Supplementary material: Confidence Aware Diffusion

Contents

1. CAD architecture

In this section, we describe the building blocks of the proposed CAD architecture.

1.1. RIN architecture for class conditional CAD

We first explain our adaptation of the RIN architecture [3], that we use in our experiments on class conditional generation in the context of coherence aware diffusion. As shown in Figure 1, the RIN block is composed of two branches: one with the latents and one with the patches. We concatenate the timestep, coherence and conditioning embeddings to the latents (olive, green and orange blocks), with the coherence embedding being an addition of our method to the original RIN architecture. Then, first, the latents gather information from the input patches via Cross-Attention. Second, the latents are processed with N self-attention layers. Finally, the patches are updated from the latents via Cross-Attention. The RIN architecture consists of stacking multiple RIN blocks, where the next RIN block receives the updated latents and patches.

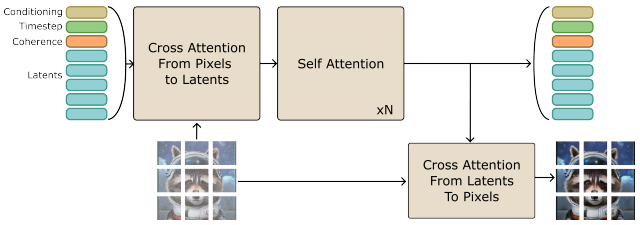

Figure 1. Architecture of the RIN Block modified to receive as input the coherence.

During inference, RIN takes as input a noisy version of the image, a class, a timestep, and a coherence token to predict the noise that has been added to the clean version of the image. To improve the sampling, output latents from a given step are forwarded as input to the next denoising step. For more details, see [3].

1.2. CAD with text conditioning

To enable text-conditioned image generation, we propose a modification to the RIN architecture, coined Text RIN Block (see Figure 2). First, the text tokens are mapped with the coherence with 2x self-attention layers initialized with LayerScale [7] (top part in the figure) and 16 registers [2]. This mapping is the same for every Text RIN block. Note that unlike the class conditional RIN block (Figure 1), in our proposed Text RIN Block, the latent branch contains only the latents; there are no concatenated tokens anymore. Instead, the mapped text tokens, coherence, and timestep embeddings provide information to the latent branch with a cross-attention layer at the beginning of the Text RIN block. Then, the rest of the architecture is the same as the RIN block.

Similarly, during inference our architecture takes as input a noisy version of the image, the timestep and coherence tokens, and a text conditioning that is mapped with a frozen FLAN T5 XL encoder. It then predicts the noise that has been added to the clean image. Output latents from one denoising step are also forwarded to the next denoising step.

2. Implementation details

In this section, we present the implementation details for each experiment as well as the training setup.

2.1. Text Conditining

For the text conditional, we use the LAMB optimizer [8] with a weight decay of 0.01. We use a learning rate of 0.001. The batch size is 1024. We use a linear warmup of the learning rate for the first 10k steps and then use a cosine decay. We train all models for 300k steps. We use an EMA decay of 0.9999 for the last 50k steps.
We use the Stable diffusion VAE encoder [6] and perform the diffusion process in its embedding space in which the image tokens have dimension 32x32x4. We have 4 RIN blocks, each having 4 self-attention units. The data tokens dimension is 256 and the latent token dimension is 768. The input data is reduced to 256 data tokens by using a patch-size of 2.

2.2. Class Conditioning

For the class conditional experiments, we follow the hyperparameters provided by the authors of RIN. We use the LAMB optimizer with weight decay of 0.01. We use a linear warmup of the learning rate for the first 10k steps and then use a cosine decay. We train all models for 150k steps. We use an EMA decay of 0.9999.
For CIFAR-10, we have 3 RIN blocks, each having 2 processing units. The data tokens dimension is 256 and the latent token dimension is 512. We use a patch-size of 2. We use a learning rate of 0.003. The batch-size is 256.
For ImageNet-64, we have 4 RIN blocks, each having 4 processing units. The data tokens dimension is 512 and the latent token dimension is 768. We use a patch-size of 8. We use a learning rate of 0.002. The batch-size is 1024.

2.3. Semantic map conditioning

Generations conditioned on semantic maps were obtained by training a ControlNet [9]. To train the ControlNet, we created a dataset of images selected from the ADE20K [10] and MS COCO [5] datasets. This dataset contains tuples of the form (image, caption, semantic map, coherence map) that were generated from the original images. The captions are obtained with BLIP2 [4], an image captioning language model. We utilize a Maskformer [1] trained either on the MS COCO dataset or ADE20k to generate the semantic maps. To extract coherence values, we use the maximum class probability obtained from the softmax output of the Maskformer model. We employ a MaskFormer trained on ADE20K to generate the coherence map for MS COCO, and conversely, use a MaskFormer trained on MS COCO to obtain the coherence map for ADE20K. This approach helps mitigate the problem of overconfidence in predictions on the training set, reducing the tendency to have only high coherence scores across all pixels. We used a batch size of 16, using the Adam optimizer with a learning rate of 1e-5.

