DefectFill: Realistic Defect Generation with Inpainting Diffusion Model for Visual Inspection

Supplementary Material

A. Training Details

We use a batch size of 4 for training. The learning rate is set to 2×10^{-4} for the UNet [7] and 4×10^{-5} for the text encoder. Training is conducted over 2000 steps, with the first 100 steps dedicated to warmup, during which the learning rate linearly increases from 0 to its specified value. Throughout the training, images I and masks M are randomly resized together by a factor between 1.0 and 1.125× and then cropped back to their original size. Random masks are generated using 30 boxes with side lengths randomly chosen between 3% and 25% of the image size. We fine-tune only the projection matrices of the text encoder and UNet using LoRA [4] with a rank of 8. The dropout rate is set to 0.1, and the LoRA scaling factor is set to 16. For the $[V^*]$ token, we use the word *"sks"*. For the DefectFill loss, we assign weights of 0.5, 0.2, and 0.05 to the defect loss, object loss, and attention loss, respectively. The adjusted mask M' used in the object loss calculation has α value set to 0.3.

B. Additional Qualitative Results

B.1. MVTec AD Dataset

We provide defect generation samples for all object and defect categories in the MVTec AD [1] dataset. As illustrated in Figs. S4 to S18, our method consistently generates realistic and naturally filled defects across all cases. The first row (blue box) displays the real defect images, while the second row (green box) contains the defect-free images used for defect generation. The third row presents the generated defects using the masks shown in the bottom-right corner, and the fourth row (red box) provides a zoomed-in view of the generated defects.

B.2. VisA Dataset

We further apply our method to another anomaly detection dataset, the Visual Anomaly (VisA) [9] dataset. Following a similar approach to its application on MVTec AD dataset, we train the model using pairs of anomalous images and their corresponding masks (limited to the first 10 pairs per object) and generate defects on defect-free images using unseen masks. As shown in Fig. S19, our method successfully generates realistic defects across all object categories. This highlights the robustness of our method in generalizing to a variety of real-world defects.

Objects	Ours w/o LFS		Ours	
	KID↓	IC-LPIPS↑	KID↓	IC-LPIPS↑
bottle	33.57	**0.12**	**30.99**	**0.12**
capsule	**5.01**	0.17	5.60	**0.18**
carpet	50.39	0.21	**50.37**	**0.22**
hazelnut	1.86	**0.31**	**1.13**	**0.31**
leather	83.06	0.29	**74.66**	**0.30**
pill	16.22	0.22	**8.76**	**0.23**
tile	49.59	**0.44**	**45.14**	**0.44**
toothbrush	**2.87**	**0.15**	3.19	**0.15**
wood	7.05	**0.35**	**4.72**	**0.35**
zipper	35.23	**0.21**	**34.91**	0.20

Table S1. **Generation Comparison with Low-Fidelity Selection.** The application of LFS demonstrates improvements in both quality (KID) and diversity (IC-LPIPS). The values represent averages calculated for each defect category.

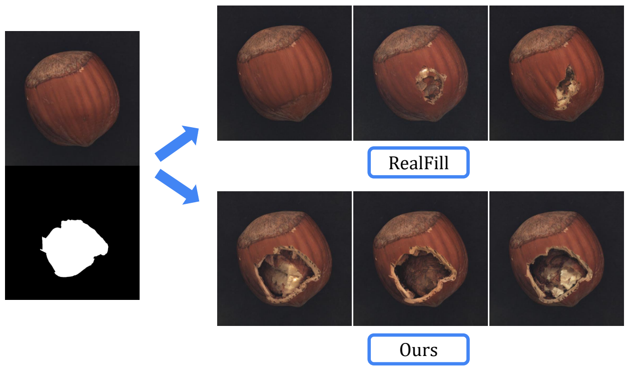

Figure S1. **Comparison to RealFill.** This figure shows a comparison of defect generation quality with another inpainting-based concept learning method, RealFill [8]. It fails to generate proper defects, either reconstructing the original region or producing unrealistic defects that are misaligned with the mask (upper images). In contrast, DefectFill (ours) generates realistic and diverse defects that align accurately with the mask (lower images).

C. Additional Quantitative Results

C.1. Low Fidelity Selection

Tab. S1 compares the quality (KID [2]) and diversity (IC-LPIPS [6]) of generated defect images with and without applying Low-Fidelity Selection (LFS). For diversity, applying LFS achieves the best performance across all objects except for the zipper. In terms of quality, applying LFS im-

Objects	DFMGAN†			AnoDiff*			AnoDiff‡			Ours		
	AUROC↑	AP↑	F_1-max↑	AUROC↑	AP↑	F_1-max↑	AUROC↑	AP↑	F_1-max↑	AUROC↑	AP↑	F_1 max↑
bottle	0.97	**1.00**	0.98	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
capsule	0.76	0.9	0.87	**1.00**	**1.00**	**0.99**	0.94	0.98	0.93	0.98	**1.00**	0.97
carpet	0.81	0.92	0.82	0.97	0.99	0.94	0.89	0.95	0.88	**1.00**	**1.00**	**1.00**
hazelnut	**1.00**	**1.00**	0.99	**1.00**	**1.00**	0.99	**1.00**	**1.00**	0.98	**1.00**	**1.00**	**1.00**
leather	0.94	0.97	0.92	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
pill	0.92	0.97	0.92	**0.98**	**1.00**	**0.97**	0.97	0.99	0.95	0.97	0.99	0.95
tile	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
toothbrush	0.97	0.98	0.93	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.98
wood	0.89	0.94	0.87	0.98	0.99	0.99	0.98	0.99	0.98	**1.00**	**1.00**	**1.00**
zipper	0.99	**1.00**	**0.99**	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**0.99**	**1.00**	**1.00**	**0.99**

