Supplementary Materials for
RoSteALS: Robust Steganography using Autoencoder Latent Space

A. Training details

RoSteALS is easy to train as long as it priorities the secret recovery loss at the early training phase. In Section 4.1 we propose a training method to overcome the complexity of the cover image domain (*e.g.* MIRFlickR is harder to train than FFHQ), the gradient flow between pretrained and learning-from-scratch modules, the challenges of large secret size, and the difficulties for the secret decoder to 'learn' perceptually invisible secret signals present in already high-quality images but corrupted with various perturbations. We adopt curriculum learning in our training schedule, starting from a fixed minibatch of cover images without noise corruptions, before unleashing the full training database and eventually enabling perturbations and linear loss weight ramping.

A successful training pipeline should be similar to Figure 1. We only experiment with few $(t_1, t_2, \beta_{\text{max}})$ tuples and settle with $(t_1 = 0.90, t_2 = 0.98, \beta_{\text{max}} = 10.0)$, therefore believe that performance could potentially be improved further with more careful parameter tuning.

B. Architecture details

RoSteALS has a very light-weight secret encoder and can be constructed using just 1 line of code using the Pytorch library. For example, for a 100-bit secret encoder:

```
secret_encoder = nn.Sequential(
  nn.Linear(100,32*32*3), nn.SiLU(),
  Lambda(lambda x: x.view(-1,3,32,32)),
  nn.Upsample((2,2)),
  nn.Conv2d(3,3,3,padding=1)
)
```

We experimented with more advanced architectures and found no clear benefits over this simple module.

C. Joint cover-secret conditioning

Existing works often model secrets and covers jointly, arguing the secret embedding should depend on the cover image for optimal stego quality. We observe that is not the case for RoSteALS, as shown in Figure. 1, 6 and discussed in Section. 4.3. Here, we implemented a RoSteALS alternative with the secret encoder E taking both the secret and cover as inputs. Specifically, the cover image is first blurred and downsampled to $H' \times W' \times C$, retaining only low frequency components. We then concatenate it with the upsampled secret embedding and passing to a sequence of convolution layers with SiLU activation. The weights of the last convolution layer is initialized with 0, in the same way as the proposed RoSteALS.

	PSNR	LPIPS	Bit acc.	Word acc.
		CLIC		
Proposed	**32.68 ± 1.75**	**0.04 ± 0.02**	0.94 ± 0.07	**0.93**
Joint cond.	32.45+-1.67	0.05+-0.02	0.94+-0.09	0.92
		MetFace		
Proposed	**34.46 ± 1.91**	0.04 ± 0.02	**0.94 ± 0.08**	**0.91**
Joint cond.	33.99+-1.81	0.04+-0.02	0.93+-0.09	0.90
		Stock1K		
Proposed	**33.27 ± 2.32**	**0.03 ± 0.02**	0.92 ± 0.10	0.86
Joint cond.	33.00+-2.18	0.04+-0.02	0.92+-0.11	0.86

Table 1. Joint cover-secret conditioning provides no benefit in RoSteALS design.

Table 1 shows the performance of this joint conditioning configuration, which is equal or slightly worse than the proposed approach in all metrics.

D. Perturbations

Figure 2 shows examples of 14 ImageNet-C perturbations used in our work. Note that there are 19 perturbations in ImageNet-C in total, we exclude 5 of them which are too slow to be included in training. Each perturbation has 5 levels of severity and its performance breakdown per level is shown in Figure 3. RoSteALS is most sensitive to degradation due to Gaussian, shot, impulse and speckle noises as well as jpeg compression; while being most resilient to brightness, pixelate and saturation effects.

E. More qualitative results

Figure 4 shows more qualitative examples of stego images created by RoSteALS and other baselines. The artifacts on StegaStamp generated images are perceptually visible, as if the image is covered with a transparent layer of fog. RoSteALS performance is comparable with other methods.

Figure 5 depicts several examples of our novel text-based steganography application. We note the glimpse of semantic objects visible in the residual image, however these artifacts are inevitable around the strongest edges during image generation and do not represent the secret artifacts to be picked up by the secret decoder (c.f. Figure. 6 in the main paper).

