

# Appendix

## A Derivations

### A.1 Extended Discussion of Proposition 1

**Why a Single Block ( $T = 1$ ) Cannot Be Universal.** With only one autoregressive-flow block,

$$\mathbf{x}_d = \mu_\theta(\mathbf{x}_{<d}) + \sigma_\theta(\mathbf{x}_{<d}) \mathbf{z}_d, \quad \mathbf{z}_d \sim \mathcal{N}(0, 1), \quad d = 1, \dots, D,$$

each conditional  $p(\mathbf{x}_d | \mathbf{x}_{<d})$  is *necessarily a single Gaussian*. Because no latent variable influences the affine parameters *beyond* the current coordinate, the model cannot represent multimodal densities or heavy tails. Consequently,  $T = 1$  flows are *not* dense in  $L^1(\mathbb{R}^D)$  and fail the universal-approximation criterion.

**Why  $T = 2$  Is Almost Sufficient.** For  $T = 2$  blocks with *opposite* orderings, all coordinates except the last ( $d < D$ ) are expressed as infinite Gaussian mixtures (Eq. (5)) and hence enjoy the universal-approximation property via the density of Gaussian mixtures (Goodfellow et al., 2016). The principal reason why  $\mathbf{x}_D$  fails to possess the universal approximation property lies in the structure:

$$\mathbf{x}_D = \mu_\theta^b(\mathbf{x}_{<D}) + \sigma_\theta^b(\mathbf{x}_{<D}) \cdot \mathbf{y}_D, \quad (10)$$

where  $\mathbf{y}_D$  is defined as

$$\mathbf{y}_D = \mu_\theta^a(\mathbf{y}_{>D}) + \sigma_\theta^a(\mathbf{y}_{>D}) \cdot \mathbf{z}_D \quad (11)$$

$$= \mu_\theta^a(\emptyset) + \sigma_\theta^a(\emptyset) \cdot \mathbf{z}_D. \quad (12)$$

It is evident that  $\mathbf{y}_D$  follows a unimodal Gaussian, since  $\mathbf{z}_D$  is sampled from a unimodal Gaussian prior and the functions  $\mu_\theta^a$  and  $\sigma_\theta^a$  receive no random variable input, regardless of their nonlinearity. Consequently,  $\mathbf{x}_D$  also becomes a unimodal Gaussian, inheriting this limitation from  $\mathbf{y}_D$ .

**Why  $T \geq 3$  Restores Universality.** Introducing a *third* block injects fresh latent variables that feed into the affine parameters of the final coordinate. In effect,  $\mathbf{x}_D$  now depends on a random input produced by the second block, exactly as  $\mathbf{x}_d$  ( $d < D$ ) depends on  $\mathbf{y}_{>d}$  in the  $T = 2$  proof. Consequently every conditional  $p(\mathbf{x}_d | \mathbf{x}_{<d})$  becomes an (infinite) Gaussian mixture, and the entire joint density is dense in  $L^1(\mathbb{R}^D)$ . Additional blocks ( $T > 3$ ) only enlarge the model class and do not degrade this property.

In summary,  $T = 1$  flows are fundamentally limited to unimodal Gaussians;  $T = 2$  flows with alternating orderings achieve universality on  $D - 1$  coordinates but leave the final one unimodal; and  $T \geq 3$  flows overcome this last obstacle, granting full universal approximation power.

### A.2 Proof of Proposition 2

*Proof.* For an isotropic Gaussian  $p(\mathbf{x}) = \mathcal{N}(\mu, \sigma^2 I)$  the score is

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}) = -\frac{\mathbf{x} - \mu}{\sigma^2}.$$

Hence

$$\nabla_{\mathbf{x}} \log p_c(\mathbf{x}) = -\frac{\mathbf{x} - \mu_c}{\sigma_c^2}, \quad \nabla_{\mathbf{x}} \log p_u(\mathbf{x}) = -\frac{\mathbf{x} - \mu_u}{\sigma_u^2}.$$

**Step 1: Guided score.** Insert these into Eq. (8) (CFG):

$$\begin{aligned} \nabla_{\mathbf{x}} \log \tilde{p}_c(\mathbf{x}) &= (1 + \omega) \left( -\frac{\mathbf{x} - \mu_c}{\sigma_c^2} \right) + \omega \left( \frac{\mathbf{x} - \mu_u}{\sigma_u^2} \right) \\ &= - \left[ \left( \frac{1+\omega}{\sigma_c^2} - \frac{\omega}{\sigma_u^2} \right) \mathbf{x} - \left( \frac{1+\omega}{\sigma_c^2} \mu_c - \frac{\omega}{\sigma_u^2} \mu_u \right) \right]. \end{aligned} \quad (13)$$

850 **Step 2: Match to a Gaussian form.** Any Gaussian  $\mathcal{N}(\tilde{\mu}_c, \tilde{\sigma}_c^2 I)$  has score  $-(\mathbf{x} - \tilde{\mu}_c)/\tilde{\sigma}_c^2$ . Equating  
 851 with Eq. (13) gives, for all  $\mathbf{x}$ ,

$$\frac{1}{\tilde{\sigma}_c^2} = \frac{1 + \omega}{\sigma_c^2} - \frac{\omega}{\sigma_u^2}, \quad (14)$$

$$\frac{\tilde{\mu}_c}{\tilde{\sigma}_c^2} = \frac{1 + \omega}{\sigma_c^2} \mu_c - \frac{\omega}{\sigma_u^2} \mu_u. \quad (15)$$

