Your Text Encoder Can Be An Object-Level Watermarking Controller

Supplementary Material

A. Post-generation vs. In-generation Watermarking: Limitations and Advances

In *post-generation watermarking*, also known as post-hoc watermarking, watermarks are injected into images after generation [1]. This approach, while straightforward, incurs additional computational overhead and is vulnerable to circumvention. For example, in cases of model leakage, attackers can easily detect and bypass the postprocessing module [10]. Additionally, post-diffusion methods often result in poorer image quality, introducing artifacts, and their separable nature makes them easily removable in open-source models, such as by commenting out a single line of code in Stable Diffusion code base [4].

In contrast, *in-generation watermarking* integrates the watermarking process into the image generation pipeline, improving stealthiness and computational efficiency. This approach embeds watermarks directly into generated images without requiring separate post-hoc steps. It is also less susceptible to diffusion denoising or removal via simple modifications.

The closest related work is [6], which also embeds watermarking bits during generation. However, their method requires retraining a UNet layer, limiting its plug-and-play capability, as discussed in Section 2 and Figure 4. Our approach, by comparison, is more convenient, embedding watermarks directly within the text prompt as a watermarking token.

Focusing on object watermarking, our method differs from [11] and [8] in several key aspects. While these methods rely on segmentation maps for supervised watermarking of specific objects in pre-existing images, ours is unsupervised, leveraging attention maps generated automatically during text-to-image generation. Unlike post-hoc methods, which watermark only pre-existing images, our approach simultaneously performs text-to-image generation and watermarking. This enables compatibility with techniques like textual inversion and ensures robustness against attacks, overcoming significant limitations of post-hoc methods.

B. Algorithm for Watermark Heatmap Generation

We present an algorithm for generating the heatmaps depicted in Figure 4 of the main paper. The process involves iterating a defined patch across the image to construct the heatmap. Details regarding the minimum allowable size of the patch can be found in Section 4.4. The patch represents the smallest region of the image that can reliably extract bits using a message detector with high confidence; for further explanation, refer to Section 4 in the main paper. Notably, the detector employed is an off-the-shelf solution. The resulting heatmap is normalized to a range of $[0, 1]$, where a value of 0 indicates regions with 0% bit accuracy, and a value of 1 corresponds to regions achieving 100% bit accuracy. Lastly, we apply Gaussian blur that preserves the edges and boundaries better than other uniform blurring filters, which is important for maintaining the structure of the heatmap.

Method	Imperceptibility			Robustness to Basic Attacks (BA):				
	PSNR ↑	SSIM ↑	FID ↓	None ↑	Bright. ↑	Contrast ↑	Blur ↑	JPEG ↑
Dct-Dwt	39.50	0.97	15.93	0.96	0.89	0.91	0.90	0.55
SSL Watermark	31.50	0.86	21.82	0.95	0.91	0.84	0.88	0.55
HiDDeN	31.57	0.88	22.67	0.99	0.93	0.88	0.80	0.88
Ours with [12]	**37.92**	**0.95**	**15.83**	**0.99**	**0.99**	**0.97**	**0.96**	**0.96**
Ours with [3]	**40.92**	**0.97**	**14.83**	**0.99**	**0.98**	**0.99**	**0.97**	**0.96**

Table 1. **Performance comparison of our method with post generation watermarking methods.** We evaluate imperceptibility and robustness against basic attacks. Higher PSNR and SSIM, and lower FID, indicate better imperceptibility. Higher robustness scores indicate greater resistance to attacks.

C. Number of Bits v/s Bit Accuracy

We perform an ablation study on the number of bits that can be embedded into images by our method, without sacrificing on bit accuracy. Methods such as [2] can embed upto 100 bits during watermarking. Extending the effort to push the benchmark for number of bits, we show our results on watermarking upto 128 bits.

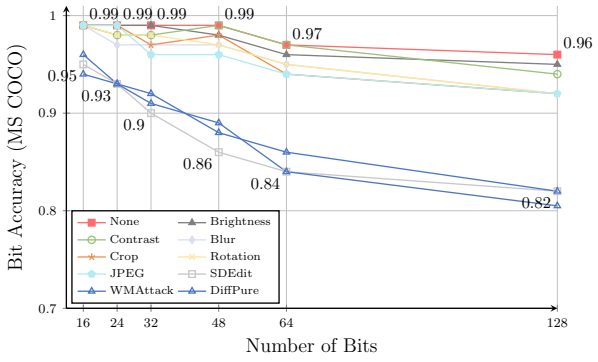

Figure 1. **Number of bits v/s Bit Accuracy:** It can be seen from the above plot that our method can embed 128 bits without loss in bit accuracy. We observe a bit accuracy of over 90% with 128 bits.

Algorithm 1 Watermark Heatmap Generation

Input: Watermarked image $\mathbf{I}$; Watermark Detector $\mathcal{D}_w$; Ground Truth Key m.

Select a patch dimensions $h \times w$.

Divide $\mathbf{I}$ into overlapping patches $\{P_{ij}\}$, where P_{ij} is the patch at location (i, j) of size $h \times w$.

