



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DOTTORATO DI RICERCA IN
COMPUTER SCIENCE AND ENGINEERING

Ciclo 38

Settore Concorsuale: 09/H1 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

Settore Scientifico Disciplinare: ING-INF/05 - SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI

NEURAL IMAGE SYNTHESIS FOR INDUSTRIAL COMPUTER VISION
APPLICATIONS

Presentata da: Musawar Ali

Coordinatore Dottorato

Paola Salomoni

Supervisore

Luigi Di Stefano

Co-supervisore

Nicola Fioraio

Esame finale anno 2026

Università di Bologna

Abstract

Facoltà di Ingegneria ed Architettura
Dipartimento di Informatica - Scienza e Ingegneria

Dottorato di Ricerca

Neural Image Synthesis for Industrial Computer Vision Applications

by Musawar Ali

Neural image synthesis has advanced significantly in recent years, driven by powerful generative models such as Generative Adversarial Networks (GANs), Denoising Diffusion Probabilistic Models (DDPMs), Latent Diffusion Models (LDMs), Neural Radiance Fields (NeRFs), and Gaussian Splatting. Among these, LDMs have delivered strong capabilities in generating high-fidelity, realistic images across diverse domains with applications ranging from artistic creation and data augmentation to the generation of rare events. These methods are being adopted in industrial settings, where synthetic data helps mitigate challenges associated with data scarcity. For example, to generate synthetic industrial defects for training anomaly inspection models, where collecting real defective samples is often impractical due to cost. Neural image synthesis is a promising tool to generate synthetic anomaly images for industrial quality control. Additionally, NeRFs and Gaussian Splatting have shown significant progress for novel view synthesis, where accurate camera poses are necessary for rendering realistic viewpoints. This capability aligns with industrial requirements, as many applications demand reliable methods for synthesizing handheld objects from multiple viewpoints. These advances highlight the growing role of neural image synthesis in both research and industrial practices. Following these lines of research, this thesis first demonstrates how synthetic images can improve downstream anomaly inspection through stable diffusion inpainting using AnomalyControl and VarianceGenerator. Then a benchmark for novel view synthesis of handheld objects is developed and presented. Through experiments with adapted baselines, we show the need

for more accurate pose estimation techniques in scenarios involving partial hand occlusions and particularly in setups where only RGB images are available.

Dedicated with respect and gratitude to the late Prof. Dr. Sajid Anwar Khan who supervised my Master's Thesis and inspired me to pursue my Ph.D.

Acknowledgements

First and foremost, I am deeply grateful to my supervisors, Professor Luigi Di Stefano and Professor Samuele Salti, and to Nicola Fioraio, for providing such a stimulating and supportive research environment. Their guidance and encouragement were essential to the completion of this thesis.

I am also sincerely grateful to EyeCan.Ai for generously funding my PhD studies, without which this research would not have been possible

I would like to thank my friend Paul Parks for his constant moral support, the many moments of encouragement, and the unforgettable Italian pasta fresca (zucca) and prosecco on weekends. I am especially grateful for allowing me to collect the dataset in his apartment.

I also thank Manuel Carranza-Garcia for graciously hosting me in Spain during my six-month stay abroad.

This work would not have been possible without the unwavering support of my family. I am profoundly grateful to my mother and to Mussadiq Ali for their encouragement, and belief in me throughout the most challenging moments of my PhD.

I extend my gratitude to all members of CV Lab; Andrea Amaduzzi, Iacopo Curti, Matteo Poggi, Alex Costanzino, Pier Luigi, and Luca Barttolomei for their continuous help and support. I also acknowledge Rubin Carkaxhia for contributing experimental results for an extension of our work.

Finally, I thank my friends Syed Muhammad Farrukh Shahid, Zeshan Khan, Shaharbano Asif, Shoaib Rauf, Muhammad Waqas Sheikh for their companionship and support during my transition to Italy.

This research is funded by the European Union - Next Generation EU, Mission 4, Component 2, Investment 3.3 (Ministerial Decree 117/2023) CUP J33C22001400009.