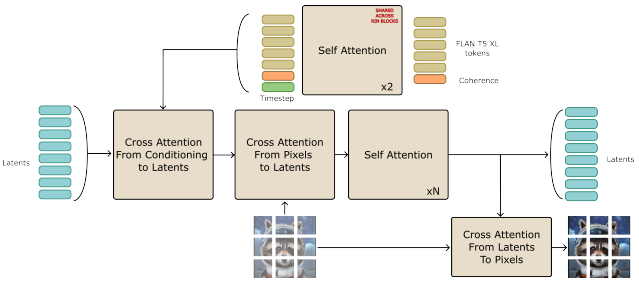

Figure 2. Architecture of the proposed Text RIN Block used in CAD.

2.4. Computational cost

Our method adds negligible training and inference time because we either modify existing architectures or add non-computationally expensive components. Specifically, in terms of architecture, we are replacing one of the latent tokens with an embedded coherence score. For the text-conditional, we do add a new cross-attention layer, but most of the compute is still in the self-attention blocks.

In this project, we have used approximately 25,553 V100 hours for preliminary experiments including the CIFAR-10 experiments and 29,489 A100 hours for ImageNet and text-conditional experiments. Each GPU hour accounts for roughly 259 Wh for a total of 14,255kWh. For semantic segmentation, the training of the different ControlNets was performed using about 1,800 hours of Nvidia A100 GPUs in total, or about 470kWh. The training process required approximately 100 GPU hours for each model trained on ADE20k [10] and 200 GPU hours for MS COCO [5].

3. Image from semantic map additional experiments

In this section, we examine the effectiveness of incorporating coherence maps for semantic segmentation. We qualitatively and quantitatively compare the results of our CAD method against a baseline approach not using the coherence information.

3.1. Quantitative results

Here, we quantitatively evaluate the effectiveness of incorporating coherence maps for semantic segmentation. For this, we experiment on two of the most popular datasets with segmentation: ADE20K [10] and MS COCO [5]. To evaluate our results, we employ the Frechet Inception Distance (FID) and Inception Score (IS) for evaluating image quality. Additionally, Precision (P), Recall (R), Density (D),

Table 1. **Quantitative results on ADE20K when conditioning on semantic maps.** 'CAD bin' encodes the coherence into 5 equally distributed discrete bins. 'CAD scalar' uses a scalar coherence score for the whole image. CAD achieves better FID due to its enhanced ability to generate realistic objects in low coherence regions and superior mIoU as the leaked spatial information from the coherence map and the caption assist it to generate better samples (see Section 3.1 for more details).

	ADE20K						
ControlNet	**FID**	**IS**	**mIoU**	**P**	**R**	**D**	**C**
Baseline	33.67	**14.82**	22.6	0.785	0.757	1.029	0.904
CAD	**30.88**	14.79	**23.7**	**0.844**	**0.824**	**1.0755**	**0.934**
Baseline w/o text	74.37	5.88	5.25	0.657	0.351	0.789	0.515
CAD w/o text	60.21	7.93	11.8	0.619	0.536	0.789	0.682
CAD bin	63.47	7.97	10.8	0.5875	0.4925	0.757	0.663
CAD scalar	74.69	6.15	3.11	0.6495	0.347	0.858	0.536

Table 2. **Quantitative results on MS COCO when conditioning on semantic maps.** 'CAD bin' encodes the coherence into 5 equally distributed discrete bins. 'CAD scalar' uses a scalar coherence score for the whole image (see Section 3.1 for more details).

	COCO						
ControlNet	**FID**	**IS**	**mIoU**	**P**	**R**	**D**	**C**
Baseline	20.1	**32.6**	35.1	0.7876	0.6760	**1.0811**	0.8956
CAD	**18.1**	32.0	**35.3**	**0.8404**	**0.8060**	1.0687	**0.9304**
Baseline w/o text	54.93	15.40	8.36	0.4884	0.4402	0.5297	0.5260
CAD w/o text	37.06	18.04	12.53	0.6222	0.6502	0.7599	0.7052
CAD bin	44.63	16.39	9.76	0.5636	0.5614	0.7075	0.6402
CAD scalar	55.82	15.92	8.10	0.5139	0.4382	0.5294	0.5341

and Coverage (C) serve as manifold metrics, enabling an evaluation of the overlap between the generated and real image manifolds. Finally, we calculate the mean Intersection over Union (mIoU) by utilizing a pre-trained MaskFormer to predict a segmentation map from the generated image and compare it with the original semantic map. This helps illustrate the fidelity of the generated images to the ground truth.