Initialize a heatmap matrix $\mathbf{H}$ of zeros with the same spatial resolution as $\mathbf{I}$.

for each patch P_{ij} in $\mathbf{I}$ **do**

Extract watermark string from the patch:

$$m'_{ij} = \mathcal{D}_w(P_{ij})$$

Compute the bit accuracy for the patch as (where $|m|$ is the length of the bit string):

$$A_{ij} = \frac{1}{|m|} \sum_{k=1}^{|m|} [m_k = m'_{ij,k}]$$

Assign the bit accuracy value to the center pixel of the patch in the heatmap $\mathcal{H}$:

$$\mathbf{H}(i, j) \leftarrow A_{ij}$$

end for

Normalize the heatmap values to the range $[0, 1]$:

$$\mathbf{H}_{\text{norm}} = \frac{\mathbf{H} - \min(\mathbf{H})}{\max(\mathbf{H}) - \min(\mathbf{H})}.$$

Apply a smoothing filter (Gaussian blur) to $\mathbf{H}_{\text{norm}}$:

$$\mathbf{H}_{\text{smooth}} = \text{GaussianBlur}(\mathbf{H}_{\text{norm}}, \sigma).$$

Generate a heatmap image by overlaying $\mathbf{H}_{\text{smooth}}$ on the original watermarked image $\mathbf{I}$.

Output: Heatmap image highlighting regions with watermark accuracy.

D. Comparison to methods that use TPR as evaluation metric

We report True Positive Rate of our watermark detection in comparison with baselines including [5, 10]. We threshold the FPR to 0.1% and report TPR:0.1%

E. Empirical study to find the optimal timestep for watermarking

As discussed in the ablation studies section of our paper, we perform empiracal study to find the optimal timestep τ^* that provides an balance between the trade off between watermark invisibility and bit accuracy.

Method	Imperceptibility			Robustness to Basic Attacks (TPR):				
	PSNR ↑	SSIM ↑	FID ↓	None ↑	Bright. ↑	Contrast ↑	Blur ↑	JPEG ↑
TreeRings	33.75	0.91	18.93	**1.00**	0.91	0.90	0.92	0.89
ConceptWM	32.89	0.89	24.58	0.98	0.90	0.84	0.83	0.85
Ours	**40.92**	**0.97**	**14.83**	0.99	**0.99**	**0.98**	**0.99**	**0.97**

Table 2. **Performance comparison of our method with methods that use TPR as evaluation metric.** We evaluate imperceptibility and robustness against basic attacks. Higher PSNR and SSIM, and lower FID, indicate better imperceptibility. Higher robustness scores indicate greater resistance to attacks.

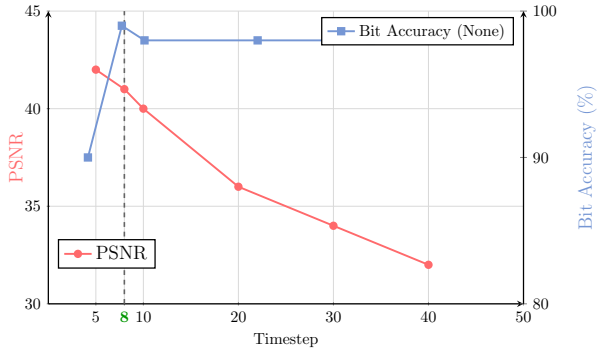

Figure 2. **Optimal sub-range of noise timesteps for invisible watermarking.** Image quality and watermark robustness are crucial evaluation metrics for watermarking techniques. Our method integrates watermarking into the core denoising module of a T2I pipeline. We study the evaluation metrics for various noise timesteps, and we find that to maintain invisible watermarking, the timesteps closest to the LDM encoder during forward process are optimal. We test this finding for multiple runs to utilize this finding into our watermarking method.

F. Details on Training and Inference time for watermarking

As discussed in our method adds τ^* timesteps of noise during the forward processing and performs denoising to train $\mathcal{W}_*$ token embeddings. We use 2000 images from MS COCO dataset for training and the entire training process takes about 2 GPU hours when benchmarked on NVIDIA RTX A6000 GPU.

During inference, the time overhead is minimal, the significant storage savings (a $10^5\times$ reduction in parameters) are noteworthy. Benchmarking on an RTX A6000, processing a single image (averaged over 1000 images) shows that for a T2I model, processing 50 diffusion steps takes 5.2 seconds without watermarking and 5.4 seconds with a minimal 0.2-second overhead due to the watermarking token.