Contents

Abstract	i
Dedication	iii
Acknowledgments	iv
1 Introduction	1
1.1 Neural Image Synthesis	1
1.2 Denoising Diffusion Probabilistic Models	2
1.3 Novel View Synthesis	3
1.3.1 Neural Radiance Fields	3
1.3.2 Gaussian Splatting	3
1.4 Anomaly Inspection	4
1.5 Content of Thesis	5
1.5.1 Image Generation for Industrial Anomaly Inspection	5
1.5.2 Novel View Synthesis	5
I Image Synthesis for Industrial Anomaly Inspection	6
2 Related Work and Background	7
2.1 Related Works	7
2.1.1 Anomaly Inspection	7
2.1.2 Generative Models	8
2.1.3 Anomaly Generation	9
2.2 Background	10
2.2.1 Stable Diffusion Inpainting	10
2.2.2 ControlNet	11
3 Few-Shot Anomaly Generation	13
3.1 Method	14
3.1.1 Training	16

3.1.2	Image Generation	16
3.1.3	Prompt Engineering	17
3.1.4	Image Filtering	18
3.1.5	Downstream tasks	19
3.2	Experiments	19
3.2.1	Dataset	19
3.2.2	Implementation details	20
3.2.3	Baseline Test Protocol	20
3.2.4	Proposed Test Protocol	20
3.2.5	Anomaly Classification	21
3.2.6	Anomaly Detection and Localization	23
3.2.7	Anomaly Generation Quality	24
3.3	Ablation Study	25
3.4	Conclusion	28
4	Learning to Inpaint Nominal Images	29
4.1	Method	30
4.1.1	Model Training	30
4.1.2	Image Generation	32
4.2	Experiments	33
4.2.1	Dataset	33
4.2.2	Implementation Details	33
4.2.3	Evaluation Protocol	33
4.2.4	Anomaly Classification	35
4.3	Conclusion	36
II	Novel View Synthesis	37
5	Related Work and Background	38
5.1	Datasets	38
5.2	Related Works	40
5.2.1	Novel view synthesis	40
5.2.2	Single-Hand objects	41
5.2.3	Two-hand 3D pose datasets	41
5.2.4	6-DoF Object Pose Estimation and Tracking	42
5.2.5	Object Reconstruction	42
6	Novel View Synthesis of Handheld Objects	43
6.1	Dataset Creation	46
6.2	Benchmark	47

6.2.1	Dataset, Task, and Metrics	47
6.2.2	Baseline Methods	51
6.3	Benchmark Results	53
6.4	Conclusions	53
7	Concluding Remarks	55
III	Appendices	57
A	FEW SHOT ANOMALY GENERATION	58
A.1	Introduction	58
A.2	Implementation Details	58
A.3	Qualitative Comparison	59
A.3.1	Anomaly Generation Quality	60
A.4	Additional Results on Anomaly Inspection	60
B	Novel View Syhthesis of Handheld Objects	82
B.1	Additional details on the benchmark	82
B.2	Additional quantitative and qualitative results	83
	List of Publications	97
	Bibliography	98

Chapter 1

Introduction

1.1 Neural Image Synthesis

Deep learning has transformed the field of neural image synthesis in recent years. Much of this progress comes from models such as Generative Adversarial Networks (GANs) [1], Denoising Diffusion Probabilistic Models (DDPMs) [2], Latent Diffusion Models (LDMs) [3], and in 3D, methods like Neural Radiance Fields (NeRFs) [4] and 3D Gaussian Splatting [5]. These approaches deliver impressive results when trained on high-quality datasets, enabling tasks ranging from high-resolution image synthesis to 3D scene reconstruction.

LDMs [3] extend the capabilities of DDPMs by encoding images into a lower-dimensional latent space rather than operating directly in pixel space. This reduces both computational cost and memory requirements while maintaining high image quality. LDMs have proven to be highly versatile, serving as the backbone for tasks such as high-resolution image synthesis, inpainting, style transfer, and domain-specific image generation. Recent techniques, including Textual Inversion [6], DreamBooth [7], and ControlNet [8], allow these pre-trained models to be fine-tuned with very few examples, making them highly practical for real-world, industrial applications. For instance, industrial defect detection systems can leverage LDMs to generate synthetic images of anomalies and to improve the performance of downstream anomaly classification, detection and localization.

In the realm of 3D generative modeling, methods such as Neural Radiance Fields (NeRFs) [4] and 3D Gaussian Splatting [5] have enabled high-resolution scene reconstruction and novel view synthesis. NeRFs represent a scene as a continuous volumetric function using multi-layer perceptrons (MLPs), mapping 3D coordinates and viewing directions to color and density values. This representation allows the rendering of photorealistic novel views points from a sparse set of images.