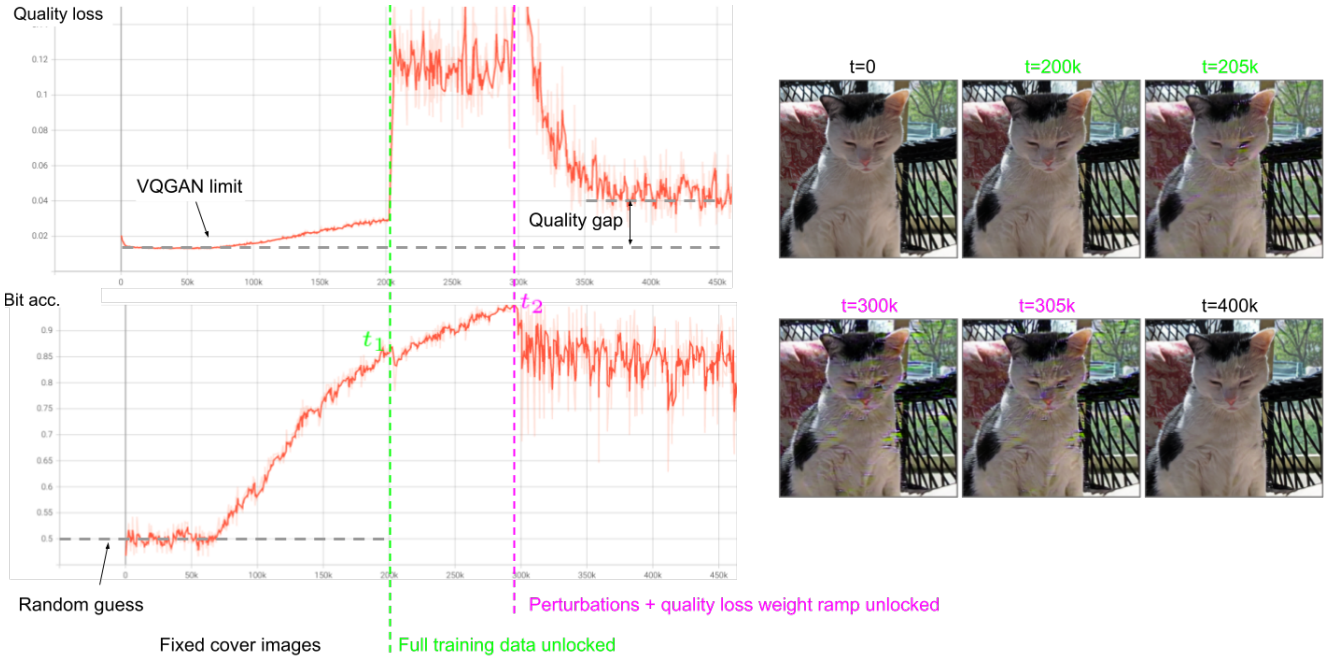

Figure 1. (Left) Quality loss and secret recovery curves for the first 18 epochs when training the 200bit-secret RoSteALS model on MIRFlickR with mini-batch size set to 4. (Right) Evolution of the stego image at different training stages.

