Image Synthesis by Image-Guided Model Inversion Supplementary Material

Pei Wang*
UC, San Diego
pew062@ucsd.edu

Yijun Li, Krishna Kumar Singh, Jingwan Lu
Adobe Research
{yijli, krishsin, jlu}@adobe.com

Nuno Vasconcelos
UC, San Diego
nuno@ucsd.edu

1. Experimental Details

Dataset: We evaluate the proposed method on three datasets, includinig ImageNet [2], Places365 [13] and DTD [1]. For each dataset, two different subsets sampled from the validation set are used as target images, one for quantitative evaluation and the other one for the user study. ImageNet is the most widely used object recognition dataset, containing $1,000$ image categories. We randomly sample one validation image per category, producing a set of $1,000$ images which are used as target images. This is denoted as the ImageNet-A set. ImageNet-B is composed of 20 randomly selected images from ImageNet-A, which are used for user studies. Places365 is a popular scene images dataset, consisting of 365 scene categories. Three samples are randomly selected from each category in the validation set, producing a set of 1095 images, denoted as Places365-A. 20 images are then randomly extracted to construct a user study dataset, denoted as Places365-B. DTD is a texture dataset, containing 47 categories inspired by human perception. One image is randomly chosen per category from the validation set (47 images in total, DTD-A) and 20 images are randomly selected from them for user studies (DTD-B).

Classifier: The default classifier used in this work is ResNet-50 [5] pre-trained on ImageNet for object images and Places365 for scene images[1]. We also trained a ResNet-50 on DTD, following the settings of [5]. To test the robustness of IMAGINE, we have used another two classifiers: the robust classifier of [10] and shape robust classifier of [4][2]. The former one is trained using adversarial examples and the latter one is trained using the Stylized-ImageNet, to overcome the shape-bias of the standard classifier. Finally, we also ablate the importance of the network architecture, by implementing IMAGINE with the VGG19 [11] and AlexNet [7] models, which are publicly available in PyTorch[3].

Evaluation metric: Several quantitative evaluation metrics are used to measure the effectiveness of IMAGINE. The inception score (IS) [9] and FID [6] are used to simultaneously characterize the quality and diversity of generated images. For each method, 5 images are synthesized per target image. For the computation of FID, the real dataset used to compare with generated results is collected by randomly sampling 5 images per categories from the training set for ImageNet, and 15 images per categories from the training set for Places365. Meanwhile, for target domains that have only a few examples available, FID is not the best metric for measuring the generation quality. Therefore, we also compute the average LPIPS [12] over multiple pairs of generated images and their corresponding targets to measure the diversity. In addition, user studies are conducted to evaluate how realistic and diverse our generated results compared with different alternatives via Amazon Mechanical Turk. Images used for user study is accessible in path "$./user\_study/object/object.html$", "$./user\_study/scene/scene.html$", "$./user\_study/texture/texture.html$". The interfaces used for user studies are shown in Figure 1.

Data pre-processing: All images are first converted to $[0.0, 1.0]$ from $[0, 255]$ and then normalized by subtracting the mean $([0.485, 0.456, 0.406])$ and divided by the standard deviation $(0.229, 0.224, 0.225])$ of each RGB color channel. Data augmentations of randomly cropping and flipping are used for the target image per iteration for image synthesis.

Structure of the discriminator: Figure 2 and Table 1 show the details of our network architecture.

*Work done during internship at Adobe Research

[1]The model pre-trained on ImageNet is provided by PyTorch (https://pytorch.org/docs/stable/torchvision/models.html) and on Places365 by (https://github.com/CSAILVision/places365)

[2]These models are available from https://github.com/MadryLab/robustness_applications and https://github.com/rgeirhos/Stylized-ImageNet

[3]https://pytorch.org/docs/stable/torchvision/models.html

2. More Comparisons

In the main paper, we mainly show the results on ImageNet and Places365. Additionally, we present more results on the texture dataset DTD in Figure 3. The goal of texture synthesis is to generate similar results with the spatial arrangement reorganized. The method of Gatys *et al*. [3] tends to break the texture pattern into pieces and fails to preserve the complete structure of the texture (e.g. "bubbly"). On the other hand, the SinGAN [8] only performs minor editing on the target image without generating significantly new spatial arrangements. For example, the four corners of the synthesized image are identical to their corresponding target sample. In contrast, the IMAGINE more effectively synthesizes a *different image of the same texture*, which is preferred by more users as shown in Table 2.