Method comparison. We compare the results of our CAD method with a baseline approach that excludes coherence information in both Table 1 and Table 2, with the complete results. We conduct experiments in two settings: with text (first two rows) and without text (last four rows). Additionally, we compare against two CAD variations. Similar to the binning strategy in text-to-image generation, 'CAD bin' encodes coherence into 5 equally distributed discrete bins. Furthermore, 'CAD scalar' utilizes a singular scalar coherence score for the entire image, equivalent to the mean of the original coherence map.

Results. In Table 1 and Table 2, show the complete results of our method on ADE20k and COCO we demonstrate that using both the segmentation maps and the coherence maps lead to a decrease in FID for both scenarios, including or not the text input. This behavior is expected as our model possesses greater freedom to generate realistic content instead of sticking to the segmentation map uniquely (see *e.g.*, the 4th column of Figure 3). Furthermore, the improvement in the mIoU score can be attributed to two factors. First, when the input segmentation map is of low quality, the baseline method fails to capture important scene information. In contrast, our method benefits from additional information from the coherence map. Secondly, our method better leverages the caption in the low coherence region, mitigating the limitation of the segmentation map's limited number of classes (as seen in 4th row in Figure 4, there is no ping-pong table class in the COCO dataset).

3.2. Additional Visualizations

We show additional results in Figure 3, where, from the left column to the right one, we highlight the segmentation input, the coherence map, the image generated by the baseline, the image generated by our methods and the reference image. The coherence map reveals spatial/shape details of the scene. For instance, when comparing our method to a ControlNet trained solely with the segmentation map, our approach, which incorporates both segmentation and coherence, accurately reconstructs the curtain's shape and the windows in the first row, or reconstructs a cloud in the back of the plane in the second row. Moreover, the efficacy of our method becomes even more apparent when the segmentation map is of poor quality and the coherence score is low. For instance, as shown in the third row, the basic ControlNet attempts to adhere to the limited information provided by the flawed segmentation map, resulting in a scene with multiple arms displayed (third column). In contrast, our approach benefits from the flexibility given by the coherence map, allowing for more consistent image generation. Interestingly, our method also exhibits localized self-correction, as shown on the fourth row. In particular, our model is refraining from generating the hand region due to its low coherence in the input. Finally, we demonstrate the model's responsiveness to textual input in the last row. We present in the last row an example where the original image does not contain snow, but have high coherence in the sky. Both models generate some snow based on the caption, but in line with the coherence map, our model does not generate snow in the sky, thus being closer to the real image.

3.3. Prompt generalization

In this subsection, we demonstrate the sensitivity of our method to the caption input. We observe in Figure 4 that our CAD method can successfully generalize to different types of captions, such as 'dining table', 'billiard', or 'ping pong table' and adjust the scene accordingly. Moreover, even when the table is not explicitly mentioned in the caption (as

