Supplementary Material

A. Overview

In the following, we discuss additional analysis of our approach, and provide further description of the model structure, implementation details, and evaluation procedures. **Appendix B** offers an overview of diffusion models' preliminaries and equations. In **Appendix C**, we then specify the chosen hyperparameters, training techniques, and sampling methods. **Appendices D–F** respectively review the datasets, metrics, and baselines we consider in this study. Finally, in **Appendix G**, we present ablation and variation studies that assess the contribution of each of our design choices, complementing the principal ones explored in the main paper.

Please see our website (*soda-diffusion.github.io*) for a variety of animations and visualizations of outputs generated by the model over different datasets, spanning image reconstructions, viewpoint traversals, latent interpolations, unsupervised attribute discovery and manipulation, demonstration of style and content (or structure) separation, qualitative impact of layer masking and variation of the initial noise map for different training data augmentation schemes, and samples conditioned on partial information.

B. Model Overview & Diffusion Preliminaries

As a denoising diffusion model [5], SODA is formally defined by a pair of forward and backward Markov chains that represent a T-steps transformation from a normal distribution $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ into the learned data distribution $\boldsymbol{x}_0 \sim p_\theta(\boldsymbol{x})$ and vice versa, where $\boldsymbol{x} \in \mathbb{R}^{H \times W \times 3}$. Each forward step t erodes $\boldsymbol{x}_t$ by adding a small Gaussian noise according to a fixed variance schedule α_t, sampling:

$$\boldsymbol{x_t} \sim \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{x_{t-1}}, (1-\alpha_t)\,\boldsymbol{I})$$

Meanwhile, each reverse step t performs image denoising, and aims to estimate $\boldsymbol{\epsilon}_t$ in order to recover $p_\theta(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{z}, \boldsymbol{c})$ where the latent representation $\boldsymbol{z} \in \mathbb{R}^D$ serves as a guidance source for denoising the image, and is produced by the encoder through $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x'}, \boldsymbol{c'})$, the image $\boldsymbol{x'}$ is a related clean input view given to the encoder, and $\boldsymbol{c}, \boldsymbol{c'}$ denote optional conditions for the encoder and decoder respectively (e.g. source and target camera perspectives of a 3D object). We note that this formulation contrasts with unconditional diffusion models, which rely on $(\boldsymbol{x_t}, t)$ only. The reverse step is realized by a denoising decoder $\mathcal{D}$ that predicts $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \boldsymbol{z}, \boldsymbol{c})$. Thanks to the reparametrization trick [5], we can then sample the following:

$$\boldsymbol{x}_{t-1} \sim \mathcal{N}\left(\frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\cdot)\right), \sigma_t^2 \boldsymbol{I}\right)$$

where $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ is the product of the variances up to step t, and σ_t^2 is either a fixed or learned variance term. To train the model, we can readily obtain $\boldsymbol{x}_t$ with the closed-form computation (where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$):

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_{t-1} + (1-\bar{\alpha}_t)\,\boldsymbol{\epsilon}$$

and couple it with the simplified re-weighted MSE training objective (where $\boldsymbol{\epsilon}_\theta$ is estimated by the model):

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}, t}[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon_\theta}]\|_2^2$$

In terms of the architecture, our model consists of an image encoder $\mathcal{E}$ (ResNet or ViT), and a denoising decoder $\mathcal{D}$ that follows the classic structural design of prior literature [5, 6], featuring a UNet implemented as a stack of residual, convolutional, and either downsampling or upsampling layers (in the encoding and decoding modules of the UNet respectively), that are further linked by symmetric skip connections. The decoder $\mathcal{D}$ notably integrates Adaptive Group Normalization layers [6, 49, 50] throughout, allowing $\boldsymbol{z}$ and t to modulate the decoder's activations of each layer $\boldsymbol{h}$, by scaling and shifting them channel-wise:

$$\text{AdaGN}(\boldsymbol{h}, \boldsymbol{z}, t) = \boldsymbol{z}_s(\boldsymbol{t}_s \text{GroupNorm}(\boldsymbol{h}) + \boldsymbol{t}_b) + \boldsymbol{z}_b$$

where $(\boldsymbol{t}_s, \boldsymbol{t}_b)$ and $(\boldsymbol{z}_s, \boldsymbol{z}_b)$ are both obtained by linear projections, the former of a sinusoidal timestep embedding of t [68], and the latter of the latent representation $\boldsymbol{z}$ created by the image encoder $\mathcal{E}$.

C. Implementation Details

Architecture. See Table 12 for our chosen hyperparameters. In terms of the training objective, optimization scheme and empirical configuration, we adopt most of the common settings of recent works [5–7], and specifically use the Adam optimizer [109], gradient accumulation, and exponential moving average for the model's weights; for the ResNet encoder [71]: variant v2 ResNet [110], Xavier initialization [111], ReLU non-linearity, dropPath [112], and mean pooling; and for the UNet decoder: truncated normal initialization (JAX default), GeLU non-linearity [113], $\sqrt{2}$ rescaling of residual connections, BigGAN re-sampling order [114], and self attention in the decoder's low-resolution layers (8-32).

Training. For each dataset, we train the model until convergence, as measured by lack of improvement over a set number of training steps along a validation metric of choice (either downstream accuracy or SSIM). For sampling, we use discrete-time DDPM [5], classifier-free guidance [27] and 1000 diffusion timestemps, practically strided into 75-250 steps [69]. We implement SODA in JAX [115], and run our experiments either on NVIDIA Tesla V100s or TPUs (v2).

Positional Encoding. We employ sinusoidal positional encoding [68] to represent both timesteps and, in the case of

Table 1. **Novel View Synthesis (FID)**, comparing different approaches and aggregation methods as we vary the number of source views. *Stochastic Conditioning* guides each sampling step with a randomly-chosen source view. *Mean Aggregation* conditions on multiple source views by averaging their latents, while *Transformer Aggregation* instead uses a shallow transformer to aggregate the view representations.

	GSO					ShapeNet				
# Source Views	**1**	**3**	**5**	**7**	**9**	**1**	**3**	**5**	**7**	**9**
NeRF-VAE [59]	74.835	79.984	76.965	81.334	80.926	45.791	42.165	36.592	35.441	34.134
SRT [80]	38.642	70.665	40.728	51.936	74.705	17.956	16.336	13.717	27.719	28.026
PixelNeRF [60]	48.721	20.659	7.934	5.906	3.622	25.341	9.679	4.591	3.607	2.557
SODA with Mean Aggregation	**1.508**	2.290	2.060	2.183	1.754	**0.736**	0.696	0.667	0.679	0.742
SODA with Stochastic Conditioning [89]	**1.508**	2.117	1.360	1.177	1.228	**0.736**	0.706	0.686	0.679	0.697
SODA (Transformer Aggregation)	**1.508**	**0.962**	**0.797**	**0.711**	**0.653**	**0.736**	**0.491**	**0.458**	**0.378**	**0.319**

