**31.43**	37.67	**45.47**	–	–	–	–
		0.7	15.70	**21.31**	30.64	**36.59**	–	–	–	–
		0.9	7.99	**9.64**	22.19	**24.26**	–	–	–	–
	Pseudo orig.	0.5	29.53	**41.16**	38.70	**47.99**	1.58/1.06	**1.06/0.68**	21.80	**17.84**
		0.7	18.93	**26.51**	30.73	**37.32**	2.09/1.54	**1.63/1.12**	26.90	**23.48**
		0.9	9.28	**11.40**	21.86	**24.11**	2.83/2.24	**2.61/2.03**	33.20	**31.94**
Flickr30K	Pseudo orig.	0.5	19.66	**30.09**	19.51	**28.73**	1.61/1.14	**1.03/0.68**	22.45	**18.25**
		0.7	11.03	**16.70**	14.41	**18.62**	2.22/1.70	**1.66/1.21**	28.12	**24.62**
		0.9	4.19	**5.90**	9.47	**10.94**	3.01/2.49	**2.77/2.31**	34.97	**33.88**

reconstruction probe: given an input image, we corrupt it at noise levels $\tau \in \{0.5, 0.7, 0.9\}$ and feed the noised image with its caption to the model. To avoid evaluating with the same models used during training, we score reconstructions with three off-the-shelf probes that are independent of the REPA teachers. For panoptic agreement (PQ [21] and mIoU), we use COCO-trained OneFormer [18] with two backbones, Swin-L [32] and DiNAT-L [11]. For depth and surface normals, we use Marigold [20]. We evaluate on COCO 2017 [28] val against both ground-truth panoptic and predictor-on-original pseudo references, and on Flickr30K [39] against pseudo references only.

As shown in Table 2, PixelDense gives stronger structural agreement than the matched PixelGen baseline across every dataset, reference type, probe, and noise level. The gap is largest at $\tau = 0.5$ and narrows at $\tau = 0.9$, where severe noise removes most of the input structure for both models. Flickr30K panoptic scores are lower than COCO because OneFormer uses the COCO vocabulary; depth and normal trends remain consistent across datasets. Figure 3 shows the same trend qualitatively: PixelDense reconstructions preserve object boundaries, depth ordering, and surface orientation more faithfully than PixelGen.

REPA Teacher Composition Analysis. As shown in Table 3, we compare four settings: adding one REPA teacher to the original PixelGen backbone, naively combining four REPA teachers, using factored REPA, and removing one REPA teacher at a time from PixelDense. From the results, each added dense REPA teacher improves GenEval Overall over DINOv2 alone. Depth Anything v2 and Metric3D v2 reach 0.8069 and 0.8060, respectively, exceeding SAM2 at 0.8020. This agrees with iREPA's finding that spatial structure carries the alignment signal [46]. The naive four-teacher REPA sum performs worse than the best single dense REPA teacher. Thus, when semantic and geometric losses update the same diffusion-denoiser projection, adding more targets does not necessarily provide better supervision. Full PixelDense reaches 0.8093, outperforming all single-teacher and unfactored

Table 3: **REPA Teacher Composition.** GenEval scores for teacher combinations. Baseline (DINOv2 only, PixelGen) is 0.7927. **DA2**: Depth Anything v2; **M3D**: Metric3D v2.

REPA teacher	GenEval	REPA teacher	GenEval
(a) Single + DINOv2		*(c) Factored streams*	
+ SAM2	0.8020	Semantic (D+S)	0.8022
+ DA2	0.8069	Geometric (DA2+M3D)	0.7960
+ M3D	0.8060	**Full PixelDense**	**0.8093**
(b) Unfactored sum		*(d) Leave-one-out*	
Equal weight	0.8036	w/o DINOv2	0.7940
L2-normalized	0.7970	w/o SAM2	0.8027
Reduced weights	0.8009	w/o DA2	0.8025
		w/o M3D	0.8037

Table 4: **Ablation on PixelDense Design Choices.** (a) Bottleneck width, (b) Alignment block, (c) REPA loss weights, and (d) Orthogonality weight λ_{orth}. Bold indicates default.

Configuration	GenEval	Configuration	GenEval
(a) Bottleneck (S/G)		*(c) REPA wts. (D/S/D/M)*	
512/512	0.8053	0.3/0.3/0.3/0.3	0.8079
768/1024 (def.)	**0.8093**	**0.5/0.3/0.3/0.3** (def.)	**0.8093**
1024/1024	0.8065	0.5/0.5/0.5/0.5	0.8081
(b) Alignment block		*(d) λ_{orth}*	
Block 6	0.8075	0	0.8053
Block 8 (def.)	**0.8093**	0.001	0.8077
Block 10	0.8007	**0.01** (def.)	**0.8093**
		0.1	0.8023

variants. Using either stream alone gives a lower score, which indicates that semantic and geometric cues are complementary under the factored objective. The leave-one-out ablation further shows that each REPA teacher contributes to the final model.

4.3 Ablation Studies

Table 4 ablates four design choices in PixelDense: the per-stream projection bottleneck width, the alignment block, the per-teacher REPA loss weights, and the orthogonality regularizer. Each row varies one factor relative to the default of Table 1.

Projection Bottleneck Width. The default widths match the native feature dimensions of the two teacher groups. Shrinking both streams to a smaller shared width removes capacity needed by the geometric teachers, while equalizing them at the larger width wastes semantic-stream parameters. Both variants underperform the default PixelDense configuration.

Alignment Block. The mid-network tap is strongest, consistent with REPA [61]. Moving the tap one block earlier causes a small drop, while moving it later causes a larger drop. A mid-network position leaves enough upstream computation to form teacher-compatible features and enough downstream capacity to decode them into pixels.

Alignment-Loss Weights. Replacing the default weighting with uniform weights yields scores close to the default, showing that PixelDense is robust to moderate weight changes. Retaining the slightly larger DINOv2 weight from the original pretrained objective still gives the best aggregate.

Orthogonality Regularization. Removing the penalty or making it $10\times$ weaker lowers GenEval Overall, and a $10\times$ stronger penalty over-constrains the projections. The default $\lambda_{\text{orth}} = 0.01$ separates the two streams enough to lift compositional alignment without sacrificing capacity.

4.4 Training from Scratch

Our main runs fine-tune from a pretrained model that already carries DINOv2 alignment. To test whether dense teachers also help under scratch training, we train a pixel diffusion model from random initialization at 128×128 resolution on BLIP3-o-60K [4], with and without PixelDense under matched optimizer, batch size, and data ordering. Figure 4 reports GenEval Overall against optimizer step. PixelDense reaches the peak score of the baseline roughly $1.23\times$ faster and stays above it through the rest of the 100K-step horizon, while the baseline drifts downward after its peak. The result is consistent with Table 1: dense alignment helps the denoiser form teacher-compatible features earlier, yielding both faster convergence and a higher final score.



Figure 4: **PixelDense converges faster on GenEval from random initialization.**

Table 5: **The PixelDense recipe transfers to the DeCo backbone without retuning.** The teachers, losses, hyperparameters, and checkpoint-selection rule used on PixelGen-XXL are applied unchanged to DeCo [34] at 512×512. Higher is better for every metric, and bold marks PixelDense entries that improve over the matched DeCo fine-tune.