We also show more comparisons on ImageNet and Places365 in Figure 4 and 5.

3. Ablation study

Several ablations are performed to evaluate the contribution of the different components of IMAGINE on image synthesis.

Classification loss: The IMAGINE enforces a class consistency in two ways: the cross-entropy loss Eq. (1) of the paper and the similarity with the target image which is obviously an example of its class. In preliminary experiments, we note that it makes little difference between using the ground truth class y^0 of the target image and the class predicted by the classifier for the target image $\mathbf{x}^0$ as y^* in the cross-entropy loss. This is because most images can be classified correctly, even when this is not the case, images of the predicted and ground truth class share similar features. We want to investigate whether the cross entropy loss is able to truly influence the synthesized image, by specifying a class different from the ground truth y^0 and predicted y^*. Figure 6 shows two examples, where the labels "zebra" and "snake" are used for a horse and grass target image, respectively. On the horse image, black-and-white stripes are generated and semantically fail on the body of the horse. On the grass, a snake outline is generated. Although the quality of the generated result is not good enough to resemble the target class in the sense of the realism, it should be noted that it is always difficult to generate a target class image without any image of this class to optimize. However, these experiments show that the cross entropy loss has an effect on the generated images.

Loss functions: Figure 7 and Table 3 summarize the results of an ablation study for the importance of the various loss components. These results are based on ResNet-50 on ImageNet-A. The *baseline* is a basic model inversion method that only uses the loss Eq. (1) in the paper. Figure 7 shows that this produces very poor results. We next sorely introduce $\mathcal{R}_{pc}$ in Eq. (9) of the paper, i.e. using $\lambda = 0$, which is denoted as *baseline+D*. This substantially improves the image quality but is insufficient to reconstruct meaningful objects. Note the "absorption by the background" problem discussed in the paper for the SinGAN.

Next, we remove the discriminator and introduce the feature distribution matching regularizer of Eq. (5), leading to the loss of Eq. (9) with $\gamma = 0$, which is denoted as *baseline+H*. This results in a dramatic improvement of object realism. It shows that, unlike patch matching, the constraint of similar activations between the target and synthesized images at *all* levels of the network is critical for object coherence. Note that, for the first time, the objects are not scattered into pieces or absorbed by the background.

Finally, we reintroduce the patch discriminator and adversarial loss, leading to the full IMAGINE algorithm. This further improves the quality of the reconstructed objects, in particular in terms of fine details. Table 3 confirms the conclusions of the qualitative analysis of Figure 7. The *baseline+H* drastically increases (decreases) IS (FID) and the addition of discriminator and adversarial loss further improves the quality of the synthesized images, but by a much smaller amount. Overall, these results show that matching distributions at the different semantic levels of the network representation is critically important to achieve object consistency. Note that *baseline+H* is different from the image-guided version of DeepInverion described in the paper, because the latter keeps the Eq. (3) that is the key speciality of DeepInverion. We observe that matching to the training distribution hurts the quality to some extent.

Choice of the classifier: IMAGINE is a generic method, which can used with any pre-trained classifier available in the literature. To determine the impact of the pre-trained classifier, we perform an ablation in terms of two aspects: the pre-training scheme and the classifier architecture. We start by comparing implementations of IMAGINE with three classifiers: AlexNet, VGG, and ResNet. Figure 8 and table 4 summarize the results of these experiments. AlexNet is significantly weaker than the other two networks which synthesize images of similar quality, with a slight advantage for the ResNet. The performance gap can be explained by the fact that AlextNet is a shallower network and thus less able to capture the diversity of semantic features extracted by deeper classifiers. This again emphasizes the importance of matching distributions across multiple semantic levels.

