Guiding a Diffusion Model by Swapping Its Tokens

Supplementary Material

This document provides additional details, results, and analysis to supplement the main paper. Additional details on the datasets, evaluation metrics, and implementation are provided in Section A. Further analysis on the properties of the proposed method and the rationality behind its designs in presented in Section B. More visualised results are given in Section C and Figures E to I.

A. Experiment Details

Datasets. The MS-COCO 2014 validation set contains a total of 40,504 images. The MS-COCO 2017 validation set includes 5,000 images sampled from the MS-COCO2014 validation set. The ImageNet validation set contains 50,000 images. For conditional image generation, we use the caption data associated with the MS-COCO 2014 and 2014 validation sets, where each image is annotated with 5 human-written captions. We randomly select one caption per image for evaluating conditional generation.

Evaluation metrics. For quantitative evaluation of image generation quality, we employ eight metrics, namely FID, CLIP Score, Inception Score (IS), Aesthetic Score (AES), PickScore, ImageReward (IR), and Improved Precision and Recall. FID evaluates both fidelity and diversity by comparing the distributions of real and generated images in the deep feature space of a pre-trained model, for which we use Inception-V3 [50]. CLIP Score evaluates the semantic alignment between generated images and their corresponding text prompts in conditional generation. We compute CLIP Score using the ViT-B/32 [51] model. For Inception Score, we use the Inception-V3 [50] model. For Improved Precision and Recall, we use a pre-trained VGG-16 classifier [44]. All images are resized to 256×256 resolution for computing evaluation metrics.

Implementation. We apply SSG to the input tokens to the self-attention module in each Transformer block of the U-Net backbone. For SDXL, we apply the proposed spatial and channel token swap operations to the early upsampling layers (*i.e.*, u0 to u5) of the U-Net backbone by default. For SD1.5, we apply token swap to the middle layer (*i.e.*, m0) of its U-Net backbone. In our experiments, SAG [19] adopts the DDIM scheduler following its official setting, while all other methods including SSG employ the Euler scheduler [20]. All experimental results are obtained using 50 inference steps.

```
# t: normalized token features [B,N,D], r: ratio
n=int(N*r) # number of swaps to make
S=bmm(t,t.transpose(1,2)) # spatial similarity
    matrix [B,N,N]
i,j=triu_indices(N,N) # get upper triangle
    indices
idx=topk(S[:,i,j], k=n, largest=False) # get most
     dissimilar n pairs
p=arange(N).repeat(B, 1)
p.scatter_(1, i[idx], j[idx]) # swap tokens
p.scatter_(1, j[idx], i[idx]) # swap tokens
t_swap=gather(t,1,p.unsqueeze(-1).expand(B,N,D))
    # get swapped token features [B,N,D]
noise,noise_pert=unet([t,t_swap]) # get noise
    predictions
noise_ssg=noise+scale*(noise-noise_pert) # guided
     noise pred at current step
```

Listing A. **PyTorch-style pseudo-code** of diffusion sampling guidance with token spatial swap.

Layer	FID↓	CLIP↑	IS↑	AES↑	PickScore↑	IR↑
Down	33.45	**0.314**	**35.12**	5.867	22.02	0.245
Mid	**30.60**	0.306	30.36	5.823	21.56	0.0607
Up	31.41	0.313	34.44	**5.901**	**22.18**	**0.297**

Table A. **Effect of applying SSG to different stages of the U-Net backbone.**

Pseudo-code. We provide the PyTorch-like pseudo-code for spatial swap in Listing A. Channel swap is implemented likewise, but along the other axis. Notably, we use ratio r to define N, the number of pairs to swap based on the number of tokens N_t. Hence, r can be larger than 1 since the total number of possible pairs $\binom{N_t}{2} \gg N_t$ for sufficiently large N_t. This also implies the same token can be swapped more than once. Empirically we set $r = 0.1$ for both swaps, and $\omega = 3$. We demonstrated that SSG is robust across a wide range of r and ω.

B. Further Analysis

We provide additional analytical studies and discussion to gain further insights into our method. Experiments in the section are conducted by generating 3k images using SDXL and evaluating on the MS-COCO 2014 validation data, unless otherwise specified.