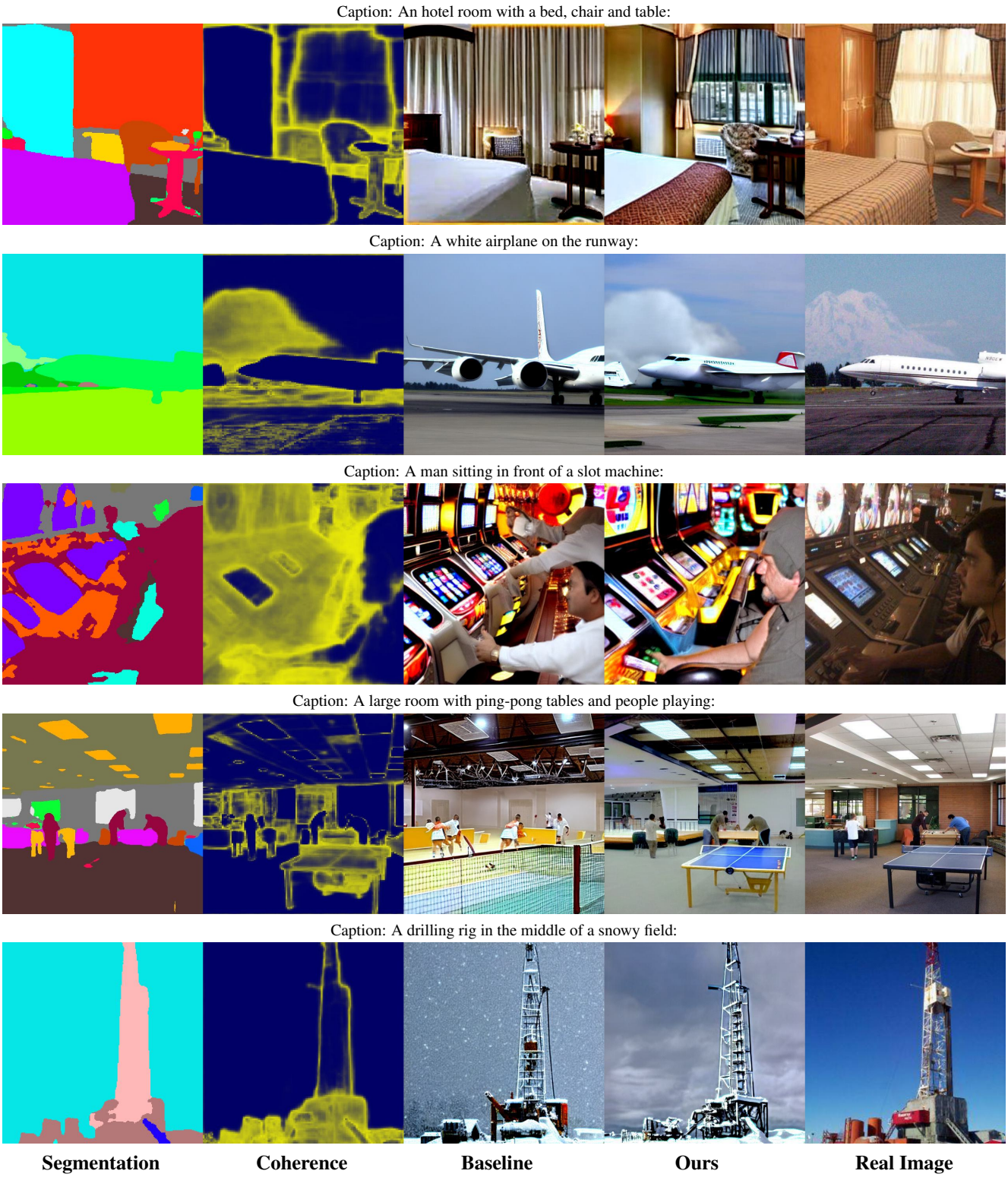

Figure 3. **Qualitative results on ADE20K:** Examples of images generated conditionally to text and semantic maps.

seen on the rightmost side of the figure), our method exhibits strong generalization capabilities and successfully generates the table.

3.4. Coherence Interpolation

In Figure 5, we demonstrate the significance of the coherence map in the conditioning of the ControlNet. In this experiment, we make an interpolation of the coherence map from the maximum coherence score everywhere (left) to very low coherence (right). When the input has high coherence throughout, the generated image lacks the presence of a ping pong table as it is not present in the semantic map. However, as the coherence score decreases, our methods recognize the shape of the ping pong table and successfully generates it in the image. It is worth noting that even at a low scale of the coherence score (second column corresponds to 1e-4 times the original value), our method is still able to reconstruct the table, even if it is not in the segmentation input. When we artificially reduce the coherence (two times less coherence, in the last column), our method is still able to generate a consistent scene without any artifacts.

4. Class conditional experiments

4.1. Simulating annotation noise

Our approach for class-conditional image generation relies on the assumption that the dataset comes with annotated coherence scores. However, such scores are not always available for traditional image-generation datasets. To address this issue, we propose to simulate annotation noise by resampling from a dataset with clean annotations.

We associate an error probability α with each label. We assume that when the annotator is wrong, they misclassify uniformly over all classes, where N is the total number of classes. This leads to the following model:

$$\bar{y} \sim p_{y,\alpha} \text{ with } p_{y,\alpha}(\bar{Y}=k) = \begin{cases} 1-\alpha & \text{if } y=k, \\ \frac{\alpha}{N-1} & \text{otherwise.} \end{cases} \quad (1)$$

We also define a strategy to remap the entire dataset using a normalized entropy-based coherence measure. We use the normalized entropy so that 0 maps to no coherence at all and 1 to total coherence in the label. We define the following coherence function:

$$E(\alpha) = \frac{-1}{\log(N)}\left((1-\alpha)\log(1-\alpha) + \alpha \log\left(\frac{\alpha}{N-1}\right)\right) \quad \text{for } \alpha \in \left[0, \frac{N-1}{N}\right]. \quad (2)$$

To ensure that the dataset has samples with varying levels of confidence, we define a target entropy cumulative distribution. To achieve this, we use a piecewise-linear function:

$$E_{\beta,th}(t) = \begin{cases} t\frac{\beta}{\kappa} & \text{if } t < \kappa, \\ 1 + (t-1)\frac{1-\beta}{1-\kappa} & \text{otherwise.} \end{cases} \quad (3)$$

where κ represents a threshold and β represents the entropy at this threshold. This function construction defines a low entropy region before the threshold and a high entropy region after the threshold.
Finally, for each sample in the dataset (X, y), we sample $t \in \mathcal{U}[0,1]$, and associate a target entropy u. We then compute the associated error probability $\alpha = E^{-1}(u)$, and resample according to $p_{y,\alpha}$ to obtain the tuple $(X, \bar{y}, 1-u)$[1]. This process allows us to generate synthetic data points with varying degrees of annotation noise and coherence.