852 **Step 3: Solve for  $\tilde{\sigma}_c$ .** Let  $s := \sigma_c^2/\sigma_u^2 (> 0)$ . Rewrite Eq. (14):

$$\frac{1}{\tilde{\sigma}_c^2} = \frac{(1 + \omega) - \omega s}{s \sigma_u^2} \implies \tilde{\sigma}_c^2 = \frac{s \sigma_u^2}{(1 + \omega) - \omega s} = \frac{\sigma_c^2}{1 + \omega - \omega s},$$

853 so that

$$\tilde{\sigma}_c = \frac{\sigma_c}{\sqrt{1 + \omega - \omega s}}.$$

854 **Step 4: Solve for  $\tilde{\mu}_c$ .** Multiplying Eq. (15) by  $\tilde{\sigma}_c^2$  and substituting the expression above yields

$$\tilde{\mu}_c = \frac{(1 + \omega)\mu_c - \omega s \mu_u}{1 + \omega - \omega s} = \mu_c + \frac{\omega s}{1 + \omega - \omega s} (\mu_c - \mu_u).$$

855 □

## 856 Additional Discussion.

- 857 • **Consistency with standard CFG.** When the two Gaussians share the same variance ( $\sigma_c = \sigma_u \implies$   
 858  $s = 1$ ), Eq. (9) reduces to  $\tilde{\sigma}_c = \sigma_c$  and  $\tilde{\mu}_c = \mu_c + \omega(\mu_c - \mu_u)$ , exactly matching the conventional  
 859 CFG used in diffusion models (Ho & Salimans, 2021).
- 860 • **Numerical stability.** The denominator  $1 + \omega - \omega s$  can approach 0 or even become negative when  
 861  $s$  is large, causing  $\tilde{\sigma}_c^2$  to blow up or change sign. Intuitively, guidance should *sharpen*  $p_c$ , which  
 862 entails  $\tilde{\sigma}_c^2 \leq \sigma_c^2$ , i.e.  $1 + \omega - \omega s \geq 1$ . We therefore clip the variance ratio<sup>3</sup> to

$$s = \text{CLIP}(s, 0, 1),$$

863 guaranteeing  $1 + \omega - \omega s \geq 1$  for any  $\omega > 0$  and ensuring both numerical stability and a genuinely  
 864 mode-seeking guided distribution.

## 865 B Implementation Details

### 866 B.1 Architecture Design

867 **Overall Structure.** We implement STARFlow with a decoder-only Transformer (Vaswani et al.,  
 868 2017). The shorthand  $l(N) - d$  (see § 3.2) denotes a single *deep* AF block of  $l$  layers followed by  
 869  $N - 1$  *shallow* blocks (two layers each) with hidden width  $d$ . Our class-conditioned baseline uses  
 870  $18(6) - 2048$  ( $\approx 1.8$  B parameters), while the text-conditioned model uses  $24(6) - 3072$  ( $\approx 3.8$  B  
 871 parameters). Layer-allocation sweeps in Fig. 8(b–e) probe scalability and convergence. Unlike Zhai  
 872 et al. (2024), we apply a final layer norm at the predictions of each Transformer block.

873 **Conditioning Mechanism.** For both conditioning modes, the context is prepended as a prefix to  
 874 the deep block, and we omit AdaLN (Peebles & Xie, 2023)—a choice that simplifies the network and  
 875 marginally improves quality. ImageNet classes are provided as one-hot vectors. Text captions (T2I)  
 876 are encoded by a frozen FLAN-T5-XL encoder (Raffel et al., 2020), truncated to 128 tokens.

877 **VAE Latent Space.** Images are first mapped to continuous latent tokens via the DiT VAE (Peebles  
 878 & Xie, 2023), which compresses spatial dimensions by  $48\times$ . Because performance is highly sensitive  
 879 to patch size, we keep  $p = 1$  for all resolutions, yielding sequences of 1024, 4096, and 16384 tokens  
 880 for  $256 \times 256$ ,  $512 \times 512$ , and  $1024 \times 1024$  images, respectively. We also applied our proposed  
 881 deep-shallow architecture in pixels (see Table 3). To match similar computation, we adopted a patch  
 882 size of  $p = 8$  for learning  $256 \times 256$  images.

<sup>3</sup>This is equivalent to clip the unconditional variance  $\sigma_u$  when it is smaller than  $\sigma_c$ .

**Positional Embeddings.** All variants employ rotary positional embeddings (RoPE) (Su et al., 2024); we adopt **3D-RoPE**, giving each token  $(x, y, t)$ , where  $(x, y)$  encodes its spatial grid location  $((0, 0)$  for text tokens) and  $t$  its caption index (0 for image tokens). During fine-tuning from  $256 \times 256$  to higher resolutions, we align positions by setting  $(x', y', t) = (x/\alpha, y/\alpha, t)$ , where  $\alpha$  is the up-sampling ratio.