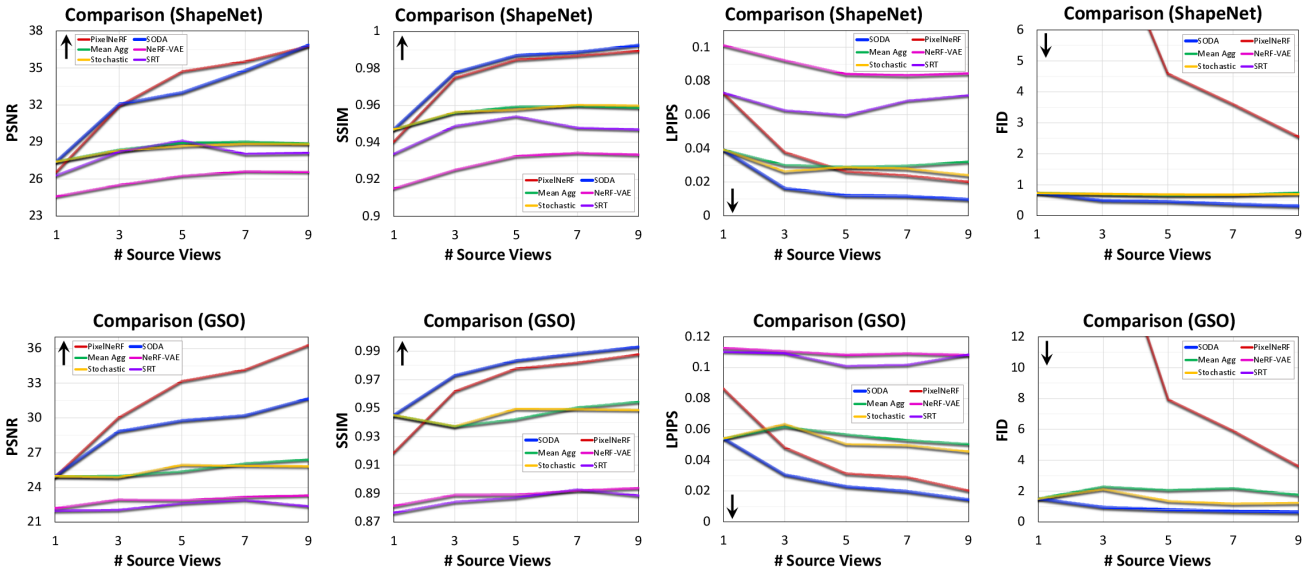

Figure 1. **Number of source views' impact on models performance**, along PSNR, SSIM, LPIPS and FID. We see that as we increase the number of views, SODA achieves a bit higher SSIM score than PixelNeRF, comparable or lower PSNR, better LPIPS score, and much better FID. Other approaches lag behind PixelNeRF and SODA. In terms of view aggregation, we see that stochastic conditioning performs similarly to averaging the view representations, and that our transformer-based aggregation performs robustly better than these alternatives.

pose-conditional view synthesis, spatial coordinates, either xy grids for the 2D case or camera rays' origins and directions for 3D, normalized to a range of $[-1, 1]$. In contrast to the original encoding scheme used to represent discrete word positions, we further scale the arguments of $\sin$ and $\cos$ by a factor of $2\pi s$ (with s being a hyperparameter), so to increase the distinction among the positional encodings (Figure 10).

Pose Conditioning. Throughout the paper, we experiment with several different flavors of the novel view synthesis task: either generating a view conditionally, matching a 3D pose or 2D coordinates, or alternatively, in a pose-unconditional fashion: where given a source view, the model is asked to generate arbitrary novel views at perspectives of its choice). For the conditional case, we represent each perspective by a $H \times W$ 2D grid – of $(x_0, y_0) \times (x_1, y_1)$ in the 2D case, and ray positions and directions in the 3D case – embedded by sinusoidal positional encoding and concatenated to linearly-mapped RGB channels of the corresponding view, after the first layer of the encoder and the denoiser respectively. In Appendix G, we compare different ways to represent the rays, such as through normalization, by casting them on a plane or a sphere, or by summing up their positions and directions.

Learning Rates. For the ImageNet dataset, we maintain a different learning rate between the encoder and the denoiser, at a ratio of $\frac{\text{lr}_\mathcal{E}}{\text{lr}_\mathcal{D}} > 1$. We practically implement it by following the idea of learning rate equalization [116], scaling down the initialized weights of the encoder by a factor of k (by scaling down the standard deviation of the initialization distribution), and then having the network itself scale them back up by k, effectively scaling the encoder's gradients by k. While the model is robust to the selection of the learning rate ratio, we find that a ratio of 2 yields optimal downstream results (Appendix G).

Table 2. **Performance Comparison on CelebA** of classification, reconstruction, and disentanglement, considering variational, adversarial and diffusion-based approaches. *Disen.* stands for Disentanglement, *Comp.* for Completeness, and *Info.* for Informativeness.

Method	Latent Dim	F1 ↑	Disen. ↑	Comp. ↑	Info. ↑	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓
Variational Approaches									
Vanilla Auto-Encoder	2048	66.35	38.94	29.82	84.52	**21.61**	**0.906**	**65.50**	0.327
AnnealedVAE [106]	2048	68.94	42.99	30.75	85.53	15.94	0.686	145.28	0.433
FactorVAE [63]	2048	71.26	46.34	29.87	88.06	20.17	0.890	87.19	0.331
DIP-VAE (I) [107]	2048	71.94	49.09	37.58	89.23	19.83	0.884	81.02	0.316
DIP-VAE (II) [107]	2048	71.85	48.56	33.37	89.12	19.95	0.887	82.41	**0.307**
β-VAE [67]	2048	**71.98**	49.04	**32.46**	**89.37**	19.94	0.886	78.65	0.314
β-TCVAE [108]	2048	71.82	**49.11**	31.63	89.10	20.18	0.892	80.33	0.315
Adversarial Approaches									
VQGAN [75]	256×16×16	-	-	-	-	**23.28***	0.773*	-	**0.311***
StyleGAN2,W [85]	512	-	**53.26**	56.15	**96.71**	16.76*	0.662*	-	0.394*
StyleGAN2,W+ [85]	512×14	-	52.71	**60.03**	94.46	21.42*	**0.813***	-	0.345*
Diffusion-based Approaches									
Unconditional Diffusion	-	63.67	41.33	30.72	84.94	-	-	25.33	-
DiffAE [48]	512	68.70	64.39	39.25	84.61	15.28*	0.681*	-	0.392*
DALL-E2 (with CLIP) [8]	1024	71.08	51.60	37.82	87.87	9.34	0.311	21.91	0.484
SODA (ResNet50×2)	2048	**72.65**	**79.93**	**53.62**	**90.44**	**18.78**	**0.859**	**9.54**	**0.273**

D. Datasets, Preprocessing & Augmentations

D.1. Datasets Overview

Throughout this work, we evaluate models over various datasets grouped into multiple tasks, as summarized by Table 10 and through the textual description below:

Representation Learning & Reconstruction:
Each image in the following datasets is associated with a category label (or for CelebA, with multiple attribute annotations).

(1) Imagenet1K [117]: includes diverse images of objects among 1,000 categories of e.g. animals, instruments, furniture and food items.
(2) CelebA-HQ [105]: features face images, annotated with 40 binary semantic properties like age, gender, or hair color; used also for quantitative disentanglement analysis.
(3) LSUN [118]: partitioned into multiple categories of objects (like cars, cats and horses) and scenes (e.g. bedrooms and churches); See Table 10 for full list.
(4) Animal Faces-HQ (AFHQ) [119]: covers various breeds of cats, dogs and wildlife.
(5) Oxford Flowers 102 [120]: features diverse flowers from the United Kingdom.

Novel View Synthesis:
Each image in the following datasets is associated with the camera perspective it was captured from, expressed as a grid of ray positions and directions $\boldsymbol{r} = (\boldsymbol{o}, \boldsymbol{d})$.