4.2. Quantitative Results

In this section we add more results on ImageNet with different levels of noise. We observe in Table 3 results on ImageNet for $\beta \in \{0.2, 0.5, 0.8\}$. We first observe the less coherent the labels are the worse the results get in both image quality (FID) but also in accuracy (Acc). However, our method CAD manages to achieve better results than other methods in the context of un-coherent labels. This is further amplified when leveraging coherence aware guidance

Table 3. Quantitative results for class-conditional image generation. Our coherence aware diffusion (CAD) is compared to a baseline model and a training set filtering strategy for different levels of label noise β. We show that CAD achieves higher fidelity and better accuracy.

		ImageNet						
β	**Method**	**FID**	**IS**	**Acc**	**P**	**R**	**D**	**C**
	Conditional	7.56	34.26	0.475	0.595	0.610	0.768	0.706
0.2	Baseline	11.09	23.54	0.264	0.562	0.598	0.670	0.594
	Filtered	8.53	29.27	0.389	0.591	0.609	0.756	0.688
	CAD	8.17	27.75	0.367	0.585	0.615	0.736	0.658
	CA-CFG $\omega = 1$	5.95	68.95	0.679	0.742	0.477	1.203	0.812
0.5	Baseline	14.38	20.46	0.168	0.539	0.579	0.595	0.505
	Filtered	10.20	26.59	0.338	0.573	0.608	0.707	0.645
	CAD	9.11	25.97	0.327	0.571	0.610	0.714	0.633
	CA-CFG $\omega = 1$	5.95	68.95	0.679	0.742	0.477	1.203	0.812
0.8	Baseline	20.10	17.28	0.100	0.502	0.535	0.526	0.417
	Filtered	12.00	24.55	0.292	0.420	0.712	0.647	0.605
	CAD	11.39	22.08	0.248	0.558	0.590	0.682	0.574
	CA-CFG $\omega = 1$	6.70	44.27	0.523	0.695	0.483	1.035	0.743

5. Theoretical analysis

In this section, we motivate the use of coherence as an additional conditioning for diffusion models. Under assumptions

[1]The coherence goes in the opposite direction of the entropy.

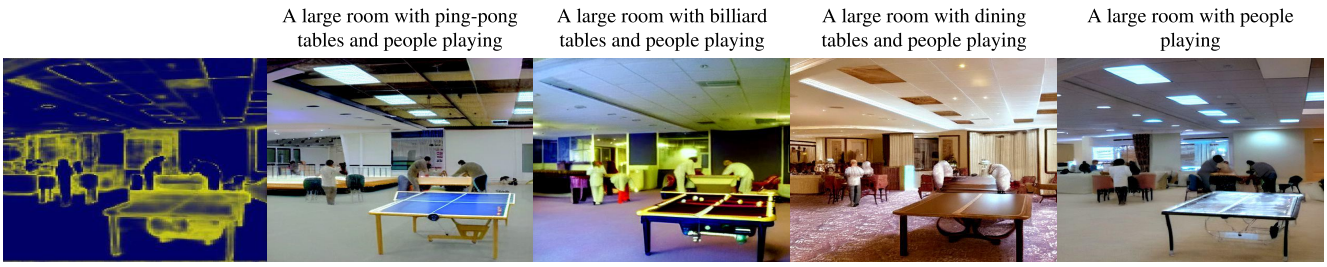

Figure 4. **Caption generalization:** Our methods demonstrate remarkable capability in leveraging the coherence map to generalize to diverse prompt inputs.