Below is the default configurations of STARFlows:

```
model config for STARFlow-1(N)-d:
    patch_size=1
    hidden_size=d
    num_layers=[1] + [2] * (N-1)
    num_channels_per_head=64
    use_swiglu_ffn=False
    use_rope=True
    use_final_rmsnorm=True
```

## B.2 Training Details

In all the experiments, we share the following training configuration for our proposed STARFlow.

```
training config:
    batch_size=512
    optimizer='AdamW'
    adam_beta1=0.9
    adam_beta2=0.95
    adam_eps=1e-8
    learning_rate=1e-4
    min_learning_rate=1e-6
    learning_rate_schedule=cosine
    weight_decay=1e-4
    max_training_images=400M
    mixed_precision_training=bf16
```

**Stability of Eq. (3).** The maximization term  $-\log \sigma$  in Eq. (3) is *unbounded*: the model can drive some  $\sigma$  values arbitrarily close to zero whenever this hardly influences  $z$ , echoing a classic pathology of normalizing-flow training. We mitigate it with two safeguards:

1. **Soft clipping.** Each raw Transformer output  $\mathbf{x}$  is mapped through  $f(\mathbf{x}) = a \tanh(\mathbf{x}/a)$ , softly limiting its magnitude to  $\pm a$ .
2. **Positive scale parameterization.** The scale is enforced positive via  $\sigma = \text{softplus}(\hat{\sigma})$ , where  $\hat{\sigma}$  is the network’s variance output.

**Mixed-Resolution Training.** During the high-resolution phase, STARFlow supports *mixed resolutions*, preserving each image’s native aspect ratio. Because the backbone is a Transformer, variable sequence lengths are handled naturally, so no aggressive cropping is required; this better retains scene content and improves caption–image alignment. We bucket images into nine aspect-ratio bins: 21:9, 16:9, 3:2, 5:4, 1:1, 4:5, 2:3, 9:16, and 9:21 with the ratio appened in the caption:

```
{original_caption}\n in a {aspect_ratio} aspect ratio.
```

Image is center-cropped and resized so that its token count roughly matches that of a square reference. For a  $512 \times 512$  target, we enforce  $H \times W \approx 512^2$ . This procedure stabilizes optimization, maximizes GPU utilization, and is used in conjunction with the 3D-RoPE alignment described above.

## B.3 Baseline Details

**Diffusion Model Baseline** We deploy the official implementation of DiT<sup>4</sup> and report the performance. To make the architecture comparable to STARFlow, we set the number of layers to 28 and

<sup>4</sup><https://github.com/facebookresearch/DiT>

Table 4: Per-block inference time (s) with a fixed batch size 16 of the 1.4B sized model for generating  $256 \times 256$  images. Sampling speed is measured with CFG. The proposed deep–shallow uses **6** blocks: an 18-layer Transformer followed by a 5 blocks of 2-layer Transformer. The hidden dimension is 2048. The equal-sized (Zhai et al., 2024) baseline uses **8** blocks where each block has 8 layer of Transformers. To match the overall parameters, we reduce the hidden dimensions to 1280.

Block ID	0	1	2	3	4	5	6	7	Total (s)
Equal-sized (Zhai et al., 2024)	9	9	9	9	9	9	9	9	72
Deep–shallow (ours)	18	2	2	2	2	2	-	-	35

hidden dimension to 2048 while keeping the number of attention heads to 16, resulting in a model size of 2.1B parameters. We kept all of the other official repository settings the same. Notably the pretrained VAE of the official repository matches the one used in STARFlow. The baseline DiT is trained for 200M samples with batch size 256 using the official implementation settings: AdamW optimizer with learning rate 0.0001 and no weight decay 0.0. In inference, we set the number of sampling steps to 250 and classifier-free guidance scale to 1.5 following the best reported setting in the original paper.

**Autoregressive Model Baseline** We deploy the official implementation of LlamaGen<sup>5</sup> (Sun et al., 2024) and report the performance. In particular, to make the architecture comparable to our STARFlow, we set the number of layers as 28, hidden dimension 2048, and number of attention heads 32, which leads to the total model size of 1.4B parameters. We also adopt the VQ-VAE from the official repository with downsample factor 8 which matches the downsample factor used in STARFlow. The baseline LlamaGen is trained for 200M samples with batch size 512 using AdamW optimizer with learning rate 0.0001, weight decay 0.05 and betas (0.9, 0.95). In inference, we set the top-k the same as the vocabulary size 16384 and temperature 1.0. We also implement classifier-free guidance with scale 1.75 following the best reported setting in the original paper.

## C Experiments on Limitations

### C.1 Inference Speed

Because STARFlow is autoregressive, tokens must be generated sequentially through every AF block, which makes inference latency the dominant bottleneck. Our deep–shallow redesign **partially** mitigates this issue: by concentrating parameters in the first few “deep” blocks and leaving the remaining ones lightweight with no condition or guidance, the incremental cost of later blocks becomes minimal. In practice, while the sampling speed is still relatively slow, this layout also outperforms the equal-sized architecture of Zhai et al. (2024) (see Table 4), and its overall runtime approaches that of a standard LLM—leaving additional head-room for techniques such as distillation or speculative decoding.

### C.2 Latent Denoising

A second limitation is that STARFlow cannot be trained directly on clean latents; adding Gaussian noise is required to keep the flow learning stable (Ho et al., 2019; Zhai et al., 2024), but this both complicates optimization and necessitates an explicit denoising stage at inference time. In this work, we investigated three strategies of denoising:

- (1) **Single-step score denoising.** Use the flow itself as a score estimator and apply one denoising step. Works only for mild noise; at  $\sigma = 0.3$  outputs are noticeably blurry.
- (2) **Multi-step diffusion denoising.** Start from the noisy latent and run unconditional DDIM steps with a pretrained DiT. Quality improves, but latency and model complexity increase substantially.
- (3) **Decoder finetuning (ours).** Finetune the VAE decoder so it can reconstruct directly from noisy latents. Training can be done very efficiently on unconditional images, and the GAN objective effectively handles the uncertainty. This option is the simplest to deploy.