(6) NMR [88]: consists of ShapeNet [87] objects' renderings at 24 fixed views, evenly spaced around a surrounding ring with constant radius and altitude; images of 64×64 resolution. We use the SoftRas data split [121].
(7) ShapeNet: our custom ShapeNet renderings dataset, featuring 120 views randomly sampled from an upper hemisphere, with random azimuth ϕ, altitude θ, and radius $r \in [r_{\text{min}}, r_{\text{max}}]$; 256×256 resolution; created by the Blender-based Kubric library [122].
(8) Google Scanned Objects (GSO) [86]: includes scans of real-world household items, which we render with Blender following the same protocol described above.

Disentanglement (Quantitative):
Each image in the following datasets is associated with discrete semantic attribute annotations.

(9) SmallNORB [101]: contains toy images belonging to 5 categories like animals and vehicles, captured from various camera perspectives and lighting conditions.
(10) 3DShapes [103]: includes images of a centered object among varied combinations of shape, color, size and orientation (4 shapes, 8 scales, 15 orientations, and 10 possible colors for the object, wall and floor).
(11) MPI3D [102]: includes 4 splits of either synthetic or real objects, hold by a robotic arm, with different discrete attributes (4-6 shapes, 4-6 colors, 2 sizes, 3 background colors, and $3 \times 40 \times 40$ camera perspectives).
(12) Caltech-UCSD Birds (CUB-200-2011) [104]: contains images of various bird species, annotated with 312 binary semantic properties.

D.2. Data Preprocessing

Resolution. We resize all images for training and evaluation to a source resolution of 256×256, inputted into the encoder $\mathcal{E}$, and target resolution 128×128, produced by the decoder $\mathcal{D}$, with the exception of CUB and ImageNet: for the former, we center-crop and pad each image based on its associated bird's bounding box; for the latter, we first resize the target images to 256×256, and then center-crop them to 224×224, matching prior literature [53, 58].

We keep the model's output resolution as 128 since according to diffusion models' practices, higher-resolution images are commonly produced through cascading [123], where a core module first generates images of resolution 64

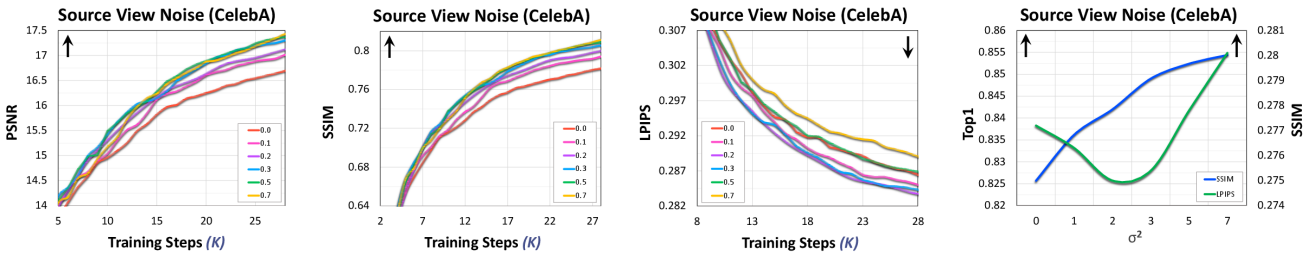

Figure 2. **Impact of source view noise** on the model's generative and representational capabilities.

or 128, and these are these are subsequently post-processed by an independent super-resolution module, rather than being created as high-resolution directly. Indeed, this technique has been shown to improve the overall sample quality, and could readily fit with our approach as well.

Normalization. We normalize the input images fed into the encoder $\mathcal{E}$ based on ImageNet mean and variance statistics [124], while linearly scaling the target images of the denoising decoder $\mathcal{D}$ to the range $[-1, 1]$, following the standard procedures.

Data Splits. For each dataset, we either use the default splits, or if not provided, split them into 80% training, 10% validation and 10% testing. Data is shuffled at training time. We note that for all the multi-view datasets: NMR, ShapeNet, GSO, and smallNorb, we intentionally keep all the views of each object exclusively grouped within one of the splits, and consequently, all the objects used for evaluation are not included in the training set.

D.3. Data Augmentation

We study several augmentation schemes, applied for different tasks and datasets: by default, we use random resize cropping, horizontal flipping and optionally RandAugment [79] data augmentation on both the source and target views $\boldsymbol{x}'$ and $\boldsymbol{x}$ (encoded and denoised respectively). Specifically, at every training step, we randomly augment each view, at the rates specified in Table 12. To train the subsequent downstream classifier, we perform cropping and flipping only, and finally, at evaluation time, perform only center-cropping, following the standard linear probing protocols of prior self-supervision learning works [53, 56]. When training the diffusion model, we also find it conducive to add low Gaussian noise to the encoded source view, similarly to the noise added to the denoised target view.

Meanwhile, for multi-view 3D datasets such as NMR, GSO and ShapeNet, we do not apply data augmentations, and instead, randomly sample one view as the source and another as the target, further supplied by their respective camera perspectives (Appendix C). In this case, we allow for conditioning on multiple source views, and conduct experiments over $k \in [1 - 9]$ sources. Lastly, to illustrate the ability of SODA to learn useful representations even without relying on data augmentation, we perform ablations on datasets used as is, forgoing augmentations of any kind.

E. Evaluation & Metrics

We explore SODA for multiple types of tasks and purposes: downstream linear-probe classification and disentanglement analysis for assessing the quality of the learned representations, as well as image reconstruction and novel view synthesis for evaluating the model's generative capabilities. These skills are measured both through qualitative inspection of the latent space, with visualizations that demonstrate its impact on the model's outputs (including in particular latent interpolations and unsupervised attribute discovery), as well as through an assortment of metrics that quantify each of the capabilities as discussed below.

E.1. Linear Probing

In Section 4.1, we analyze the model's learned latent representations by measuring their predictive performance on a downstream classification task. Following the common evaluation protocol [53, 56, 58], we first train our model on a collection of images, and then fit a linear classifier that considers the latent encodings produced by the model and use them to predict each respective image's category or semantic attributes. The classifier is either trained on the frozen representations $\boldsymbol{z}$ subsequently to the training of the diffusion model, or alternatively, trained with it in tandem by blocking the gradient flow between the two networks – we find that both approaches achieve similar results.

When training the classifier, we refrain from applying weight decay, and adhere to either light augmentation of cropping and flipping for ImageNet or no augmentation in other cases. The latents $\boldsymbol{z}$ are normalized before being fed to the classifier, concretely, by processing them with an unparameterized batch normalization [53], which only tracks mean and variance statistics and lacks the follow-up affine transformation. After normalizing the latents, we use 0.1 dropout for regularization, and for ImageNet, apply label smoothing of 0.1. Since the annotated datasets we explore

Table 3. **Disentanglement Analysis**, comparing SODA to variational approaches on various datasets. Our model achieves improvements of 27.2-58.3% in Disentanglement, 5.0-23.8% in Completeness, and comparable Informativeness. Its reconstructions are often sharper and more accurate. Metrics: *Disen.* stands for Disentanglement, *Comp.* for Completeness, and *Info.* for Informativeness. *PSNR*, *SSIM*, and *LPIPS* respectively express pixel-wise, structural and perceptual/semantic similarity, while *FID* captures sharpness and fidelity.