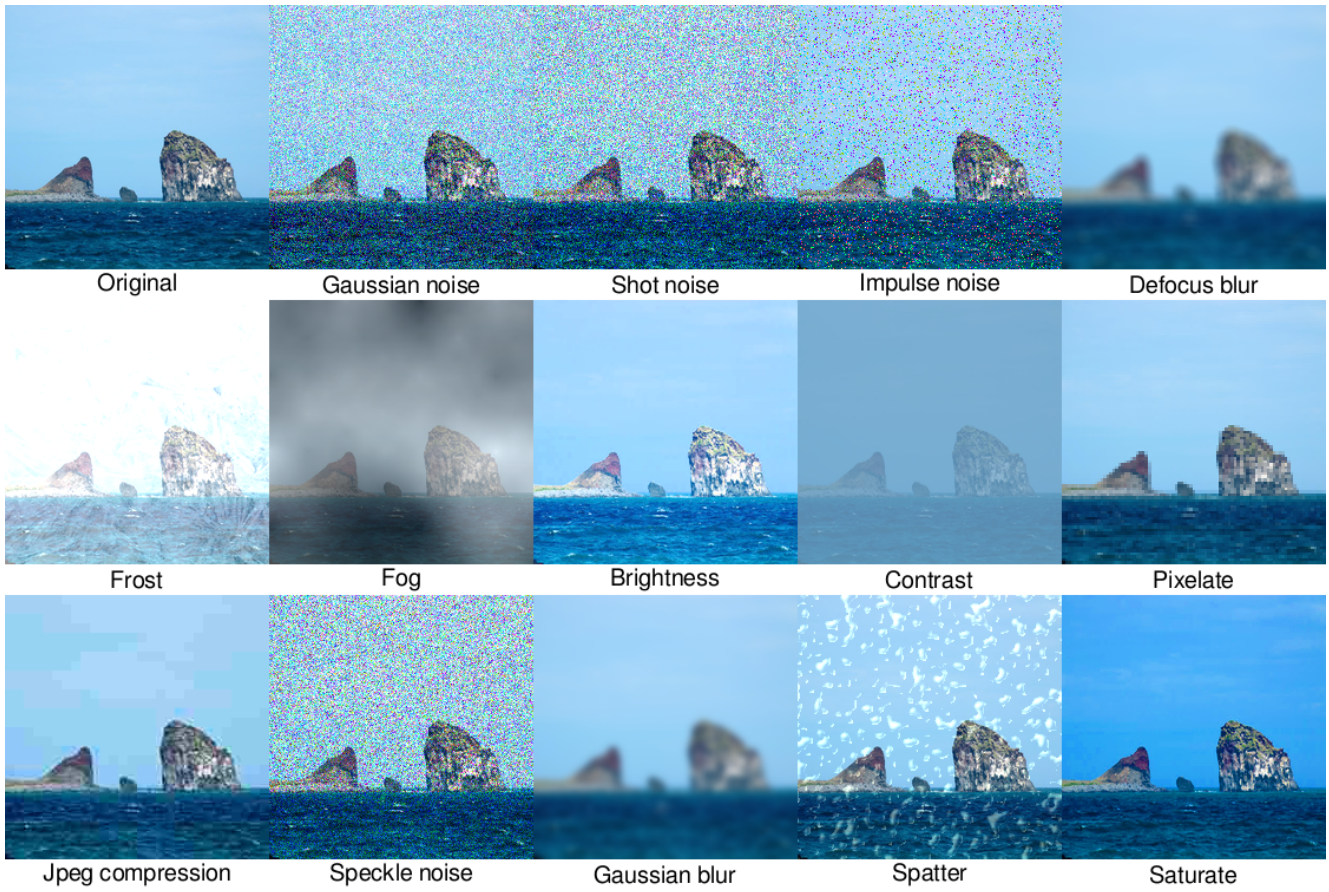

Figure 2. ImageNet-C perturbations on an example image, noise strength is set to 3 (out of 5).