Table S2. **Image-Level Detection Comparison.** The table presents AUROC, AP, and F_1-max scores for image-level anomaly detection evaluation using a UNet trained on generated defect images. Our method achieves the highest performance across most metrics and objects. The labels are defined in Tab. 2.

w/o $\mathcal{L}_{obj}$		w/o $\mathcal{L}_{def}$		w/o $\mathcal{L}_{attn}$		Ours	
KID↓	IC-LPIPS↑	KID↓	IC-LPIPS↑	KID↓	IC-LPIPS↑	KID↓	IC-LPIPS↑
26.26	**0.25**	67.34	0.23	26.51	0.24	**25.95**	**0.25**

Table S3. Results when each loss term is removed during training.

proves the KID score for all objects except the capsule and toothbrush.

C.2. Detection

Similar to the evaluation of the anomaly localization task (Tab. 2), we also evaluate our method on the image-level anomaly detection task, comparing it with defect generation baselines (DFMGAN [3], AnomalyDiffusion [5]). Tab. S2 shows our method achieves the best scores in most cases. Even in instances where it does not achieve the best score, it consistently performs well, with all scores exceeding 0.95.

C.3. Loss Ablation

Tab. S3 shows the evaluation results on the MVTec dataset after removing each loss term during training. Notably, removing $\mathcal{L}_{def}$ causes a significant increase in KID. Using all terms achieves the best scores for both KID and IC-LPIPS.

D. Comparison to RealFill

To demonstrate DefectFill's ability to learn defect features and generate realistic defects, we compare it with another inpainting-based concept learning method, RealFill [8]. While RealFill focuses on filling erased regions in a single target image, making it less suitable for defect generation tasks required in visual inspection, this comparison highlights the superior generation quality of DefectFill. As shown in Fig. S1, RealFill (upper images) fails to generate proper defects, often reconstructing the original region or producing unrealistic defects that are misaligned with the mask. In contrast, our method (lower images) generates defects that are both realistic and diverse, while precisely aligning with the mask's shape. This highlights not only the importance of leveraging an inpainting diffusion model but also the crucial role of our defect-specific loss, which is tailored for inpainting diffusion models.

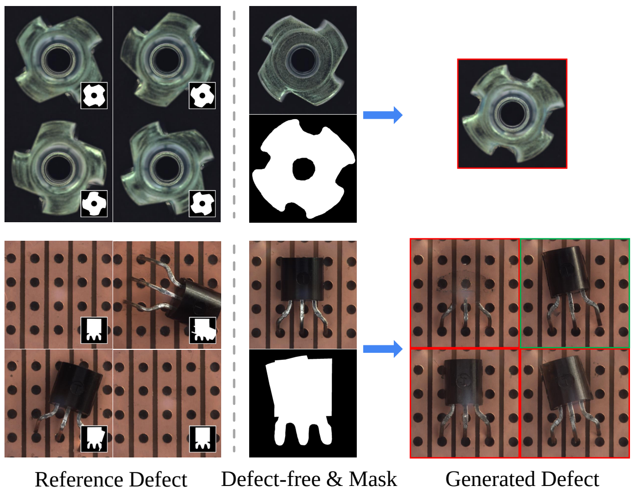

Figure S2. **Failure Cases.** DefectFill struggles with structural defects affecting the entire object. For the metal nut (top), the mask covers the flipped nut itself, so the model learns its appearance rather than its orientation. For the transistor (bottom), inpainting replaces the defect-free object, creating a stochastic mix of defect features, though it often generates proper defects (green box).

E. Failure Cases

As discussed in the conclusion, our method excels at generating local defects but is less effective at handling global structural defects. Fig. S2 illustrates failure cases of structural defects from the MVTec AD dataset. For the metal nut's *flip* defect (upper part of Fig. S2), both the refer-

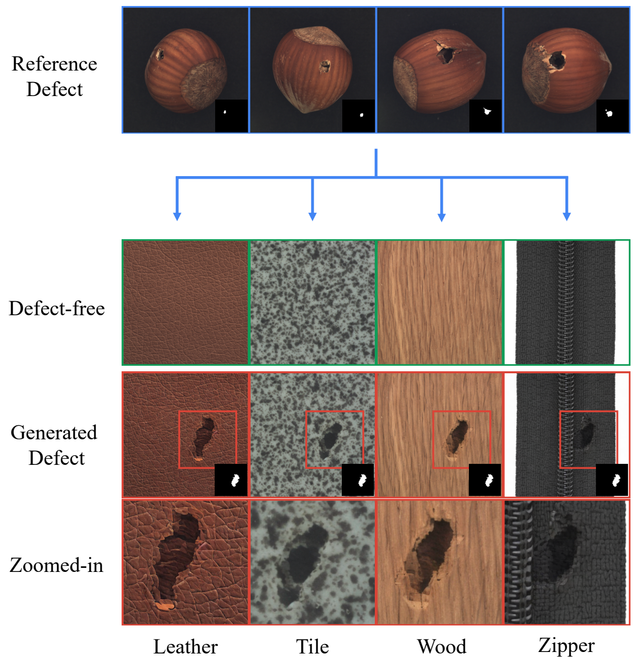

Figure S3. **Transferring defects across different objects.** The figure illustrates the results of generating hole defects in different objects after learning the features of a hole defect from a hazelnut. Defect transfer can occur when the defect features are general and plausible in the context of other objects.

