EEdit ⚡: Rethinking the Spatial and Temporal Redundancy for Efficient Image Editing

Supplementary Material

7. Analysis on Hidden States in MM-DiT

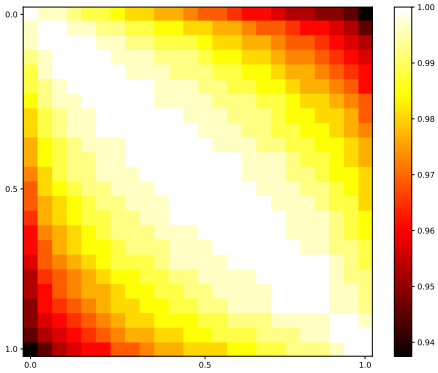


Figure 9. Cross-Attention hidden states similarity

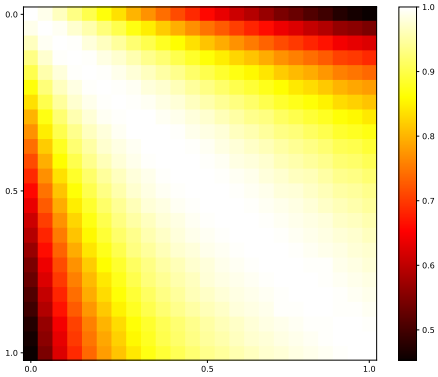


Figure 10. Self-Attention hidden states similarity

As shown in the Figure. 9 and Figure. 10, we visualize the cosine similarity of hidden states across different timesteps in both Cross-Attention and Self-Attention. It can be observed that Cross-Attention exhibits higher similarity, indicating greater redundancy in this module. Consequently, in our approach, Cross-Attention is either fully computed or entirely skipped to optimize efficiency.

8. Algorithm for Image Editing with SLoC

8.1. Cache Frequency Control

Cache Frequency Control can be formulated as

$$\mathcal{S}_{ij}^{\tau+1} \leftarrow \begin{cases} \mathcal{S}_{ij}^{\tau} + \gamma f_{ij}^{\tau} & f_{ij}^{\tau+1} \leftarrow f_{ij}^{\tau} + 1 \\ & \mathcal{S}_{ij}^{\tau} \leq Top_{R\%}(\mathcal{S}^{\tau}[1 \dots L_{WH}]) \\ \mathcal{S}_{ij}^{\tau} & f_{ij}^{\tau+1} \leftarrow 0 \\ & \mathcal{S}_{ij}^{\tau} > Top_{R\%}(\mathcal{S}^{\tau}[1 \dots L_{WH}]) \end{cases}$$

where γ is a scaling factor that controls the impact of reuse frequency on the score map $\mathcal{S}$. τ indicates current time step and $\tau+1$ indicates next time step. ij indicated index in score map $\mathcal{S}$ and frequency map $\mathcal{M}_{freq}$. We have $\mathcal{M}_{freq} = \{f_{ij}, i \in [1 \dots L_W], j \in [1 \dots L_H]\}$

8.2. Vanilla SLoC w/o ISS and TIP

Algorithm 2 Image Editing with **SLoC**

Require: Input image $\mathbf{I_s}$, Mask for editing region $\mathbf{M_s}$, Prompt for editing $\mathbf{P_m}$, Randomly initialized map $\mathcal{R}$, Bonus for edited region $\mathbf{S_E}$ and Cache dict $\mathcal{C}_{l,m}[:]$.
Ensure: The edited result $\mathbf{I}^*$