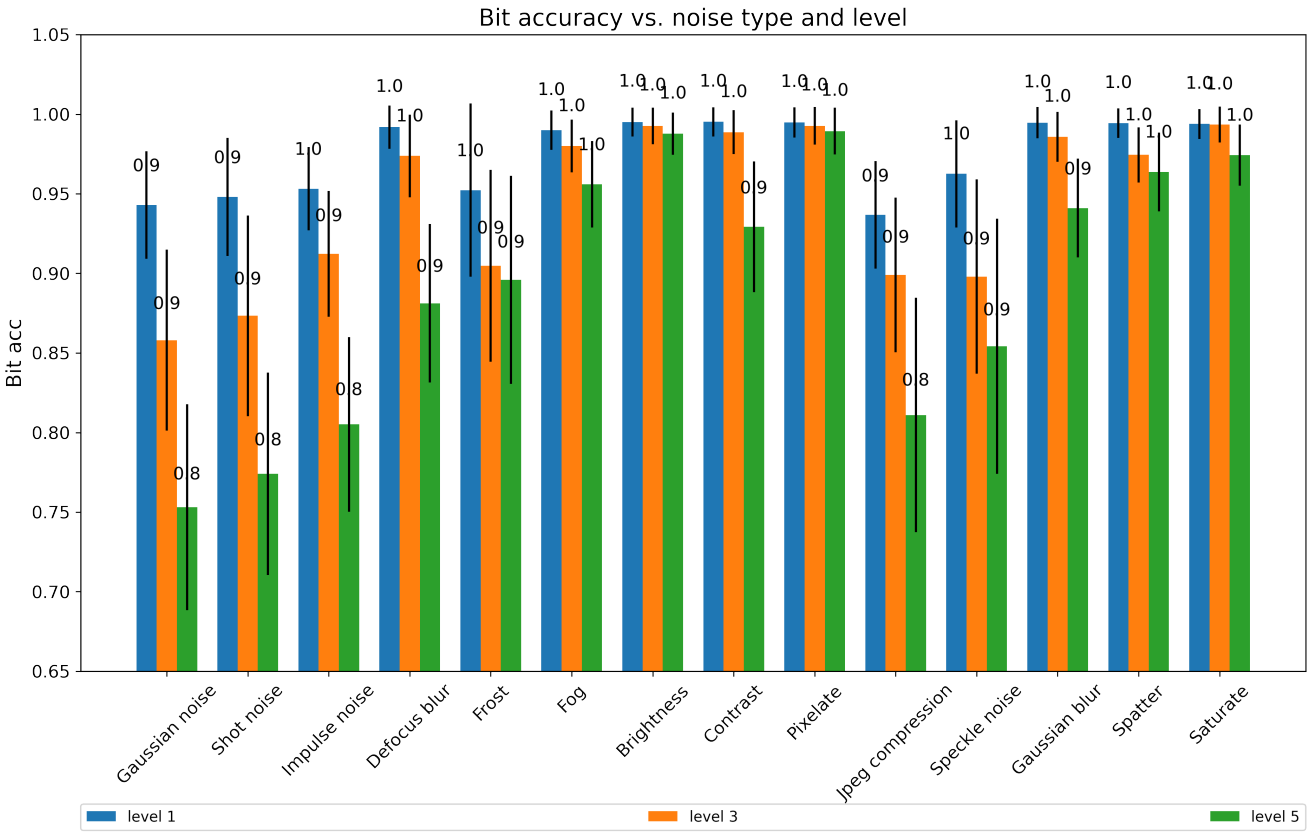

Figure 3. RoSteALS secret recovery performance breakdown for noise types and severity levels .

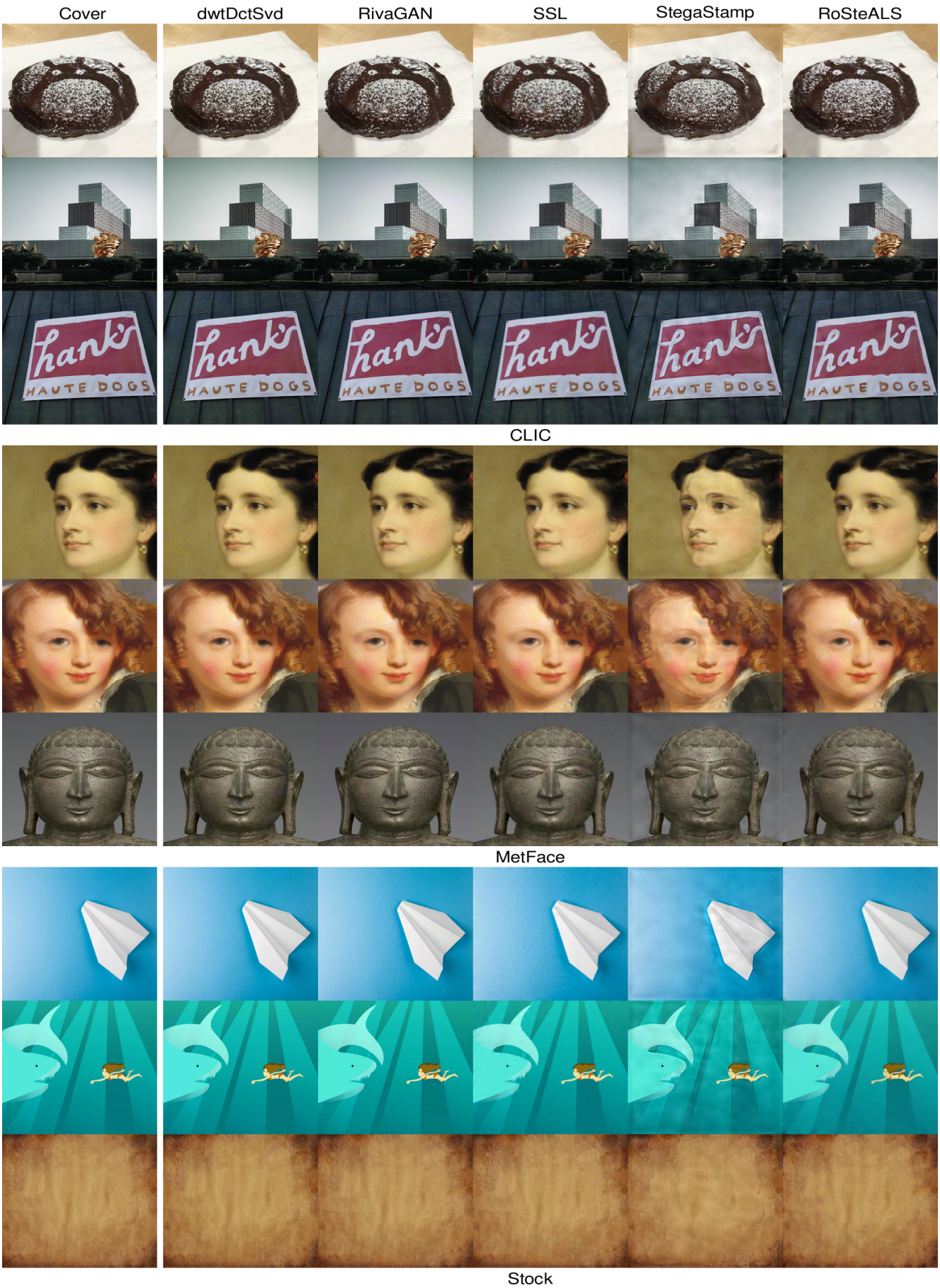

Figure 4. Stego images generated from several covers and a fixed secret. RoSteALS has better image quality than StegaStamp and perceptually comparable with other methods.