Model	GenEval ↑							DPG ↑
	Sing. Obj.	Two Obj.	Count.	Colors	Attr.	Pos.	Overall	
DeCo T2I, released checkpoint [34]	100.00	92.00	72.00	91.00	79.00	80.00	0.8600	81.4
DeCo + fine-tune	99.38	**94.44**	73.75	**93.62**	79.00	77.00	0.8620	81.4
PixelDense	**100.00**	93.43	**74.38**	**93.62**	**80.00**	**80.00**	**0.8690**	**81.8**

Table 6: **PixelDense keeps more of the source image in SDEdit editing on PIE-Bench.** Only the checkpoint differs between the two models. Background metrics are computed outside the edit mask, and FID is computed against the source images. Bold marks PixelDense entries that improve over PixelGen.

Metric	Model	Noise level τ						
		0.3	0.4	0.5	0.6	0.7	0.8	0.9
Background, outside the edit mask								
LPIPS ↓	PixelGen	0.22	0.24	0.26	0.28	0.30	0.32	0.35
	PixelDense	**0.19**	**0.21**	**0.23**	**0.24**	**0.26**	**0.29**	**0.32**
PSNR (dB) ↑	PixelGen	26.07	24.72	23.27	21.76	20.17	18.19	15.57
	PixelDense	**28.00**	**26.74**	**25.39**	**23.92**	**22.30**	**20.23**	**17.10**
SSIM ↑	PixelGen	0.80	0.77	0.74	0.71	0.69	0.66	0.62
	PixelDense	**0.84**	**0.82**	**0.80**	**0.77**	**0.74**	**0.71**	**0.66**
DINOv2 self-similarity to the source								
Full image ↑	PixelGen	0.65	0.60	0.55	0.51	0.47	0.43	0.38
	PixelDense	**0.72**	**0.67**	**0.62**	**0.58**	**0.53**	**0.48**	**0.41**
Background ↑	PixelGen	0.80	0.77	0.74	0.72	0.69	0.67	0.63
	PixelDense	**0.84**	**0.81**	**0.78**	**0.76**	**0.73**	**0.70**	**0.65**
FID ↓	PixelGen	53.99	59.93	65.28	69.21	71.11	74.19	77.54
	PixelDense	**41.85**	**48.99**	**54.72**	**60.72**	**65.17**	**69.26**	**73.51**

4.5 Transfer to DeCo

To test whether the recipe ports beyond PixelGen, we apply the same setting to DeCo without retuning. As shown in Table 5, PixelDense improves DeCo on both GenEval Overall and DPG-Bench over the matched DeCo fine-tune. The gain is smaller than on PixelGen likely because DeCo starts from a stronger model, but the direction is consistent, indicating that the recipe ports to a different pixel-space backbone.

4.6 Transfer to Image Editing

Table 2 shows that PixelDense keeps more of the input structure when it reconstructs a noised image. Editing needs the same ability, since the model should change the edited region and keep the rest. We run SDEdit [36] on PIE-Bench [19]: the source image is noised to level τ as in the reconstruction probe and then denoised with the target prompt, for seven levels $\tau \in \{0.3, 0.4, \ldots, 0.9\}$. Both models use the same source images, target prompts, noise, sampler, and number of steps, so only the checkpoint differs. We measure background preservation outside the PIE-Bench edit mask with LPIPS, PSNR, and SSIM. For structure, we use DINOv2 patch self-similarity, which compares the patch-to-patch similarity maps of the edited and source images, on the full image and on the background. We also report FID against the source images. As shown in Table 6, PixelDense is better on all six metrics at every noise level. The FID and DINOv2 gaps shrink at high τ, where little of the source is left for either model.

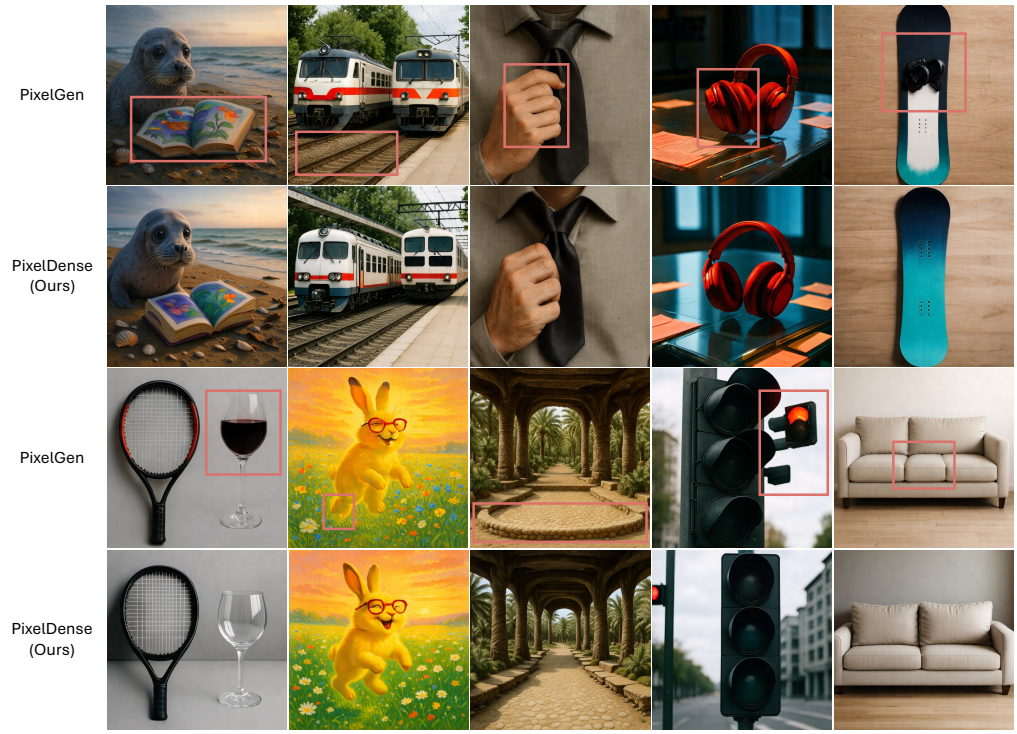

Figure 5: **Qualitative comparison between PixelGen and our PixelDense on text-to-image generation.** PixelDense better preserves object counts, spatial relations, color binding, and scene geometry than PixelGen, yielding stronger prompt alignment and visual coherence.

4.7 Qualitative Results

Figure 5 compares PixelGen and PixelDense on matched prompts and seeds, with the same 25-step sampler as the quantitative evaluation and no dense teacher at inference. Red boxes mark where PixelGen breaks scene structure, and the corrections fall into the two kinds of structure that the two streams supervise. Layout errors appear as distorted rails in front of the trains and a raised platform that cuts off the receding arcade path. PixelDense keeps continuous rails and a path that recedes to a single vanishing point, matching the depth and surface-normal structure of the geometric stream. Part-level errors appear as a malformed hand, a duplicated earcup on the headphones, a stray extra foot on the rabbit, a second signal head fused to the traffic light, and three seat cushions under a two-cushion backrest. PixelDense renders each object as one coherent instance with clean part boundaries, matching the region and instance structure of the semantic stream. The same two kinds of structure drive the panoptic, depth, and normal gains in Table 2.