ence defect image and mask represent the entire flipped nut. This causes the model to learn the flipped nut's appearance rather than the direction of the flipped teeth as a defect feature. Consequently, when generating a flipped nut from an unflipped one, the teeth's direction remains unchanged, and the model instead fills the appearance aligning with the mask shape. For the transistor's *misplaced* defect (lower part of Fig. S2), the scenario differs. The mask includes both the original and misaligned positions, enabling the model to learn misalignment features. However, the *misplaced* defect involves not only misaligned cases but also missing ones. In this situation, the inpainting process entirely removes the transistor from the original position and generates a new defect. This results in the loss of semantic information from the defect-free object, causing stochastic appearances of defect features representing both misaligned and missing cases. As shown in Fig. S2, the generated defects manifest as complete transparency, semi-transparent alignment, semi-transparent misalignment (red boxes), or proper misalignment (green box). Addressing these global structural defects is left for future research. Nevertheless, our method demonstrates strong performance in handling most practical cases, where localized defects are the primary focus in real-world scenarios.

F. Transferring Defects across Objects

We observe that if a defect in one object exhibits general features, it can be generated in other objects where such a defect is plausible. As shown in Fig. S3, after learning the hole defect from a hazelnut, our method successfully generates similar defects in various defect-free objects (*e.g*. leather, zipper, wood, and tile).

References

[1] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9592–9600, 2019. 1

[2] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 1

[3] Yuxuan Duan, Yan Hong, Li Niu, and Liqing Zhang. Few-shot defect image generation via defect-aware feature manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 571–578, 2023. 2

[4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 1

[5] Teng Hu, Jiangning Zhang, Ran Yi, Yuzhen Du, Xu Chen, Liang Liu, Yabiao Wang, and Chengjie Wang. Anomalydiffusion: Few-shot anomaly image generation with diffusion model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8526–8534, 2024. 2

[6] Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10743–10752, 2021. 1

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 1

[8] Luming Tang, Nataniel Ruiz, Qinghao Chu, Yuanzhen Li, Aleksander Holynski, David E Jacobs, Bharath Hariharan, Yael Pritch, Neal Wadhwa, Kfir Aberman, et al. Realfill: Reference-driven generation for authentic image completion. *ACM Transactions on Graphics (TOG)*, 43(4):1–12, 2024. 1, 2

[9] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In *European Conference on Computer Vision*, pages 392–408. Springer, 2022. 1