Method	Disen. ↑	Comp. ↑	Info. ↑	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓
MPI3D (Toy)							
AnnealedVAE [106]	18.43	21.44	52.47	29.49	0.852	118.18	0.169
FactorVAE [63]	22.41	27.06	60.79	34.60	0.959	31.64	0.075
DIP-VAE (I) [107]	51.12	41.33	72.05	37.34	0.977	22.33	0.057
DIP-VAE (II) [107]	24.88	27.37	64.83	36.68	0.976	25.52	0.062
β-VAE [67]	25.34	28.57	64.65	36.86	0.977	24.61	0.061
β-TCVAE [108]	32.21	40.10	64.03	35.17	0.965	30.81	0.072
SODA w/o layer mod.	83.44	**55.65**	**85.38**	50.21	0.998	2.64	0.015
SODA (*ours*)	**87.38**	54.79	84.78	**50.72**	**0.999**	**1.49**	**0.014**
MPI3D (realistic)							
AnnealedVAE [106]	18.76	19.93	51.66	32.54	0.968	55.95	0.201
FactorVAE [63]	30.80	36.63	60.82	34.51	0.980	29.76	0.181
DIP-VAE (I) [107]	45.61	42.37	66.59	36.23	0.986	24.90	0.169
DIP-VAE (II) [107]	29.63	35.85	64.85	36.33	0.986	25.14	0.170
β-VAE [67]	25.14	26.15	61.48	36.54	0.987	24.46	0.168
β-TCVAE [108]	35.81	41.07	66.83	36.71	0.988	24.41	0.167
SODA w/o layer mod.	71.73	51.07	76.53	39.40	0.995	**3.42**	0.109
SODA (*ours*)	**85.19**	**56.26**	**77.78**	**40.65**	**0.996**	3.84	**0.069**
MPI3D (real)							
AnnealedVAE [106]	17.62	17.41	53.60	31.97	0.963	40.33	0.083
FactorVAE [63]	33.99	41.48	59.74	34.19	0.978	33.70	0.062
DIP-VAE (I) [107]	53.01	40.35	72.05	36.68	0.988	18.31	0.037
DIP-VAE (II) [107]	33.54	38.72	65.57	36.37	0.987	20.36	0.041
β-VAE [67]	48.56	48.20	68.19	36.64	0.988	19.35	0.038
β-TCVAE [108]	48.60	44.68	67.03	36.47	0.987	19.74	0.040
SODA w/o layer mod.	75.47	**51.30**	77.32	41.03	**0.998**	1.57	0.007
SODA (*ours*)	**81.19**	50.92	**77.69**	**42.51**	0.997	**0.61**	**0.006**
MPI3D (complex)							
AnnealedVAE [106]	21.08	21.27	57.73	31.37	0.952	39.71	0.077
FactorVAE [63]	37.50	48.55	68.12	32.93	0.966	36.27	0.067
DIP-VAE (I) [107]	58.49	51.67	76.64	35.14	0.980	23.50	0.042
DIP-VAE (II) [107]	35.00	40.21	72.30	35.03	0.979	27.44	0.045
β-VAE [67]	51.82	50.70	74.47	35.58	0.982	22.78	0.039
β-TCVAE [108]	42.99	46.39	74.26	35.42	0.981	24.90	0.042
SODA w/o layer mod.	86.55	56.60	77.43	**44.23**	**0.998**	0.47	0.007
SODA (*ours*)	**89.76**	**56.98**	**78.22**	43.18	0.997	**0.44**	**0.006**
3DShapes							
AnnealedVAE [106]	58.14	66.97	91.23	30.77	0.994	54.83	0.063
FactorVAE [63]	87.62	**87.52**	98.64	30.37	0.994	48.89	0.058
DIP-VAE (I) [107]	99.75	82.59	99.97	34.23	0.997	31.04	0.031
DIP-VAE (II) [107]	99.22	83.01	**99.99**	33.87	0.997	32.41	0.033
β-VAE [67]	**99.87**	83.51	99.96	33.93	0.997	29.52	0.032
β-TCVAE [108]	90.92	74.78	99.81	34.62	0.998	29.03	0.031
SODA w/o layer mod.	92.18	78.21	99.27	51.75	0.9997	0.36	0.0003
SODA (*ours*)	98.60	84.76	98.88	**52.54**	**0.9999**	**0.32**	**0.0002**
SmallNORB							
AnnealedVAE [106]	21.50	17.62	60.30	28.40	0.900	164.94	0.271
FactorVAE [63]	37.01	44.58	68.52	27.67	0.885	148.40	0.267
DIP-VAE (I) [107]	34.13	35.51	**71.85**	28.92	0.907	114.48	0.226
DIP-VAE (II) [107]	37.94	45.76	69.89	29.15	0.909	122.10	0.232
β-VAE [67]	37.79	42.40	71.01	**29.25**	**0.912**	120.31	**0.229**
β-TCVAE [108]	37.43	43.51	70.75	29.20	0.909	120.28	0.230
SODA w/o layer mod.	63.12	45.82	68.36	16.06	0.756	47.90	0.253
SODA (*ours*)	**72.60**	**51.56**	70.19	15.47	0.734	**44.81**	0.235
CUB							
AnnealedVAE [106]	37.53	11.59	91.71	14.73	0.244	275.35	0.716
FactorVAE [63]	39.14	11.68	92.04	**16.26**	**0.517**	244.70	0.637
DIP-VAE (I) [107]	36.44	10.61	**92.12**	15.05	0.421	233.65	0.642
DIP-VAE (II) [107]	37.43	12.19	92.05	14.97	0.409	227.14	0.647
β-VAE [67]	38.22	11.88	92.06	15.10	0.428	234.52	0.639
β-TCVAE [108]	40.43	13.39	91.95	15.26	0.389	237.96	0.652
SODA w/o layer mod.	62.05	14.35	87.86	12.96	0.423	20.30	0.503
SODA (*ours*)	**65.40**	**17.43**	88.10	13.04	0.344	**17.21**	**0.492**

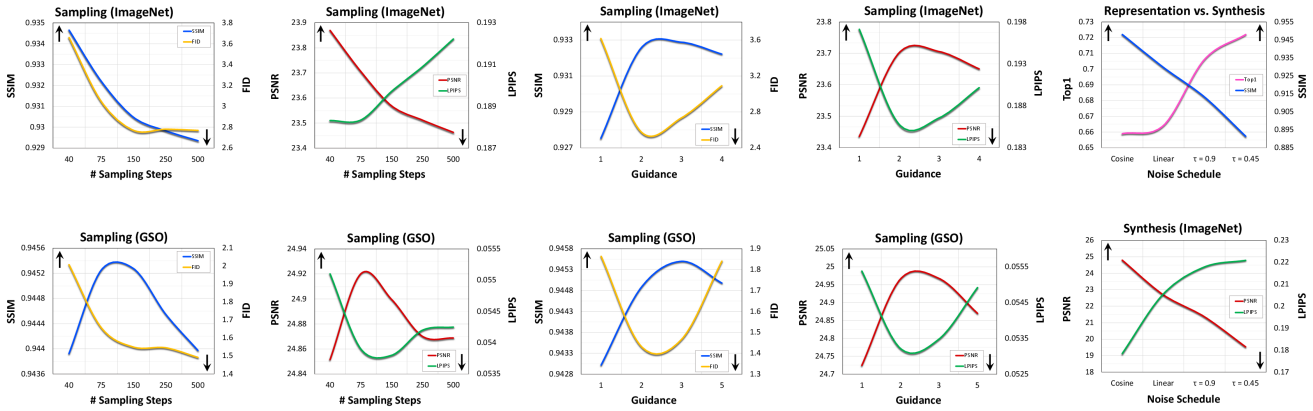

Figure 3. **Impact of classifier-free guidance and number of sampling steps** and noise schedule on the model's performance.

all have discrete labels, we use softmax cross entropy to train the classifier, and report its performance along metrics such as F1 for binary attributes, and top1 accuracy for other ones.

