A OVERVIEW

In this supplementary material, we present additional qualitative results for various domains in Section B. Next, we describe the model architecture for our approach in Section D. Finally, we include experiment details on training datasets, baselines, training, and inference in Section E.

B ADDITIONAL RESULTS

We first provide additional results on global factor decomposition and recombination in Section B.1. We then give additional results on object-level decomposition and recombination in Section B.2. Finally, we provide more results that demonstrate cross-dataset generalization in Section B.3.

B.1 GLOBAL FACTORS

Decomposition and Reconstruction. In Figure XI, we present supplemental image generations that demonstrate our approach's ability to capture global factors across different domains, such as human faces and scene environments. The left side of the figure displays how our method can decompose images into global factors like facial features, hair color, skin tone, and hair shape, which can be further composed to reconstruct the input images. On the right, we show additional decomposition and composition results using Virtual KITTI 2 images. Our method can effectively generate clear, meaningful global components from input images. In Figure XII, we show decomposition and composition results on Falcor3D data. Through unsupervised learning, our approach can accurately discover a set of global factors that include foreground, background, objects, and lighting.

Recombination. Figure XIII showcases our approach's ability to generate novel image variations through recombination of inferred concepts. The left-hand side displays results of the recombination process on Falcor3D data, with variations on lighting intensity, camera position, and lighting position. On the right-hand side, we demonstrate how facial features and skin tone from one image can be combined with hair color and hair shape from another image to generate novel human face image combinations. Our method demonstrates great potential for generating diverse and meaningful image variations through concept recombination.

B.2 LOCAL FACTORS

Decomposition and Reconstruction. We present additional results for local scene decomposition in Figure XIV. Our proposed method successfully factorizes images into individual object components, as demonstrated in both CLEVR (**Left**) and Tetris (**Right**) object images. Our approach also enables the composition of all discovered object components for image reconstruction.

Recombination. We demonstrate the effectiveness of our approach for recombination of local scene descriptors extracted from multi-object images such as CLEVR and Tetris. As shown in Figure XV, our method is capable of generating novel combinations of object components by recombining the extracted components (shown within bounding boxes for easy visualization). Our approach can effectively generalize across images to produce unseen combinations.

B.3 CROSS DATASET GENERALIZATION

We investigate the recombination of factors inferred from multi-modal datasets, and the combination of separate factors extracted from distinct models trained on different datasets.

Multi-modal Decomposition and Reconstruction. We further demonstrate our method's capability to infer a set of factors from multi-modal datasets, *i.e.*, a dataset that consists of different types of images. On the left side of Figure XIX, we provide additional results on a multi-modal dataset that consists of KITTI and Virtual KITTI 2. On the right side, we show more results on a multi-modal dataset that combines both CelebA-HQ and Anime datasets.

Multi-modal Recombination. In Figure XX, we provide additional recombination results on the two multi-modal datasets of KITTI and Virtual KITTI 2 on the left hand side of the Figure, and CelebA-HQ and Anime datasets on the right hand side of the Figure.

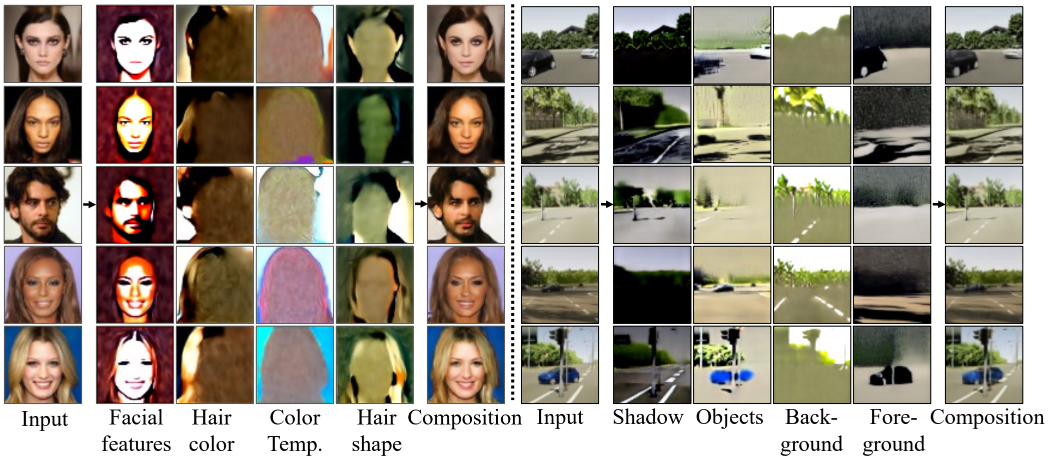


Figure XI: **Global Factor Decomposition.** Global factor decomposition and composition results on CelebA-HQ and Virtual KITTI 2. Note that we name inferred concepts for easier understanding.