Applying SSG to different layers of the backbone. The U-Net backbone in existing diffusion models [32, 36] is typically organised into downsampling, middle, and upsampling stages. For SDXL, these stages contain 24, 10, and 36 layers, respectively. Amongst prior methods, both PAG [1] and SEG [18] apply their perturbations to the middle layers of the backbone. Since SSG's perturbation operates through

SA	CA	FF	FID↓	CLIP↑	IS↑	AES↑	PickScore↑	IR↑
✓	✗	✗	31.41	0.313	34.44	5.901	22.18	0.297
✗	✓	✗	**30.96**	0.313	35.76	5.900	22.18	0.279
✗	✗	✓	31.26	0.313	**36.04**	**5.906**	**22.19**	**0.302**
✓	✓	✓	31.50	0.313	35.29	5.901	22.18	0.289

Table B. **Effect of applying SSG to different modules of the U-Net backbone.**

Guidance	FID↓	CLIP↑	IS↑	AES↑	PickScore↑	IR↑
SAG [19]	43.97	0.295	22.12	5.756	20.65	-0.483
SAG + **SSG**	**32.40**	**0.315**	**31.40**	**5.914**	**22.01**	**0.295**
SEG [18]	38.15	0.303	26.15	5.888	21.38	-0.0239
SEG + **SSG**	**31.60**	**0.313**	**34.98**	**5.901**	**22.20**	**0.308**
PAG [1]	36.79	0.306	29.42	5.827	21.57	0.00171
PAG + **SSG**	**32.53**	**0.313**	**35.25**	**5.897**	**22.18**	**0.292**

Table C. **Compatibility of SSG and previous condition-free guidance methods.**

a different mechanism, we explore applying it at various locations. Quantitative results in Table A reveal that placing SSG in the upsampling layers yields the best performance overall, whereas middle layers perform the worst—contrary to the observations made in PAG and SEG. Qualitatively, we find that applying SSG in the upsampling layers leads to less hallucination and unrealistic structures. It also produces images that are less susceptible to distribution drift, such as changing hue and style. Therefore, we choose to adopt the upsampling layers of SDXL's backbone model as the default location for SSG. Some visual comparisons are made in Figure E. Note that in our experiments, we avoid using the very first few downsampling layers or the final upsampling layers, as perturbing them severely disrupts image content and style.

Applying SSG to different Transformer modules. The U-Net backbone of SD1.5 and SDXL contains a stack of Transformer blocks throughout different network stages. Each Transformer block comprises a cascade of self-attention, cross-attention, and feed-forward modules. Table B quantitatively examines the effect of applying SSG to the token input to each module. We find that applying SSG immediately before the self-attention modules yields the best metric results overall. Visual inspection suggests that applying SSG to different modules generally produce very similar images, although applying it before self-attention occasionally offers better structures and photorealism. Some cases of choosing SA outperforming other modules are selected and presented in Figure F.

Compatibility with existing condition-free guidance. As SSG's token-level operation is orthogonal to the input- [19, 40] or attention-level [1, 18] perturbations in previous approaches, it can be integrated with these methods

Dataset	Method	FID↓	CLIP↑	IS↑	AES↑	PickScore↑	IR↑
COCO14	TPG	22.67	**0.313**	32.38	5.815	21.84	0.234
	SSG	**21.73**	**0.313**	**34.63**	**5.902**	**22.17**	**0.276**
COCO17	TPG	33.04	**0.312**	32.09	5.811	21.83	0.200
	SSG	**31.92**	**0.312**	**34.15**	**5.890**	**22.14**	**0.253**

Table D. **Comparison with Token Perturbation Guidance (TPG) [35].**

Method	FID↓	CLIP↑	IS↑	AES↑	PickScore↑	IR↑
TPG	32.41	0.312	32.17	5.816	21.84	0.221
SSG†	33.64	0.313	34.40	5.867	22.02	0.238
SSG	**31.41**	**0.313**	**34.44**	**5.901**	**22.18**	**0.297**

Table E. **Performance of SSG using TPG's perturbed layers.**

to achieve further quality improvements. As demonstrated by the empirical results in Table C, when added on top, SSG complements and considerably improves over existing condition-free guidance methods. We also provide some example images for visualisation in Figure G. It is clear than SSG consistently improves the quality of images generated by existing methods, offering improved structures, textures, semantics, and overall photorealism.