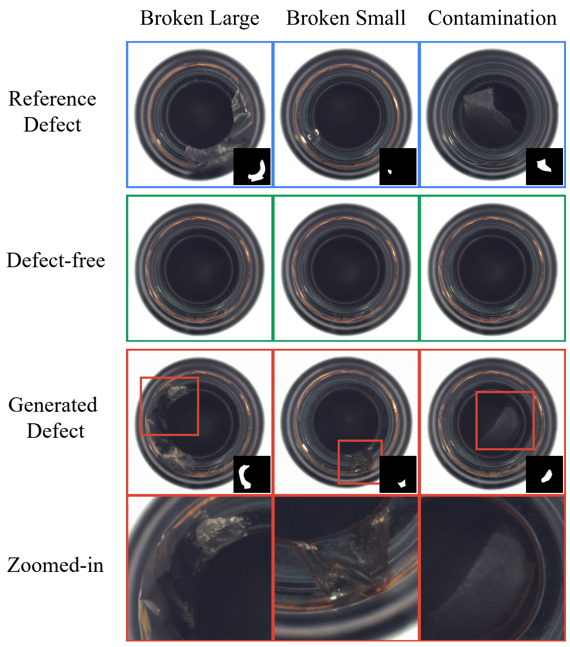

Figure S4. **Defect generation results on MVTec AD dataset (object: bottle).**

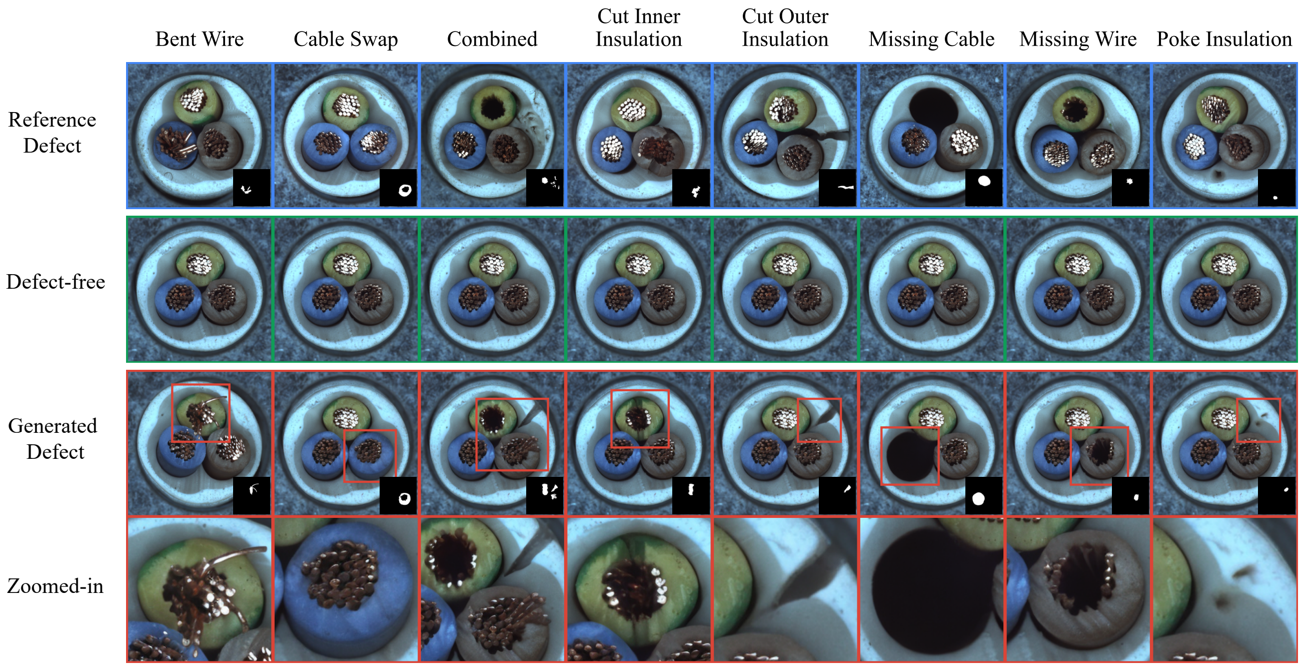

Figure S5. **Defect generation results on MVTec AD dataset (object: cable).**

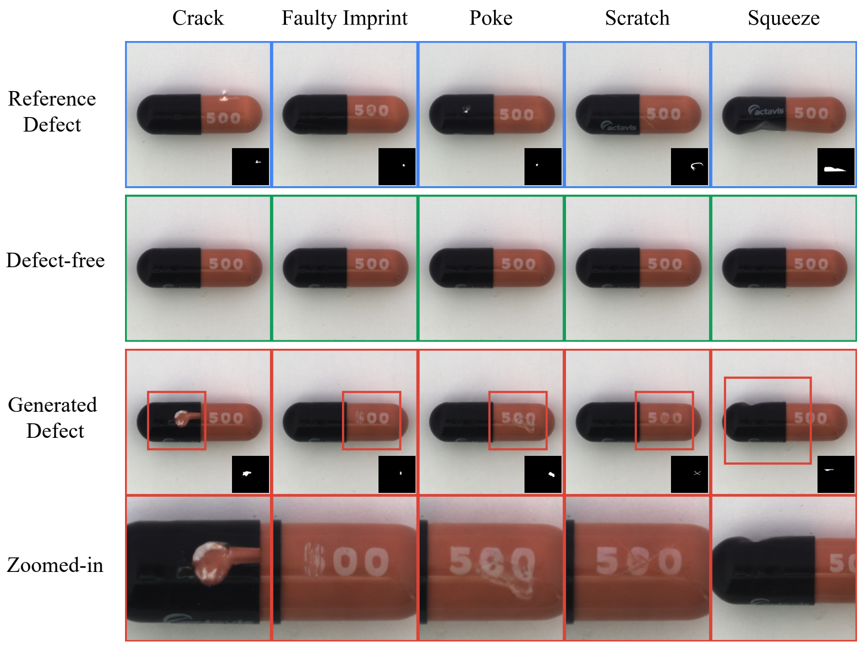

Figure S6. **Defect generation results on MVTec AD dataset (object: capsule).**