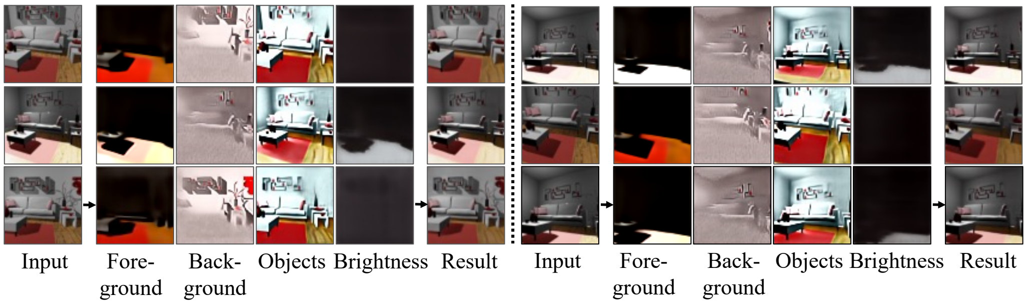
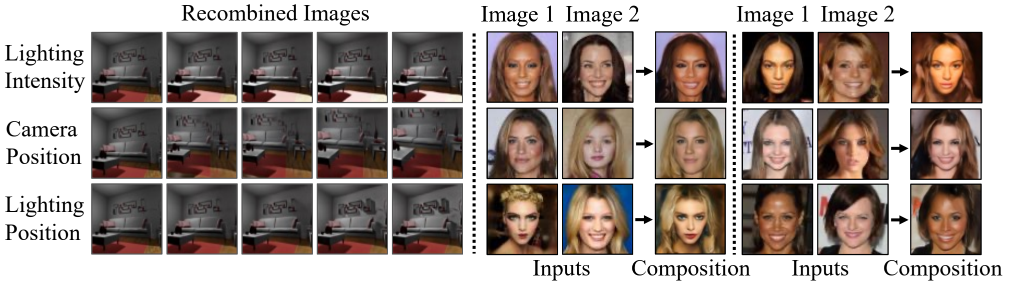


Figure XII: **Global Factor Decomposition.** Global factor decomposition and composition results on Falcor3D. Note that we name inferred concepts for easier understanding.



Facial features, Color temp. (Image 1) + Hair color, Hair shape (Image 2)

Figure XIII: **Global Factor Recombination.** Recombination of inferred factors on Falcor3D and CelebA-HQ datasets. In Falcor3D (**Left**), we show image variations by varying inferred factors such as lighting intensity. In CelebA-HQ (**Right**), we recombine factors from two different inputs to generate novel face combinations.

Cross Dataset Recombination. We also show more results for factor recombination across two different models trained on different datasets. In Figure XXI, we combine inferred object components from a model trained CLEVR images and components from a model trained on CLEVR Toy images. Our method enables novel recombinations of inferred components from two different models.

C Additional Experiments

Impact of the Number of Components K. We provide qualitative comparisons on the number of components K used to train our models in Figure XVI and Figure XVII.

Decomposition Comparisons. We provide qualitative comparisons of decomposed concepts in Figure XVIII.