<sup>5</sup><https://github.com/FoundationVision/LlamaGen>



Table 5: Comparison of latent-denoising strategies at  $\sigma_L = 0.3$  on ImageNet  $256 \times 256$ .

Method	Extra Model	Extra Steps	FID 50K ↓	Remarks
Single-step score	—	1	2.96	Blurry
Multi-step DiT (from 0.3)	DiT-XL	30	2.53	Slowest
Decoder finetune	Finetuned Decoder	0	<b>2.40</b>	Best, simplest

968 Future work will aim for a principled solution that trains directly on clean data, eliminating the  
969 denoising stage entirely.

## 970 D Application Details

### 971 D.1 Training-free Image Inpainting with STARFlow

972 Let  $M \in \{0, 1\}^{H \times W}$  be a binary mask that selects the pixels to be filled and let  $\mathbf{x}_{\text{gt}} \in \mathbb{R}^{H \times W \times C}$  be  
973 the ground-truth image (available only at evaluation time for measuring fidelity). We split the image  
974 into the *observed* part  $\mathbf{x}_O = (1 - M) \odot \mathbf{x}_{\text{gt}}$  and the *missing* part  $\mathbf{x}_M = M \odot \mathbf{x}_{\text{gt}}$ . The pretrained  
975 flow  $\mathbf{f}_\theta : \mathbf{x} \mapsto \mathbf{z}$  induces a tractable density  $p_\theta(\mathbf{x}) = \mathcal{N}(\mathbf{f}_\theta(\mathbf{x}); \mathbf{0}, I) |\det \nabla_{\mathbf{x}} \mathbf{f}_\theta|$ . To sample from  
976 the conditional  $p_\theta(\mathbf{x}_M | \mathbf{x}_O)$  *without retraining*, we construct a Metropolis–Hastings (MH) chain in  
977 latent space:

978 1. **Init.** Replace the missing region by Gaussian noise,  $\tilde{\mathbf{x}}^{(0)} = \mathbf{x}_O + M \odot \epsilon$ ,  $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ , and  
979 map to latent space  $\mathbf{z}^{(0)} = \mathbf{f}_\theta(\tilde{\mathbf{x}}^{(0)})$ .

980 2. **Proposal.** Draw fresh noise in the *same* masked region of latent space

$$\mathbf{z}' = \mathbf{z}^{(t)} + M \odot \gamma, \quad \gamma \sim \mathcal{N}(0, \tau^2 I),$$

981 and obtain the candidate image  $\mathbf{x}' = \mathbf{f}_\theta^{-1}(\mathbf{z}')$ . We then restore the context pixels,  $\tilde{\mathbf{x}}' = \mathbf{x}_O +$   
982  $M \odot \mathbf{x}'$ , ensuring every proposal satisfies the observed evidence.

983 3. **Acceptance.** Because the forward and reverse proposals are symmetric, the MH acceptance  
984 probability reduces to the ratio of conditional probabilities:

$$\alpha = \min \left\{ 1, \frac{p_\theta(\tilde{\mathbf{x}}' | \mathbf{x}_O)}{p_\theta(\tilde{\mathbf{x}}^{(t)} | \mathbf{x}_O)} \right\} = \min \left\{ 1, \exp[\log p_\theta(\tilde{\mathbf{x}}') - \log p_\theta(\tilde{\mathbf{x}}^{(t)})] \right\}.$$

985 Accept with probability  $\alpha$ ; otherwise keep the current state.

986 4. **Iteration.** Set  $t \leftarrow t + 1$  and repeat steps (ii)–(iii) until convergence; the final sample  $\tilde{\mathbf{x}}^{(T)}$  is  
987 reported as the inpainted image.

988 Intuitively, each step perturbs only the masked latents, letting the powerful flow prior propose  
989 content that is *globally* coherent with the context while the MH test enforces exact consistency with  
990 the joint density. The chain is ergodic—Gaussian noise gives non-zero probability to every latent  
991 configuration—and its stationary distribution is precisely  $p_\theta(\mathbf{x}_M | \mathbf{x}_O)$ . In practice we set both  $\sigma = 1$   
992 and  $\tau = 1$ . Since our STARFlow is well-trained on large-scale text-to-image data with sufficient  
993 capacity, it yields high acceptance rates and we set the total iterations to **20**. No additional training,  
994 guidance network, or data-specific tuning is required. effective plug-in for image inpainting with  
995 pretrained autoregressive flows.

### 996 D.2 Interactive Image Editing with STARFlow

997 STARFlow can be naturally extended to multi-round tasks such as interactive image editing. We start  
998 from a pretrained text-to-image checkpoint and finetune on the ANYEDIT corpus<sup>6</sup>. For simplicity, we  
999 use only the subset that provides text instructions. Each training quadruple  $(\mathbf{x}^{\text{src}}, \mathbf{t}^{\text{cap}}, \mathbf{t}^{\text{inst}}, \mathbf{x}^{\text{tgt}})$   
1000 contains a source image, its caption, a free-form instruction, and the edited target (see Fig. 9b).