We next choose the ResNet and investigate the impact of two types of robust training. The SIN ResNet [4] is trained on a stylized version ImageNet. This aims to combat the bias towards texture representations of standard CNN training, favoring the learning of a stronger representation of shape. The robust ResNet [10] uses adversarial training

to learn more class-discriminative features. Although visually it is hard to identify the best of three ResNet training schemes, the quantitative comparison indicates that robust models can generate higher quality and more diverse results. However, the differences are not large. Overall, these results suggest that IMAGINE is quite robust to the classifier architecture and its training scheme. As long as the classifier is not too shallow, other variables are less likely to greatly affect the quality of synthesized images.

4. Limitations

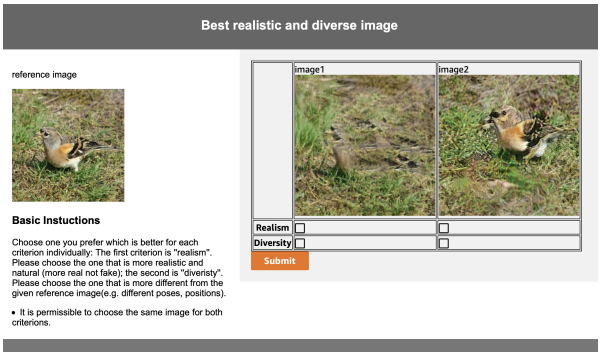
Though we have shown that our approach outperforms existing GAN-based and inversion-based methods across different image domains, honestly 'still' might be better, there are a few limitations.

"Ghost" issue: One limitation of IMAGINE is that sometimes some ghosts are generated, e.g. "fox squirrel" in Figure 4 and "jellyfish" in Figure 6 in the main paper. We visualized the attribution map of the results given in Figure 9 and found that the classifier treats those ghosts as background and so ignores them. We argue that this can be avoided by using a more robust foreground-sensitive classifier and can benefit from future more advanced classifier.

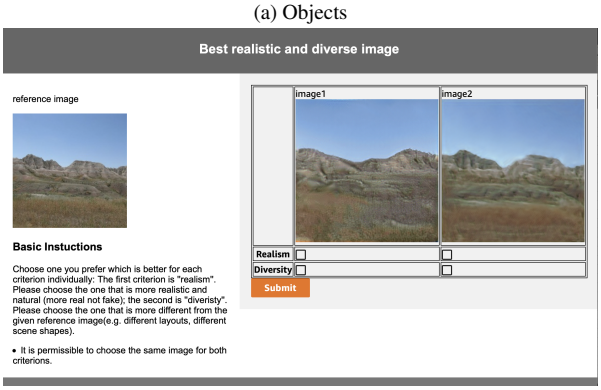
Computational cost: Here we discuss the time cost based on a NVIDIA TITAN Xp and image resolution of 224×224. SinGAN and DGP train a model one time per target image and generate results by feedforward passing, which is faster than the optimization-based IMAGINE (about 10 minutes for one optimization to generate a batch of high-quality synthesized results, per target image).

However, for SinGAN, training a pyramid GAN of 9 scales of images takes nearly 2 hours. Hence, our approach (IMAGINE) is a faster option if we have fewer images to generate. Also, IMAGINE can generate multiple images in parallel. For example, IMAGINE supports maximum batch size of 16 on a single NVIDIA TITAN Xp. So 16 images can be synthesized simultaneously in 10 minutes by one optimization. What is the worse for SinGAN is that the time and memory increase exponentially with the image size because of its multi-scale pyramid structure. While DGP takes only 3 minutes to finetune the generator, it still requires to learn individual generator (i.e., one BigGAN) for each target example, which is super heavy for disk storage. This is also true for SinGAN, training individual generators for individual target images.

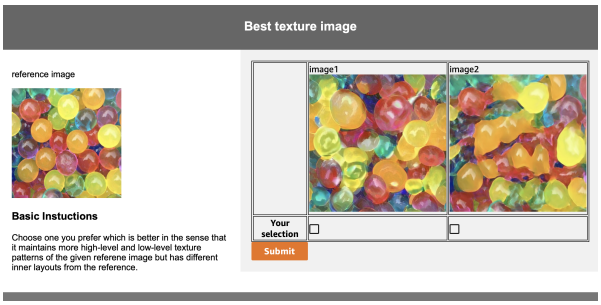
For future work, we plan to evacuate more unknown knowledge inside the classifier that can benefit the image synthesis which in turn might also reward improvements on classification performance.