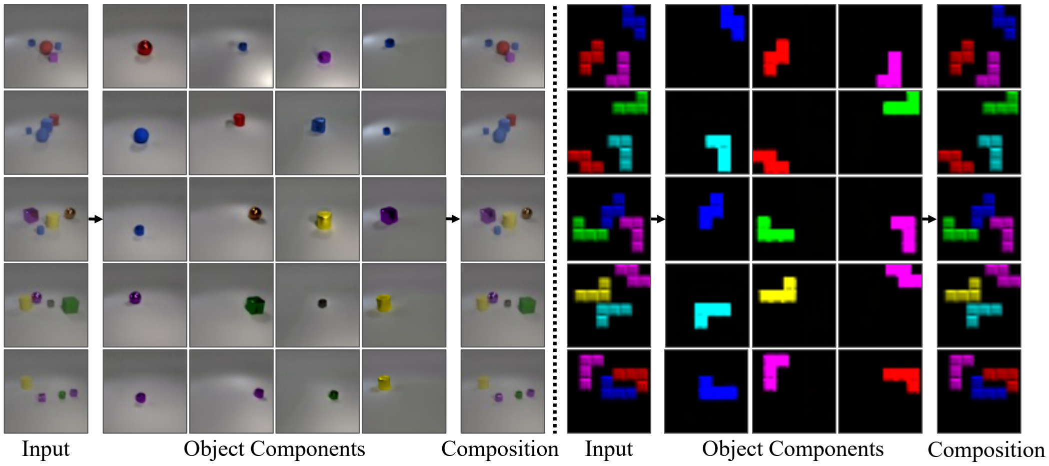


Figure XIV: **Local Factor Decomposition.** Object-level decompositions results on CLEVR (**left**) and Tetris (**right**).

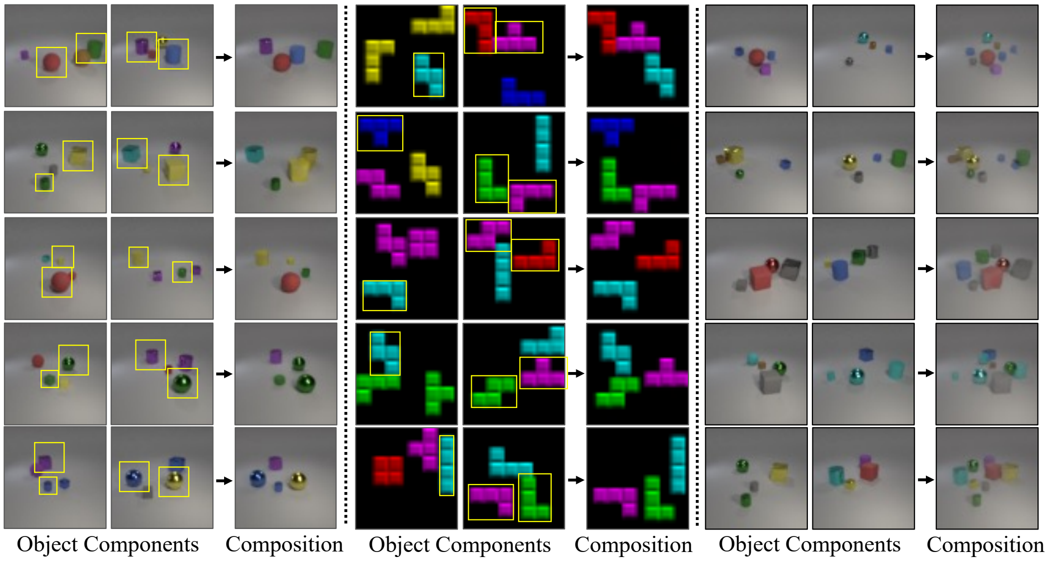


Figure XV: **Local Factor Recombination.** Recombination results using object-level factors from different images.

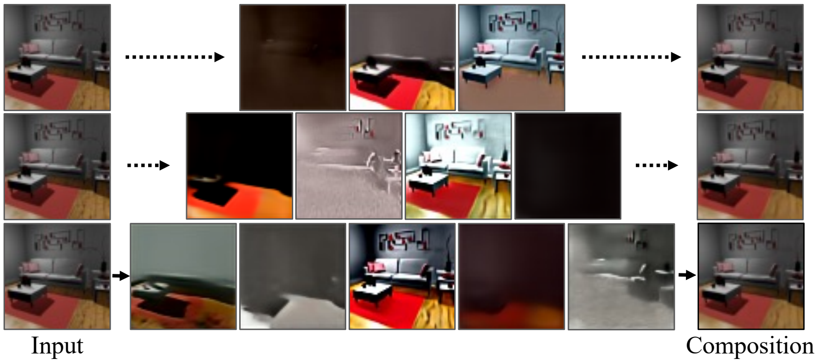


Figure XVI: Decomp Diffusion trained on Falcor3D dataset with varying number of components $K = 3, 4,$ and 5