However, training NeRFs can be computationally intensive, and rendering times may be prohibitive for real-time applications.

Gaussian Splatting, by contrast, represents 3D scenes as a set of anisotropic 3D Gaussians, which are efficiently rasterized to produce images from new viewpoints. While this approach may slightly compromise reconstruction accuracy compared to NeRFs, it provides significant speed advantages, making it suitable for interactive applications and real-time visualization. Both techniques have applications in industrial inspection, AR/VR, robotics, and digital twin creation, where accurate 3D modeling is critical for simulation and analysis.

Industrial applications highlight the practical relevance of these generative methods. For instance:

- **Diffusion models** can generate synthetic anomolous images in manufacturing lines, thereby allowing significant improvement in the downstream anomaly detection, localization and classification.
- **NeRFs and Gaussian Splatting** enable the generation of high-fidelity 3D digital twins of industrial assets, which can be analyzed from previously unseen perspectives to assess wear, detect faults, and guide maintenance operations.

1.2 Denoising Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models (DDPMs) [2] represent a popular paradigm of generative models that have recently demonstrated remarkable success in photo-realistic image generation tasks. In the forward process, DDPMs gradually add random Gaussian noise to images, while in the reverse process they reconstruct the original image. The forward noising process can be defined as:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (1.1)$$

where an image x_0 is transformed to a noisy image x_t , with t uniformly sampled from $\{1, \dots, T\}$, ϵ drawn from $\mathcal{N}(0, I)$ and $\bar{\alpha}_t = \prod_{i=0}^t \alpha_i = \prod_{i=0}^t (1 - \beta_i)$, with $\beta_i \in (0, 1)$ representing the noise variance schedule.

At training time, the Markov chain of noisy images is used to learn the reverse process, i.e. to learn a neural network that predicts x_{t-1} given x_t . It can be shown [2] that this is equivalent to predict the noise added to the image at time t , which is usually done by learning a U-Net [9] modeling the noise predictor $\epsilon_\theta(x_t; t)$.

Given the learned noise predictor, during the inference stage $\hat{x}_{t-1}$ is predicted from $\hat{x}_t$ using:

$$\hat{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\hat{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(\hat{x}_t; t) \right) + \sigma_t \epsilon \quad (1.2)$$

where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The process starts at $\hat{x}_T \sim \mathcal{N}(0, I)$ and occurs in an iterative way, progressing from $\hat{x}_T$ to $\hat{x}_{T-1}$ and so forth until reaching $\hat{x}_0$.

1.3 Novel View Synthesis

1.3.1 Neural Radiance Fields

NeRF [4] offers an efficient approach for synthesizing novel views in 3D environments. It models a continuous volumetric scene function $F_\Theta : (\mathbf{x}, \mathbf{d}) \mapsto (\mathbf{c}, \sigma)$, which takes as input a spatial location $\mathbf{x} \in \mathbb{R}^3$ and a viewing direction $\mathbf{d} \in S^2$, and outputs the emitted color $\mathbf{c} \in \mathbb{R}^3$ and volume density $\sigma \in \mathbb{R}^+$. Using this formulation, the color $C(\mathbf{r})$ observed along a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ can be computed via volume rendering as:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \sigma(\mathbf{r}(t)) \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt,$$

where t_n and t_f are the near and far bounds of the ray, and the accumulated transmittance $T(t)$ is defined as:

$$T(t) = \exp \left(- \int_{t_n}^t \sigma(\mathbf{r}(s)) ds \right).$$

NeRF parameters are optimised by minimising the difference between the ground-truth pixel color $C(\mathbf{r})$ and the rendered color $\hat{C}(\mathbf{r})$ from the model, minimizing the loss:

$$\mathcal{L}_I = \|C(\mathbf{r}) - \hat{C}(\mathbf{r})\|_2^2.$$

1.3.2 Gaussian Splatting

3D-Gaussian Splatting (3D-GS) [5] represents a scene using a set of anisotropic 3D Gaussians that preserve the differentiability of volumetric methods, while allowing for fast and efficient rendering through a tile-based rasterization pipeline.