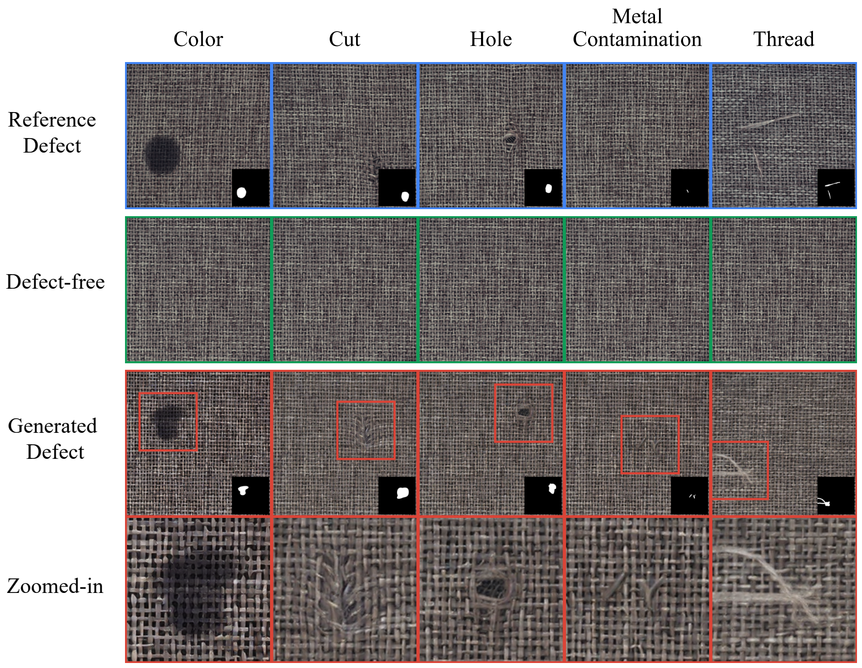

Figure S7. **Defect generation results on MVTec AD dataset (object: carpet).**

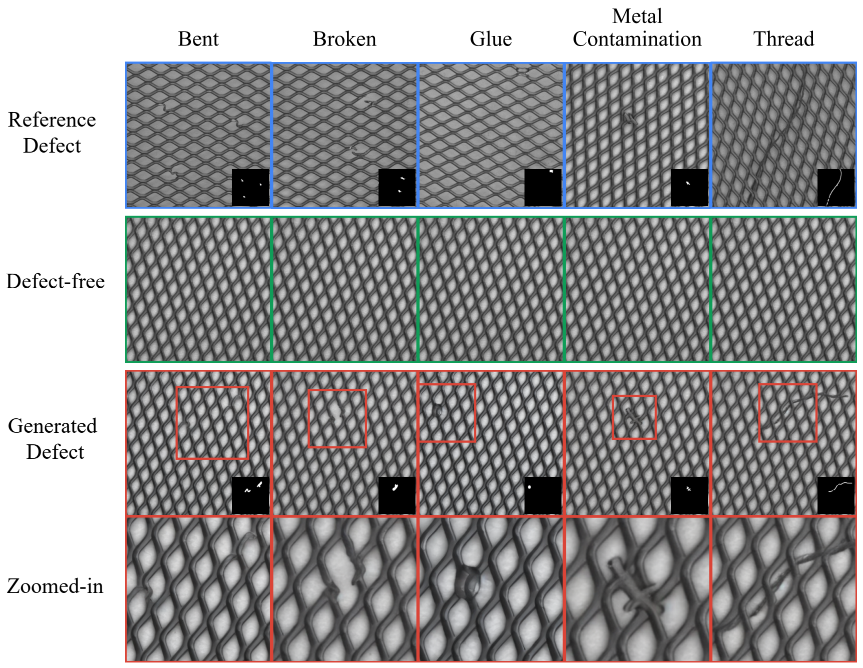

Figure S8. **Defect generation results on MVTec AD dataset (object: grid).**

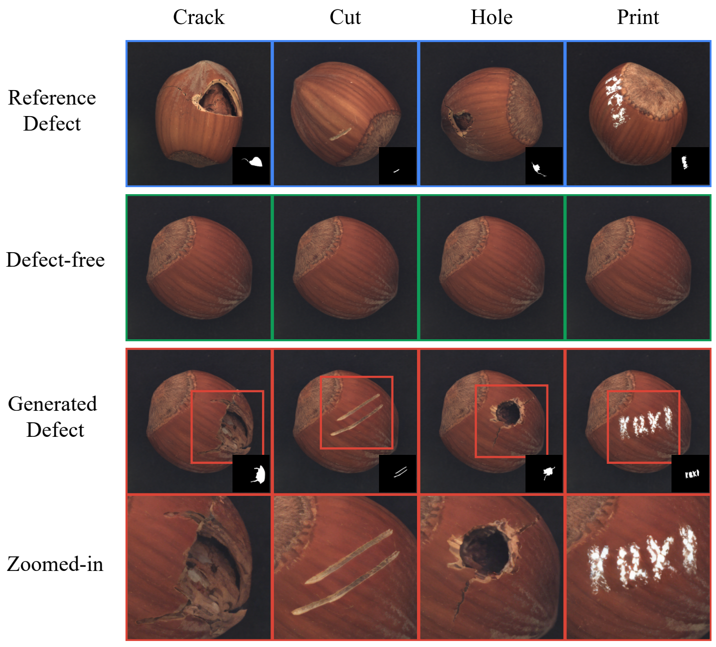