E.2. Image synthesis

In Section 4.2.1, we analyze the capacity of SODA to both reconstruct an input view and generate novel views. Given a target image x, we assess the quality of a synthesized output $\hat{x}$ through multiple complementary metrics that range from visual to semantic similarity:

(1) **Peak Signal-too-Noise Ratio (PSNR)** $\uparrow$ *(measured in dB)* [81]: is directly derived from the mean MSE between x and $\hat{x}$, and it thereby measures **pixel-wise** similarity. It may rate a blurry estimation as highly consistent with the target, as long as they match well with each other *on average*.

(2) **Structural Similarity Index Measure (SSIM)** $\uparrow$ *(ranges between* $[-1, 1]$*)* [82]: compares images along three perceptual factors: luminance, contrast and **structure**, and is thus better correlated with the Human Visual System (HVS).

(3) **Learned Perceptual Image Patch Similarity (LPIPS)** $\downarrow$*(often normalized to be in* $[0, 1]$*)* [84]: computes the distance between the target and synthesized images in the feature space of a supervised pre-trained network, such as VGG [125], and therefore serves as an indicator for **semantic** similarity.

(4) **Fréchet inception distance (FID) score** $\downarrow$ *(is* ≥ 0*)* [83]: quantifies **realism and sharpness** of the generated images by comparing their distribution to that of the target ground-truth images. It concretely achieves it by considering the mean and variance of each, in a latent feature space, e.g. of the Inception model [126]. When assessing unconditionally-generated images, the FID score further expresses their diversity, but in the case of conditional synthesis, either as reconstructions or with pose conditioning, it mainly reflects their fidelity, sharpness and lack of distortions (also known as R-FID in this context).

For fair comparison, we compute these metrics over all approaches using the same metrics' implementations, and specifically, casting the images to the range of $[0, 1]$ for PSNR, $[-1, 1]$ for FID and LPIPS, and using a uniform kernel to calculate SSIM scores.

E.3. Disentanglement

In Section 4.3, we examine the latent space of our model and assess its degree of disentanglement and controllability through quantitative and qualitative evaluation methods:

DCI metrics (Disentanglement, Completeness & Informativeness) $\uparrow$ *(at a range of* $[0, 100\%]$*)* [94]: measures the 1:1 alignment between the latent representation $\boldsymbol{z}$ and the natural (ground-truth) factors of variation $\boldsymbol{c}$. **Disentanglement** reflects the extant to which each latent variable $\hat{z}_j$ (the j'th axis of the vector $\boldsymbol{z}$) corresponds to a unique natural factor c_j. **Completeness** inversely measures the extant to which each natural factor c_j is captured by a single latent variable $\hat{z}_j$. Finally, **Informativeness** indicates the predictability of the natural factors c from the latent encoding $\boldsymbol{z}$. These metrics are derived from the normalized importance matrix of a learned classifier and its performance, where the classifier is based on either gradient boosting or Lasso (we use the former). Our implementation of these metrics closely follows Locatello *et al.* [61].

Latent Interpolation: we randomly pick two images $\boldsymbol{x}_1, \boldsymbol{x}_2$ from each dataset, encode them to obtain $\boldsymbol{z}_1, \boldsymbol{z}_2$, and then decode back the latents along a linear segment that connects between the endpoints: $\boldsymbol{z}_1 + (\boldsymbol{z}_2 - \boldsymbol{z}_1) \cdot t$ for $t \in [0, 1]$, which results in a visualization of the latent traversal.

Principal Component Analysis (PCA) [100, 127]: we

Table 4. **Ablations on ShapeNet**, varying the **camera viewpoint encoding schemes**, including **coordinate** system: Polar or Cartesian, rays' representation **method**: by origin and direction $[\boldsymbol{o}, \boldsymbol{d}]$, or with a weighted sum $\boldsymbol{o} + s_d \cdot \boldsymbol{d}$, and **conditioning** mode: through 2D grid concatenation or vector-based modulation (either of *absolute* or *relative* camera perspective).

Method	Coordinates	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓
Concat	Cartesian	27.21	0.945	0.82	0.040
Concat	Polar	27.10	0.940	0.93	0.041
Sphere	Cartesian	27.12	0.946	0.75	0.040
Normalized	Cartesian	27.00	0.941	0.78	0.041
Plane	Cartesian	27.03	0.943	0.78	0.041
Sphere	**Polar**	**27.42**	**0.947**	**0.74**	**0.039**

Conditioning	Coordinates	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓
Absolute	Polar	23.81	0.870	2.65	0.059
Grid+Relative	Polar	26.81	0.941	0.87	0.040
Grid+Absolute	Polar	27.27	0.946	0.80	0.040
2D Grid	**Polar**	**27.42**	**0.947**	**0.74**	**0.039**

Table 5. **Ablations on ImageNet** for feature modulation (*mod.*), evaluated through reconstruction and classification.

Ablation	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓	Top1 ↑
w/o scale mod.	21.33	0.910	5.33	0.224	70.31
w/o layer mod.	17.61	0.800	9.99	0.377	68.10
sum mod.	13.78	0.541	15.24	0.483	61.46
concat mod.	13.57	0.522	16.68	0.494	59.87
SODA (*default*)	**23.63**	**0.931**	**2.77**	**0.191**	**72.24**

encode a sample set of N images (1,000-10,000), and perform PCA decomposition over the obtained latents $\boldsymbol{z}_{i=1}^{N}$, which yields the latent directions $\boldsymbol{s}_j$ of the greatest variation. We then traverse the latent space along the discovered directions: $\boldsymbol{z} + \boldsymbol{s}_j t$ for $t \in [-\sqrt{\lambda_j}, \sqrt{\lambda_j}]$ where λ_j is the respective eigenvalue and $\sqrt{\lambda_j}$ is the standard deviation along direction $\boldsymbol{s}_j$. Doing so allows us to visualize the impact of these latent directions on the model's generated images, and indeed, we find they strongly correlate with semantically-meaningful manipulations.

Thanks to layer modulation (Section 3.2.1), we can further perform PCA over chosen sub-vectors of $\boldsymbol{z}$ that are responsible for guiding decoder's layers of interest. This enables the discovery of latent directions that control particular levels of granularity, from low-frequency structural aspects to high-frequency factors like texture and color, enhancing the model's overall controllability.

Classifier-based Attribute Manipulation: For datasets with binary attributes annotations, such as CelebA and CUB, we can produce similar visualizations to the ones described above by examining the weight matrix's rows of the linear probes we train for the downstream classification experiments (Section 4.1). Indeed, these probes are trained to capture the latent directions that correspond to the presence or absence of the semantic attribute annotations that accompany the datasets we study. The key difference between the PCA-based approach and this technique is that the former is unsupervised while the latter is not.