$\mathcal{M}_{freq} \leftarrow \boldsymbol{zero}[\mathcal{F}_\mathbf{i}, \mathbf{t}, \mathbf{l}]$
$\mathbf{Z}_0 \leftarrow \mathbf{cat}(\text{VQ-Encoder}(\mathbf{I_s}), \text{Txt-Encoder}(\mathbf{P_m}))$
// Image Latent Inversion
for $t = 1, \dots, T$ **do**
  $\mathbf{Z}_t \leftarrow \text{RF-inversion}(\mathbf{Z}_{t-1}, t-1, \phi)$
end for
$Z_T^* \leftarrow Z_T$
// Image Editing Steps with Caching
for $t = T, T-1, \dots L+1$ **do**
  for $\mathcal{F}_i \leftarrow \mathbf{SA}_l, \mathbf{CA}_l, \mathbf{MLP}_l, l \in [1, \dots \mathcal{L}]$ **do**
    $\mathcal{S}_l \leftarrow (\mathcal{R} \odot \mathbf{S_E}) \oplus \mathcal{M}_{freq}$
    $\mathcal{I}_{i,l,t} \leftarrow \mathbf{Sel}_{topR\%}(\mathcal{S}_l)$
    $Z_{l+1}^* \leftarrow \mathbf{scatter}(\mathcal{F}_i(Z_l^*, \mathcal{I}_{i,l,t}), \mathcal{C}_{t+1}[l, \mathcal{F}_i])$
    $\mathbf{Update}(\mathcal{C}_{t+1}[l, \mathcal{F}_i], \mathcal{M}_{freq})$
  end for
  $\mathbf{Z}_{t-1}^* \leftarrow \mathbf{Z}_{t-1}^* \odot \mathbf{M}_s + \mathbf{Z}_t \odot (1 - \mathbf{M}_s)$
end for
$\mathbf{I}^* \leftarrow \text{VQ-Decoder}(\mathbf{x}_0)$
return $\mathbf{I}^*$

9. Discussion on TIP

We adopt Token Index Preprocessing. This design offers an additional advantage by reducing the number of function

Table 4. **Ablation study for cache**. Comparisons of different cache methods in terms of FG fidelity, and computational efficiency.

Method	FG preservation				Efficiency					Speed Up	
	FID↓	PSNR↑	$MSE^{\downarrow}_{10^{-3}}$	$SSIM^{\uparrow}_{10^{-1}}$	Steps	Inv.(s)↓	Fwd.(s)↓	Inference (s)↓	FLOPs(T)↓	Latency↑	FLOPs↑
No Cache	-	-	-	-	28	5.44	5.67	11.30	932.95	1 ×	1 ×
50% Step	90.60	24.46	6.66	8.77	14	2.63	2.87	5.69	456.55	**2.04**×	**1.99**×
FORA [50]	39.96	31.62	1.74	9.47	28	2.42	**2.45**	5.05	318.09	**2.24**×	**2.93**×
ToCa [76]	84.77	26.16	5.76	8.88	28	3.52	3.52	7.29	332.93	**1.55**×	**2.80**×
DuCa [77]	84.85	26.16	5.76	8.88	28	3.26	3.26	6.87	313.00	**1.67**×	**2.98**×
SLoC	39.50	**31.75**	1.72	**9.48**	28	2.86	2.91	5.96	384.03	**1.90**×	**2.43**×
SLoC$_{+TIP+ISS}$	**39.21**	**31.75**	**1.71**	**9.48**	28	**1.92**	2.49	**4.60**	**264.50**	**2.46**×	**3.53**×

calls within the cache module. Specifically, with a preprocessing overhead of no more than **150ms**, we achieve a reduction of over **1000ms** in cache-induced inference latency. Since the token selection in our algorithm is independent of the internal properties of individual tokens or their mutual interactions, this decoupling is logically equivalent in the temporal sequence. Consequently, it enables further lossless acceleration on top of SLoC.

9.1. Proof of TIP Equivalence with Original Operations

We maintain a cache of intermediate features for a set of tokens in SLoC. At each iteration step, each token is assigned a score based on (i) a random or seed-based component and (ii) a function of its selection frequency (the $\mathcal{M}_{freq}$). The top $R\%$ of tokens are selected for updating the cache. It follows algorithm 2.

We prove that this preprocessing-based approach is mathematically equivalent to performing scoring, sorting, and token selection *online* at each iteration.

9.2. Notation and Problem Setup

- N: Total number of tokens, indexed as $\{1, 2, \ldots, N\}$.
- T: Total number of diffusion iterative steps.
- $s_i^{(t)}$: Score of token i at step t, given by

$$s_i^{(t)} = f(r_i^{(t)}) + \mathcal{M}_{freq,i}^{(t)}$$

  where:
  - $r_i^{(t)}$: Random (or seed-based) component.
  - $\mathcal{M}_{freq,i}^{(t)}$: Frequency of times token i has been selected before step t.
  - $f(\cdot)$: A deterministic function adjusting scores, region score bonus adopted in SLoC here.
- After computing $\{s_i^{(t)}\}_{i=1}^{N}$, the top $R\%$ tokens are selected for cache updates.
- For simplicity in our proof, we have omitted the layer index and module type (Cross-Attention, Self-Attention, MLP).