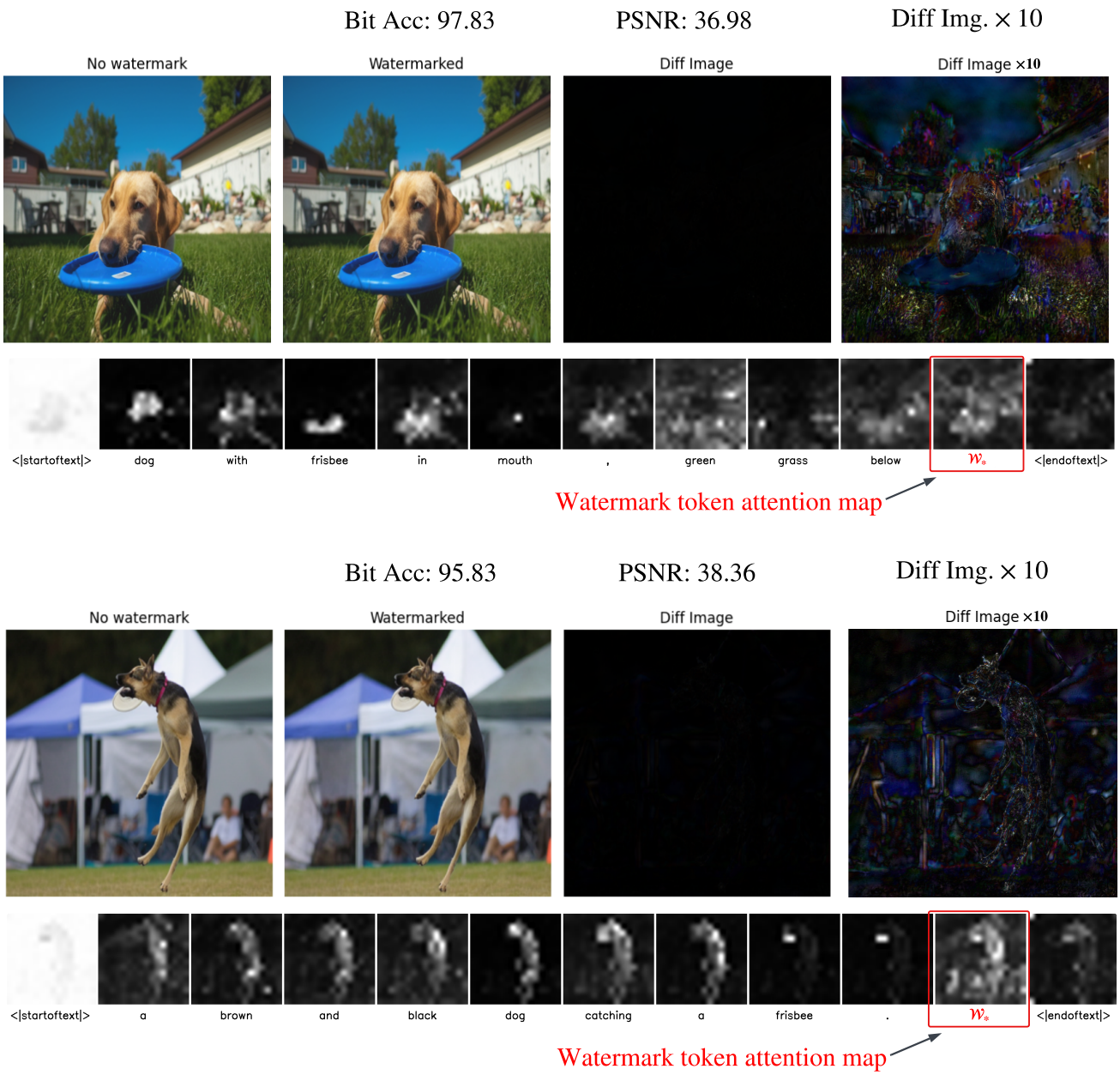

Figure 3. **Watermark Invisibility and $\mathcal{W}_*$ Attention Map** *We provide the attention map of our watermarking token $\mathcal{W}_*$. Our method achieves a very high PSNR of 38.36 while maintaining a bit accuracy of over 95%. The attention map of $\mathcal{W}_*$ does not interfere with the attention maps of other tokens, preserving overall image quality.*