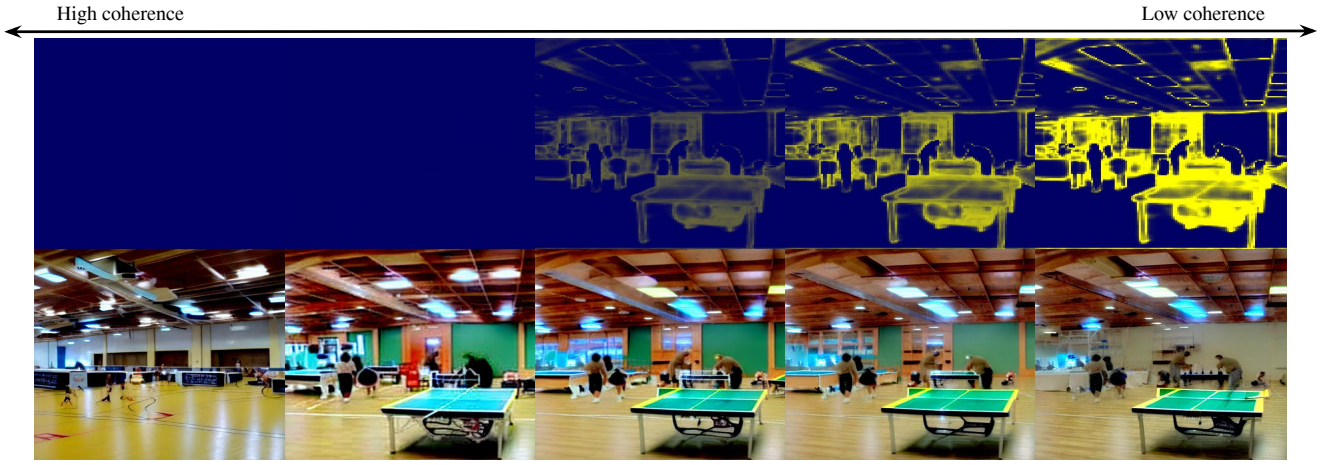

Figure 5. **Coherence interpolation:** In the first column, we artificially provide our ControlNet with a coherence map having the maximum value everywhere. Indeed, our models do not generate the ping pong tables. But, as soon as we decrease the coherence toward its original value, the ping pong tables start to appear. Finally, in the last column, we provide the model with a coherence map that is half as confident as the original value and demonstrate that we can generate an image without artifacts

that are verified empirically, we show that coherence aware diffusion can transition from an unconditional model to a conditional model simply by varying the coherence passed to the model. First, we define a consistency property of the coherence embedding as follows:

Definition 5.1. *We denote* coherence consistent *a conditional embedding $h(y, c)$ of the condition $y \in \mathcal{Y}$ under coherence $c \in [0, 1]$, if $\forall y_1, y_2 \in \mathcal{Y}$ we have*

$$\lim_{c \to 0} \|h(y_1, c) - h(y_2, c)\| = 0 \, . \tag{4}$$

In other words, an embedding is *coherence consistent* if it tends to produce the same vector as the coherence approaches 0. This property is a sufficient condition to constrain the behavior of the diffusion model. Indeed, the following proposition is easily derived from it:

Proposition 5.1. *Lipschitz continuous conditional neural diffusion models that leverage* coherence consistent *embeddings for the conditioning are equivalent to unconditional models at low coherence.*

Proof. We have to prove the following equivalent statement: Let $\epsilon_\theta : x_t, t, h(y, c) \mapsto \hat{\epsilon}_t$ be a Lipschitz continuous neural diffusion model that predicts the noise $\hat{\epsilon}_t$ at time t from

the noisy sample x_t with the help of the condition y embedded using the *coherence consistent* embedding h under coherence c. Then, $\forall \eta > 0$ and $\forall x_t, t, y_1 \neq y_2$, there exists $C > 0$ such that for all $0 < c \leq C$, we have

$$\|\epsilon_\theta(x_t, t, h(y_1, c)) - \epsilon_\theta(x_t, t, h(y_2, c))\|^2 < \eta \,. \quad (5)$$

By Lipschitz property of ϵ_θ, we have $\|\epsilon_\theta(x_t, t, h(y_1, c)) - \epsilon_\theta(x_t, t, h(y_2, c))\|^2 \leq L^2 \|h(y_1, c) - h(y_2, c)\|^2$. From the *coherence consistent* property, there exists $C > 0$ such that for all $0 < c \leq C, \|h(y_1, c) - h(y_2, c)\| < \sqrt{\eta}/L$. □

The following contrapositive necessary condition on the coherence directly follows from this proposition:

Corollary 5.2. *Lipschitz continuous conditional neural diffusion models that leverage* coherence aware *embeddings require high coherence to behave like conditional models.*

In practice, we show in the experiments that the coherence consistency property tends to naturally emerge during training and that consequently coherence aware diffusion provides a tunable prompt parameter to sample from unconditional to conditional models.

6. Additional Qualitative Results

In this section, we provide additional samples from our method. Most of the prompts are sampled from the Lexica.art website.