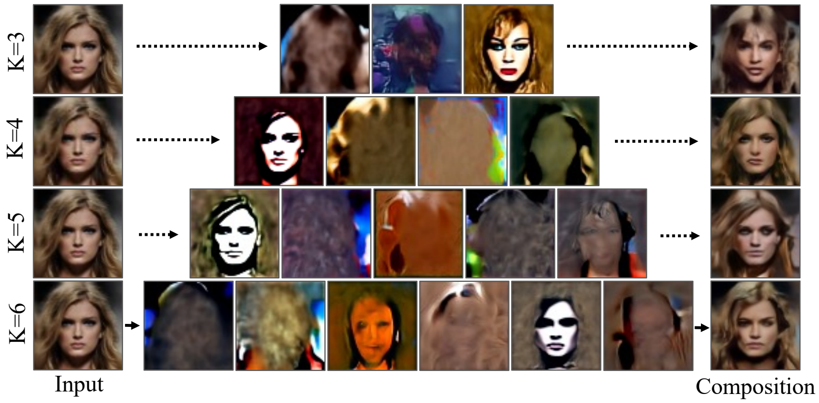


Figure XVII: Decomp Diffusion trained on CelebA-HQ with varying number of components $K = 3, 4, 5,$ and 6

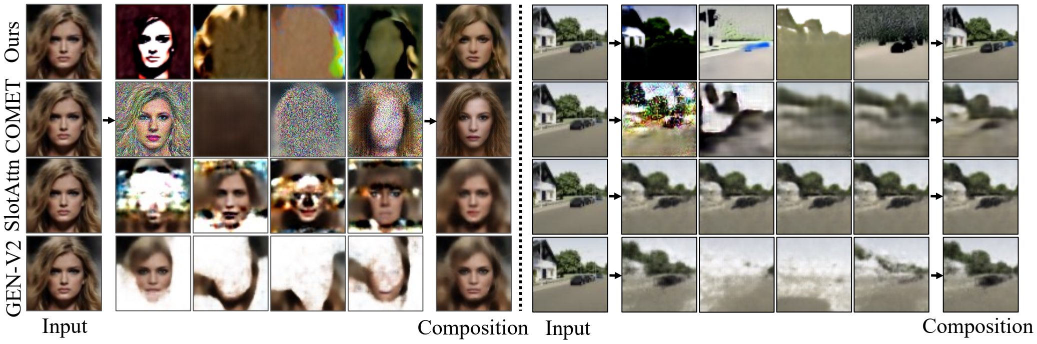


Figure XVIII: **Qualitative comparisons on CelebA-HQ and VKITTI datasets**. Decomposition results on CelebA-HQ (**Left**) and Virtual KITTI 2 (**Right**) on benchmark object representation methods. Compared to our method, COMET generates noisy components and less accurate reconstructions. SlotAttention may produce identical components, and it and GENESIS-V2 cannot disentangle global-level concepts.

D Model Details

We used the standard U-Net architecture from Ho et al. (2020) as our diffusion model. To condition on each inferred latent z_k, we concatenate the time embedding with encoded latent z_k, and use that as our input conditioning. In our implementation, we use the same embedding dimension for both time embedding and latent representations. Specifically, we use 256, 256, and 16 as the embedding dimension for both timesteps and latent representations for CelebA-HQ, Virtual KITTI 2, and Falcor3D, respectively. For datasets CLEVR, CLEVR Toy, and Tetris, we use an embedding dimension of 64.

To infer latents, we use a ResNet encoder with hidden dimension of 64 for Falcor3D, CelebA-HQ, Virtual KITTI 2, and Tetris, and hidden dimension of 128 for CLEVR and CLEVR Toy. In the encoder, we first process images using 3 ResNet Blocks with kernel size 3×3. We downsample images between each ResBlock and double the channel dimension. Finally, we flatten the processed residual features and map them to latent vectors of a desired embedding dimension through a linear layer.

E Experiment Details

In this section, we first provide dataset details in Section E.1. We then describe training details for our baseline methods in Section E.2. Finally, we present training and inference details of our method in Section E.3 and Section E.4.