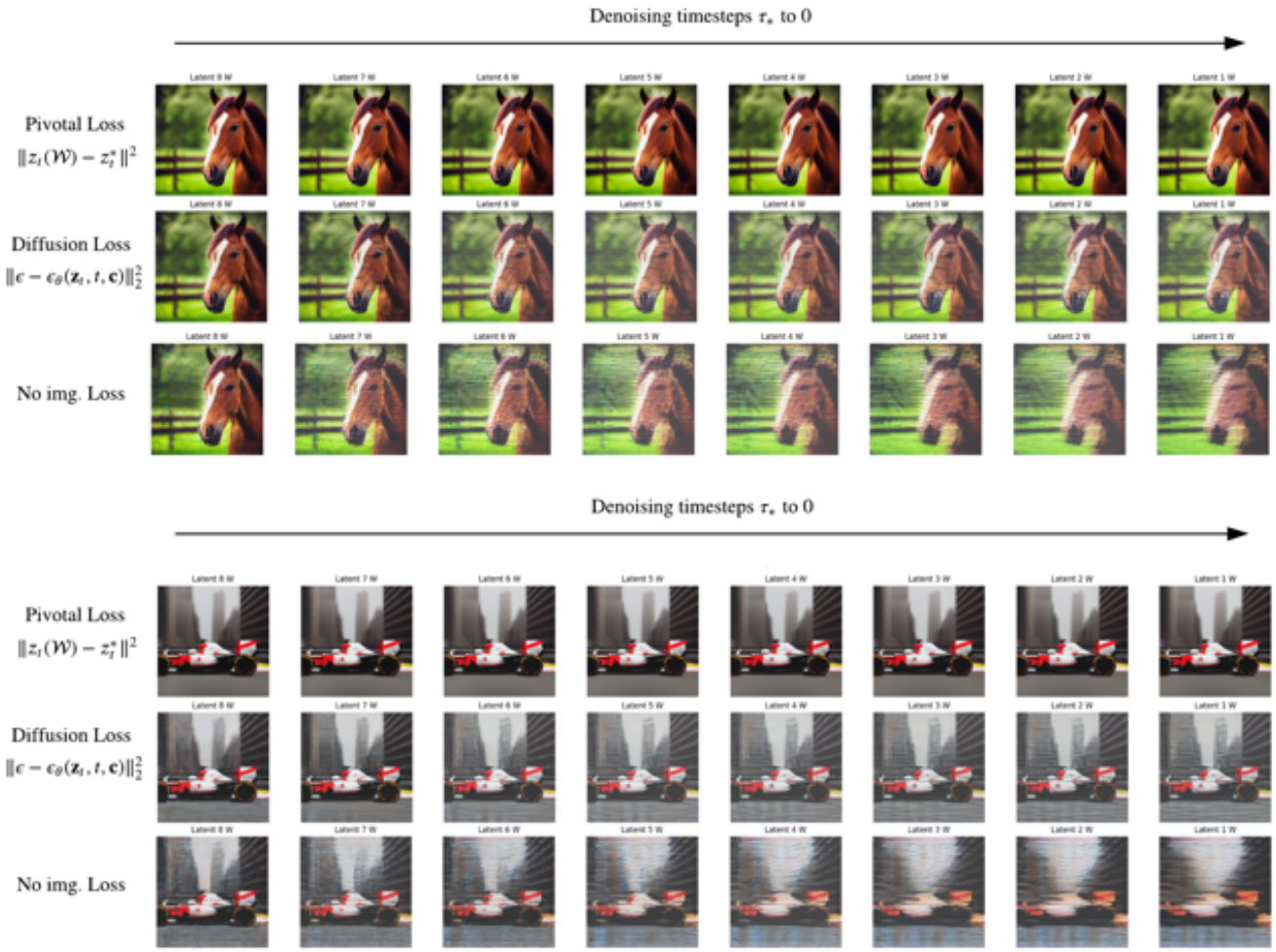

Figure 4. **Diffusion Loss v/s Latent Loss** *We provide qualitative results for our choice of latent loss in our method. This figure shows decoded latents at each step of denoising from τ_* to* 0 *within T2I generation. The first row corresponds to our method, which achieves the best image quality while watermarking. The second row represents training using Diffusion Loss, while the last row shows image quality when $\mathcal{W}_*$ is trained only on watermarking loss without any image loss applied.*