<sup>6</sup><https://dcd-anyedit.github.io>

1001 **Input Linearization.** We serialize every sample into the sequence

$$[\text{T5}(\mathbf{t}^{\text{cap}}), \text{AFs}[\text{VAE}(\mathbf{x}^{\text{src}})], \text{T5}(\mathbf{t}^{\text{inst}}), \text{AFs}[\text{VAE}(\mathbf{x}^{\text{tgt}})]] ,$$

1002 where image segments are tokenized by our VAE ( $p = 1$ ) and text segments are embedded by a  
 1003 frozen FLAN-T5-XL (Raffel et al., 2020; Chung et al., 2022). Image latents first pass through  
 1004 the shallow-AF blocks independently, after which all tokens are processed by the shared deep-AF  
 1005 Transformer. Because the deep block is strictly causal, the edited image and all later tokens can attend  
 1006 to the entire prefix—including the source image—without any special masking. During inference the  
 1007 prefix is written once into the  $KV$  cache; sampling the edited tokens simply reads from this cache,  
 1008 mirroring the behavior of language-only LLMs.

1009 **Joint Training Objective.** Instead of optimizing a single conditional likelihood, we maximize the  
 1010 *joint* log-likelihood of both images:

$$\max_{\theta} \mathcal{L}_{\text{joint}} = \mathbb{E}_{(\mathbf{x}^{\text{src}}, \mathbf{t}^{\text{cap}}, \mathbf{t}^{\text{inst}}, \mathbf{x}^{\text{tgt}})} \left[ \log p_{\theta}(\mathbf{x}^{\text{src}} | \mathbf{t}^{\text{cap}}) + \log p_{\theta}(\mathbf{x}^{\text{tgt}} | \mathbf{t}^{\text{inst}}, \mathbf{x}^{\text{src}}, \mathbf{t}^{\text{cap}}) \right],$$

1011 where each term is evaluated via the change-of-variables formula (Eq. (3)). This objective maintains  
 1012 maximum-likelihood training, allows gradients to propagate across *all* modalities, and enables the  
 1013 same network to generate from scratch (empty image prefix) or perform edits (given image prefix).

1014 **Invertibility Advantage.** Unlike diffusion-based MLLMs that first *generate* pixels and then re-  
 1015 encode them with a separate vision backbone, our autoregressive flow is invertible: a single forward  
 1016 pass encodes the user image, and a single reverse pass decodes the edited result. Encoding and decod-  
 1017 ing share parameters, introduce no information loss, and integrate seamlessly with the Transformer’s  
 1018  $KV$  cache. This *single-pass round-trip* property sharply reduces latency and highlights autoregressive  
 1019 flows as a compelling choice for tightly coupled vision–language applications.

1020 **Empirical Results.** We show interactive image generation and editing examples in Fig. 10.

## 1021 E Additional Samples

1022 We show more generated samples from STARFlow in Figs. 11 and 12.

### Prompts for Fig. 1

1. vw bus, canvas art, abstract art printing, in the style of brian mashburn, light red and light brown, theo prins, charming character illustrations, pierre pellegrini, vintage cut-and-paste, rusty debris –ar 73:92 –stylize 750 –v 6"
2. A red fox curled up asleep in a snowy woodland clearing, with delicate snowflakes falling gently around it, watercolor style.
3. A landscape featuring mountains, a valley, sunset light, wildlife and a gorilla, reminiscent of Bob Ross’s artwork
4. Vibrant abstract painting with colorful swirling brushstrokes in blue, orange, red, yellow, and green, resembling a dynamic river of paint. In the center, bold hand-painted text reads “Go with the StarFlow” in a playful and modern font. The style is fluid, energetic, and uplifting, like a digital acrylic artwork.
5. A rustic kitchen table set with freshly baked bread, an assortment of cheeses, a bowl of ripe fruit, and a bouquet of lavender, sunlight streaming through a nearby window casting soft shadows, detailed still life painting with warm tones and textures.
6. A panda eating bamboo in a lush green forest, with soft sunlight filtering through the leaves, realistic painting.

## 1024 F Discussion and Broader Impacts

### 1025 F.1 Autoregressive Model v.s. Normalizing FLOW

1026 Connections between the two families emerge in masked autoregressive flows (MAF, (Papamakar-  
 1027 ios et al., 2017)), which impose invertibility on an autoregressive factorisation, yet fundamental