Distinct patterns of SS and CS guidance. As can be observed from the plot in Figure A, SS and CS produce increasingly divergent guidance directions towards later steps (averaged over 50 samples), again revealing their complementarity. On the right of Figure A, further analysis examines the orthogonal component between $\Delta\epsilon$ and ϵ_{ori}, theoretically shown to drive quality improvements in previous research [39]. The visualised guidance maps at different timesteps unveil the unique dynamics of SS and CS.

Comparison with TPG. We compare SSG with the recent method of Token Perturbation Guidance (TPG) [35]. Experiments are conducted under the conditional generation setting on full COCO 2014 and COCO 2017 validation data. As shown in Table D, SSG consistently generates images with higher quality and diversity across both datasets. In terms of prompt alignment, SSG matches TPG by achieving the same CLIP scores on both datasets.

Furthermore, we investigate whether SSG's superior performance over TPG stems from their different choices of layers for applying perturbation. To this end, we apply SSG to the same layers selected by TPG, *i.e.*, downsampling layers 6 to 23. The results, reported in Table E and marked by "†", show that SSG still outperforms TPG under this configuration. This indicates that SSG's superiority is not merely a result of identifying more suitable layers for perturbation.

Computational cost. We report the computational cost of different condition-free sampling guidance methods in Table F, comparing both the GPU memory consumption

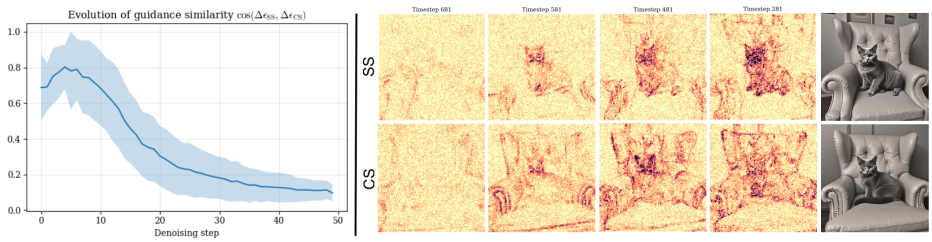

Figure A. **Visualised distinct characteristics of spatial swap (SS) and channel swap (CS).** Left: evolution of guidance vector similarity between SS-induced guidance and CS-induced guidance. Right: visualised SS- and CS-induced guidance maps at different timesteps.

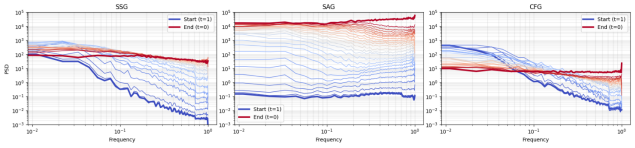

Figure B. **PSD profile of guidance vectors across different timesteps.**

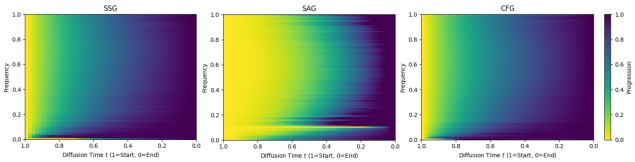

Figure C. **Progression of frequencies in denoised latents across different timesteps.**

and the per-sample inference time. In terms of memory usage, SSG is consistent with PAG and SEG, and is only marginally higher than SAG and the guidance-free vanilla SDXL model. For latency, SSG runs faster than the prior approaches of SAG and SEG. The random swap variant of SSG (*i.e.*, SSG-R), discussed earlier in Section 5.3 and Table 5, is even more efficient, while retaining a considerable performance advantage over existing methods (see Table 5).