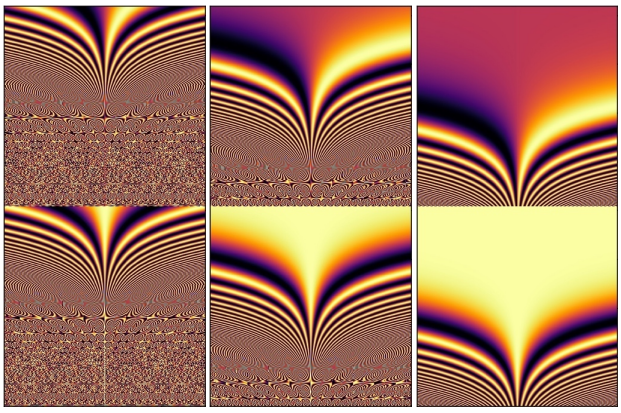

Figure 4. **Positional Encoding Scales.** We visualize the positional encodings in the range of [-1,1], with each embedding visualized vertically within each plot. When the positional encoding scale is set to a too low value (**right column**), the encodings are less distinctive and their capacity is under-utilized (as many of the features (rows) get the the same value for all positions). Meanwhile, a too high value (**left column**) damages the positional encodings empirical performance. We thus make sure to select a medium scale (**middle column**) for optimal performance.

F. Baselines

For each of the tasks we explore, we compare our model to the respective leading approaches, as well as to additional ablated baselines that we design. Here, we list and review all the baseline methods we compare to.

F.1. Task-Agnostic Baselines

First, we implement multiple baselines and ablated models within our diffusion codebase, and report their performance across the range of tasks:

(1) EncDec: a vanilla encoder-decoder $\hat{\boldsymbol{x}} = \mathcal{D}(\mathcal{E}(\boldsymbol{x'}))$, sharing the same encoder and decoder architectures as SODA (for $\mathcal{D}$, the decoding module of the UNet), but being trained to generate the target image $\boldsymbol{x}$ from scratch, with no denoising. We train this model both with and without ($\boldsymbol{x} = \boldsymbol{x'}$) data augmentation.

(2) Uncond: an unconditional diffusion model $\mathcal{D}$ (see also DDAE [19]). To obtain an encoding $\boldsymbol{z}$ for an image $\boldsymbol{x}$, we compute $\mathcal{D}(\tilde{\boldsymbol{x}})$ over a lightly-noised version of $\boldsymbol{x}$ and pool the activations from the middle layer of the UNet denoiser (The layer index and noise degree are hyperparameters chosen for highest performance).

(3) Palette: a diffusion model that instead of having a dedicated encoder $\mathcal{E}$, concatenates the source image $\boldsymbol{x'}$ to the denoised image $\boldsymbol{x}_t$ directly, and inputs both of them to a UNet denoiser $\mathcal{D}(\boldsymbol{x}_t, \boldsymbol{x'})$ (also known as Image-to-Image diffusion model [30]).

(4) unCLIP (Dall-E2) [8]: a diffusion model that relies on a frozen pretrained CLIP [128] as the encoder $\mathcal{E}$.

Table 6. **Datasets Configuration & Statistics**. ($\star$) LSUN sizes per partition: Bedrooms (3.03M), Church Outdoors (126K), Bird (2.31M), Car (5.52M), Cat (1.66M), Dog (5.05M), Horse (2.00M).

Dataset	Size	Resolution (Raw)	# Categories / Attributes	Augmentation / Views	Source View Noise Scale	ResNet Size	Learning Rate	Batch Size	Guidance
Imagenet1K [117]	1.28M	Varied	1000	RandAugment	0.10	50×2	4×10^{-4}	4096	2
CelebA-HQ [105]	30K	1000	2×40	Gaussian Noise	0.22	50×2	4×10^{-4}	4096	2
LSUN [118]	($\star$)	Varied	-	Gaussian Noise	0.22	50×2	4×10^{-4}	4096	2
AFHQ [119]	15K	512	3	Gaussian Noise	0.22	50×2	4×10^{-4}	4096	2
NMR [88]	1.05M	64	13	24×43.8K	0.00	50	2×10^{-4}	2048	2
ShapeNet [87]	6.28M	256	55	120×52K	0.00	50	2×10^{-4}	2048	2
GSO [86]	120K	256	17	120×1K	0.00	50	2×10^{-4}	2048	2.5
SmallNORB [101]	24.3K	96	18,5,9,6	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
3D-Shapes [104]	480K	64	4,10,10,10,15,8	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
MPI3D (Toy) [102]	1.03M	64	6,6,2,3,3,40,40	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
MPI3D (Realistic) [102]	1.03M	64	6,6,2,3,3,40,40	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
MPI3D (Real) [102]	1.03M	64	6,6,2,3,3,40,40	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
MPI3D (Complex) [102]	461K	64	4,4,2,3,3,40,40	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
Caltech-UCSD Birds (CUB) [104]	11.8K	Varied	2×312	Gaussian Noise	0.05	18	1×10^{-4}	1024	3
Oxford Flowers [120]	8.2K	Varied	102	Gaussian Noise	0.05	18	1×10^{-4}	1024	3

We emphasize that we do not refer here to the already trained Dall-E2 model, but rather to its architecture, and so we train its denoiser (in a comparable size to our model) from scratch along with the frozen pre-trained CLIP encoder, for each dataset of interest.

(5) w/o bottleneck: an ablation of SODA with no bottleneck, which rather encodes the input image $\boldsymbol{x'}$ into a 2D feature grid $\boldsymbol{z}^{w\times h}$, with no global pooling, and conditions the denoising on it through cross-attention (similarly to text-to-image diffusion models [7]).

(6) w/o modulation: an ablation of SODA that broadcasts and concatenates the latent $\boldsymbol{z}$ to linearly-mapped RGB channels of the denoised image $\boldsymbol{x_t}$, instead of applying modulation through adaptive group normalization (also called a spatial broadcast decoder [129]).

F.2. Linear-Probe Classification

For downstream classification, we compare our model to a diverse array of leading self-supervised learning approaches: **generative** methods like MAE [53], BEIT [54] and iGPT [55] split each image into a grid of tokens or patches, mask some patches and predict them back from the unmasked ones, oftentimes using a transformer backbone.

Meanwhile, **discriminative** approaches leverage contrastive learning (as in SimCLR [58]), clustering techniques (as in SwAV [57]), and distillation (as in DINO [56] and BYOL [72]) to derive visual representations. At the core of these methods is a strong reliance on rich data augmentations, which are essentially the driving force that allows the to perform unsupervised clustering.

Consequently, contrastive learning approaches operate well at tasks that involve identification of an image's category, as is the case for ImageNet, but may struggle to capture finer traits that are altered by the augmentations. The semantic properties they may or may not encode into the learned representations heavily depend on the particularities of the data augmentation scheme they employ, since they are basically encouraged to form a latent space that is invariant to the augmentation applied, instead casting different augmentations into similar representations.

Contrary to these two kinds of approaches, both of which are unsuitable for high-quality image generation, SODA stands out being able to both encode input images into meaningful latents, and also **synthesize** back crisp output images, conditionally and unconditionally. It learns **compact** and disentangled representations, which contrast with the large, potentially discrete, 2D grids learned by alternative approaches, and as demonstrated in Section 4.1, is **robust to the chosen data augmentation** scheme, operating well even in its absence.

Our comparison to the approaches discussed in this subsection relies on the performance reports in their respective publications over the ImageNet1K dataset, with the exception of the crop+flip accuracy for SwAV and DINO's for which we retrain the models.