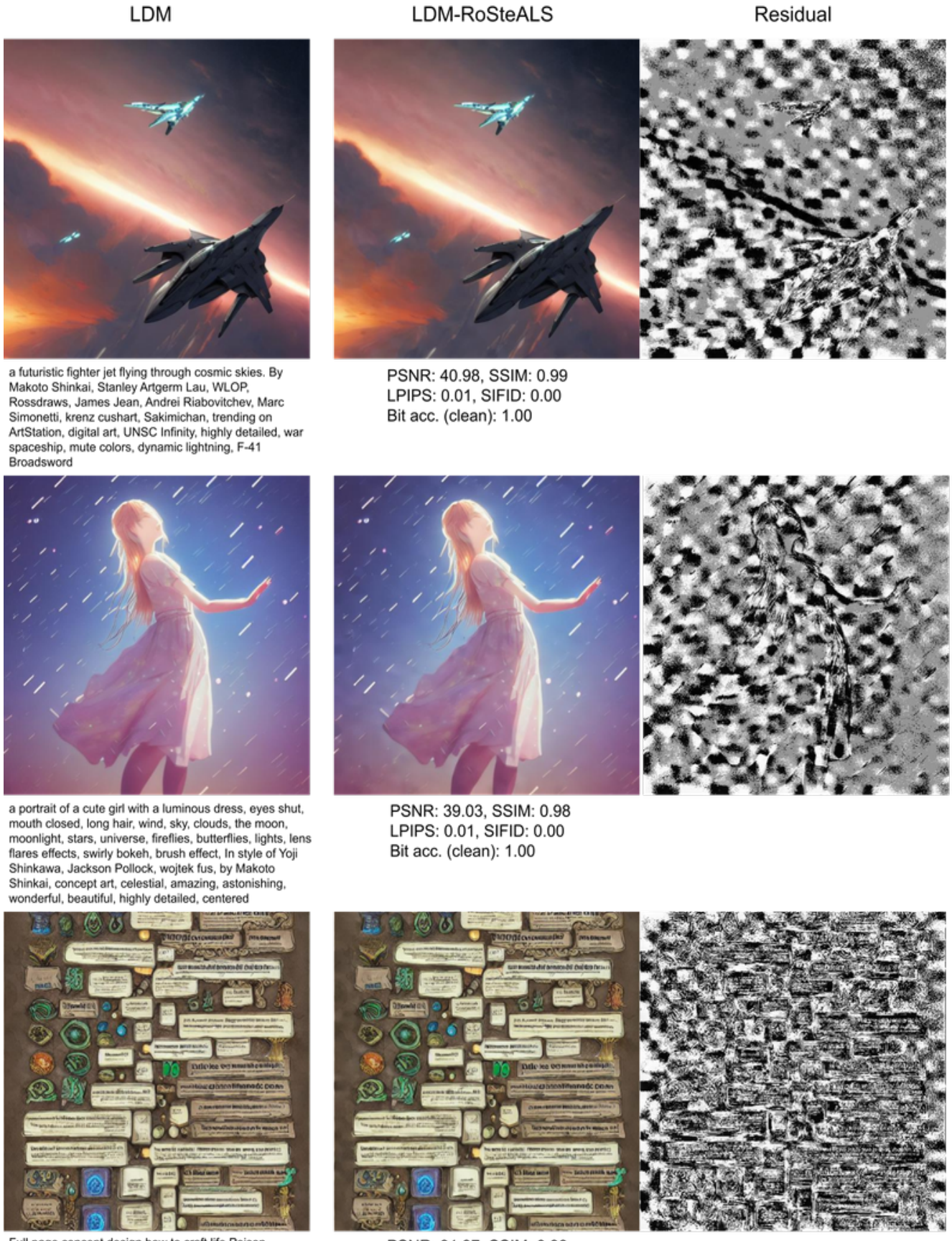

Figure 5. Text-based steganography with LDM-RoSteALS. (Left)- 512x512 images sampled from Stable Diffusion using the given prompts. (Middle) - Stegos with secret word “RoSteALS” injected. (Right) - Residual image scaled to [0,255] range.