References

[1] Bowen Cheng, Alexander G. Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *NeurIPS*, 2021. 2

[2] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. *arXiv preprint arXiv:2309.16588*, 2023. 1

[3] Allan Jabri, David Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. *Int. Conf. Mach. Lear.*, 2022. 1

[4] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. *Int. Conf. Mach. Lear.*, 2023. 2

[5] 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. *ECCV*, 2014. 2

[6] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *CVPR*, 2022. 2

[7] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *Int. Conf. Mach. Lear.*, 2021. 1

[8] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *ICLR*, 2019. 2

[9] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *ICCV*, 2023. 2

[10] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. *CVPR*, 2017. 2

Figure 6. Samples from our CAD model at 512 resolution with associated caption

An old-world galleon navigating through turbulent ocean waves under a stormy sky lit by flashes of lightning

an oil painting of rain at a traditional Chinese town

portrait photo of a asia old warrior chief tribal panther make up blue on red side profile looking away serious eyes 50mm portrait photography hard rim lighting photography

a blue jay stops on the top of a helmet of Japanese samurai background with sakura tree

A cute little matte low poly isometric cherry blossom forest island waterfalls lighting soft shadows trending on Artstation 3d render monument valley fez video game.

Underwater cathedral

A cozy gingerbread house nestled in a dusting of powdered sugar snow adorned with vibrant candy canes and shimmering gumdrops

a teddy bear wearing blue ribbon taking selfie in a small boat in the center of a lake

Pirate ship trapped in a cosmic maelstrom nebula rendered in cosmic beach whirlpool engine volumetric lighting spectacular ambient lights light pollution cinematic atmosphere art nouveau style illustration art artwork by SenseiJaye intricate detail.