Figure S9. **Defect generation results on MVTec AD dataset (object: hazelnut).**

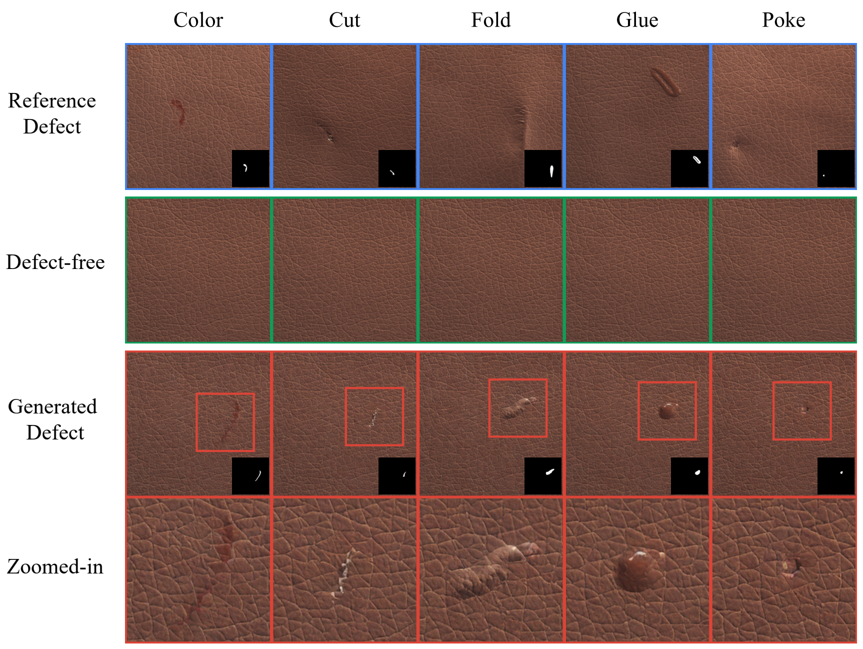

Figure S10. **Defect generation results on MVTec AD dataset (object: leather).**

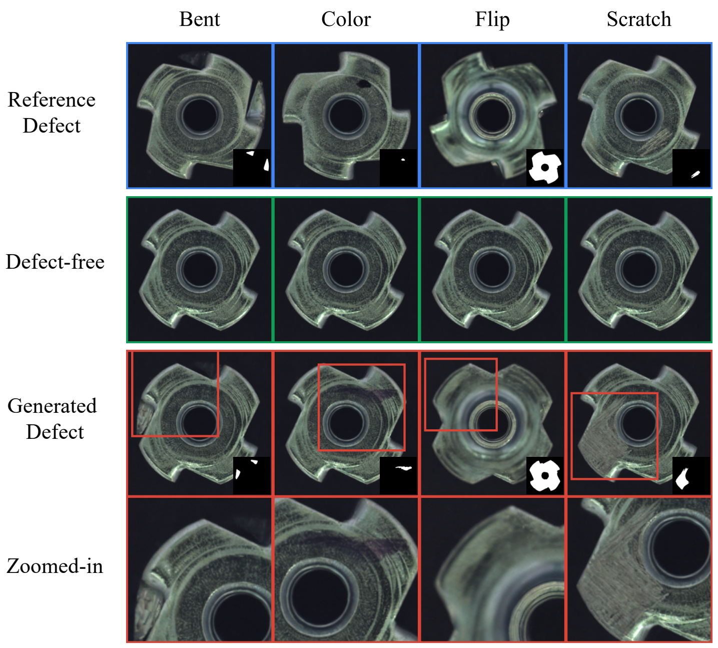

Figure S11. **Defect generation results on MVTec AD dataset (object: metal nut).**