5 Limitation and Conclusion

This paper introduces PixelDense, a dense-perception representation-alignment framework for pixel-space diffusion. Our approach routes semantic and geometric teachers through separate streams, using weight-space orthogonality to prevent collapse into a single denoiser subspace. Experiments on PixelGen and DeCo show improved compositional alignment at no inference-time cost, while reconstruction probes show stronger preservation of segmentation, depth, and surface-normal structure. Image editing typically requires modifying content while maintaining source image identity, layout, and 3D structure. With SDEdit on PIE-Bench, PixelDense keeps more of the source background and layout at every edit strength, so it is a promising approach for scenarios that demand strict structure preservation. Our editing tests cover only SDEdit, and other editing methods, such as inversion-based and instruction-based editing, are left to future work. We hope PixelDense encourages using dense perception models as practical training-time supervision for image generators.

Acknowledgments and Disclosure of Funding

This work is supported in part by the U.S. Office of Naval Research Award under Grant Number N00014-24-1-2668, the National Science Foundation under Grants IIS-2316306 and CNS-2330215, the National Institutes of Health (NIH) under Grant R01EB293388, Merck BARDS Academic Collaboration Grant, and gifts from Adobe Research. **Competing interests.** This work was conducted as an academic research project at the University of Virginia. Co-authors affiliated with Michigan State University, Adobe, and Arcade AI participated as collaborators and advisors in the capacities indicated by their listed affiliations. The authors declare no competing interests.

References

[1] Alan Baade, Eric Ryan Chan, Kyle Sargent, Changan Chen, Justin Johnson, Ehsan Adeli, and Li Fei-Fei. Latent forcing: Reordering the diffusion trajectory for pixel-space image generation. *arXiv preprint arXiv:2602.11401*, 2026.

[2] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22669–22679, 2023.

[3] Hesen Chen, Junyan Wang, Zhiyu Tan, and Hao Li. Sara: Structural and adversarial representation alignment for training-efficient diffusion models. *arXiv preprint arXiv:2503.08253*, 2025.

[4] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

[5] Shoufa Chen, Chongjian Ge, Shilong Zhang, Peize Sun, and Ping Luo. Pixelflow: Pixel-space generative models with flow. *arXiv preprint arXiv:2504.07963*, 2025.

[6] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[7] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In *European Conference on Computer Vision*, pages 241–258. Springer, 2024.

[8] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 23164–23173, 2023.

[9] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.

[10] Yasaman Haghighi, Bastien van Delft, Mariam Hassan, and Alexandre Alahi. Layersync: Self-aligning intermediate layers. *arXiv preprint arXiv:2510.12581*, 2025.

[11] Ali Hassani and Humphrey Shi. Dilated neighborhood attention transformer. *arXiv preprint arXiv:2209.15001*, 2022.

[12] Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Zhang, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. *arXiv preprint arXiv:2409.18124*, 2024.

[13] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[15] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning*, pages 13213–13232. PMLR, 2023.

[16] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10579–10596, 2024.

[17] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

[18] Jitesh Jain, Jiachen Li, Mang Tik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. Oneformer: One transformer to rule universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2989–2998, 2023.

[19] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Pnp inversion: Boosting diffusion-based editing with 3 lines of code. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=FoMZ4ljhVw.

[20] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9492–9502, 2024.

[21] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9404–9413, 2019.

[22] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[23] Dohun Lee, Hyeonho Jeong, Jiwook Kim, Duygu Ceylan, and Jong Chul Ye. Improving video diffusion transformer training by multi-feature fusion and alignment from self-supervised vision encoders. *arXiv preprint arXiv:2509.09547*, 2025.

[24] Jiachen Lei, Keli Liu, Julius Berner, Haiming Yu, Hongkai Zheng, Jiahong Wu, and Xiangxiang Chu. There is no vae: End-to-end pixel-space generative modeling via self-supervised pre-training. *arXiv preprint arXiv:2510.12586*, 2025.

[25] Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. Repa-e: Unlocking vae for end-to-end tuning of latent diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18262–18272, 2025.

[26] Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. *arXiv preprint arXiv:2511.13720*, 2025.

[27] Han Lin, Xichen Pan, Zun Wang, Yue Zhang, Chu Wang, Jaemin Cho, and Mohit Bansal. V-co: A closer look at visual representation alignment via co-denoising. *arXiv preprint arXiv:2603.16792*, 2026.

[28] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[29] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

[30] Deyuan Liu, Peng Sun, Xufeng Li, and Tao Lin. Efficient generative model training via embedded representation warmup. *arXiv preprint arXiv:2504.10188*, 2025.

[31] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[33] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pages 23–40. Springer, 2024.

[34] Zehong Ma, Longhui Wei, Shuai Wang, Shiliang Zhang, and Qi Tian. Deco: Frequency-decoupled pixel diffusion for end-to-end image generation. *arXiv preprint arXiv:2511.19365*, 2025.

[35] Zehong Ma, Ruihan Xu, and Shiliang Zhang. Pixelgen: Pixel diffusion beats latent diffusion with perceptual loss. *arXiv preprint arXiv:2602.02493*, 2026.

[36] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=aBsCjcPu_tE.

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[38] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[39] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proceedings of the IEEE international conference on computer vision*, pages 2641–2649, 2015.

[40] 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.

[41] Lu Qi, Lehan Yang, Weidong Guo, Yu Xu, Bo Du, Varun Jampani, and Ming-Hsuan Yang. Unigs: Unified representation for image generation and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6305–6315, 2024.

[42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[43] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[44] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[45] Jaeyo Shin, Jiwook Kim, and Hyunjung Shim. Representation alignment for just image transformers is not easier than you think. *arXiv preprint arXiv:2603.14366*, 2026.

[46] Jaskirat Singh, Xingjian Leng, Zongze Wu, Liang Zheng, Richard Zhang, Eli Shechtman, and Saining Xie. What matters for representation alignment: Global information or spatial structure? *arXiv preprint arXiv:2512.10794*, 2025.

[47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

[48] Chaoyang Wang, Xiangtai Li, Lu Qi, Henghui Ding, Yunhai Tong, and Ming-Hsuan Yang. Semflow: Binding semantic segmentation and image synthesis via rectified flow. *Advances in Neural Information Processing Systems*, 37:138981–139001, 2024.

[49] Chenyu Wang, Cai Zhou, Sharut Gupta, Zongyu Lin, Stefanie Jegelka, Stephen Bates, and Tommi Jaakkola. Learning diffusion models with flexible representation guidance. *arXiv preprint arXiv:2507.08980*, 2025.

[50] Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. *arXiv preprint arXiv:2504.05741*, 2025.

[51] Ziqiao Wang, Wangbo Zhao, Yuhao Zhou, Zekai Li, Zhiyuan Liang, Mingjia Shi, Xuanlei Zhao, Pengfei Zhou, Kaipeng Zhang, Zhangyang Wang, et al. Repa works until it doesn't: Early-stopped, holistic alignment supercharges diffusion training. *arXiv preprint arXiv:2505.16792*, 2025.