Frequency perspective analysis. In Figure B, tracing the PSD profile of guidance vector $\Delta\epsilon_{\text{SSG}} = \epsilon_{\text{ori}}(x_t) - \epsilon_{\text{pert}}(x_t)$ reveals SSG guides more discriminatively, especially over early steps where it focus more on low-freq content. SAG shows monotonically increasing PSD across timesteps over all frequencies, which means its guidance globally amplifies instead of selective refinement. We also visualise the progression of frequencies in denoised latent across diffusion timesteps in Figure C. SSG exhibits faster, more uniformly paced progression across different frequencies, indicating more steady guidance process. Notably, for both analyses, SSG more closely aligns with the behaviours of CFG, possibly also explaining its effectiveness.

Limitations. While SSG greatly improves image quality, occasionally it generates stylised images, even without explicit instruction, such as vintage, cartoon, or watercolour appearances. This style drift is also observed in other guidance methods and appears linked to the pretraining of

Method	w/o Guidance	SAG	PAG	SEG	SSG	SSG-R
Memory (MB)	6,901	6,980	7,060	7,060	7,060	7,060
Run-time (s)	14.77	29.74	15.10	18.94	18.09	17.06

Table F. **A comparison of computational cost** of different sampling guidance methods.

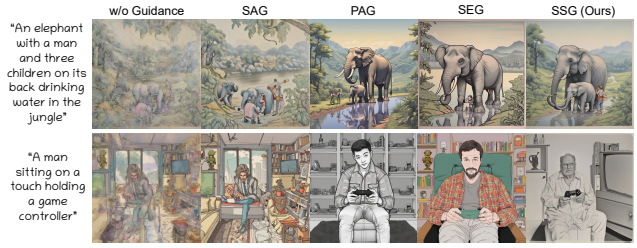

Figure D. **Some cases of drifted style.** SSG and previous condition-free guidance methods [1, 18, 19] occasionally generate images in non-photorealistic styles not explicitly instructed.

vanilla diffusion models for unguided sampling. While it may be mitigated by specifying "photorealistic style" or using negative prompts, understanding why guidance methods sometimes fail to recover from unnatural distribution of unguided sampling remains an open research question.

C. Additional Qualitative Results

Unconditional generation by SD1.5. Figure H showcases unconditional generation results by SD1.5. Similar to the observations made for SDXL, under the unconditional setting, SSG is less prone to generating images of oversimplified structures and layouts with unnatural style, and is more likely to generate photorealistic images from the baseline model's outputs.

Conditional generation by SD1.5. Images generated by SD1.5 are generally in lower fidelity and prompt alignment quality, due to the limited capability of the SD1.5 model. Nonetheless, in most cases SSG is able to produce images with improved quality compared to existing methods, which also corroborates its better quantitative results. Some examples are provided in Figure I.

Demo video. We also include a video in the supplementary package to better illustrate image generation with SSG.

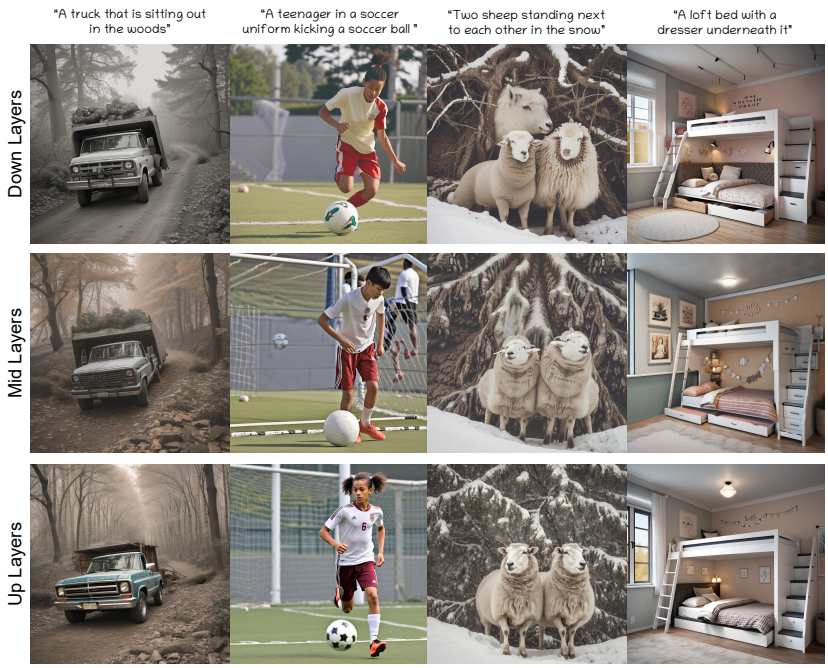

Figure E. **Effect of applying SSG to downsampling, middle, and upsampling layers of SDXL's U-Net backbone.**