(a) Objects



(b) Scene



(c) Texture

Figure 1: Interface

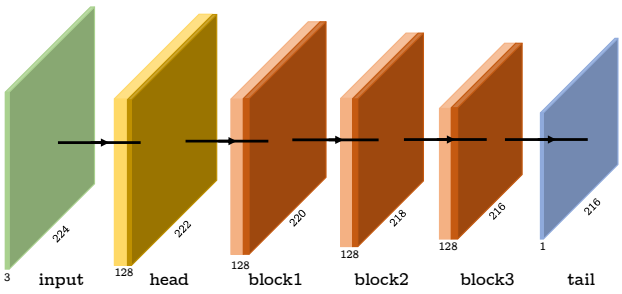


Figure 2: Architecture details of the discriminator exemplified by an input image with resolution of 224×224.

References

[1] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of*

Table 1: Architecture details of the discriminator

	layer	details
head	Conv2d	input channel=3, output channel=128, kernel size = (3,3), stride=(1,1)
	BatchNorm2d	channel number=128, eps=1e-5, momentum=0.1, affine=True, track running stats=True
	LeakyReLu	negative slope=0.2, inplace=True
block1	Conv2d	input channel=128, output channel=128, kernel size = (3,3), stride=(1,1)
	BatchNorm2d	channel number=128, eps=1e-5, momentum=0.1, affine=True, track running stats=True
	LeakyReLu	negative slope=0.2, inplace=True
block2	Conv2d	input channel=128, output channel=128, kernel size = (3,3), stride=(1,1)
	BatchNorm2d	channel number=128, eps=1e-5, momentum=0.1, affine=True, track running stats=True
	LeakyReLu	negative slope=0.2, inplace=True
block3	Conv2d	input channel=128, output channel=128, kernel size = (3,3), stride=(1,1)
	BatchNorm2d	channel number=128, eps=1e-5, momentum=0.1, affine=True, track running stats=True
	LeakyReLu	negative slope=0.2, inplace=True
tail	Conv2d	input channel=128, output channel=1, kernel size = (1,1), stride=(1,1)

Table 2: Comparison of user preference (%), $\text{mean}_{\text{std, confidence interval}}$ (conf. interval at 95% conf. level).

	preference (%)
Gatys/Ours	$14.0/\mathbf{86.0}_{7.0,4.8}$
SinGAN/Ours	$10.0/\mathbf{90.0}_{3.7,2.7}$

Table 3: Comparison of different components of the loss function (mean(std))

method	IS↑	FID↓
baseline	6.0(0.2)	175.8(4.3)
baseline+D	8.2(0.5)	156.7(1.6)
baseline+H	99.1(5.4)	55.4(2.0)
IMAGINE	117.1(6.2)	38.3(1.1)

Table 4: Comparison of different models and architectures (mean(std))

method	IS↑	FID↓
AlexNet	7.0(0.2)	178.1(7.3)
VGG	64.3(4.2)	81.6(2.1)
SIN ResNet	102.1(2.4)	39.2(2.4)
Robust ResNet	122.2(7.0)	36.1(1.7)
ResNet	117.1(6.2)	38.3(1.1)

the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2014.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[3] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in neural information processing systems*, pages 262–270, 2015.

[4] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *ICLR*, 2019.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[8] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Computer Vision (ICCV), IEEE International Conference on*, 2019.

[9] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[10] Shibani Santurkar, Andrew Ilyas, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Image synthesis with a single (robust) classifier. In *Advances in Neural Information Processing Systems*, pages 1262–1273, 2019.