Starting from a set of points obtained via Structure-from-Motion (SfM), each point is initialized as the mean $\boldsymbol{\mu}$ of a 3D Gaussian defined as:

$$\mathcal{G}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

where $\mathbf{x}$ denotes an arbitrary location in 3D space, and Σ is the covariance matrix of the Gaussian. To ensure that Σ is positive semi-definite, it is parameterized using a rotation matrix R and a scaling matrix S as:

$$\Sigma = RSS^\top R^\top$$

Each 3D Gaussian is further assigned a color $\mathbf{c}$, represented using spherical harmonics, and an opacity value α , which modulates the Gaussian’s contribution during blending via $\alpha \cdot \mathcal{G}(\mathbf{x})$.

Unlike traditional volumetric methods that rely on computationally expensive ray marching, 3D-GS renders scenes efficiently using a tile-based rasterization approach. Each 3D Gaussian $\mathcal{G}(\mathbf{x})$ is projected onto the image plane to form a corresponding 2D Gaussian $\mathcal{G}'(\mathbf{x})$, following the projection procedure described in [10]. The rasterizer is then employed to sort the projected 2D Gaussians and to perform alpha blending.

The color at a pixel location $\mathbf{x}'$ is computed using alpha compositing over the sorted set of 2D Gaussians N that influence the pixel:

$$C(\mathbf{x}') = \sum_{i \in N} \mathbf{c}_i \sigma_i \prod_{j=1}^{i-1} (1 - \sigma_j), \quad \text{where} \quad \sigma_i = \alpha_i \mathcal{G}'_i(\mathbf{x}'),$$

where α_i is the opacity of the i -th Gaussian, $\mathbf{c}_i$ is its color, and $\mathcal{G}'_i(\mathbf{x}')$ is the value of the projected 2D Gaussian at the pixel location. This differentiable rasterization process enables all Gaussian parameters to be optimized end-to-end via reconstruction loss from training views.

1.4 Anomaly Inspection

Anomaly inspection usually involves detection, localization and classification [11]. Due to the scarcity of defective images, the first two tasks are mainly addressed by unsupervised approaches which learn models based only on nominal samples. Unsupervised anomaly detection and localization falls in two broad-categories. The first deploys features obtained by deep neural networks [12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]. The second category focuses on learning the reconstruction of nominal samples using auto-encoders [31, 32, 33, 34], diffusion models [35] and image inpainting [36]. Then, at the inference time, the model fails to reconstruct the defective images correctly, leading to a discrepancy between the reconstructed and original images, which can be used to compute per-pixel anomaly map. More recently, some methods have shown promising results in addressing zero-shot or few-shot anomaly detection and localization [37, 38, 39, 40, 41, 42, 43].

It is worth pointing out that anomaly classification, a key task in manufacturing scenarios where visual inspection is necessary for both quality control as well as process monitoring, in that it can not be tackled by an unsupervised paradigm. As such, it stands out as the anomaly inspection task most severely affected by the data scarcity problem.

1.5 Content of Thesis

This thesis is organized into two parts, which are briefly outlined below.

1.5.1 Image Generation for Industrial Anomaly Inspection

In Chapter 3, we introduce AnomalyControl [44], a novel framework for generating images for industrial anomaly inspection. Our approach is based on training ControlNet [8], which leverages the production-ready Stable Diffusion inpainting [3] model. To ensure the quality of the generated data, we employ DINO-V2 [45] to filter out images that either lack realistic defects or where Stable Diffusion inpainting fails to produce plausible anomalies. In Chapter 4, we present an extension of AnomalyControl, developed in collaboration with a Master’s student [46], which focuses on learning to inpaint nominal images to further improve the performance of downstream anomaly classification.

1.5.2 Novel View Synthesis

In Chapter 6, we introduce NVS-HO, a benchmark designed for Novel View Synthesis (NVS) of handheld objects in real-world environments. The dataset is collected in two complementary sequences:

- **Handheld Sequence (HS)** – The object is freely manipulated by hand while the camera remains static, realizing natural human interaction. This sequence is used to train a NVS model that can capture the full appearance of the object.
- **Board Sequence (BS)** – The object is placed on a ChArUco board while the camera is moved across a set of capture positions. The markers present on the board enable accurate pose annotations, referred to as ground-truth poses. This sequence is used to evaluate the trained NVS model.

The ultimate goal of this benchmark is to create Neural Twins of handheld objects and to evaluate them by rendering view points corresponding to the board sequence.