Figure F. **Effect of applying SSG to different modules within SDXL's transform blocks.**

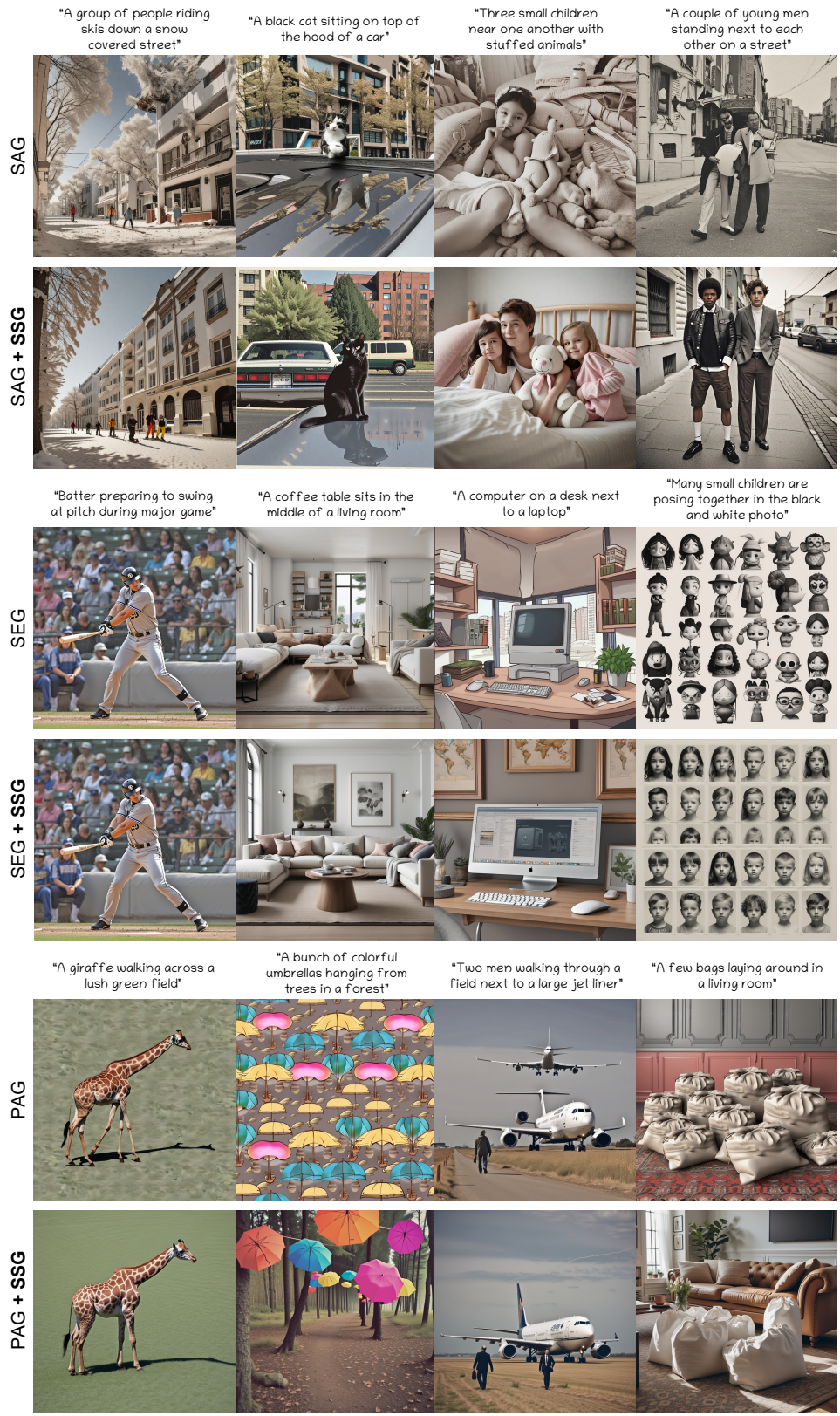

Figure G. **Effect of applying SSG on top of existing methods SAG [19], SEG [18], and PAG [1].**