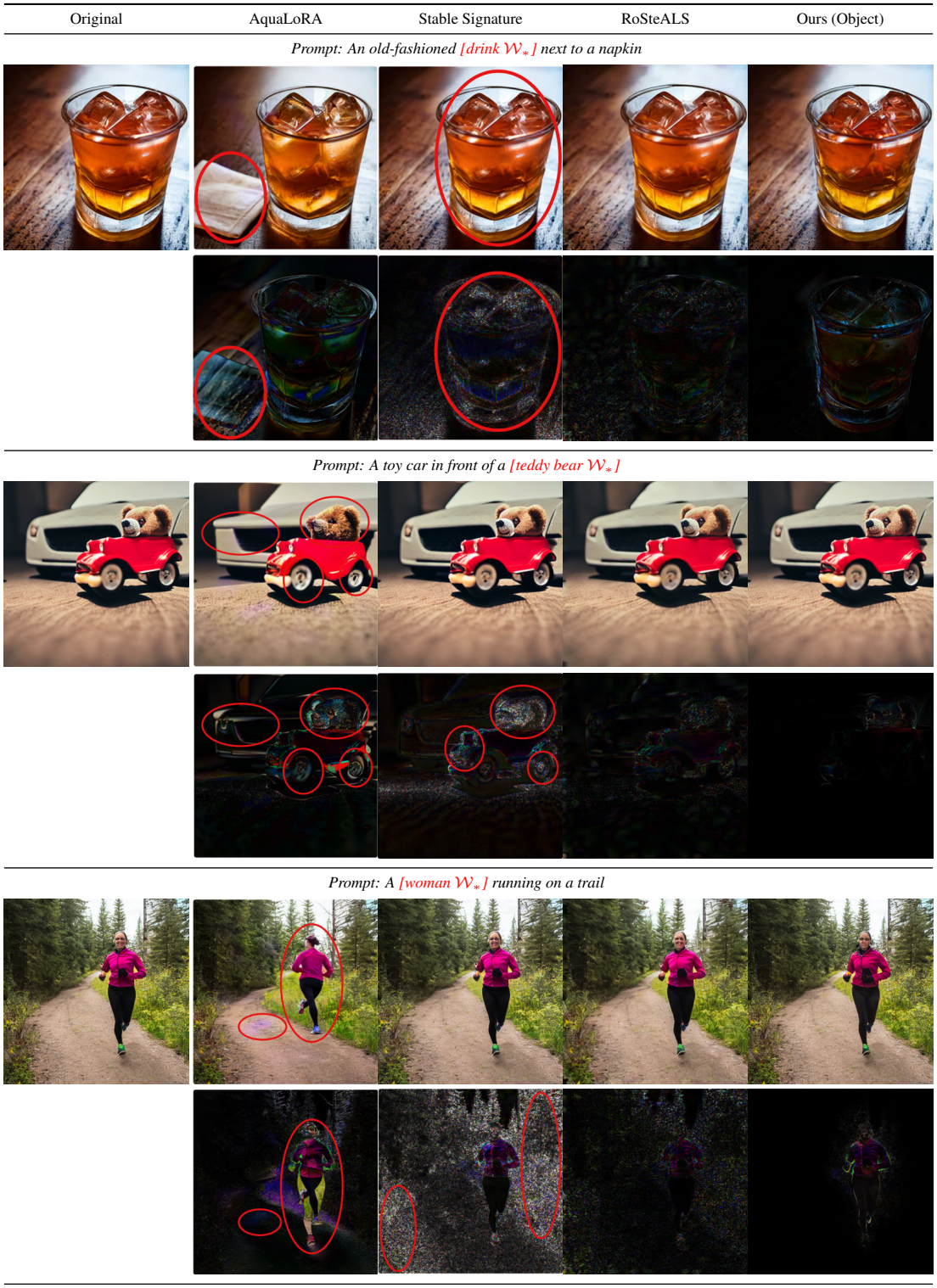

Figure 5. **Qualitative results 1 for different watermarking methods on generated images** *We provide several comparative examples with existing watermarking techniques for a qualitative analysis of watermark invisibility. The above images compare the performance of our method with AquaLoRA [3], Stable Signature [4] and RoSteALS [2]. We see that we surpass existing watermark methods in maintaining invisibility while watermarking an object within an image. In addition to the watermarked image, we also provide difference image below each watermarked image. Text in red denotes the object being watermarked by our method.*