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[12] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[13] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

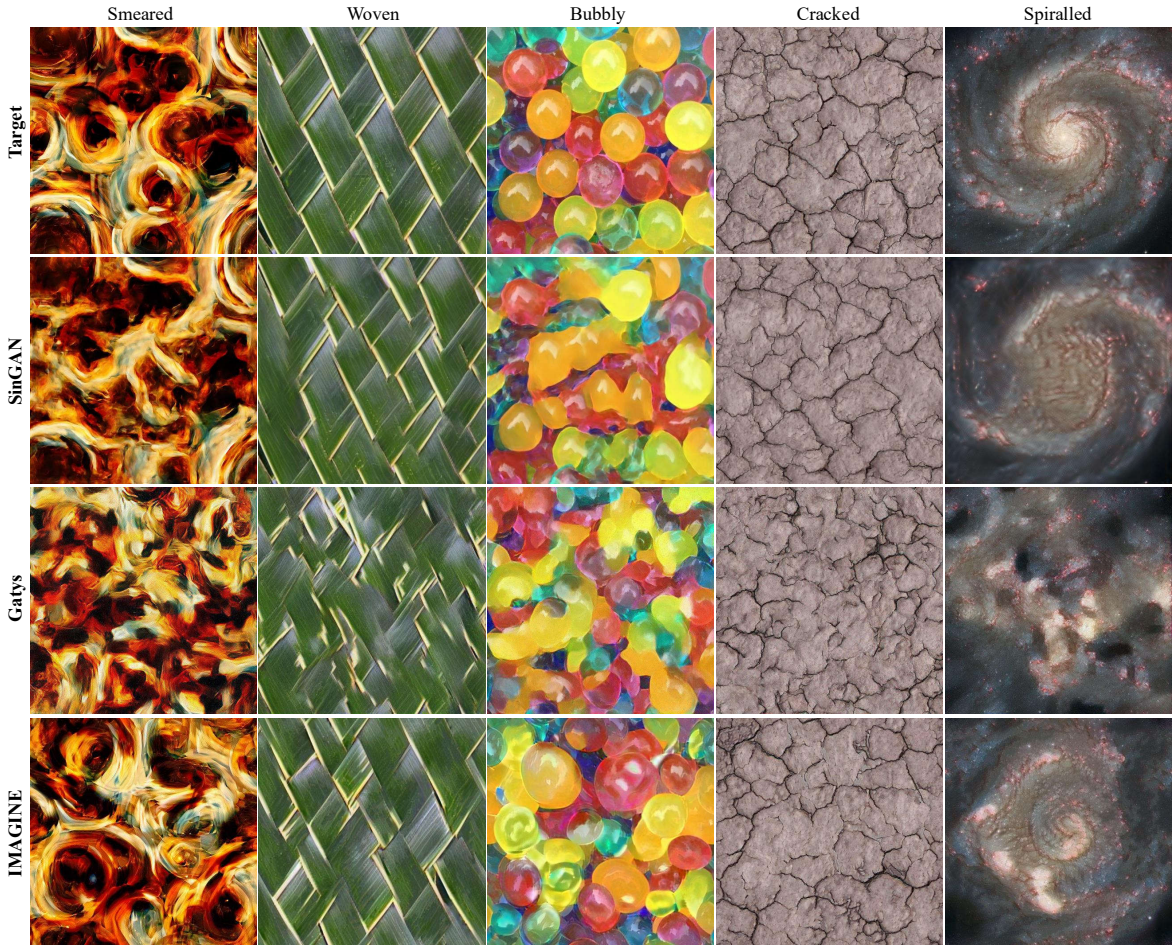


Figure 3: Texture synthesis comparison of different methods.