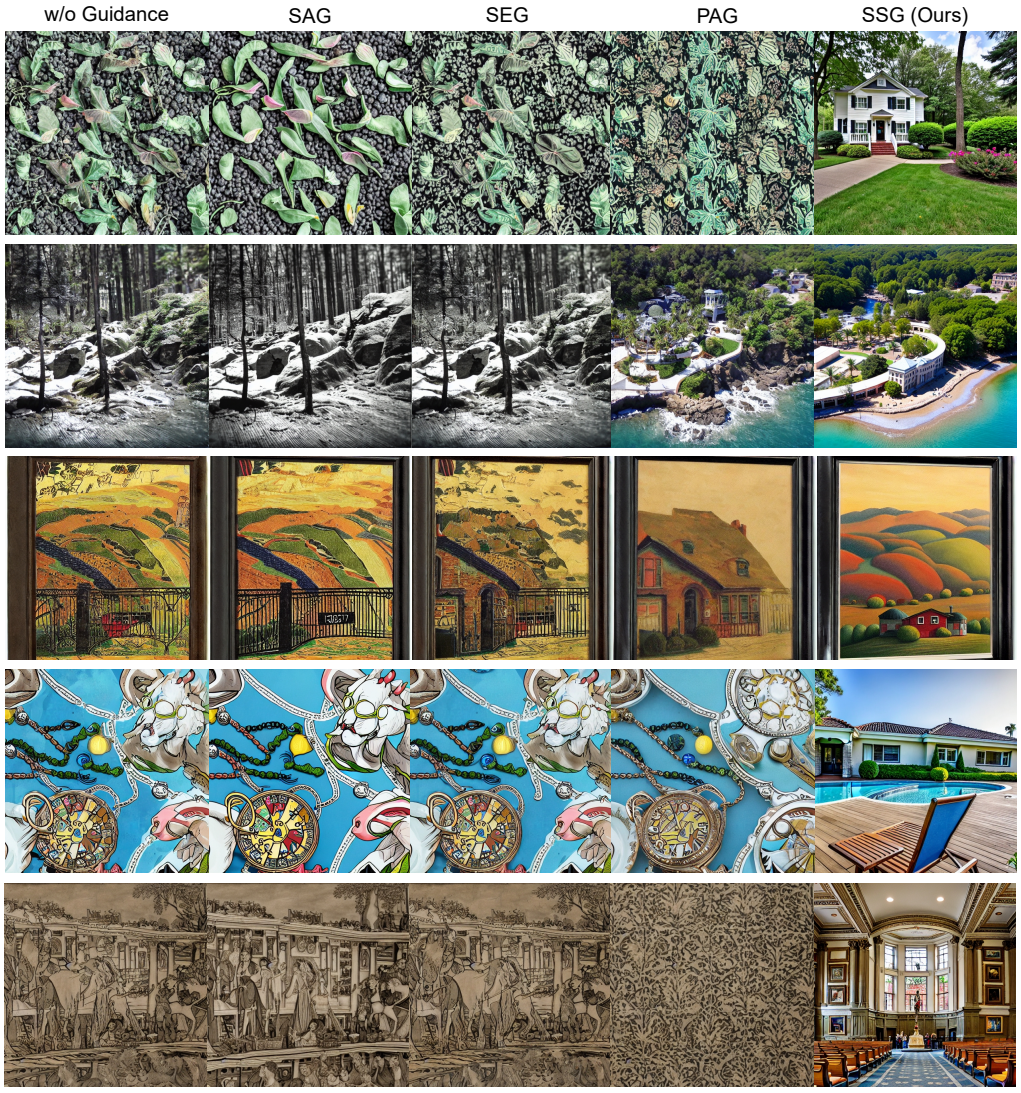

Figure H. **Qualitative comparison of unconditional generation by SD1.5.**