CLEVR	CLEVR Toy	CelebA-HQ	Anime	Tetris	Falcor3D	KITTI	Virtual KITTI 2
10K	10K	30K	30K	10K	233K	8K	21K

Table IV: **Training dataset sizes.**

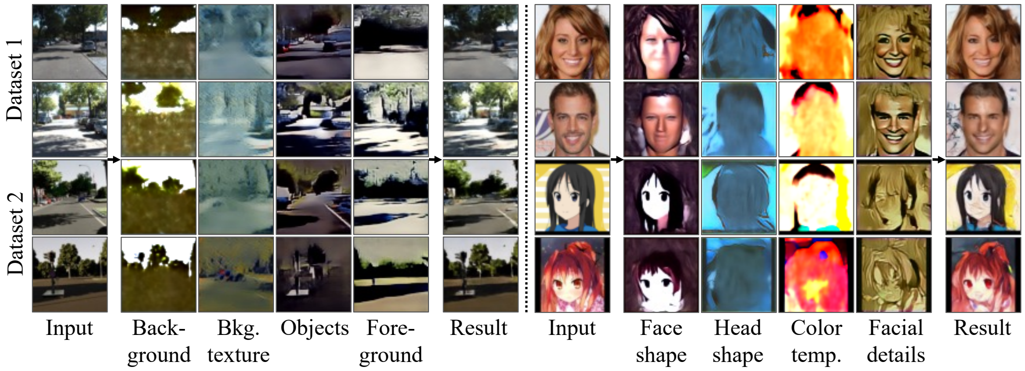


Figure XIX: **Multi-modal Dataset Decomposition.** Multi-model decomposition and composition results on hybrid datasets such as KITTI and Virtual KITTI 2 scenes (**Left**), and CelebA-HQ and Anime faces (**Right**). The top 2 images are of the first dataset, and the bottom 2 images are of the second dataset. Inferred concepts are named for better understanding.

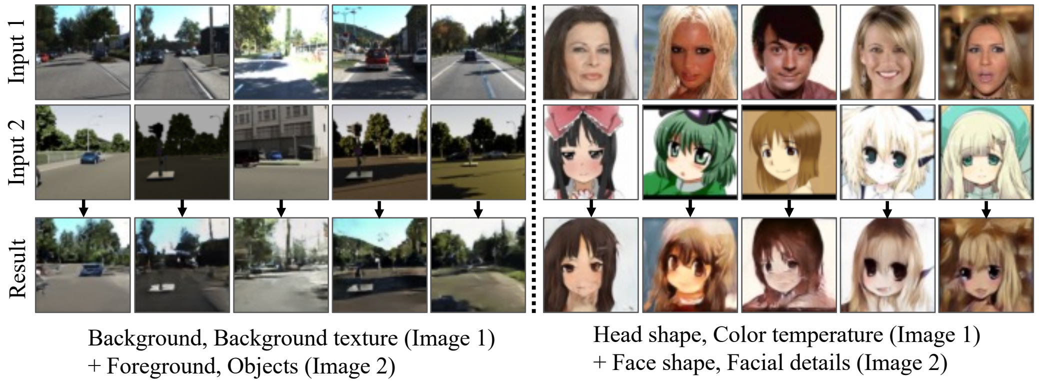

Input 1
Input 2
Result
Background, Background texture (Image 1)
+ Foreground, Objects (Image 2)
Head shape, Color temperature (Image 1)
+ Face shape, Facial details (Image 2)

Figure XX: **Multi-modal Dataset Recombination.** Recombinations of inferred factors from hybrid datasets. We recombine different extracted factors to generate unique compositions of KITTI and Virtual KITTI 2 scenes (**Left**), and compositions of CelebA-HQ and Anime faces (**Right**).

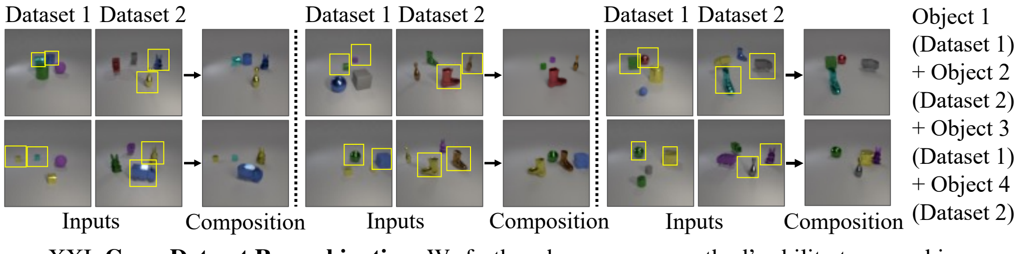


Figure XXI: **Cross Dataset Recombination.** We further showcase our method's ability to recombine across datasets using 2 different models that train on CLEVR and CLEVR Toy, respectively. We compose inferred factors as shown in the bounding box from two different modalites to generate unseen compositions.

E.1 Dataset Details

Our training approach varies depending on the dataset used. Specifically, we utilize a resolution of 32×32 for Tetris images, while for other datasets, we use 64×64 images. The size of our training dataset is presented in Table IV and typically includes all available images unless specified otherwise.