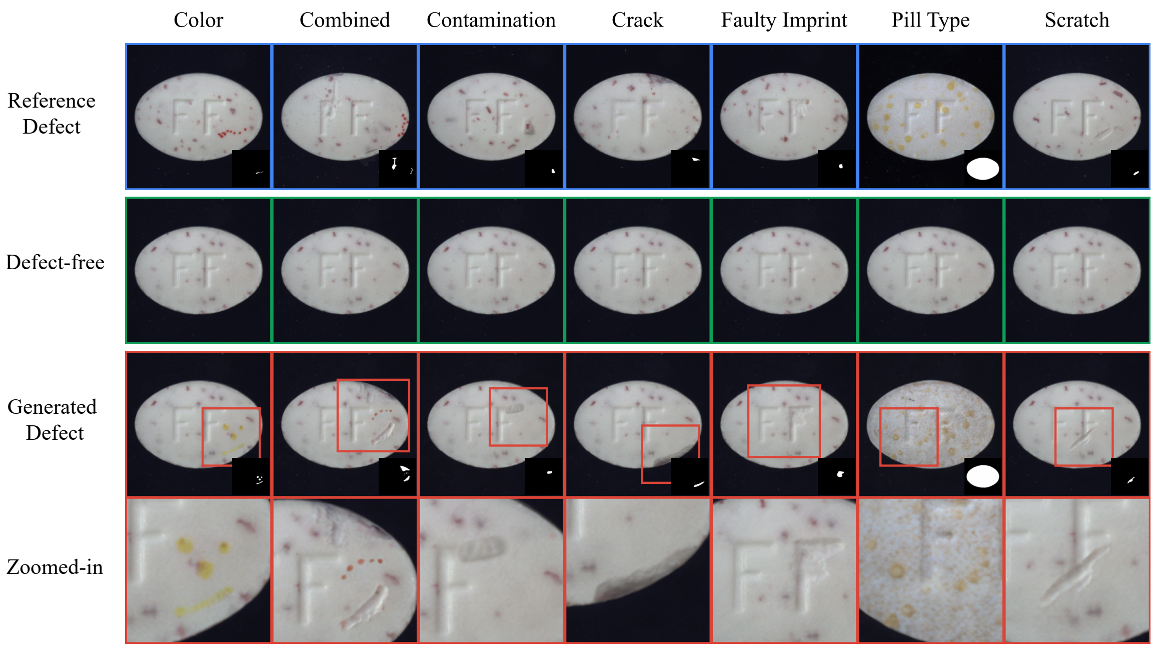

Figure S12. **Defect generation results on MVTec AD dataset (object: pill).**

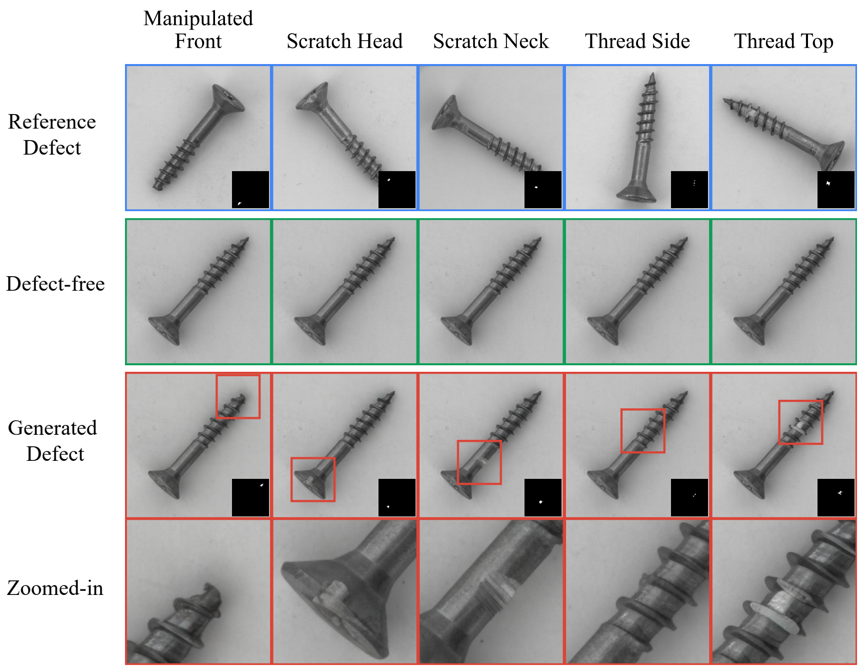

Figure S13. **Defect generation results on MVTec AD dataset (object: screw).**

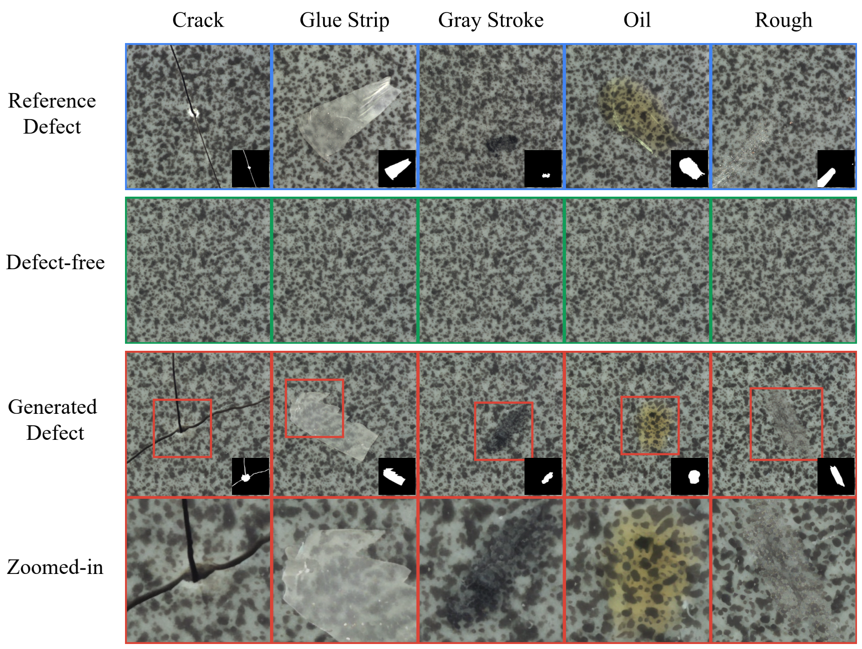

Figure S14. **Defect generation results on MVTec AD dataset (object: tile).**

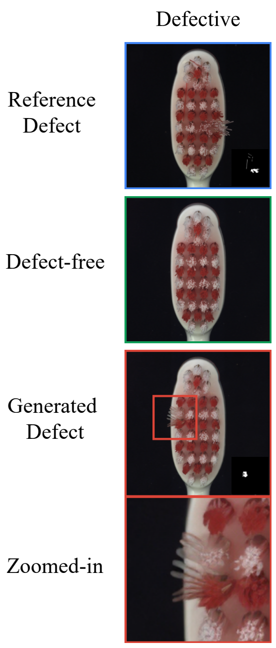

Figure S15. **Defect generation results on MVTec AD dataset (object: toothbrush).**

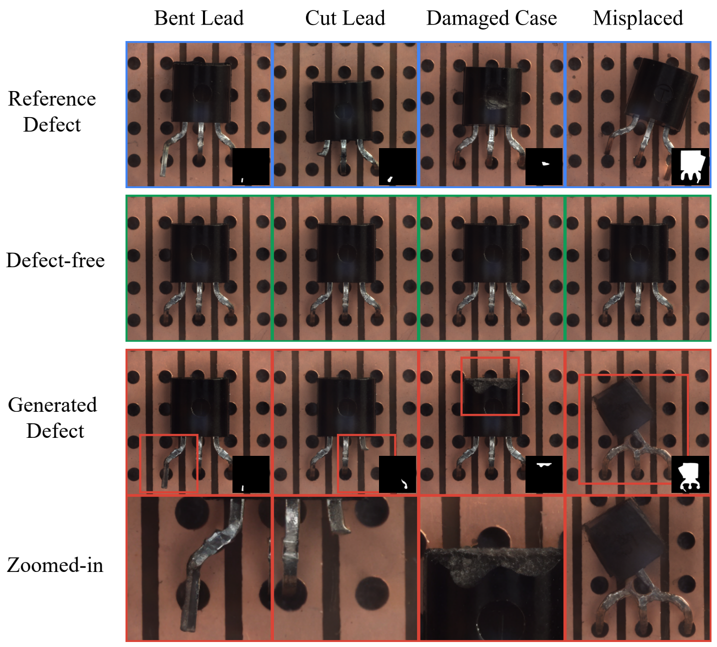

Figure S16. **Defect generation results on MVTec AD dataset (object: transistor).**