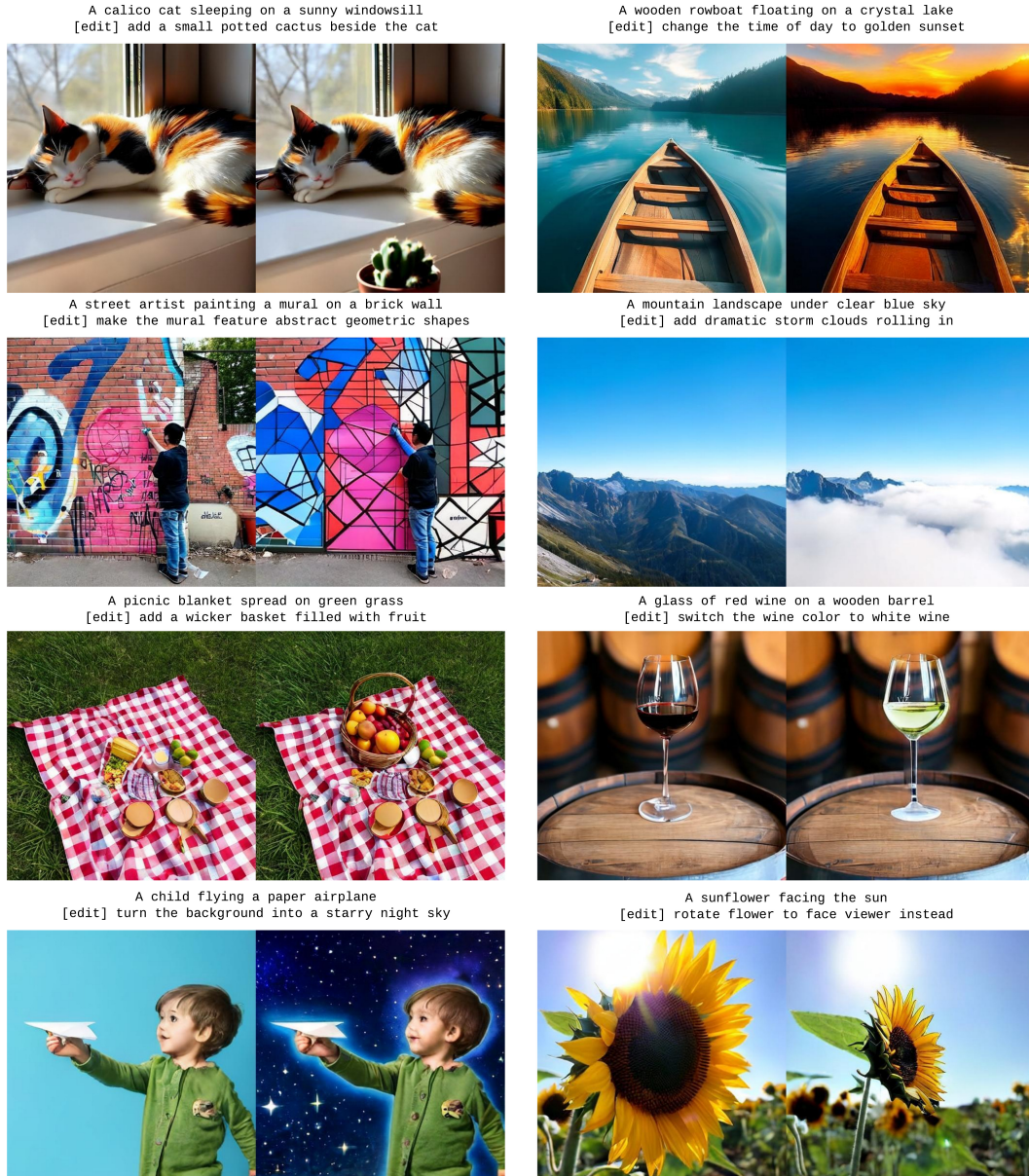


Figure 10: Interactive editing with STARFlow. Starting from an initial caption, STARFlow generates a base image. Given a subsequent user-provided editing instruction, the model then modifies the image accordingly—without requiring re-encoding. Each example illustrates a generic instruction applied to a generated image. All images are synthesized at a resolution of  $512 \times 512$ .

1028 differences remain. Autoregressive models dispense with any latent prior; each conditional distri-  
 1029 bution is learned directly in the data domain, which is typically discrete—tokens, integer pixels,  
 1030 or quantized audio samples—and generation proceeds strictly one element at a time. Normalizing  
 1031 flows, by contrast, begin from an explicit Gaussian prior in a continuous latent space and learn an  
 1032 invertible transformation that warps this prior into the target distribution. This design delivers exact  
 1033 log-likelihoods, parallel one-shot sampling, and bidirectional latent inference, but at the cost of  
 1034 enforcing invertibility and differentiability in every layer. While MAF narrows the gap by marrying  
 1035 an autoregressive factorisation with invertibility, the reliance on a Gaussian base and a continuous  
 1036 formulation remains the defining hallmark of normalizing flows, whereas the absence of a prior and  
 1037 the natural alignment with discrete data continue to characterise pure autoregressive models.

## 1038 **F.2 Flow Matching v.s. Normalizing FLOW**

1039 Normalizing flows (NF) and Flow Matching (FM) both map a simple latent prior to the data dis-  
1040 tribution, but they differ fundamentally in what they optimise and how they realise the map. A  
1041 normalizing flow learns a time-independent bijection whose parameters are updated by directly  
1042 minimising the data’s negative log-likelihood (NLL); the change-of-variables formula provides an  
1043 exact, unbiased gradient, so every parameter update moves the model toward the true maximum-  
1044 likelihood solution. Flow Matching instead specifies a time-dependent vector field that transports  
1045 probability mass along a chosen path and trains this field with a velocity-matching loss. In short,  
1046 Flow Matching reduces per-iteration cost by relaxing the objective, but Normalizing Flows retain the  
1047 rigorous maximum-likelihood foundation, and exact densities.

## 1048 **F.3 Broader Impacts**

1049 **Positive societal impacts.** STARFlow shows—for the first time—that *exact* likelihood-based  
1050 models can scale to the same resolutions and sample quality previously dominated by diffusion and  
1051 discrete autoregressive methods. The invertibility of normalizing flows enables interactive image  
1052 editing (See Fig. 9b for examples) making STARFlow suitable for assistive technologies (e.g., real-  
1053 time diagram manipulation for education or accessibility) and for professional design workflows that  
1054 demand faithful, iterative refinement.