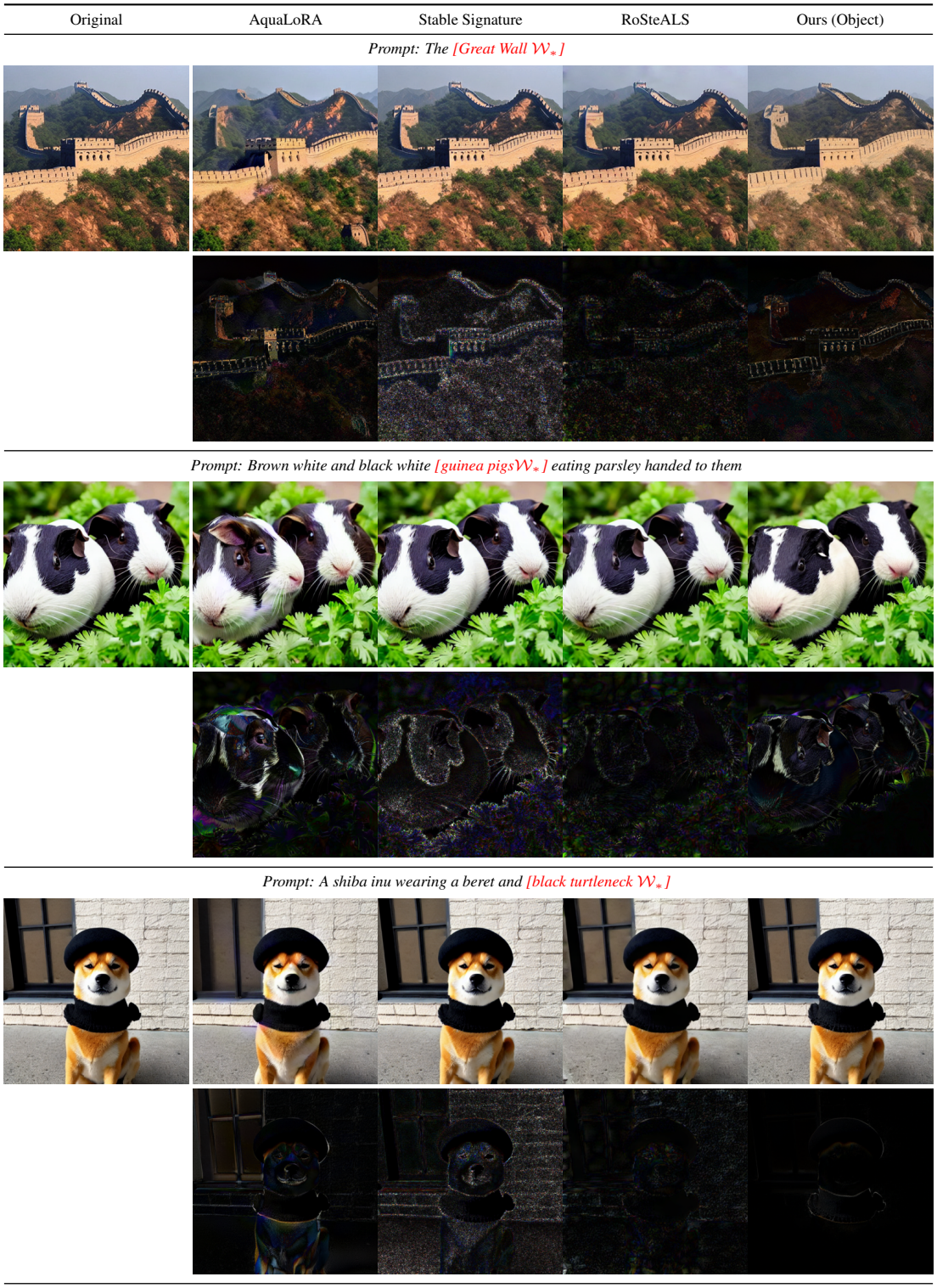

Figure 6. Qualitative results for different watermarking methods on generated images (part 2).

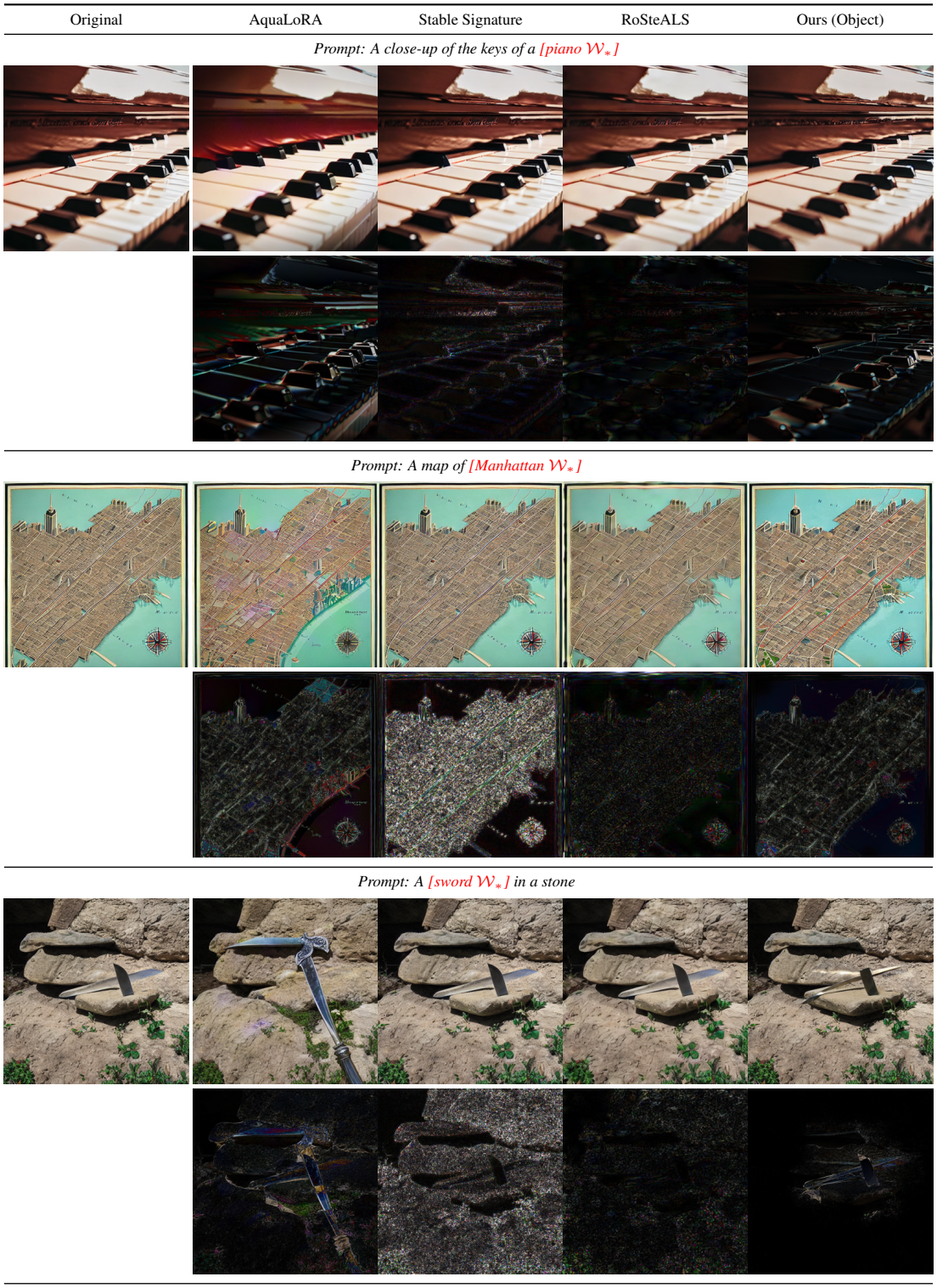

Figure 8. Qualitative results for different watermarking methods on generated images (part 3).