F.3. Image Reconstruction

We examine the performance of varied models for the task of image reconstruction: Dall-E [74] and VQGAN [75] employ a **discrete variational** auto-encoder [130], which casts input images into 2D token grids, based on a trainable codebook. These approaches then couple the auto-encoder with a prior-distribution model, to enable unconditional image synthesis. However, for our purposes (image reconstruction), we consider the auto-encoder module only.

The **adversarial** StyleGAN model [49] can also be used for image reconstruction, by applying optimization-based inversion techniques to infer back latents from images. Given an image x, they leverage gradient descent to reverse engineer the latent z that gives rise to an output $\hat{x}$ that is as close as possible to the image x while still staying on the model's learned manifold. While these techniques tend to produce samples that share semantic properties with the source images, they oftentimes fail to reconstruct them

Table 7. **Ablations on CelebA**, evaluted through classification, reconstruction, and disentanglement. The no-bottleneck ablation ($\star$) has skip connections between the model's input and output, making the reconstruction task trivial, but simultaously damaging the learned representations' quality. *Disen.* stands for Disentanglement, *Comp.* for Completeness, and *Info.* for Informativeness.

Ablation	F1 ↑	Disen. ↑	Comp. ↑	Info. ↑	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓
w/o bottleneck	58.89	36.61	26.12	83.15	27.29*	0.985*	6.61*	0.099*
w/o modulation	60.38	64.51	40.85	84.11	14.53	0.734	20.58	0.332
w/o layer modulation	70.90	67.25	42.53	87.06	16.79	0.821	10.35	0.288
w/o layer mask	70.36	68.41	43.41	87.45	16.98	0.820	11.39	0.285
w/o scale modulation	69.65	67.41	44.84	87.63	16.53	0.816	10.49	0.290
sum modulation	68.31	61.12	40.53	85.88	16.71	0.816	17.95	0.290
concat modulation	67.58	61.34	40.67	86.02	16.67	0.816	17.35	0.289
SODA (ResNet50, *default*)	**71.63**	**73.90**	**48.81**	**87.64**	**18.24**	**0.842**	**10.09**	**0.275**

faithfully. Finally, we compare our model to the **diffusion-based** DiffAE [48], which, in contrast to our study, focuses on auto-encoding only, and can be viewed as a predecessor of our approach, as discussed in Section 2.

We assess the reconstruction capabilities of the approaches described in this subsection by evaluating a sample set of images produced by their associated public pre-trained checkpointed models.

F.4. Novel View Synthesis

For novel view synthesis of 3D objects, we compare SODA to a collection of **geometry-free and -aware** approaches designed for few-shot settings: PixelNeRF [60] learns to translate a small number of source views into a neural radiance field, and then use volumetric rendering techniques to generate new ones. NeRF-VAE [59] extends this idea by leveraging amortized variational inference to learn probablistic neural scene representations. In contrast to these specialized methods, designed specifically for 3D environments, SODA proves considerably more versatile, successfully addressing a broader spectrum of tasks and datasets.

As an alternative to differentiable rendering, geometry-free approaches often use attention mechanisms to directly transform source views into targets: Scene Representation Transformer (SRT) [80] parametrizes scenes with the computationally lighter and faster Light-Field formulation [131], and synthesize output views from new perspectives by directly attending to the input views' encodings. The diffusion-based 3DiM [89] goes further and makes extensive use of cross-attention throughout all of its network's layers so to directly map sources to targets. In contrast to these approaches, we intentionally introduce a bottleneck into our model that induces a meaningful and compact latent space. This, in turn, offers much tighter control over the model's generative process, opening the door for both semantic manipulation of given scenes, as well as unconditional synthesis of new ones – two new capabilities that are out of these prior works' reach.

We evaluate the methods described in this subsection either using the authors' official implementations (for NeRF-VAE), or with our own re-implementations (for PixelNeRF and SRT), matching the originally reported performance.

F.5. Disentanglement

In terms of disentanglement, we analyze SODA over a suite of semantically-annotated datasets, and compare it with a series of **variational** approaches [66], which are traditionally known for encouraging the formation of disentangled representations: β-VAE [67], re-weights the KL regularization term to constrain the latents' capacity; AnnealedVAE [106] slowly relaxes the encoder-decoder bottleneck so to foster gradual learning; FactorVAE [63] and β-TCVAE [108] encourage factorization of the latent distribution by reducing the correlations among the axes; DIP-VAE [107] (variants I and II) penalizes the mismatch between the prior and the posterior, so to similarly encourage factorization within the latter. We evaluate these methods using the official *disentanglement-lib* TensorFlow repository [61], while modifying the backbone encoder and decoder architectures to match the ones used in SODA, for better comparability.

G. Ablation Studies

To gain better insight into the relative contributions our design decisions make, we conduct thorough ablation and variation studies for each of the model's components, inspecting the (1) **feature modulation** used to propagate information between the encoder and the denoiser, (2) **data augmentation** strategies for the source and target views, encoding and conditioning schemes of (3) **positional information** for our 3D multi-view experiments, (4) **sampling configurations** of the denoising process and its classifier-free guidance, and finally, (5) the encoder and denoiser's respective **sizes, dimensions and learning rates**.

This study joins ablations presented through the main paper (Sections 4.1 and 4.3.2) that attest to the strengths and benefits of the model's core aspects and key innovations, like bottleneck compactness (Section 3.2), layer modulation (Section 3.2.1), redesigned noise schedule (Section 3.4), and incorporation of novel view synthesis as a self-supervised training objective (Section 3.3).

G.1. Feature Modulation

We explore multiple modulation variants and examine how they fare in terms of generative skills and downstream per-

Table 8. **Hyperparameters** of our model, including the encoder, denoiser, linear probe, and view aggregation transformer (for 3D experiments), as well as optimization, sampling and augmentation settings. ($\star$) Depends on the dataset. (†) Applied for ImageNet only. ($*$) We use a base value of 64 for ImageNet pre-training for downstream classification, and 128 otherwise.

Hyperparameter	Value
Optimization	
Learning Rate*	(1-4)$\times 10^{-4}$
Batch Size*	1024-4096
Learing Rate Schedule†	Cosine Decay
Learing Rate After Decay†	0.25$\times$LR
Learning Rate Decay Steps†	1.2×10^5
Weight Decay	0.05
EMA Decay Rate	0.9999
Warmup Steps	5000
Gradient Clipping Norm	0.5
Optimizer	Adam
β_1	0.9
β_2	0.95
Model	
Latent Dimension*	128-2048
Bottleneck Dropout	0.1
Classifier-Free Guidance Masking Rate	0.12
Layer Masking Rate	0.15
Positional Encoding Dimension	512
Positional Encoding Scale (Figure 10)	0.0001
Encoder	
Architecture	ResNet
Size*	18, 50, 50$\times$2
Version	v2
Resolution	256 (224†)
LR multiplier	2
DropPath [112]	0.1
View Aggregation Transformer	
Depth	2
Attention Heads Number	4
Hidden Layer Multiplier	4
Denoiser	
Architecture	UNet
Resolution	128
Base Channels*	64-128
Channels multipliers	1,1,2,3,4
Residual blocks per resolution	2
Selt-Attention resolution	8,16,32
Attention Head Dimension	64
Normalization Type	GroupNorm [51]
Dropout Rate	0.1
Sampling	
Classifier-Free Guidance*	2-3
Diffusion Training Steps	1000
Sampling Strided Steps*	75-250
Linear Probe	
Weight Initialization Scale	0.02
Bias Initialization Scale†	-10.0
Dropout Rate	0.1
Augmentation Rate (RandomResizedCrop)	0.65
Label Smoothing†	0.1
Data Augmentation	
Augmentation Rate (Cropping+Flipping)	0.95
Distortation Rate (RandAugment [79])	0.65
Distortion Layers Number	2
Distortion Magnitude	9
Distortion Magnitude STD	0.5
Source View Noise Scale*	0-0.22

formance (Tables 7, 9 and 11). As the results suggest, **modulation-based conditioning** proves considerably more effective than alternative mechanism such as input concatenation $[\boldsymbol{x}_t, \boldsymbol{x'}]$ (Palette [30]) or spatial broadcasting (*w/o modulation*) [129], with respective deltas of 15.5% and 12.3% at classification over ImageNet (top1) and CelebA (F1), and 0.32 (out of 1.0) mean SSIM improvement at novel view synthesis. **Layer modulation** proves beneficial too, enhancing disentanglement scores, with up to 13.9% improvement, and generative capabilities, with 0.13 increase in SSIM and halving of LPIPS for ImageNet reconstructions.