9.3. Original (Online) Algorithm Description

The online method iterates as follows 2:

1. For $t = 1$ to T:
   (a) Compute $s_i^{(t)}$ for each token i.
   (b) Sort tokens by $s_i^{(t)}$ and select the top $R\%$.
   (c) Update the cache for these selected tokens.
   (d) Increment $\mathcal{M}_{freq,i}^{(t+1)}$ for each selected token i.

Here, $r_i^{(t)}$ is reproducible when using a fixed seed.

9.4. Proposed Optimization (TIP)

The optimized approach precomputes the cache update and selection process 1:

1. Generate and iterate all $r_i^{(t)}$ for $i = 1, \ldots, N$ and $t = 1, \ldots, T$.
2. Simulate the selection process offline:
   (a) Initialize $\mathcal{M}_{freq,i}^{(1)} = 0$ for all i.
   (b) For each $t = 1$ to T:
   - Compute $s_i^{(t)} = f(r_i^{(t)}) + \mathcal{M}_{freq,i}^{(t)}$.
   - Sort and select the top $R\%$, recording indices as $\mathcal{I}_{\text{top}}^{(t)}$.
   - Update $\mathcal{M}_{freq,i}^{(t+1)}$ for selected tokens.
3. Store $\{\mathcal{I}_{\text{top}}^{(t)}\}_{t=1}^{T}$ for later use.

At inference, we read precomputed $\mathcal{I}_{\text{top}}^{(t)}$ instead of recomputing scores.

9.5. Proof of Equivalence

We prove that both methods select identical tokens at each step.

Step 1 Equivalence. At $t = 1$, we have $\mathcal{M}_{freq,i}^{(1)} = 0$, so

$$s_i^{(1)} = f(r_i^{(1)}) + 0.$$

Since $r_i^{(1)}$ is identical in both methods (fixed seed), sorting $s_i^{(1)}$ gives the same top $R\%$ tokens, ensuring identical updates and increments for $\mathcal{M}_{freq,i}^{(2)}$.

Inductive Hypothesis. Assume for steps $k < t$ that

$$\mathcal{I}_{\text{top}}^{(k)}(\text{offline}) = \mathcal{I}_{\text{top}}^{(k)}(\text{online}).$$

Thus, $\mathcal{M}_{freq,i}^{(t)}$ is identical in both methods.

Step t Equivalence. At step t,

$$s_i^{(t)} = f(r_i^{(t)}) + \mathcal{M}_{freq,i}^{(t)}$$

Since $r_i^{(t)}$ and $\mathcal{M}_{freq,i}^{(t)}$ are identical (by induction), we get

$$s_i^{(t)}(\text{offline}) = s_i^{(t)}(\text{online}),$$

ensuring that sorting and selecting the top $R\%$ gives identical indices sets:

$$\mathcal{I}_{\text{top}}^{(t)}(\text{offline}) = \mathcal{I}_{\text{top}}^{(t)}(\text{online}).$$

Conclusion by Induction. By induction, token selection and cache updates remain identical for all $t = 1, \ldots, T$. Thus, preprocessing achieves the same outcome as the online approach.

10. Implementation Details

The experiments were conducted on a machine with the following hardware and software specifications:

10.1. Hardware Specifications

- **Architecture:** x86_64
- **CPU Op-Modes:** 32-bit, 64-bit
- **Address Sizes:** 52 bits physical, 48 bits virtual
- **Byte Order:** Little Endian
- **Total CPU(s):** 128
- **On-line CPU(s) List:** 0-127
- **Vendor ID:** AuthenticAMD
- **Model Name:** AMD EPYC 9K84 96-Core Processor
- **CPU Family:** 25

10.2. Software Specifications

- **Operating System:** Ubuntu 22.04.3 LTS
- **Python** 3.12.3
- **huggingface-hub:** 0.26.2
- **numpy:** 1.26.4
- **torch:** 2.5.1
- **torchmetrics:** 1.6.1
- **transformers:** 4.46.1

11. Metrics

Our experiments employ a selection of the most widely used image quality, instruction adherence, and efficiency metrics.

Frechet Inception Distance (FID) and **Learned Perceptual Image Patch Similarity (LPIPS)** are feature-based similarity metrics computed using pretrained neural networks. Lower values indicate higher similarity. We use `InceptionV3` for FID and `AlexNet` for LPIPS measurements. **Peak Signal-to-Noise Ratio (PSNR)** and **Mean Squared Error (MSE)** are pixel-space similarity metrics.