[52] Ge Wu, Shen Zhang, Ruijing Shi, Shanghua Gao, Zhenyuan Chen, Lei Wang, Zhaowei Chen, Hongcheng Gao, Yao Tang, Jian Yang, et al. Representation entanglement for generation: Training diffusion transformers is much easier than you think. *arXiv preprint arXiv:2507.01467*, 2025.

[53] Haoyu Wu, Diankun Wu, Tianyu He, Junliang Guo, Yang Ye, Yueqi Duan, and Jiang Bian. Geometry forcing: Marrying video diffusion and 3d representation for consistent world modeling. *arXiv preprint arXiv:2507.07982*, 2025.

[54] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

[55] Guangkai Xu, Yongtao Ge, Mingyu Liu, Chengxiang Fan, Kangyang Xie, Zhiyue Zhao, Hao Chen, and Chunhua Shen. What matters when repurposing diffusion models for general dense perception tasks? *arXiv preprint arXiv:2403.06090*, 2024.

[56] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2955–2966, 2023.

[57] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[58] Lehan Yang, Lu Qi, Xiangtai Li, Sheng Li, Varun Jampani, and Ming-Hsuan Yang. Unified dense prediction of video diffusion. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 28963–28973, 2025.

[59] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10371–10381, 2024.

[60] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024.

[61] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=DJSZGGZYVi.

[62] Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. Pixeldit: Pixel diffusion transformers for image generation. *arXiv preprint arXiv:2511.20645*, 2025.

[63] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.

[64] Wenliang Zhao, Yongming Rao, Zuyan Liu, Benlin Liu, Jie Zhou, and Jiwen Lu. Unleashing text-to-image diffusion models for visual perception. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5729–5739, 2023.

[65] Shuhong Zheng, Zhipeng Bao, Ruoyu Zhao, Martial Hebert, and Yu-Xiong Wang. Diff-2-in-1: Bridging generation and dense perception with diffusion models. *arXiv preprint arXiv:2411.05005*, 2024.

Appendix

A Implementation Details

A.1 Pixel-Space Backbones

The primary backbone is PixelGen-XXL [35], a 1.1B-parameter pixel-space diffusion transformer with 16 visual blocks. Inputs are 512×512 RGB images split into 32×32 patches, giving $N = 256$ visual tokens of hidden width $d_h = 1536$. The text branch uses a frozen Qwen3 text encoder [57] with cross-attention into the visual stream. The transfer backbone is DeCo-T2I [34], a 1.1B pixel diffusion transformer with the same patch grid. Both backbones are pretrained with single-teacher DINOv2 [37] REPA [61] and continue to receive that signal during PixelDense fine-tuning. The deployed sampler reads only the three RGB channels, so PixelDense adds no parameters at inference.

A.2 Frozen Dense Teachers

The four teachers and the block used for alignment are summarized in Table 7. Each teacher consumes the clean image $\mathbf{r}$ scaled to $[0, 1]$ at 512×512, and emits a token grid that is bilinearly resampled to the denoiser's 16×16 patch lattice and channel-wise L2 normalized before the cosine alignment of Equations (1) and (5). Teacher weights remain frozen, no augmentation is applied at the teacher input, and gradients flow only through the denoiser, the projection streams, and the per-teacher heads.

Table 7: **Frozen dense-prediction teachers used by PixelDense.** All teachers operate on the clean image $\mathbf{r}$ at 512×512, are frozen during training, and are dropped at inference. Feature dimension is the per-token width before normalization. Group is the semantic-versus-geometric assignment used by Section 3.3.

Teacher	Backbone	Tap	Feature dim.	Group
DINOv2 [37]	ViT-B/14 patch tokens	last layer	768	semantic
SAM2 [43]	Hiera-L image encoder	last layer	256	semantic
Depth Anything v2 [60]	ViT-L encoder	layer 17	1024	geometric
Metric3D v2 [16]	ViT-L encoder	layer 17	1024	geometric

A.3 Projection Streams and Teacher Heads

The semantic stream π_s and the geometric stream π_g both read the block-8 token state $\mathbf{h}_\ell \in \mathbb{R}^{N \times d_h}$ of the denoiser. Each stream is a two-layer MLP of the form $W^{(2)}\sigma(\mathrm{LN}(W\mathbf{h}_{\ell,n}))$, with SiLU nonlinearity σ and channel-wise LayerNorm. The semantic stream uses $W_{\text{sem}} \in \mathbb{R}^{768 \times d_h}$ followed by $W^{(2)}_{\text{sem}} \in \mathbb{R}^{768 \times 768}$, and the geometric stream uses $W_{\text{geo}} \in \mathbb{R}^{1024 \times d_h}$ followed by $W^{(2)}_{\text{geo}} \in \mathbb{R}^{1024 \times 1024}$. The four teacher heads $g_{\text{DINO}}, g_{\text{SAM}}, g_{\text{DA2}}, g_{\text{M3D}}$ are single linear maps from the corresponding stream width to the teacher feature dimension in Table 7. The orthogonality penalty of Equation (4) acts on the row-normalized matrices $\bar{W}_{\text{sem}}$ and $\bar{W}_{\text{geo}}$ and adds no inference-time cost.

A.4 Training Recipe

Fine-tuning runs use 2× NVIDIA H200 GPUs with a per-GPU batch size of 16 and 8 gradient-accumulation steps, giving an effective batch size of 256. The optimizer is AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 0.01, learning rate 1e-5, and a constant schedule after a 200-step linear warmup. Each fine-tune runs for 10,000 optimizer steps, which is the budget reported in Tables 1 and 3 to 5. The full objective of Equation (7) combines the flow-matching loss $\mathcal{L}_{\text{FM}}$, the four cosine alignment losses with the weights in Equation (6), the orthogonality penalty with $\lambda_{\text{orth}} = 0.01$, and the auxiliary LPIPS and Perceptual-DINO losses inherited from PixelGen. We use bf16 mixed precision, gradient norm clipping at 1.0, and exponential moving average over denoiser parameters with decay 0.9999. The same recipe transfers to DeCo without retuning. The from-scratch run of Figure 4 trains a pixel diffusion model from random initialization at 128×128 on BLIP3-o-60K [4] with the same effective batch size and optimizer, for 100,000 steps with and without PixelDense.

A.5 Sampling

At inference, the dense teachers and the alignment-side parameters Φ are dropped. We use the Adam LM solver with 25 steps and classifier-free guidance scale 4.0. Generation resolution is 512×512 for PixelGen-XXL and DeCo, and 128×128 for the from-scratch run. Reported numbers are seed-matched per evaluation, so PixelGen, the matched fine-tune, and PixelDense receive the same prompts and sampler.

A.6 Datasets

Fine-tuning uses BLIP3-o-60K [4], the third-stage dataset of PixelGen training, with no extra filtering. The from-scratch run uses the same dataset at 128×128. Evaluation uses GenEval [9], DPG-Bench [17], and HPS v2.1 [54] for text-to-image quality, COCO 2017 [28] val and Flickr30K [39] test for partial-noise reconstruction. All evaluation images are pre-resized with bilinear interpolation to 512×512 followed by a center crop, and GT panoptic masks are nearest resampled with the same crop so that pixel indices align across reconstruction, prediction, and reference. We did not retrain or fine-tune any teacher, probe, or evaluation model.