1055 **Potential risks and negative impacts.** Higher-quality image generation lowers the barrier to  
1056 fabricating realistic—but false—visual evidence. Interactive editing magnifies this risk by enabling  
1057 rapid revision cycles. We advocate the concurrent development of reliable flow-specific watermarking  
1058 and provenance tools.



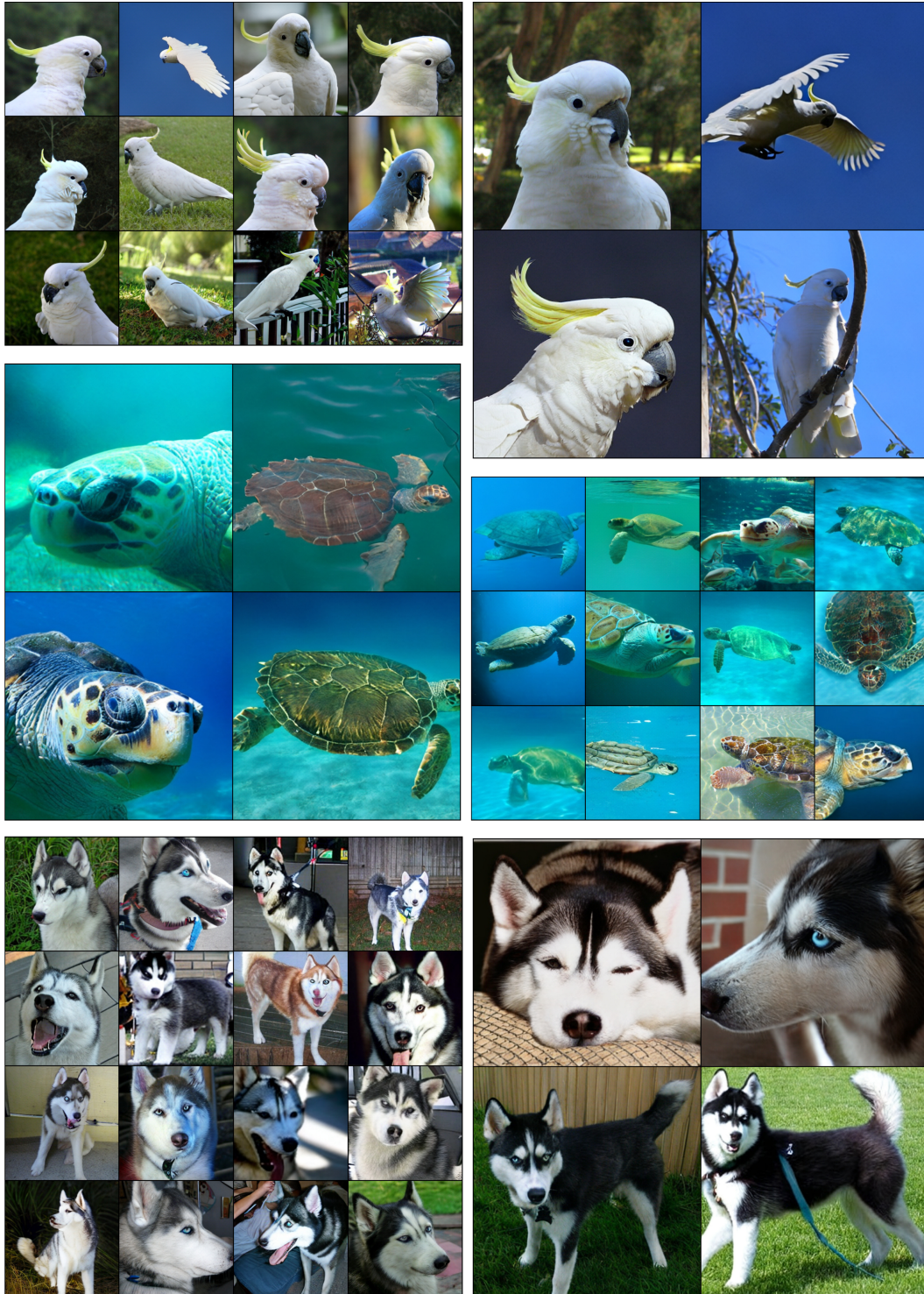
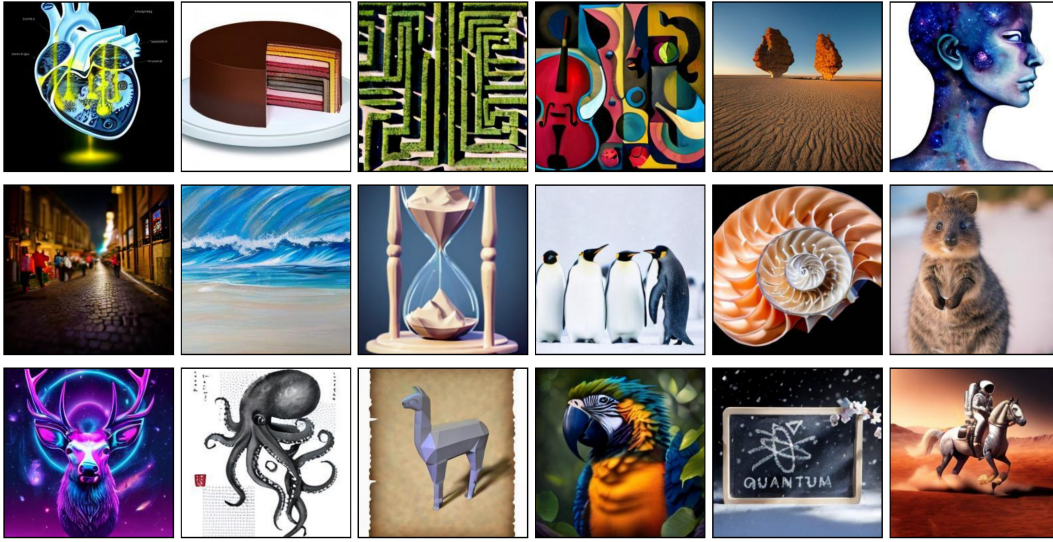
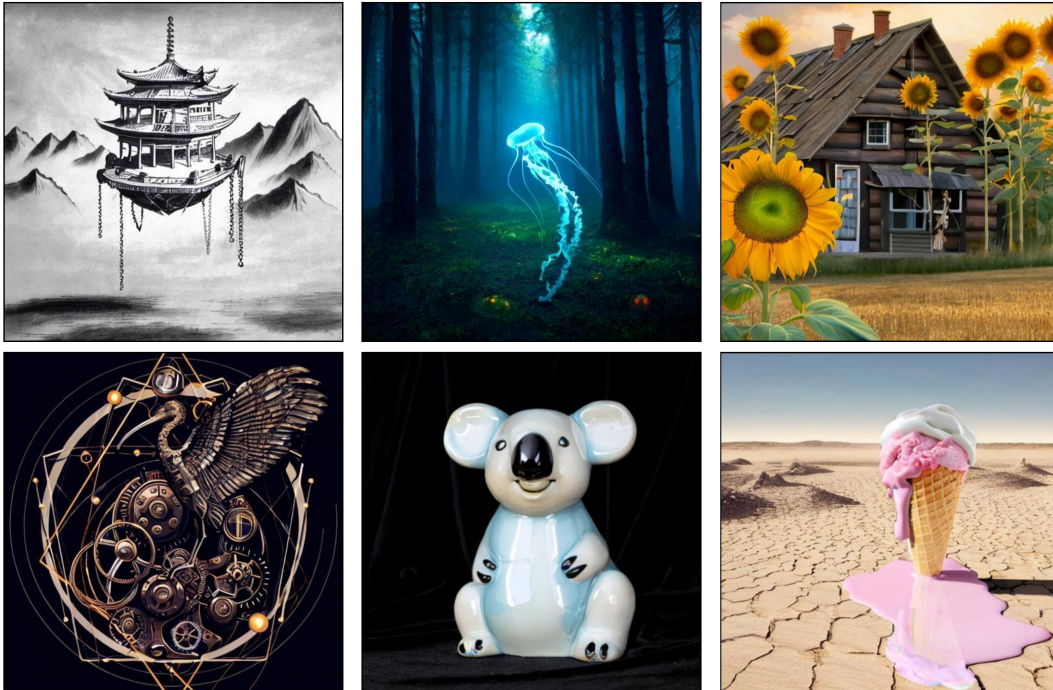