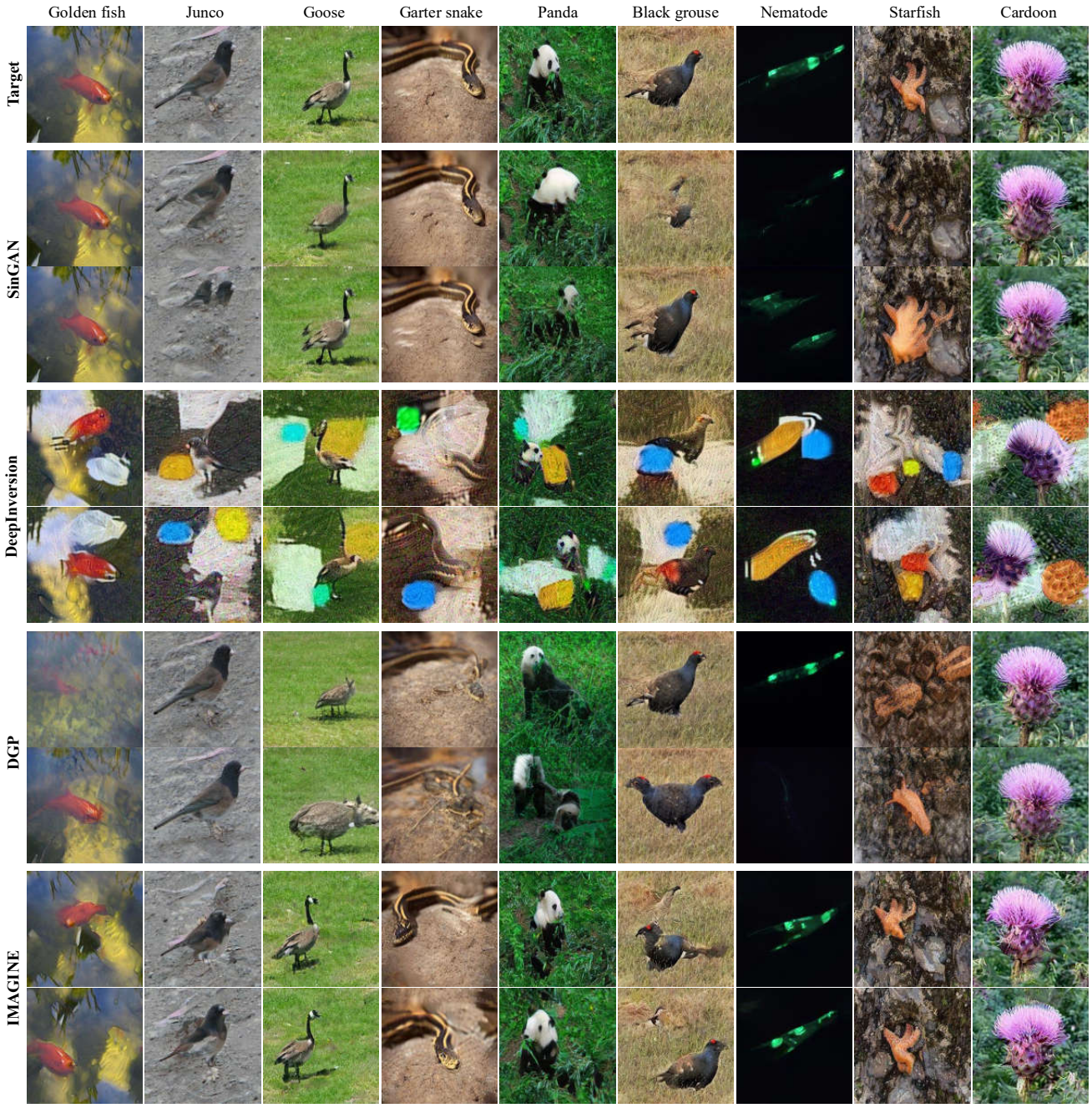


Figure 4: Object generation comparison of different methods, where two randomly results are shown for each method.