Part I

Image Synthesis for Industrial Anomaly Inspection

Chapter 2

Related Work and Background

Automated quality inspection is a key requirement in modern manufacturing. Yet, the difficulty and costs associated with acquiring sizable and representative collections of defective images in production lines pose a data scarcity problem. Hence, unsupervised approaches which tackle anomaly detection and localization by training models only on non-defective images -also known as nominal images- have become popular in literature [47, 48, 12, 15]. Along a different path, researchers have been trying to address the data scarcity problem by techniques aimed at generating synthetic images of anomalies. Model-free methods crop-and-paste anomalies from defective images onto nominal ones [49], or, alternatively, swap random patches between nominal images [18]. Conversely, model-based approaches, e.g., relying on Generative Adversarial Network (GAN) [1], learn to render synthetic anomalies from a training set of real defective images [11, 50, 51]. A recent diffusion model based technique [52] achieved the state-of-art performance in anomaly inspection by generating synthetic defects by leveraging textual inversion [6]. Following this line of research, we first introduce AnomalyControl [44] in Chapter 3, where we show that controlled generation of synthetic anomalies improves downstream anomaly inspection. Then, in Chapter 4, we demonstrate how inpainting nominal images [46] can further enhance classification performance by addressing the limitations of AnomalyControl [44].

2.1 Related Works

2.1.1 Anomaly Inspection

Anomaly inspection usually involves anomaly detection, segmentation and classification [11]. Due to data scarcity of industrial anomaly images, the first

two tasks are addressed with the help of unsupervised approaches where the models are trained solely on nominal samples. Anomalies are detected based on the distribution gap between predictions and nominal samples. Unsupervised approaches fall into two categories: (1) feature-based methods using deep models [12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23, 27, 28], and (2) reconstruction-based methods leveraging autoencoders [31, 32, 33, 34], inpainting [36], or diffusion models [35]. More recently, zero-shot and few-shot methods [37, 38, 39, 40, 41, 42, 43] have achieved encouraging results. However, anomaly classification remains limited by data scarcity, and this can not be tackled by unsupervised paradigms.

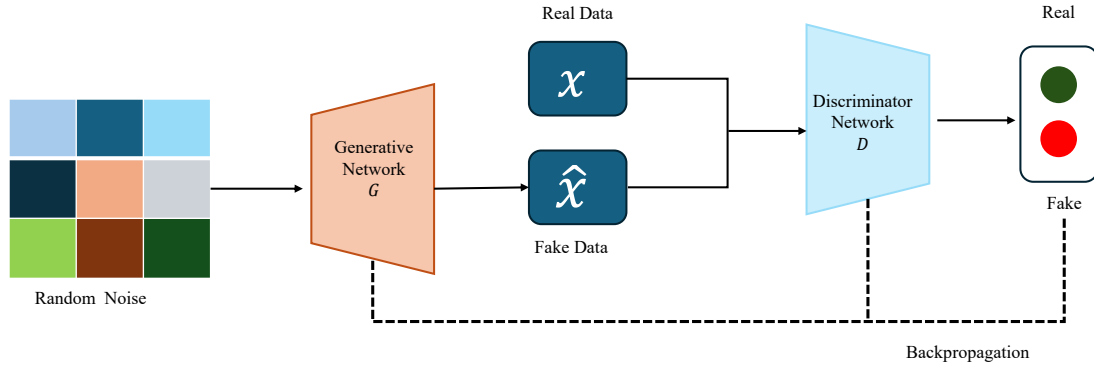


Figure 2.1: Illustration of the Generative Adversarial Network (GAN), consisting of a generator trained to produce realistic samples and a discriminator trained to distinguish real from generated data [1].

2.1.2 Generative Models

Image generation is a central part of our work. One of the first methods to achieve this goal was Generative Adversarial Networks (GANs) [1], which are, however, hard to train due to the potentially unstable min-max loss. We illustrate the GAN framework in Fig. 2.1. Then, more recently, diffusion models [53] have gained success in generating images across a wide range of domains. Latent diffusion models [3] have further improved generation capabilities by compressing images into a latent space and effective text-conditioned models trained on the huge LAION dataset [54] have been released. Moreover, the same architecture with more input channels has been trained to learn how to generate realistic images while inpainting a given picture in the area indicated by a mask. This model, known as Stable Diffusion Inpainting [3], has shown impressive results in replacing specific image areas without causing modifications to the rest of the image. Intuitively,

it appears as a promising tool to generate anomalies as one may inpaint nominal images conditioned on masks borrowed from available defective ones.