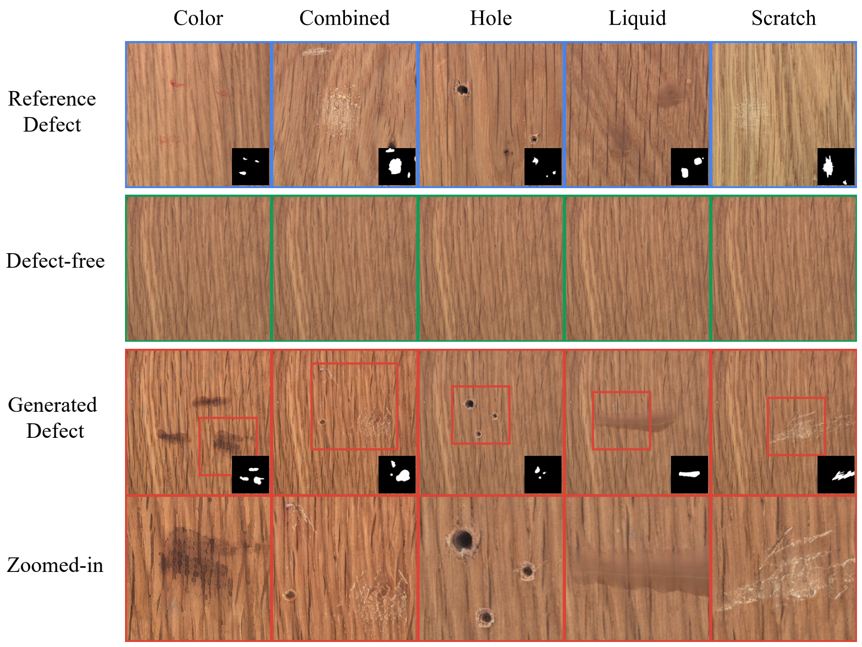

Figure S17. **Defect generation results on MVTec AD dataset (object: wood).**

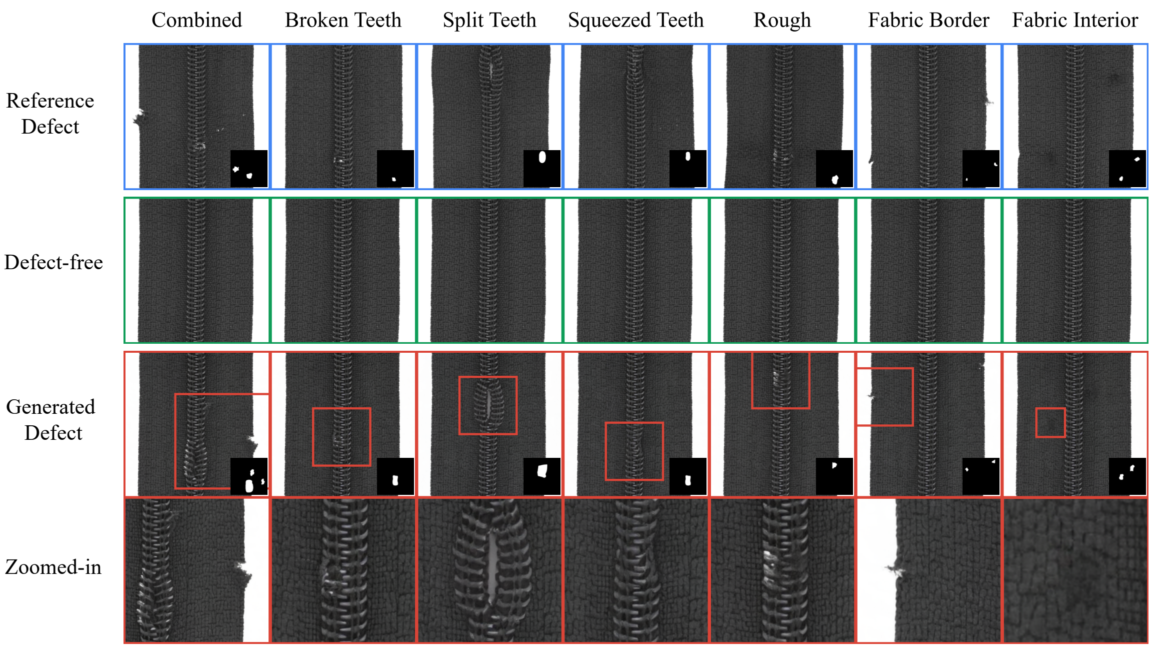

Figure S18. **Defect generation results on MVTec AD dataset (object: zipper).**

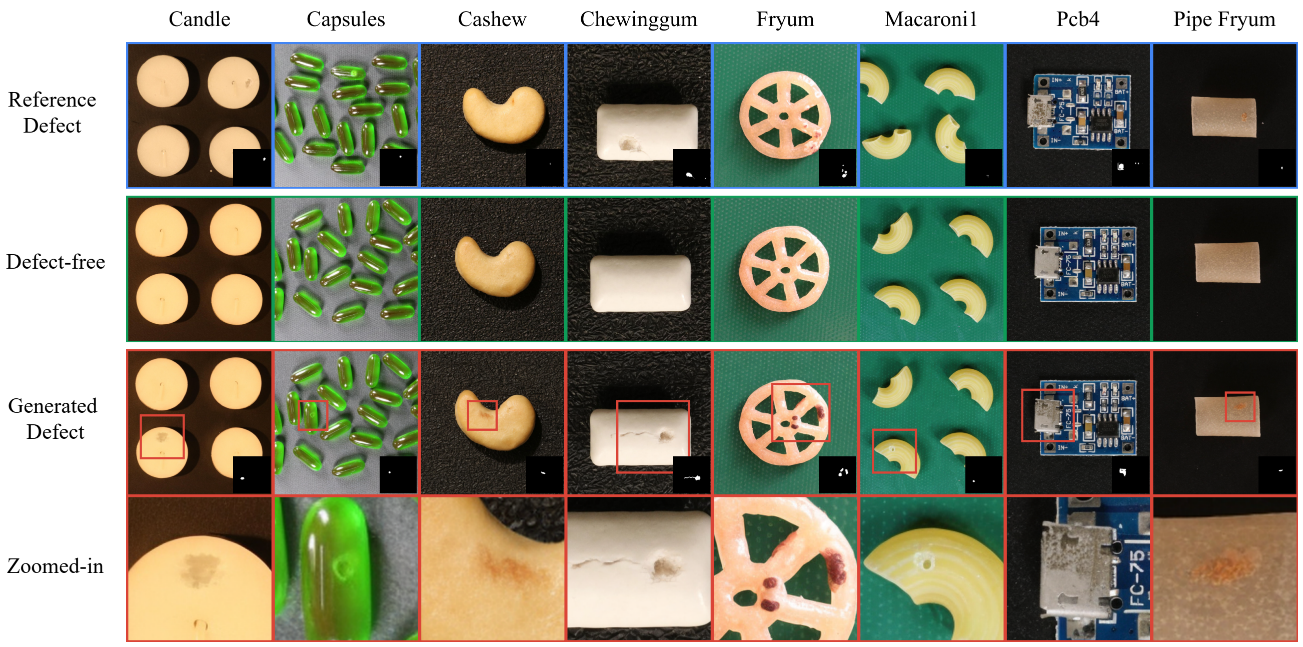

Figure S19. **Defect generation results on VisA dataset.**