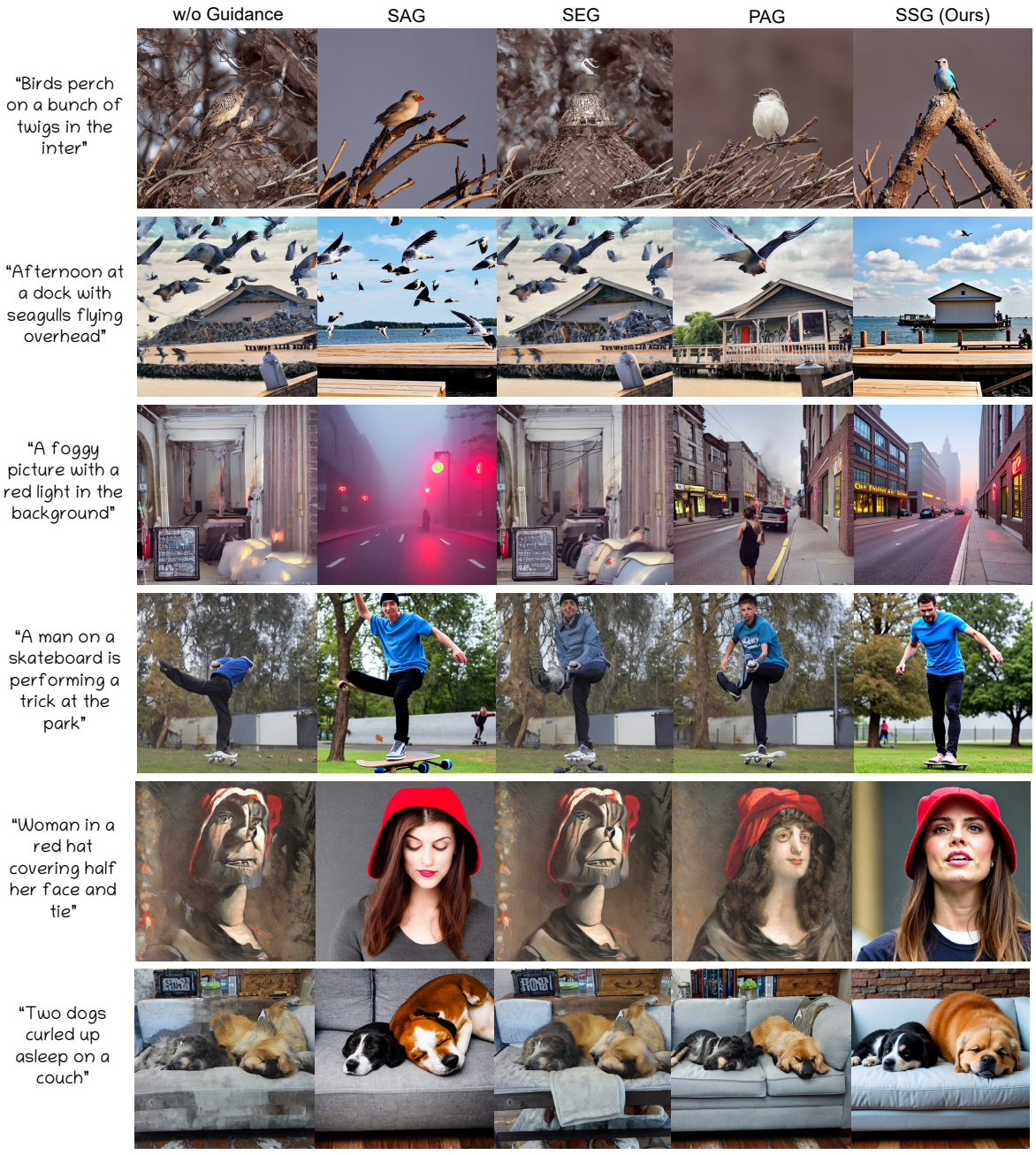

Figure I. **Qualitative comparison of conditional generation by SD1.5.**