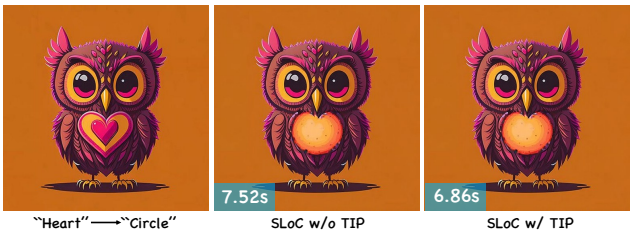


Figure 11. A qualitative example. Token index preprocessing shows loss-less acceleration for editing quality.

A higher PSNR and a lower MSE indicate greater image similarity. **CLIPScore** measures how well an image generation or editing result follows a given prompt using a pretrained CLIP model. A higher score indicates stronger adherence to the prompt. In our experiments, we use the `clip-vit-base-patch16` model. **FLOPs** quantify the computational cost associated with model inference. A higher value indicates greater computational overhead.

12. More Experiments

Table 5. **Performance comparison**. An ablation study is conducted on imbalanced inversion and denoising for background preservation, foreground fidelity and inference time.

Inversion	Denoising	BG Preservation	FG Fidelity			Inference ↑
		LPIPS ↓ $_{\times 10^{-2}}$	LPIPS ↓ $_{\times 10^{-3}}$	PSNR ↑	FID↓	Time (s)
Full Step	**Full Step**	1.98	-	-	-	13.27
2-step skip	**Full Step**	31.38	1.98	31.93	3.35	10.16
3-step skip		31.38	1.98	25.79	3.23	9.31
4-step skip		31.38	1.98	26.50	3.31	8.76
Full Step	2-step skip	1.97	50.40	31.93	28.51	11.79
	3-step skip	1.97	121.44	25.79	64.10	9.28
	4-step skip	1.96	102.67	26.50	56.93	8.54

We compared the edited results in terms of image similarity and efficiency. The Table. 4 demonstrate that our approach, when incorporating all optimizations (TIP + ISS), achieves the best performance in both fidelity and efficiency. SLoC achieves a **2.46**× significant improvement in inference latency and a **3.53**× acceleration in computational efficiency compared to the original unaccelerated version. Additionally, our method demonstrates either improved fidelity or remains at a state-of-the-art level across various editing tasks.

13. Related work

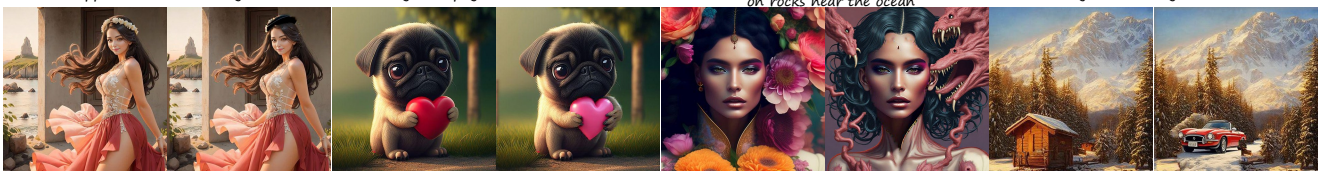
We thanks the related work about editing both image and video, such as VideoGrain [67], FastVAR [15], Int-Lora [14], FollowFamilty [5, 11, 32, 34, 37, 39, 40, 42, 56, 66, 69, 74], Tpsence [72], Pointnorm [71], MotionDiff [41], FastScene [38], DreamRelation [61] and Dreamvideo [60], and Lazymar [65].

14. Gallery

We present additional editing results, including prompt-guided, drag-guided, and reference-guided editing. Furthermore, we adapted the community-developed Redux model for img2img tasks, enabling the generation of impressive image variations.