B Evaluation Protocols

B.1 Text-to-Image Benchmarks

GenEval [9] is run with the released 553 prompt set across the six standard categories of single-object, two-object, counting, colors, attribute binding, and position. Score with the official GenEval scorer, and report Overall together with the six per-category scores. DPG-Bench [17] is run with its official prompt set, four images per prompt, and the released grader. HPS v2.1 [54] is run on the four 800-prompt style splits of Anime, Concept-art, Paintings, and Photo, with one image per prompt. The per-style breakdown for our PixelGen-XXL fine-tunes is given in Table 15.

B.2 Partial-Noise Reconstruction Probes

The probe of Table 2 corrupts an input image $\mathbf{x}_0$ by $\mathbf{x}_t = (1-\tau)\mathbf{x}_0 + \tau\varepsilon$ with $\varepsilon \sim \mathcal{N}(0, I)$, feeds $\mathbf{x}_t$ together with a caption to the model, and reads the predicted clean image $\hat{\mathbf{x}}_0$. Captions are the original COCO caption with the lowest sentence index for COCO, and the first non-empty Flickr30K caption otherwise. For each $\tau \in \{0.5, 0.7, 0.9\}$ we use a single fixed Gaussian sample ε shared across PixelGen and PixelDense, so the two models reconstruct from identical noisy inputs.

Panoptic probes. The same OneFormer [18] model with two backbones, Swin-L [32] and DiNAT-L [11] pretrained on COCO panoptic for 100 epochs, is applied to the reconstruction $\hat{\mathbf{x}}_0$ and the reference, both at 512×512. Real-COCO references are the public panoptic GT after nearest resize and the same center crop. Pseudo references are the OneFormer prediction on the resize-cropped original. We score with PQ [21] and mIoU under the COCO panoptic vocabulary, and average across Swin-L and DiNAT-L for the cells of Table 2. Per-backbone numbers and $\mathrm{PQ}_{\mathrm{th}}$/$\mathrm{PQ}_{\mathrm{st}}$ breakdowns are in Tables 11 and 12.

Depth probe. We run Marigold v1.1 depth [20] on the reconstruction $\hat{\mathbf{x}}_0$ and on the resize-cropped original. The original prediction is treated as a pseudo-reference because COCO and Flickr30K do not ship dense metric depth. For each image we solve the closed-form least-squares scale and shift (a, b) that align $a \cdot d_{\text{recon}} + b$ to d_{orig}, exclude pixels with $d_{\text{orig}} \leq 10^{-3}$, and report mean and median of pixelwise AbsRel, $|a \cdot d_{\text{recon}} + b - d_{\text{orig}}|/d_{\text{orig}}$.

Surface-normal probe. We run Marigold v1.1 normals on the reconstruction and the resize-cropped original, defensively re-normalize both to unit length, and report the mean pixelwise angular error $\arccos(\langle \hat{n}_{\text{recon}}, \hat{n}_{\text{orig}} \rangle)$ in degrees. Per-image median and mean-of-per-image-medians statistics are in Table 14.

Datasets and counts. COCO val 2017 contributes $5{,}000$ images to every probe, real-GT and pseudo-reference. Flickr30K test contributes $1{,}000$ images filtered to non-empty captions, pseudo reference

only. The same image set is used for PixelGen and PixelDense at every τ, so the comparisons in Table 2 are paired across models.

C Additional Results

C.1 Semantic versus Geometric Teacher Features

Figure 6 visualizes the per-patch responses of the four dense teachers on the same image. Semantic teachers fire on object regions and instance boundaries, while geometric teachers trace depth-ordered silhouettes and surface-normal layout. The split motivates the semantic-versus-geometric factorization of Section 3.3.

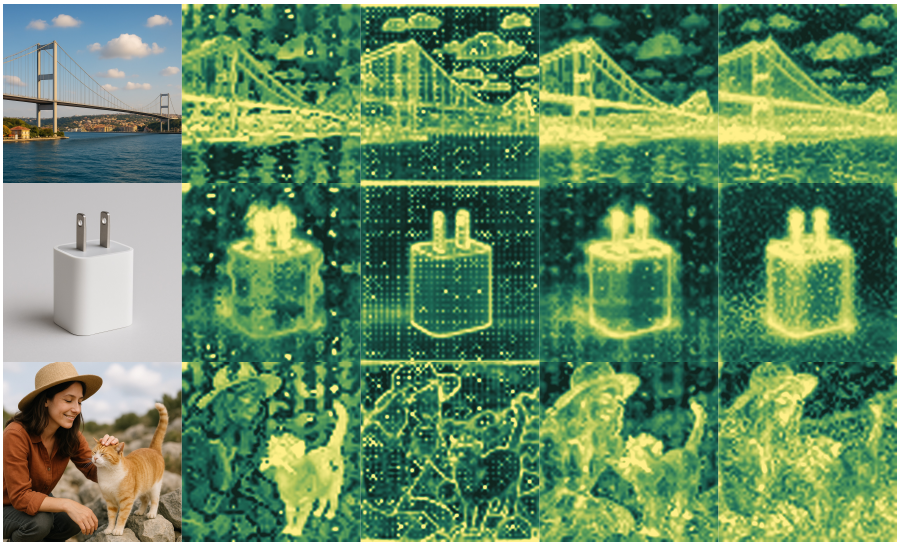

Figure 6: **Semantic and geometric teachers light up complementary spatial structure on the same image.** The leftmost column shows the RGB input, and the next four columns show the per-patch RMS magnitude of spatially normalized features from DINOv2, SAM2, Depth Anything v2, and Metric3D v2, with yellow marking high response. DINOv2 fires on object regions and SAM2 fires on instance boundaries, while Depth Anything v2 traces depth-ordered silhouettes and Metric3D v2 traces surface-normal layout.

C.2 Training-Dynamics Diagnostic for Unfactored versus Factored Alignment

This subsection looks at the training trajectories. We log the four per-teacher cosine alignment losses every 50 optimizer steps and read the four curves jointly across runs from Table 3, the naive equal-weight sum and our default PixelDense configuration. The diagnostic examines three properties simultaneously, namely whether the four teachers co-improve over the full training, whether they move in lockstep at the optimizer-step scale, and whether each teacher's signal is being absorbed at a similar rate.

Setup. Both runs share the recipe of Section 4.1. We log every 50 optimizer steps from step 49 to step 9,999, giving 200 points per run for each of $\mathcal{L}_{\text{DINO}}$, $\mathcal{L}_{\text{SAM}}$, $\mathcal{L}_{\text{DA2}}$, and $\mathcal{L}_{\text{M3D}}$. We compute three diagnostics on this 200×4 matrix per run. Pearson correlation on the centered curves measures whether the four losses co-improve over the full schedule. Cosine similarity of step-to-step loss deltas measures whether the four losses move together or apart at the optimizer-step scale. Per-teacher relative loss reduction, taken between the first ten and last ten logged points, measures how much of each teacher's signal the denoiser has absorbed.