All these models can be tuned by Dreambooth [7] or Textual Inversion [6] to represent visual concepts specified by a few images as special words in the embedding space of a frozen text-to-image model. Finally, ControlNet [8] has emerged as a new effective paradigm to add precise spatial conditioning controls to large pre-trained diffusion models. In particular, when deployed with Stable Diffusion, the ControlNet architecture is realized by copying the U-Net encoder into a trainable network that leverages on zero-convolutions to internalize the spatial conditioning signal and to add its contributions to the skip connections toward the frozen U-Net decoder.

2.1.3 Anomaly Generation

To tackle the scarcity issue related to real defective images, researchers have been exploring a variety of approaches geared towards generation of plausible synthetic images that may be used to train downstream models pursuing anomaly inspection tasks. Among them, model-free methods [18, 49, 34, 55] crop and paste textures either within the same image, or from anomalous images into nominal ones, in order to generate new defective images, but the generated anomalies are not always realistic.

Other model-based approaches try to achieve better image realism by learning a generative model. For instance, Defect-GAN [50] generated the defective images on nominal samples by training a GAN [1] using defacement and restoration processes. Similarly, SDGAN [51] used two generators aimed at generating defective images from nominal images and vice versa. Differently, DFMGAN [11] trains StyleGAN2 [56] on nominal samples as a backbone, and subsequently attaches defect-aware residual blocks to learn representations of type-specific defects.

The advancements in diffusion models have recently led to novel frameworks to generate defective images. RealNet [57] trains a diffusion model on nominal images and then, to generate synthetic anomalies, adds perturbations in the reverse diffusion process. Thus, the method generates type-agnostic anomalous images that can improve downstream anomaly detection and localization but cannot be deployed to pursue anomaly classification. AnomalyDiffusion [52] proposes a few-shot anomaly generation method based on latent diffusion models [3]. They train two diffusion models: one for mask generation and one for anomaly generation. Specifically, they learn a textual embedding via a masked formulation of textual inversion [6] and use blended diffusion [58] to generate the synthetic anomaly images. Their framework aims to generate synthetic images closely aligned to real defective ones, and they report experimental results showing

state-of-the-art performance on anomaly inspection when only generated images are deployed to train the downstream, task-specific models. Nevertheless, their generated anomalies still exhibit significant misalignment with respect to real defective images. Finally, we wish to point out how [59] offers a different perspective to the deployment of diffusion model for anomaly inspection. Indeed, AnomalyControl [44] proposes a method focused on generating nominal images by a text-guided diffusion model [3]. As such, it realizes a data augmentation process to increase the variability of the nominal images and improve unsupervised anomaly detection and localization but does not help to address anomaly classification.

2.2 Background

2.2.1 Stable Diffusion Inpainting

Stable Diffusion [3] is a text-to-image diffusion model that is more efficient than DDPM since it applies the diffusion process to latent vectors computed by a VQ-VAE [60] autoencoder. The model is trained by first encoding an image x_0 with a learned encoder to generate a latent vector $z_0 = E(x_0)$. Then the DDPM forward process is applied in latent space to generate the Markov chain z_t with $t = 0, \dots, T$. Training learns to estimate z_{t-1} from z_t by predicting the noise $\epsilon_\theta(z_t, t, c)$, by conditioning it on the text prompt c . Hence, a real image $\hat{x}_0$ is generated by first sampling a noisy latent vector $\hat{z}_T \sim \mathcal{N}(0, I)$, and then iteratively denoising it using the learned denoiser ϵ_θ to produce a noise-free latent $\hat{z}_0$. Finally, the noise-free latent is passed through a decoder to obtain the generated image $\hat{x}_0 = D(\hat{z}_0)$. Fig. 2.2 illustrates the functionality of stable diffusion.

The same framework can be used to inpaint a masked image [3], i.e., generate photorealistic completions for a masked area given a text prompt c , a mask m and a masked image x^m . The only modification required is that the noise estimator $\epsilon_\theta(z_t, z_0^m, m; t, c)$ expects more input channels, since the input tensor should contain the concatenation of the latent to be denoised z_t with the latent vector $z_0^m = E(x^m)$, obtained by encoding the masked input image, and the mask $m_\downarrow$, i.e., the original binary mask m downsampled to the resolution of the latents. Stable Diffusion Inpainting has shown impressive results in replacing specific image areas without causing modifications to the rest of the image. Intuitively, it appears as a promising tool to generate anomalies as one may inpaint nominal images conditioned on masks borrowed from available defective ones.