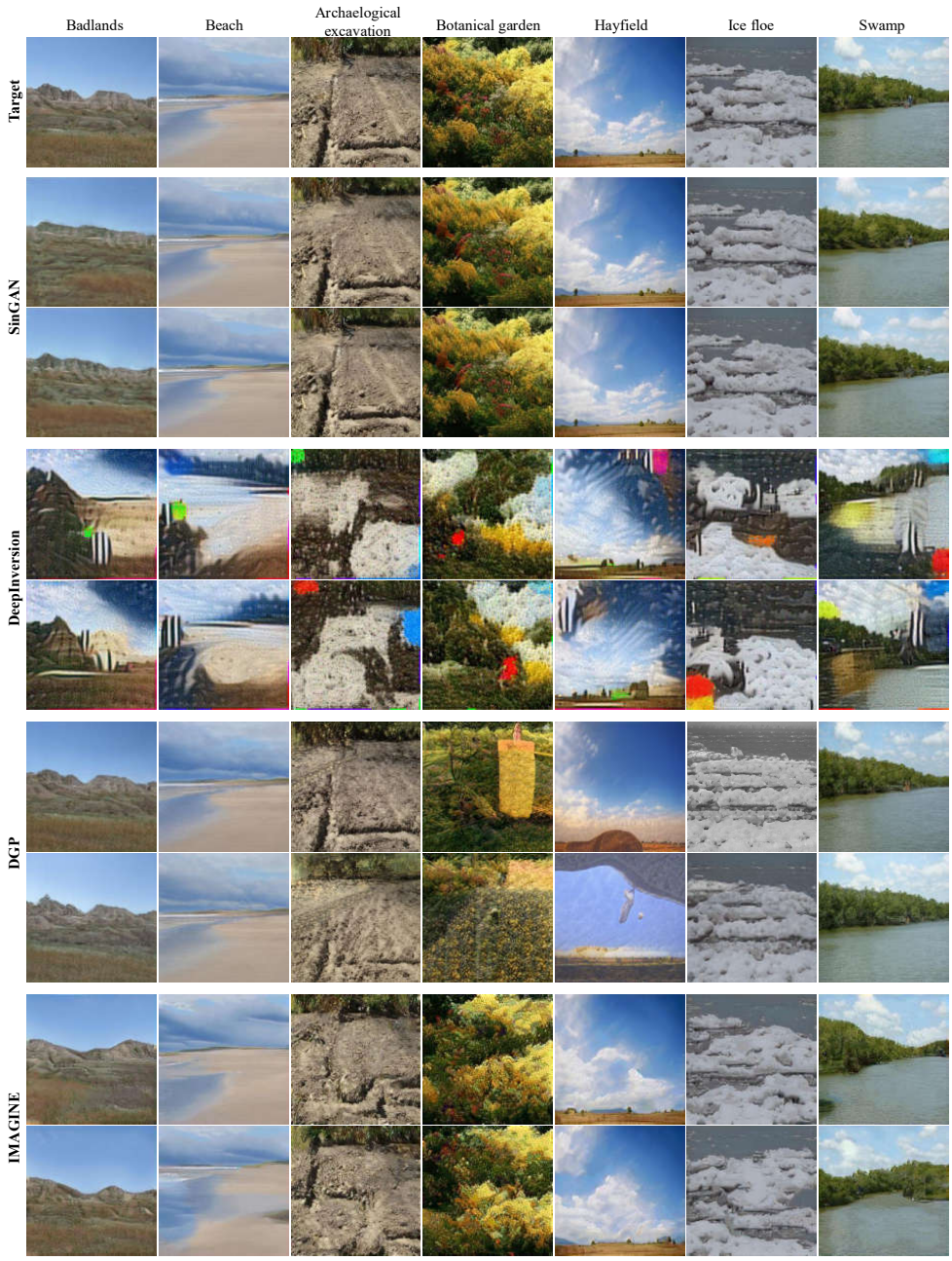


Figure 5: Scene generation comparison of different methods, where two randomly results are shown for each method.

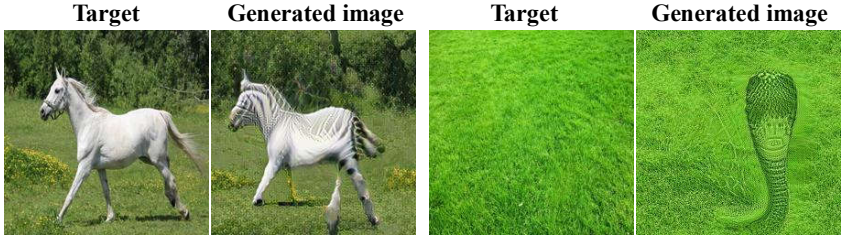


Figure 6: Ablation study on classification loss of IMAGINE. Conflict labels are assigned between the classification loss and target examples.