Figure 11: Additional uncrated class-conditioned generation from STARFlows trained on  $256 \times 256$  and  $512 \times 512$ , respectively. The classes are *sulphur-crested cockatoo*, *Kakatoe galerita*, *Cacatua galerita*, *loggerhead*, *loggerhead turtle*, *Caretta caretta*, and *Siberian husky*.





0. Schematic cutaway of a clockwork heart pumping luminous liquid, technical drawing style 1. Cross-section illustration of a layered cake that resembles planetary strata 2. Surreal desert with floating sandstone monoliths casting long shadows at golden hour, ultra-wide lens 3. Cubist still life of fruit and musical instruments, vivid complementary colors 4. Top-down shot of a labyrinth garden trimmed into Escher-like impossible geometry 5. Watercolor portrait of an abstract humanoid with translucent skin revealing galaxies 6. Tilt-shift photo of a festival lantern parade through narrow cobblestone streets 7. Oil-on-canvas seascape where waves are brush strokes of pure geometry 8. Design an hourglass where sand forms miniature mountains in low-poly 3-D model style 9. Juvenile emperor penguins huddling together on Antarctic ice shelf, gentle snowfall 10. Low-key studio shot of concentric nautilus shell cross-section revealing logarithmic spiral 11. Quokka standing on hind legs engaging camera with curious expression, beach background 12. neon synthwave poster featuring a deer amid swirling galaxies 13. Japanese ink wash of an octopus surrounded by geometric patterns 14. low-poly 3-D render of an alpaca on vintage parchment 15. Photorealistic close-up of a macaw hunting silently in the tropical rainforest canopy 16. High-resolution chalkboard typography sketch spelling 'Quantum' amid cherry-blossom snowfall, dramatic lighting 17. vibrant A photorealistic image of an astronaut riding a horse on Mars



0. Ink-on-parchment concept art of a floating pagoda tethered by chains to mountain peaks 1. Glowing jellyfish drifting through a misty pine forest at dawn, photoreal composite 2. Timber-frame hobbit-style cottage under giant sunflowers, golden afternoon 3. steampunk clockwork version of a ibis surrounded by geometric patterns 4. ceramic glazed statue of a koala against black velvet backdrop 5. dreamlike An ice cream cone melting into a desert landscape, surrealism

Figure 12: Additional uncrated text-conditioned generation from STARFlows trained on  $256 \times 256$  and  $512 \times 512$ , respectively.