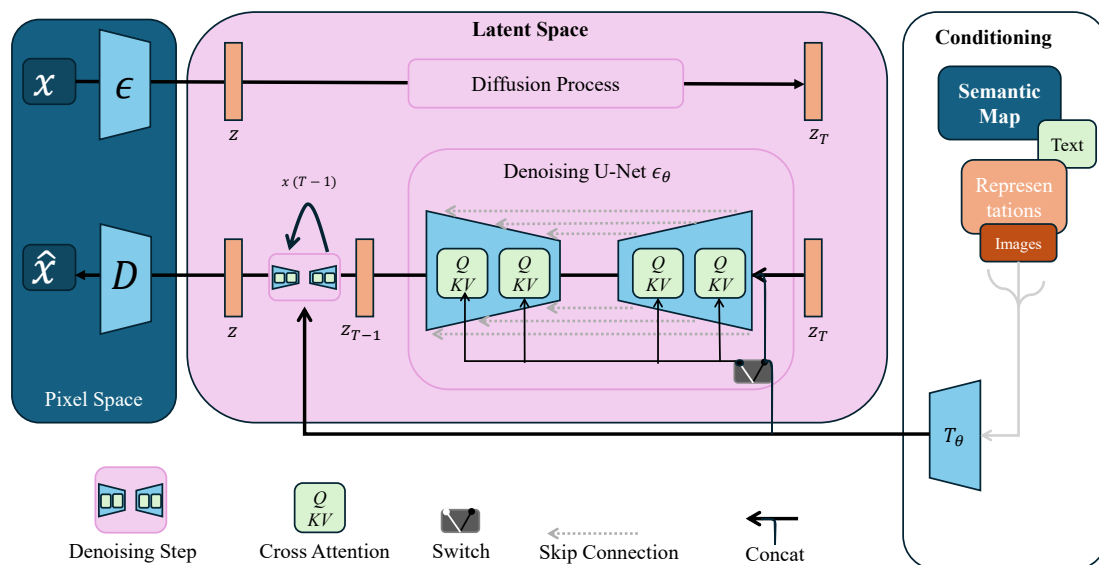


Figure 2.2: Illustration of the Latent Diffusion Model (adapted from Figure 3. in [3])

2.2.2 ControlNet

Obtaining images following a precise spatial arrangement with only text conditioning is impossible. ControlNet [8] is a neural network architecture designed to make it easy to control the generation of latent diffusion models by feeding them with image-based inputs detailing the desired spatial arrangement. In particular, the ControlNet architecture is realized by copying the U-Net encoder into a trainable network that leverages on zero-convolutions to internalize the spatial conditioning signal and add its contributions to the skip connections of the frozen U-Net decoder. Fig. 2.3 illustrates the architecture of ControlNet.