We further assess ways to integrate the guidance of the timestep t and the latent $\boldsymbol{z}$, and as an alternative to our **two-stage guidance** approach, where the denoiser's activations $\boldsymbol{h}$ are modulated first by t and subsequently by $\boldsymbol{z}$, we map them instead to a single pair $(\boldsymbol{w}_s, \boldsymbol{w}_b)$ either through summation or concatenation (*sum/concat mod.*) which is then used to modulate the activations: $\text{AdaGN}(\boldsymbol{h}, t, \boldsymbol{z}) = \boldsymbol{w}_s\text{GroupNorm}(\boldsymbol{h}) + \boldsymbol{w}_b$ However, our two-step strategy proves stronger than this variant. Likewise, **scaling the features multiplicatively** with $\boldsymbol{z}_s$, as opposed to adding a bias term $\boldsymbol{z_b}$ only, leads to small improvements across different datasets.

G.2. Data Augmentation

We examine the impact of data augmentations on the model's performance, and analyze variations of the augmentation method itself as well as the inputs it is applied to (Figure 6). At training, our model receives two inputs: a clean view $\boldsymbol{x'}$ processed by the encoder $\mathcal{E}$, and a noisy view $\boldsymbol{x}_t$ denoised by the decoder $\mathcal{D}(\boldsymbol{x}_t|\boldsymbol{x'})$, aiming to recover $\boldsymbol{x} = \boldsymbol{x}_0$. With the exception of native multi-view datasets (e.g. ShapeNet), we create the source and target views $\boldsymbol{x'}$ and $\boldsymbol{x}_t$ by applying random data augmentations at each training step on the original image $\boldsymbol{x}$ (from the dataset).

We test the impact of applying augmentations either just to the source view, just to the target view, to both, or to none of them. We observe that **augmenting the source** is more critical than the target in terms of its influence on downstream classification performance, and that the model still achieves 55.1 when the source and the target views remain equal. Moreover, we find it valuable to add **low Gaussian noise** to the source views read by the encoder, yielding 1.3% improvement in ImageNet classification accuracy and improving LPIPS scores relatively by 33%.

G.3. Pose Conditioning

We compare different encoding schemes of the camera perspectives for the 3D novel view synthesis task (Table 8). Given a camera pose $\boldsymbol{p}$, we can use a closed-form calculation to derive a 2D grid of rays $\boldsymbol{r} = (\boldsymbol{o}, \boldsymbol{d})$ of dimension $H\times W\times 6$ with origins $\boldsymbol{o}$ and directions $\boldsymbol{d}$. We can then represent each ray through concatenation: $[\boldsymbol{o}, \boldsymbol{d}]$ (*concat*), as commonly done in prior works [60, 80], or instead, express them with a parametric sum: $\boldsymbol{o} + s_d{\cdot}\boldsymbol{d}$, where s_d is

Table 9. **Ablations on ShapeNet**, varying the **classifier-free guidance** settings: either masking the latent $\boldsymbol{z}$ that encodes the source image view, masking the pose information $\boldsymbol{r}$ of the source and target views, or independently masking both.

Masking	PSNR ↑	SSIM ↑	FID ↓	LPIPS ↓
ShapeNet				
Latent	**27.42**	**0.947**	**0.74**	**0.039**
Pose	27.16	0.940	0.96	0.041
Latnet + Pose	27.11	0.938	0.95	0.041
GSO				
Latent	24.12	0.937	2.22	0.065
Pose	24.25	0.939	2.48	0.062
Latnet + Pose	**24.97**	**0.945**	**1.51**	**0.054**

a scaling factor that can be chosen in different ways: either normalizing $\boldsymbol{d}$ to a length of 1, casting it onto the image plane, or, as we propose, on a sphere that centers at the object, i.e. the origin (*Normalized*, *Plane* and *Sphere*). We can further describe the rays either using Polar or Cartesian coordinates, embedded with sinusoidal positional encoding [68] as explained above (Appendix C). We compare these alternatives, and find that **casting the rays on a sphere** performs most effectively, and that for this case, **Polar coordinates** outperform the Cartesian ones.

We further experiment with representing the camera pose as a single vector $\boldsymbol{p}$ that captures its position and direction in Polar coordinates, either considering $\boldsymbol{p}_t - \boldsymbol{p}_s$, the relative camera transformation from the source to the target, or concatenating the two absolute viewpoints $[\boldsymbol{p}_s, \boldsymbol{p}_t]$. We then encode the information with sinusoidal positional encoding [68]), and use the resulting vector to guide the denoiser's operation through feature modulation, similarly to the latent $\boldsymbol{z}$. However, the ablations show that integrating the camera perspective by concatenating a 2D grid of rays surpasses both the modulation-based pose conditioning as well as a hybrid alternative that simultaneously uses both techniques.

Finally, we experiment with different masking techniques as part of the classifier-free guidance (Section 3.3), either randomly masking the latent representation $\boldsymbol{z}$ that encodes the source image view, masking the pose information (namely, the rays 2D grid $\boldsymbol{r}$), or independently masking both. We interestingly note that the ideal masking vary for different datasets: while masking of both the latent and pose improves performance for the real-world Google Scanned Objects, it reduces the performance for ShapeNet. Qualitatively, masking both the pose $\boldsymbol{r}$ and the latent $\boldsymbol{z}$ enhances the model's generative flexibility, allowing it to synthesize either novel objects at requested camera perspectives, or arbitrary novel views even at the absence of source or target's pose information (supplementary figures will be added very soon).

G.4. Sampling Configuration

We vary the guidance strength g and timesteps striding l (i.e. number of timesteps used at sampling) and analyze their impact on the generated images' quality along different metrics (Figure 9). The model is robust to variation in both settings, with optimal values commonly achieved at $g = 2$ and $l = 150$ (considering different metrics and datasets). Classifier-free guidance consistently yields higher-quality images than unguided sampling, while too strong guidance (like ≥ 5) results in a slight reduction in scores and potential visual artifacts.

As per the number of sampling steps, while PSNR and LPIPS scores tend to remain constant, we interestingly observe an inverse correlation between the process length's influence on FID vs. SSIM, the former reflecting sharpness and fidelity while the latter capturing similarity to the target: Sampling images over more steps tends to improve their realism, but may simultaneously induce subtle variations, as the samples begin to slightly move away from the mean estimated target. As aforementioned, we find that $l = 150$ offers a favorable balance between these two qualities.