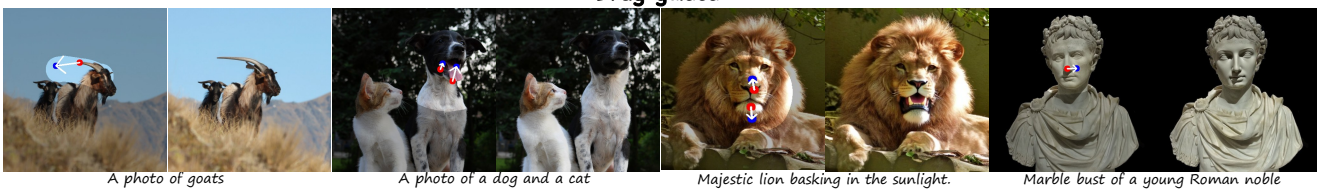
Prompt-guided

A red apple and a bird sitting on it | A golden pagoda in the rain | Photo of a horse and a cat standing on rocks near the ocean | A closed eyes cat sitting on wooden floor



A beautiful woman with hat on head | A cute dog holding a pink heart | A woman with monster around her face | A painting of a car in the snow with mountains in the background

Drag-guided



A photo of goats | A photo of a dog and a cat | Majestic lion basking in the sunlight. | Marble bust of a young Roman noble

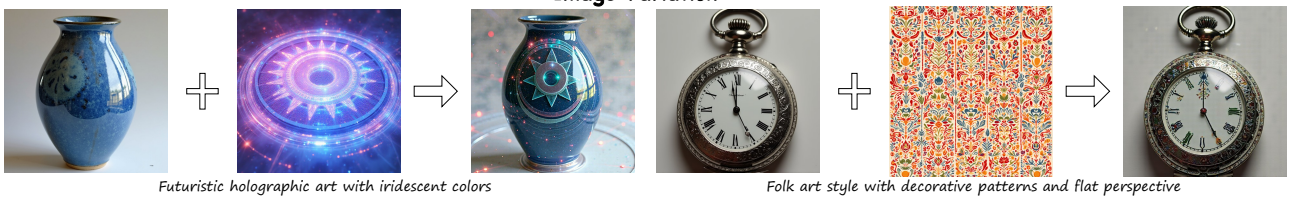
An oil painting of a female | A photo of a man holding a crocodile | Pastoral scene with horses and dogs in a countryside setting | Mountain peaks glowing in the sunset

Reference-guided

A cartoon animation of a castle in the distance | A cartoon animation of a panda in the forest | A professional photograph of a fire hydrant on the grass, ultra realistic | A professional photograph of a tiger on the beach, ultra realistic

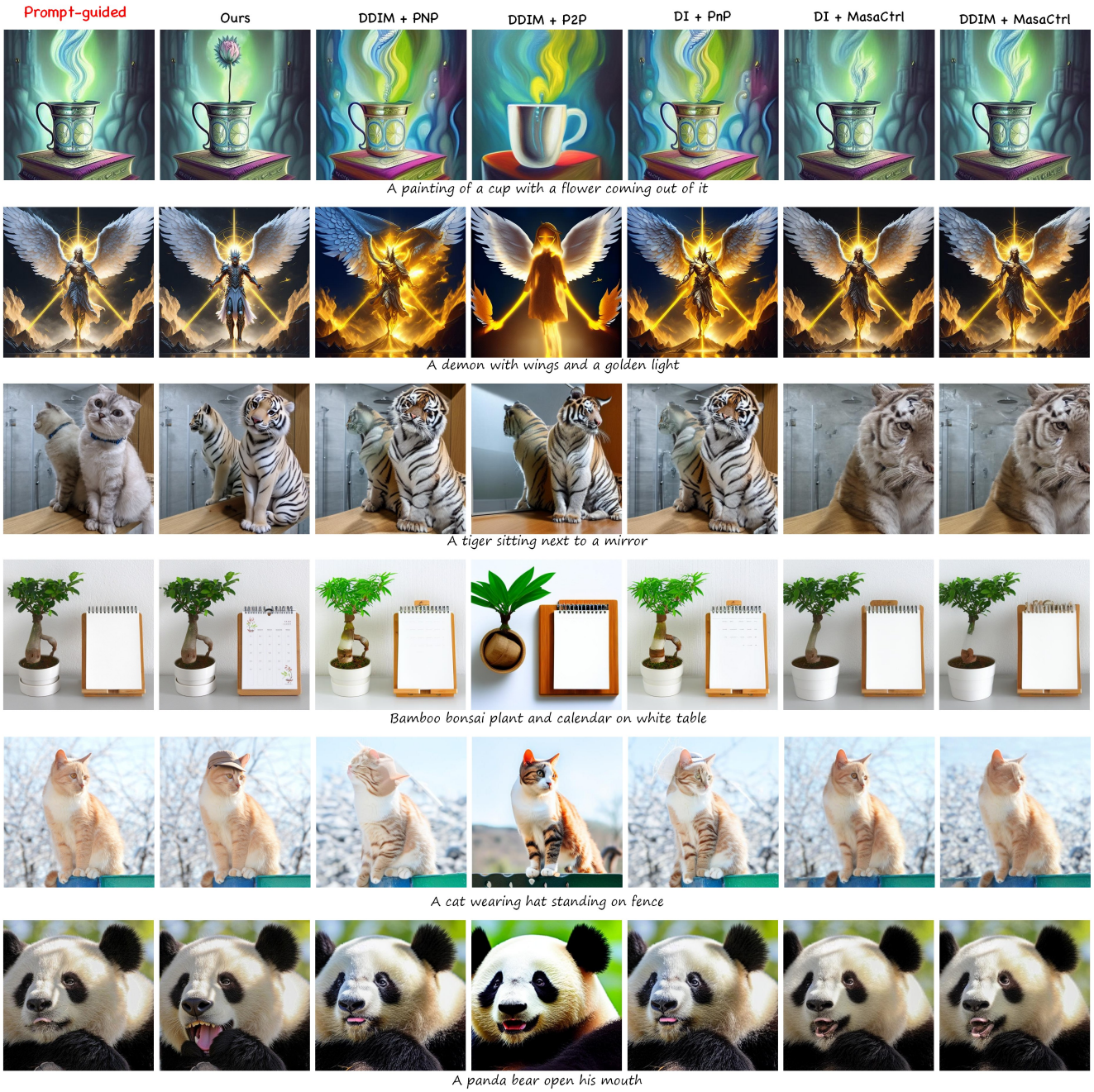
A cartoon animation of a squirrel in the forest | A cartoon animation of a goose in the forest | A cartoon animation of a panda in the forest | A professional photograph of a puppy on the grass, ultra realistic