Figure 7. Samples from our CAD model at 512 resolution

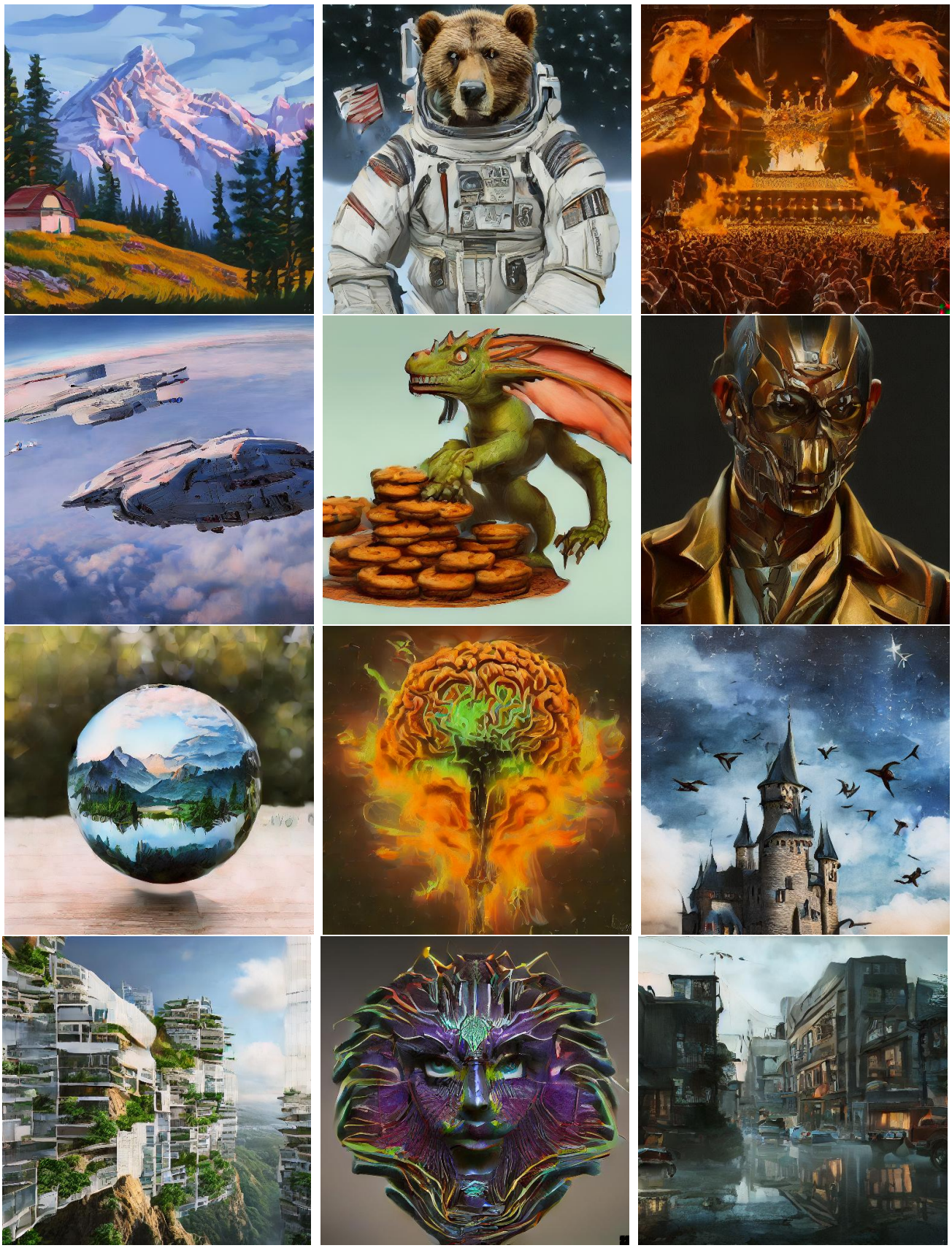

Figure 8. Samples from our CAD model at 512 resolution

Figure 9. Samples from our CAD model at 512 resolution

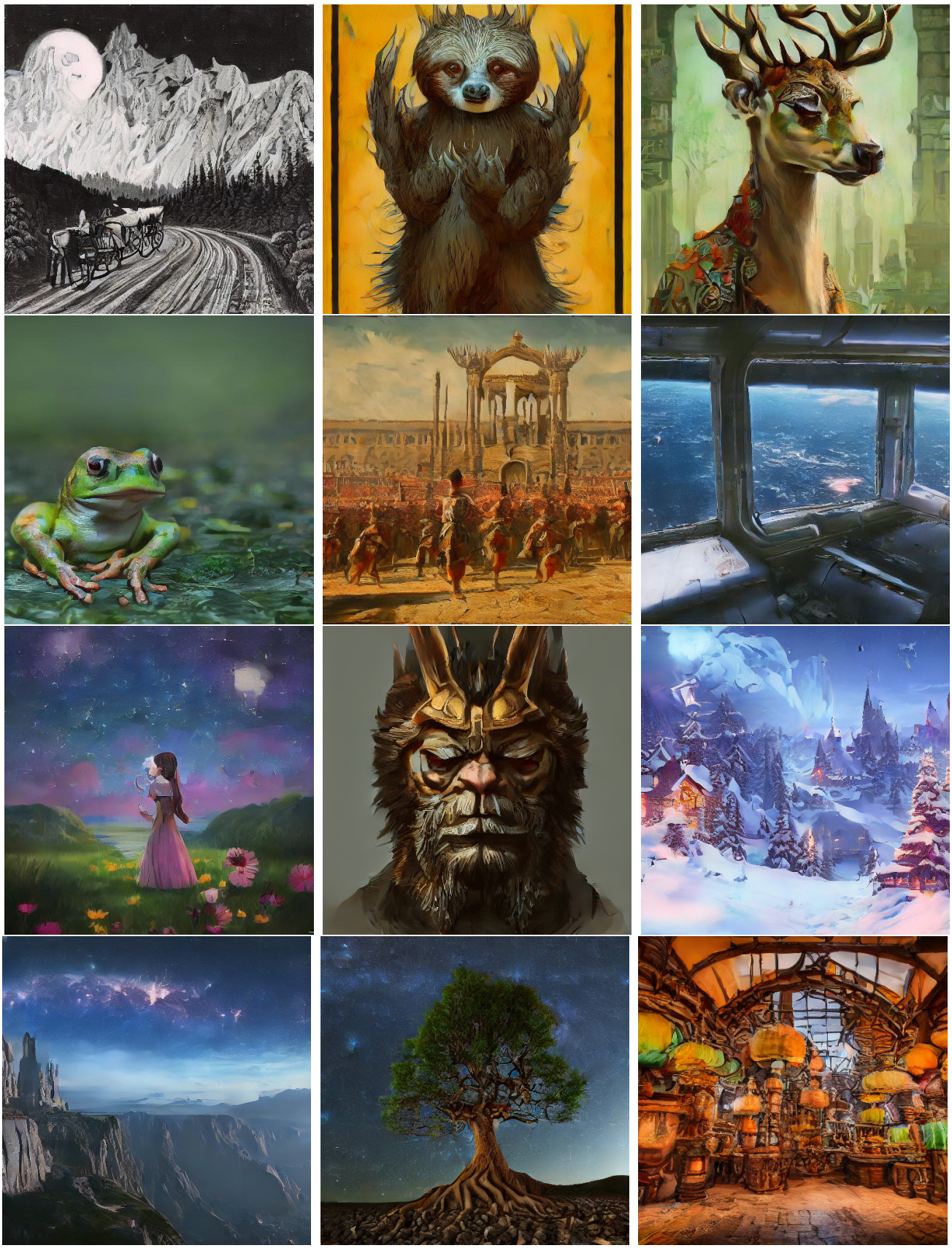

Figure 10. Samples from our CAD model at 512 resolution