Anime. (Branwen et al., 2019) When creating the multi-modal faces dataset, we combined a $30,000$ cropped Anime face images with $30,000$ CelebA-HQ images.

Tetris. (Greff et al., 2019) We used a smaller subset of 10K images in training, due to the simplicity of the dataset.

KITTI. (Geiger et al., 2012) We used $8,008$ images from a scenario in the the Stereo Evaluation 2012 benchmark in our training.

Virtual KITTI 2. (Cabon et al., 2020) We used $21,260$ images from a setting in different camera positions and weather conditions.

E.2 Baselines

Info-GAN (Chen et al., 2016). We train Info-GAN using the default training settings from the official codebase at https://github.com/openai/InfoGAN.

β-VAE (Higgins et al., 2017). We utilize an unofficial codebase to train β-VAE on all datasets til the model converges. We use $\beta = 4$ and 64 for the dimension of latent $\boldsymbol{z}$. We use the codebase in https://github.com/1Konny/Beta-VAE.

MONet (Burgess et al., 2019). We use an existing codebase to train MONet models on all datasets until models converge, where we specifically use 4 slots, and 64 for the dimension of latent $\boldsymbol{z}$. We use the codebase in https://github.com/baudm/MONet-pytorch.

COMET (Du et al., 2021a). We use the official codebase to train COMET models on various datasets, with a default setting that utilizes 64 as the dimension for the latent variable $\boldsymbol{z}$. Each model is trained until convergence over a period of $100,000$ iterations. We use the codebase in https://github.com/yilundu/comet.

SlotAttention (Locatello et al., 2020b). We use an existing PyTorch implementation to train SlotAttention from https://github.com/evelinehong/slot-attention-pytorch .

GENESIS-V2 (Engelcke et al., 2021b). We train GENESIS-V2 using the default training settings from the official codebase at https://github.com/applied-ai-lab/genesis .

E.3 Training Details

We used standard denoising training to train our denoising networks, with 1000 diffusion steps and squared cosine beta schedule. In our implementation, the denoising network ϵ_θ is trained to directly predict the original image $\boldsymbol{x}_0$, since we show this leads to better performance due to the similarity between our training objective and autoencoder training.

To train our diffusion model that conditions on inferred latents $\boldsymbol{z}_k$, we first utilize the latent encoder to encode input images into features that are further split into a set of latent representations $\{\boldsymbol{z}_1, \dots, \boldsymbol{z}_K\}$. For each input image, we then train our model conditioned on each decomposed latent factor $\boldsymbol{z}_k$ using standard denoising loss.

Each model is trained for 24 hours on an NVIDIA V100 32GB machine or an NVIDIA GeForce RTX 2080 24GB machine. We use a batch size of 32 when training.

E.4 Inference Details

When generating images, we use DDIM with 50 steps for faster image generation.

Decomposition. To decompose an image $\boldsymbol{x}$, we first pass it into the latent encoder Enc_θ to extract out latents $\{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_K\}$. For each latent $\boldsymbol{z}_k$, we generate an image corresponding to that component by running the image generation algorithm on $\boldsymbol{z}_k$.

Reconstruction. To reconstruct an image $\boldsymbol{x}$ given latents $\{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_K\}$, in the denoising process, we predict ϵ by averaging the model outputs conditioned on each individual $\boldsymbol{z}_k$. The final result is a denoised image which incorporates all inferred components, *i.e.*, reconstructs the image.

Recombination. To recombine images $\boldsymbol{x}$ and $\boldsymbol{x}'$, we recombine their latents $\{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_K\}$ and $\{\boldsymbol{z}'_1, \cdots, \boldsymbol{z}'_K\}$. We select the desired latents from each image and condition on them in the image generation process, *i.e.*, predict ϵ in the denoising process by averaging the model outputs conditioned on each individual latent.

To additively combine images $\boldsymbol{x}$ and $\boldsymbol{x}'$ so that the result has all components from both images, *e.g.*, combining two images with 4 objects to generate an image with 8 objects, we modify the generation procedure. In the denoising process, we assign the predicted ϵ to be the average over all $2 \times K$ model outputs conditioned on individual latents in $\{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_K\}$ and $\{\boldsymbol{z}'_1, \cdots, \boldsymbol{z}'_K\}$. This results in an image with all components from both input images.