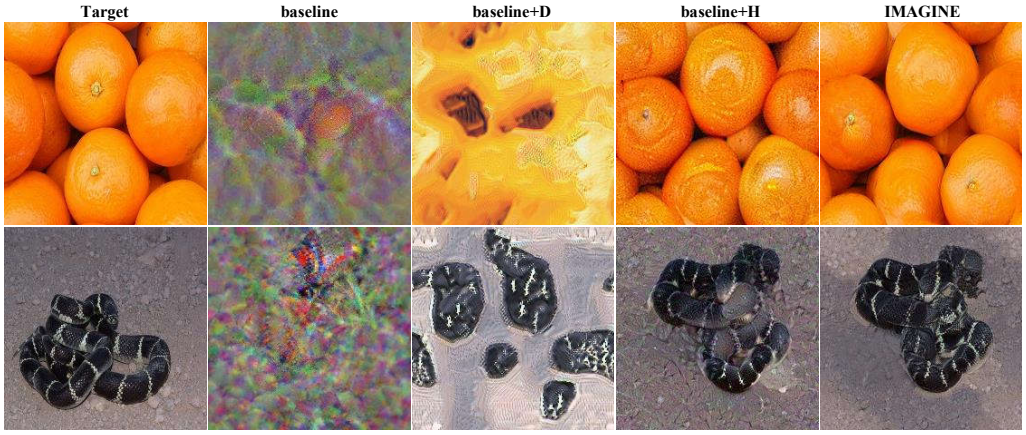


Figure 7: Ablation study on different components of IMAGINE.

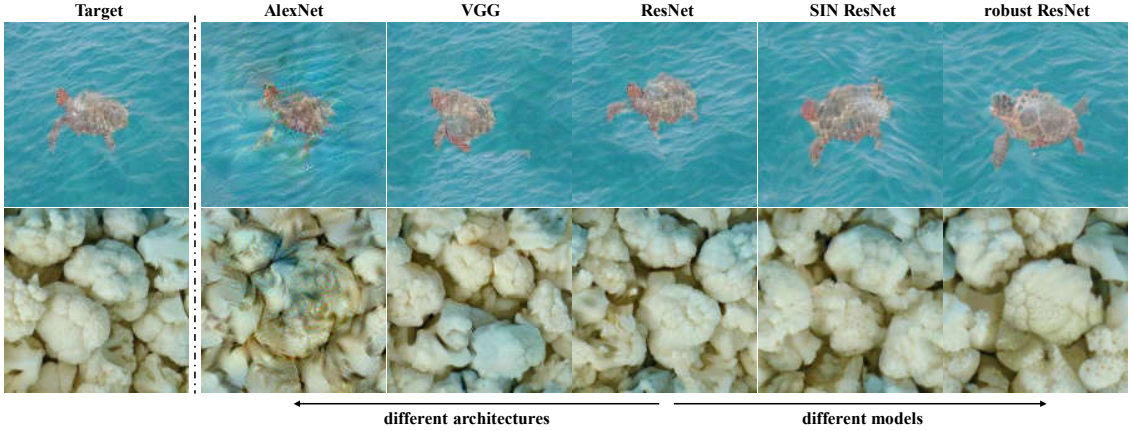


Figure 8: Ablation study on different models and architectures of IMAGINE.

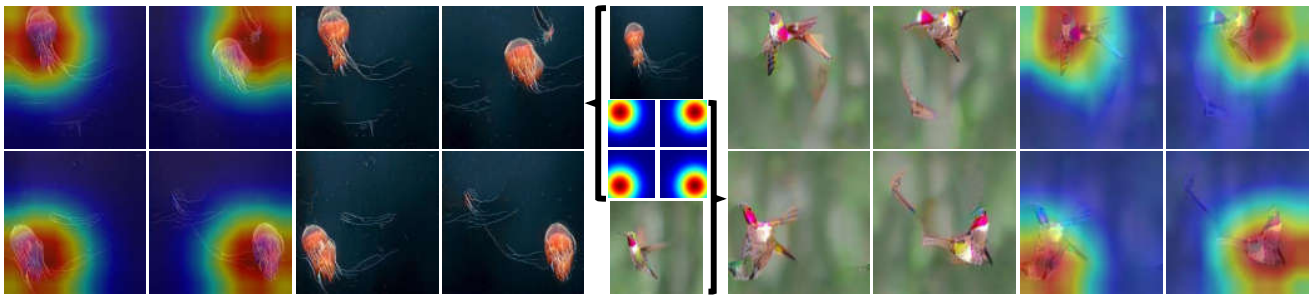


Figure 9: Object position control. Four different target positions (center) are used to supervise the position of objects, jellyfish (top-center) or hummingbird (bottom-center) on the generated images. The corresponding attribution maps are associated at two sides.