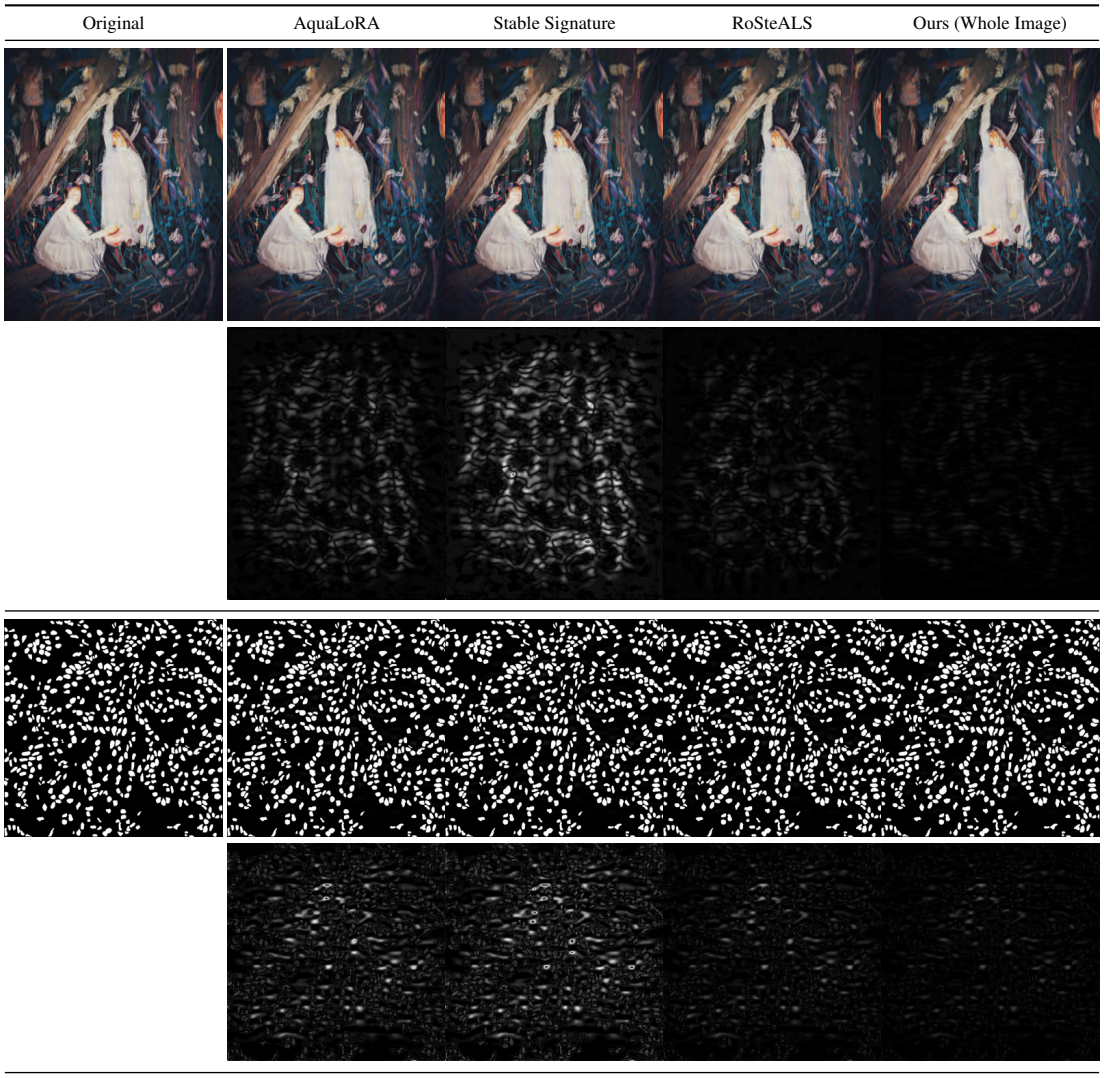

Figure 9. **Qualitative results for different watermarking methods on generated images** *We present our results compared to the baselines on images from WikiArt [9] and TNBC [7] datasets.*

G. Attention Map of Watermarking Token

Our method appends a token $\mathcal{W}_*$ to the text prompt to integrate watermarking within the T2I generation. We utilize attention maps of each token in the text prompt to overlay the attention map of watermarking token $\mathcal{W}_*$ to control object-level watermarking. Here we provide the attention map of $\mathcal{W}_*$ token during T2I generation. We see that the difference image of watermarked and non-watermarked images is negligible. We amplify the difference image by 10 and provide as the image in the last column.