Long-term co-improvement is preserved. As show in Table 8, both runs have a high off-diagonal mean Pearson correlation across the four loss curves, 0.9057 for the naive sum and 0.9083 for

Table 8: **PixelDense preserves long-term co-improvement across teachers while breaking the short-term lockstep that the naive sum imposes.** Run-level coupling and balance diagnostics on the four cosine alignment losses logged every 50 optimizer steps for 10,000 steps, giving 200 points per run. Pearson is the off-diagonal mean of the 4×4 cross-loss correlation on those 200 points. $\Delta_{\cos}$ is the off-diagonal mean of the cosine on the 199 consecutive step-to-step loss deltas. Pearson $- \Delta_{\cos}$ isolates long-term co-improvement from short-term lockstep. Mean reduction is the average per-teacher relative loss reduction between the first and last ten logged points. Sem. and Geo. reduction restrict that average to the two semantic teachers and the two geometric teachers, and Sem./Geo. gap is their absolute difference. Reduction CV is the across-teacher coefficient of variation of the four reductions. Higher Pearson, higher Pearson $- \Delta_{\cos}$, higher mean reduction, lower Sem./Geo. gap, and lower CV are better.

Run	Pearson ↑	$\Delta_{\cos}$ ↓	Pearson $- \Delta_{\cos}$ ↑	Mean red. ↑	Sem. red.	Geo. red.	Sem./Geo. gap ↓	Reduction CV ↓
Naive sum, equal weight	0.9057	0.7430	0.1627	0.7213	0.7573	0.6854	0.0719	0.0973
PixelDense	**0.9083**	**0.5966**	**0.3116**	**0.7340**	0.7337	0.7343	**0.0006**	**0.0629**

Table 9: **Step-to-step loss-delta cosine matrices on the four cosine alignment losses, naive sum versus PixelDense.** Each off-diagonal entry is the cosine of step-to-step deltas of the two named per-teacher losses, computed on the 199 consecutive deltas of the 200 logged points per run. The top sub-block reports the naive equal-weight sum, and the bottom sub-block reports PixelDense. The within-geometric Depth Anything v2 against Metric3D pair stays near 0.95 under both objectives, reflecting the shared 3D structure of the two geometric teachers. The other five off-diagonal entries collapse only under PixelDense, with the largest drop on the within-semantic DINOv2 against SAM2 pair.

	DINOv2	SAM2	DA2	M3D
Naive sum, equal weight, off-diagonal mean 0.7430				
DINOv2	1.0000	0.7287	0.7120	0.6950
SAM2	0.7287	1.0000	0.6799	0.6671
DA2	0.7120	0.6799	1.0000	0.9756
M3D	0.6950	0.6671	0.9756	1.0000
PixelDense, off-diagonal mean 0.5966				
DINOv2	1.0000	**0.3567**	**0.5323**	**0.4969**
SAM2	**0.3567**	1.0000	**0.6618**	**0.5843**
DA2	**0.5323**	**0.6618**	1.0000	0.9477
M3D	**0.4969**	**0.5843**	0.9477	1.0000

PixelDense. The minimum cross-teacher Pearson is also high in both runs, 0.8079 for naive and 0.8366 for factored. The factoring does not break the basic property that all four cosine alignment losses descend together over the 10,000-step horizon. The long-term coupling is marginally tighter under PixelDense.

Short-term lockstep weakens where teacher targets are not intrinsically correlated. The picture inverts at the optimizer-step scale. The off-diagonal mean of the step-delta cosine drops from 0.7430 under the naive sum to 0.5966 under PixelDense. Table 9 reports the full 4×4 matrices. The drop is concentrated on the five pairs that span teacher targets with weaker intrinsic correlation, the four cross-stream pairs and the within-semantic DINOv2-versus-SAM2 pair. The Depth Anything v2-versus-Metric3D pair stays at 0.9756 under naive and 0.9477 under PixelDense, which reflects the genuinely correlated 3D structure of the two geometric teachers and is the only pair that remains strongly coupled under both objectives. The DINOv2-versus-SAM2 step-delta cosine falls from 0.7287 to 0.3567 even though both losses share the semantic projector π_s, which says the per-teacher heads g_{DINO} and g_{SAM} are doing real decoupling work on top of the stream split. Read with the previous paragraph, this says that PixelDense preserves long-term co-progress while breaking the short-term lockstep that an unfactored summed objective imposes on the shared denoiser feature. The naive sum imposes that lockstep on teacher pairs whose targets are not intrinsically correlated. The combined effect is captured by the Pearson $- \Delta_{\cos}$ column of Table 8, which rises from 0.1627 to 0.3116.

Table 10: **Per-teacher relative loss reduction over $10{,}000$ training steps, naive sum versus PixelDense.** The reduction is computed between the first ten and last ten logged points of the 200-point training log per run. Higher values indicate that more of a teacher's signal has been absorbed. Under the naive sum, Metric3D is the slowest teacher in the bank at 64.26% and DINOv2 is the fastest at 83.10%. PixelDense narrows the spread to within 10 points across the four teachers and raises the mean from 72.13% to 73.40%.

Run	DINOv2	SAM2	DA2	M3D	Mean
Naive sum, equal weight	**83.10%**	**68.35%**	72.81%	64.26%	72.13%
PixelDense	78.49%	68.26%	**77.49%**	**69.37%**	**73.40%**

Table 11: **OneFormer Swin-L panoptic scores per dataset, reference type, and noise fraction.** Reconstructions of $\hat{\mathbf{x}}_0$ are scored against the indicated reference at 512×512. PQ is reported on All, Things, and Stuff. Higher is better, and bold marks the PixelDense column.

Dataset	Reference	τ	PQ_{All} ↑		PQ_{Th} ↑		PQ_{St} ↑		mIoU ↑	
			PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense
COCO val	Real panoptic	0.5	23.25	**31.54**	26.32	**35.96**	18.62	**24.86**	37.53	**45.36**
		0.7	15.69	**21.31**	17.16	**23.93**	13.48	**17.37**	30.61	**36.49**
		0.9	8.02	**9.72**	7.92	**9.96**	8.17	**9.36**	22.16	**24.32**
	Pseudo orig.	0.5	29.72	**41.33**	31.90	**44.39**	26.43	**36.71**	38.62	**47.79**
		0.7	19.01	**26.65**	19.95	**28.14**	17.59	**24.39**	30.76	**37.38**
		0.9	9.38	**11.56**	8.87	**11.17**	10.15	**12.15**	21.75	**24.13**
Flickr30K	Pseudo orig.	0.5	19.35	**29.19**	16.54	**25.50**	23.29	**34.35**	19.07	**28.03**
		0.7	11.20	**16.71**	7.71	**12.95**	16.08	**21.95**	14.26	**18.22**
		0.9	4.33	**6.11**	2.26	**4.09**	7.23	**8.92**	9.30	**10.79**

Per-teacher progress is more balanced and the mean rises. Table 10 reports per-teacher relative reduction. The naive run reduces $\mathcal{L}_{\text{DINO}}$ by 83.10% and $\mathcal{L}_{\text{M3D}}$ by only 64.26%. The 19-point gap matches the picture of negative transfer between teacher losses sharing one denoiser feature, the summed update mostly follows the dominant gradient and leaves slower teachers under-served. Under PixelDense the DINOv2 reduction settles at 78.49%, the Depth Anything v2 reduction rises from 72.81% to 77.49%, and the Metric3D normal reduction rises from 64.26% to 69.37%. Mean reduction across the four teachers rises from 0.7213 to 0.7340. The semantic-versus-geometric reduction gap drops from 0.0719 to 0.0006, and the across-teacher coefficient of variation drops from 0.0973 to 0.0629. Each loss is being absorbed at a rate more uniform across the bank, with no teacher under-served, and the higher mean reduction is achieved without requiring more than a 4.6-point relative drop on the dominant DINOv2 term.