Image Variation

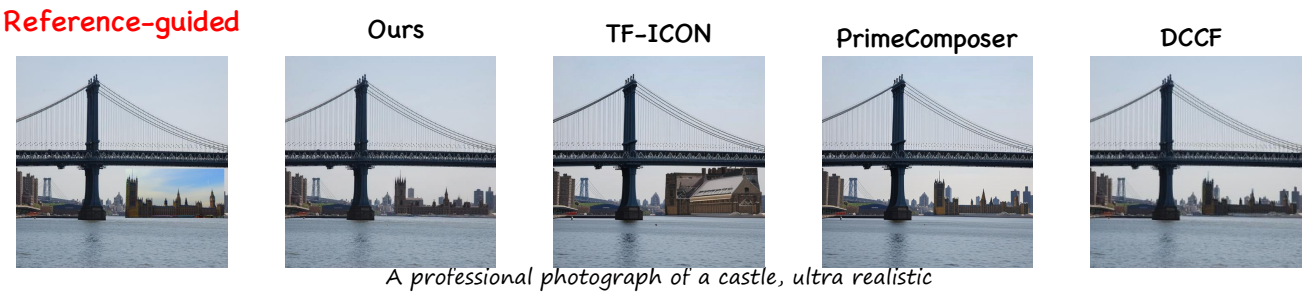


Futuristic holographic art with iridescent colors | Folk art style with decorative patterns and flat perspective

Traditional Chinese painting with fine ink brushwork and subtle gradients | Digital art in cyberpunk style with neon colors and high contrast


Prompt-guided
Ours
DDIM + PNP
DDIM + P2P
DI + PnP
DI + MasaCtrl
DDIM + MasaCtrl
A painting of a cup with a flower coming out of it
A demon with wings and a golden light
A tiger sitting next to a mirror
Bamboo bonsai plant and calendar on white table
A cat wearing hat standing on fence
A panda bear open his mouth


Drag-guided
Ours
Drag Diffusion
FastDrag
RegionDrag
A photo of a cat
Lighthouse, coast, grassland, sea
A photo of a leaf
A boy holding an umbrella
The oil painting of a beautiful scene
A photo of mountains covered with snow


Reference-guided
Ours
TF-ICON
PrimeComposer
DCCF
A professional photograph of a castle, ultra realistic
A cartoon animation of a panda in the forest
A professional photograph of a skyscraper in the distance, ultra realistic
An oil painting of a hamburger, Van Gogh Style
An oil painting of a chocolate doughnut, Van Gogh Style
An oil painting of a hot dog, Van Gogh Style