H. Choice of Latent loss for Trajectory Alignment

Our method uses Latent loss $\|z_t(\mathcal{W}) - z_t^*\|^2$ (where $z_t(\mathcal{W})$ represent watermarked latents with $\mathcal{W}_*$ appended to the text prompt and z_t^* denote non-watermarked latents with good image quality) for trajectory alignment during watermarking. In this section, we provide comparative of different loss functions to ensure trajectory alignment (preserve image quality) during watermarking while denoising. We see that in Fig. 4 loss maintains the best invisibility for watermarking. While diffusion loss performs better the case without any image loss, we observe that, for black-box watermarking, diffusion loss performs sub-par compared to the chosen latent loss that controls the trajectory of the latents.

I. Robustness attacks implementation details

We follow standard implementation for simulation of various attacks for watermarking as mentioned in [4] and [3]. Crop 0.1 removes 10% from the image retaining the center. Resize 0.2 scales the image to 20% of its original size. Rotate 25 rotates the image by 25 degrees. Operations are performed using PIL and torchvision utilizing standard implementations from [3, 4].

J. Medical image watermarking

We tested our method to watermark medical images where invisibility of watermark is critical. Our method performs invisible watermarking our medical images with a high PSNR of **35.89** on the TNBC-Seg [7] dataset while maintaining a bit accuracy of **0.99**.

K. Qualitative results compared to baselines

In this section, we provide additional qualitative results compared to different watermarking methods that perform in-generation watermarking. We consider Stable Signature [4], AquaLoRA [3], and RoSteALS [2] for comparing qualitative results.

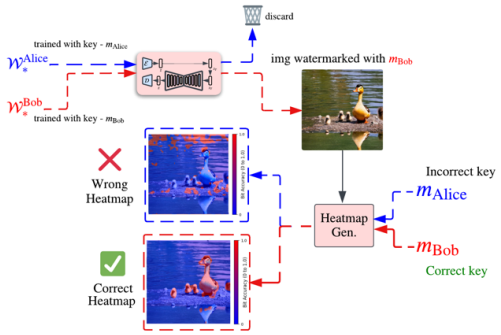

Figure 10. We show our watermark heatmap generation module's ability to retrieve heatmaps from watermarked images with different keys.

L. Heatmap detection based on user provided key

We present the ability of our watermark heatmap generation module's ability to retrieve correct watermarked regions based on user key provided.

References

[1] Bang An, Mucong Ding, Tahseen Rabbani, Aakriti Agrawal, Yuancheng Xu, Chenghao Deng, Sicheng Zhu, Abdirisak Mohamed, Yuxin Wen, Tom Goldstein, et al. Benchmarking the robustness of image watermarks. *arXiv preprint arXiv:2401.08573*, 2024. 1

[2] Tu Bui, Shruti Agarwal, Ning Yu, and John Collomosse. Rosteals: Robust steganography using autoencoder latent space, 2023. 1, 5, 9

[3] Weitao Feng, Wenbo Zhou, Jiyan He, Jie Zhang, Tianyi Wei, Guanlin Li, Tianwei Zhang, Weiming Zhang, and Nenghai Yu. AquaLoRA: Toward white-box protection for customized stable diffusion models via watermark LoRA. In *Proceedings of the 41st International Conference on Machine Learning*, pages 13423–13444. PMLR, 2024. 1, 5, 9

[4] Pierre Fernandez, Guillaume Couairon, Hervé Jégou, Matthijs Douze, and Teddy Furon. The stable signature: Rooting watermarks in latent diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22409–22420, 2023. 1, 5, 9

[5] Liangqi Lei, Keke Gai, Jing Yu, Liehuang Zhu, and Qi Wu. Conceptwm: A diffusion model watermark for concept protection, 2024. 2

[6] Rui Min, Sen Li, Hongyang Chen, and Minhao Cheng. A watermark-conditioned diffusion model for ip protection. *arXiv preprint arXiv:2403.10893*, 2024. 1

[7] M. Mehdi Owrang O, Fariba Jafari Horestani, and Ginger Schwarz. Survival analysis of young triple-negative breast cancer patients, 2024. 8, 9

[8] Tom Sander, Pierre Fernandez, Alain Durmus, Teddy Furon, and Matthijs Douze. Watermark anything with localized messages. *arXiv preprint arXiv:2411.07231*, 2024. 1

[9] Wei Ren Tan, Chee Seng Chan, Hernan Aguirre, and Kiyoshi Tanaka. Improved artgan for conditional synthesis of natural image and artwork. *IEEE Transactions on Image Processing*, 28(1):394–409, 2019. 8

[10] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-rings watermarks: Invisible fingerprints for diffusion images. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2

[11] Chengxin Zhao, Hefei Ling, Sijing Xie, Han Fang, Yaokun Fang, and Nan Sun. Ssyncoa: Self-synchronizing object-aligned watermarking to resist cropping-paste attacks. *arXiv preprint arXiv:2405.03458*, 2024. 1

[12] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks, 2018. 1