What the diagnostic says about knowledge transmission. The diagnostic is consistent with the structural argument of Section 3.3. An unfactored objective sums four teacher cosine gradients on the same denoiser hidden state without reserving semantic and geometric subspaces. This can align per-step loss deltas across the bank and leave slower teachers behind on the longer horizon, which is why the naive run shows both higher cross-teacher step-delta cosine and a larger semantic-versus-geometric reduction gap. Factoring the objective across the semantic-versus-geometric supervision split lets the per-step gradient match the active teacher group with less drag from the other group, which is what the lower cross-stream step-delta cosine measures. The separated update shows up as larger and more uniform per-teacher loss reductions over $10{,}000$ steps, which we read as more even knowledge transmission from the four frozen teachers into the denoiser. The training-time advantage in Tables 8 and 10 is the mechanism behind the GenEval gap reported in Table 3.

C.3 Per-Backbone Geometry Preservation Numbers

Table 2 averages the OneFormer panoptic scores across the Swin-L and DiNAT-L backbones. Tables 11 and 12 report the two backbones separately and break PQ into PQ_{th} for thing classes and PQ_{st} for stuff classes. The two backbones disagree by at most 0.005 on PQ and mIoU per cell, so the averaged numbers in Table 2 are not driven by either backbone in particular. Tables 13 and 14 report the dense-geometry probes with both mean and median statistics; the median AbsRel and the mean of per-image median angular errors agree with the means on every dataset and τ cell, so the ranking in Table 2 is not driven by tail outliers.

Table 12: **OneFormer DiNAT-L panoptic scores per dataset, reference type, and noise fraction.** Same protocol as Table 11, with DiNAT-L replacing Swin-L. The two backbones agree to within 0.005 on PQ and mIoU per cell.

Dataset	Reference	τ	PQ_{All} ↑		PQ_{Th} ↑		PQ_{St} ↑		mIoU ↑	
			PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense
COCO val	Real panoptic	0.5	23.21	**31.32**	26.06	**35.68**	18.91	**24.75**	37.80	**45.59**
		0.7	15.70	**21.31**	17.07	**23.63**	13.65	**17.80**	30.67	**36.70**
		0.9	7.97	**9.57**	7.90	**9.67**	8.08	**9.42**	22.22	**24.20**
	Pseudo orig.	0.5	29.34	**40.99**	31.23	**43.66**	26.49	**36.95**	38.77	**48.19**
		0.7	18.86	**26.37**	19.66	**27.57**	17.65	**24.55**	30.69	**37.26**
		0.9	9.18	**11.25**	8.71	**10.79**	9.89	**11.94**	21.97	**24.10**
Flickr30K	Pseudo orig.	0.5	19.96	**30.98**	17.53	**28.88**	23.40	**33.90**	19.94	**29.42**
		0.7	10.86	**16.69**	7.74	**13.69**	15.33	**20.94**	14.56	**19.03**
		0.9	4.05	**5.69**	1.87	**3.55**	7.09	**8.76**	9.64	**11.10**

Table 13: **Marigold v1.1 depth AbsRel against pseudo references, mean and median.** Mean and median per-image AbsRel after closed-form per-image scale and shift fitting, lower is better. The median AbsRel agrees with the mean on every cell, so the comparison is not driven by a few badly aligned outliers.

Dataset	τ	Mean AbsRel ↓		Median AbsRel ↓	
		PixelGen	PixelDense	PixelGen	PixelDense
COCO val	0.5	1.5805	**1.0639**	1.0559	**0.6752**
	0.7	2.0919	**1.6286**	1.5393	**1.1179**
	0.9	2.8269	**2.6105**	2.2362	**2.0290**
Flickr30K	0.5	1.6061	**1.0257**	1.1443	**0.6770**
	0.7	2.2186	**1.6639**	1.7021	**1.2144**
	0.9	3.0102	**2.7662**	2.4926	**2.3138**

C.4 HPS v2.1 Per-Style Breakdown

Table 15 reports the HPS v2.1 scores for each of the four 800-prompt style splits of Wu et al. [54]. PixelDense matches or exceeds the matched PixelGen DINOv2-only fine-tune on every split and on the four-split average, with the largest gain on Paintings and Concept-art. The released PixelGen-XXL T2I checkpoint is included for reference.

C.5 Additional Qualitative Visualization

Figure 7 extends the prompt range of the main-paper teaser, covering object-centered photographs, painterly portraits, atmospheric landscapes, and stylized fantasy scenes at 512×512. All samples use the 25-step Adam LM solver and classifier-free guidance scale 4.0 of Section 4.1, and the dense teachers are absent at inference. Figure 8 extends Figure 3 with additional partial-noise reconstruction comparisons between PixelGen and PixelDense at the same noise fractions as Table 2.

D Asset Licenses and Use

We use the public assets listed in Table 16 without modification, in accordance with their published licenses. We did not collect new data, did not redistribute any of these assets.

E Broader Impact and Limitations

PixelDense improves the training-time supervision of pixel-space text-to-image diffusion models without changing the deployed sampler. Like other text-to-image generators, the resulting models can support creative tools and accessible visual prototyping, and can also be misused for fabricated imagery, identity-targeted content, or copyright-sensitive style mimicry. The teachers we add are perception models trained on natural images, so PixelDense inherits whatever biases are present in those teachers' training distributions on top of the biases of the pixel backbone. We did not

Table 14: **Marigold v1.1 surface-normal angular error against pseudo references, mean and per-image-median statistics.** Mean pixelwise angular error in degrees, median per-image angular error, and mean of per-image medians. Lower is better. Per-image medians track the means on every cell, so the comparison is not driven by tail pixels.

Dataset	τ	Mean (°) ↓		Median per-image (°) ↓		Mean of per-image medians (°) ↓	
		PixelGen	PixelDense	PixelGen	PixelDense	PixelGen	PixelDense
COCO val	0.5	21.80	**17.84**	20.79	**16.48**	15.25	**11.76**
	0.7	26.90	**23.48**	26.61	**22.80**	19.97	**16.50**
	0.9	33.20	**31.94**	33.70	**32.36**	26.50	**24.93**
Flickr30K	0.5	22.45	**18.25**	21.03	**16.88**	16.27	**12.26**
	0.7	28.12	**24.62**	27.55	**23.59**	21.88	**17.84**
	0.9	34.97	**33.88**	35.39	**34.52**	29.18	**27.98**

Table 15: **HPS v2.1 per-style breakdown across the four prompt categories of Wu et al. [54].** Each split contains 800 prompts, and the rightmost column averages the four splits. Higher is better, and bold marks PixelDense entries that improve over the matched PixelGen fine-tune.

Model	Anime	Concept-art	Paintings	Photo	Average
PixelGen-XXL T2I, released checkpoint	0.292	0.278	0.269	0.284	0.281
DeCo T2I, released checkpoint	0.285	0.267	0.259	0.277	0.272
PixelGen + DINOv2-only fine-tune	0.290	0.277	0.268	0.285	0.280
PixelGen + PixelDense, ours	**0.292**	**0.280**	**0.272**	0.284	**0.282**

change the training dataset of the backbone, so PixelDense does not introduce a new data-collection footprint, and the dense teachers are dropped at inference, so deployment cost and downstream filtering pipelines are unchanged.

The current alignment is applied at a single mid-network block of the denoiser. Multi-scale and multi-block alignment, scaling the backbone and the training corpus, and exploring teacher banks beyond the four perception models studied here are natural extensions. Image editing often requires preserving instance identity, scene layout, object boundaries, depth, and surface orientation while changing only the requested content. Under SDEdit, which uses the trained denoiser as is, PixelDense keeps more of the source background and layout (Table 6). Other editing methods, such as inversion-based and instruction-based editing, add inversion or extra training on top of the denoiser. We leave them to future work, along with metrics for identity and 3D geometry after editing.



Figure 7: **Additional PixelDense samples on the PixelGen-XXL backbone at** 512×512**.** Object-centered photographs, painterly portraits, atmospheric landscapes, and stylized fantasy scenes. The dense teachers are dropped at inference, and sampling matches the protocol of Section 4.1.

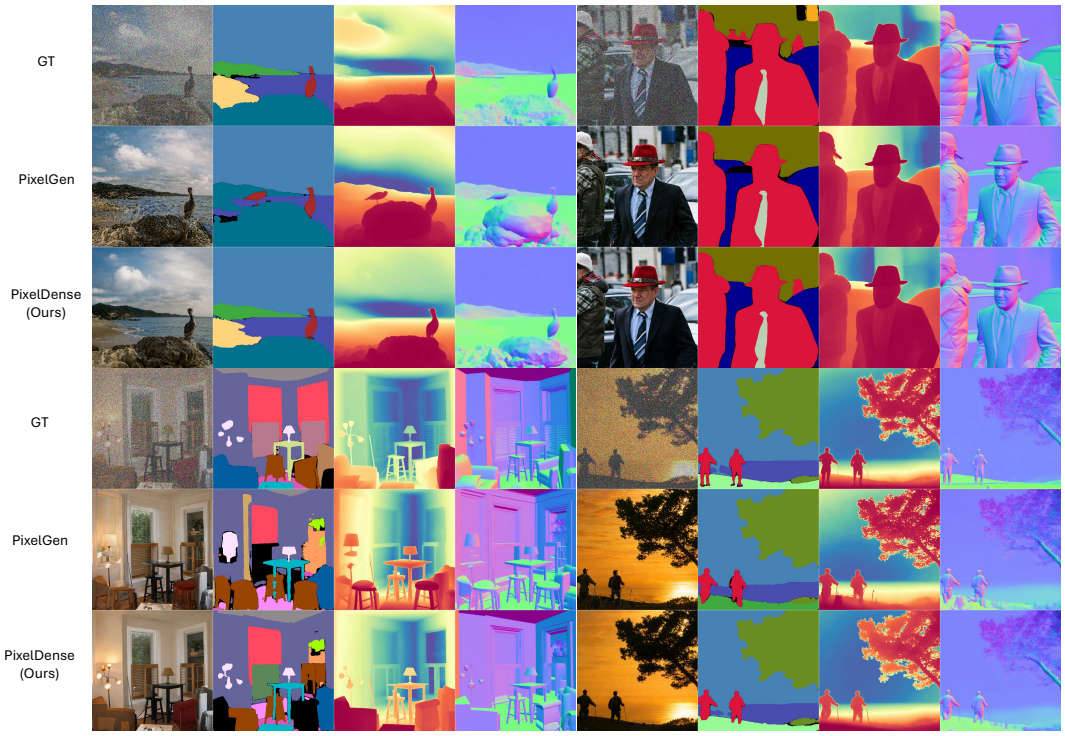

Figure 8: **Additional partial-noise reconstruction comparisons between PixelGen and PixelDense.** Each row shows the noised image at the indicated τ, the two reconstructions, and the panoptic, depth, and surface-normal probes from OneFormer [18] and Marigold [20]. PixelDense reconstructions track object boundaries, depth ordering, and surface orientation more faithfully than PixelGen at every τ.

Table 16: **Public assets used in this paper, with the license under which each is released.** All assets are used without modification. The training corpus and evaluation prompt sets are used through their official releases.

Role	Asset	License
Pixel backbone	PixelGen-XXL [35]	Apache 2.0 use
Pixel backbone	DeCo-T2I [34]	Apache 2.0 use
Semantic teacher	DINOv2 [37]	Apache 2.0
Semantic teacher	SAM2 [43]	Apache 2.0
Geometric teacher	Depth Anything v2 [60]	Apache 2.0
Geometric teacher	Metric3D v2 [16]	BSD 2-Clause
Panoptic probe	OneFormer [18]	MIT
Panoptic backbone	Swin-L [32]	MIT
Panoptic backbone	DiNAT-L [11]	MIT
Depth and normal probe	Marigold v1.1 [20]	Apache 2.0
Training corpus	BLIP3-o-60K [4]	Apache 2.0
Evaluation	GenEval [9]	MIT
Evaluation	DPG-Bench [17]	Apache-2.0
Evaluation	HPS v2.1 [54]	Apache 2.0
Evaluation	COCO 2017 [28]	CC BY 4.0, Flickr terms
Evaluation	Flickr30K [39]	research use

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: As shown in abstract and introduction,

   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: Yes we discussed both in the Limitation and Conclusion section and also in the appendix.

   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: We provided detailed equations of the framework design, loss functions, and etc.

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: We provided details about major aspect of the design and implementation details.

   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: [Yes]

Justification: Code and checkpoint released.

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: We described in the experiments section, and also the supplementary in the appendix

   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: In appendix, we did a statistical exploration on the correlation study of training-dynamics diagnostic for unfactored versus Factored Alignment. But in the main paper experiments, as the heavy computational resources need for the experiments we conducted, we conducted our experiments through rigorous control of variables and ablation studies, thereby serving as a substitute for traditional statistical analyses such as error bars.

   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: Yes we provided detailed infomation on the compute resources in the experiments section and appendix.

   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: Confirmed with the 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: We discussed broader impacts in the appendix.

    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: [Yes]

    Justification: We discussed in the appendix, and we can provide the standard generation filter after the model, but this is not one of the contribution of this paper.

    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 used open sources code / model / data, we summarized in appendix.

    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: [Yes]

    Justification: We will provide the checkpoint of our model.

    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: This paper does not involve crowdsourcing nor research with human subjects.

    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: his paper does not involve crowdsourcing nor research with human subjects.

    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: The core method development in this research does not involve LLMs

    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.