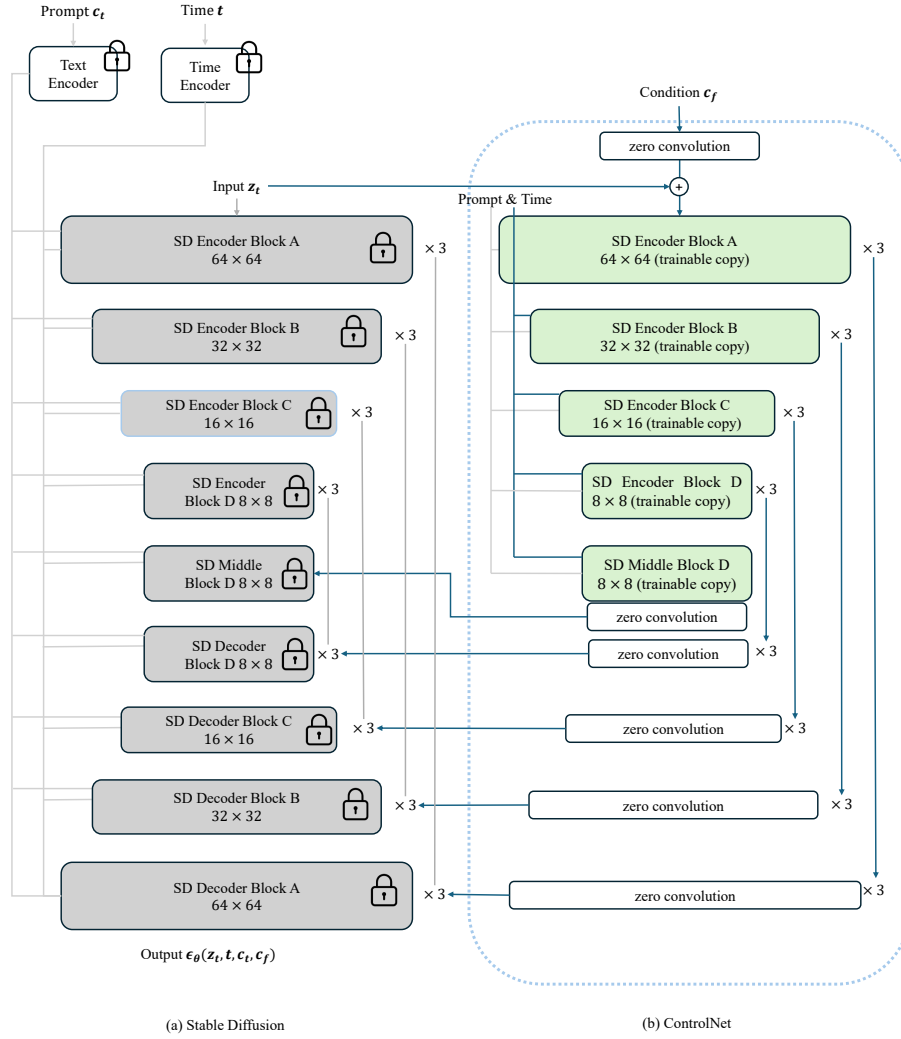


Figure 2.3: Illustration of the ControlNet architecture (adapted from Figure 3. in [8])

Chapter 3

Few-Shot Anomaly Generation

This chapter extends the foundations laid by AnomalyDiffusion [52]. Before, discussing our contributions, we will briefly present the key aspects of the prior work. AnomalyDiffusion [52] achieves state-of-the-art anomaly classification, detection and localization in the challenging setting where only a few real images are available to learn the image generator while the downstream, task-specific models are trained solely with the images sampled from the generator. However,

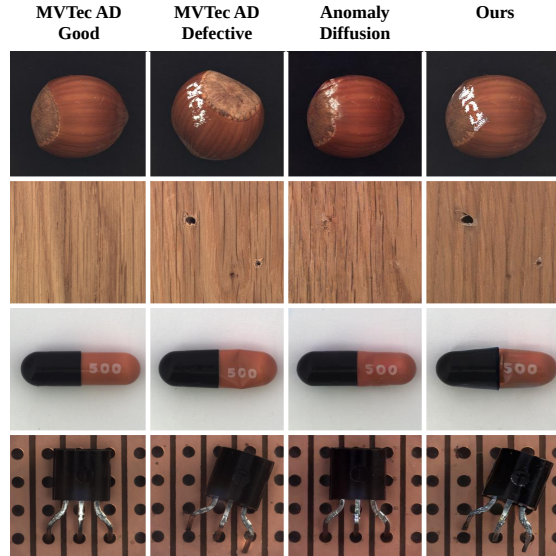


Figure 3.1: Nominal (first column) and defective images (second column, from top to bottom: hazelnut-print, wood-hole, capsule-squeeze, and transistor-misplaced) from the MVTec AD dataset [61]. Defective images generated by AnomalyDiffusion [52] (third column) and our framework (fourth column) for the same types of anomaly as in the second column.

generated images¹ looks quite unsatisfactory in terms of resemblance to real defective ones, as exemplified in Fig. 3.1 (third column). We therefore argue that existing techniques struggle in creating defective images aligned to real ones, which may significantly hinder performance on downstream tasks.

Therefore, we propose AnomalyControl [44], a novel framework based on stable diffusion inpainting which learns to generate images of anomalies from nominal images and a few anomaly masks taken from real defective images. We leverage the publicly available stable diffusion inpainting model² [3] trained on LAION-400M [62] as the backbone while training ControlNet [8] conditioned on masks from real defective image and text prompts describing the types of anomalies to synthesize. Besides, to improve alignment of the generated data with the real training dataset as well as to prevent generation of nominal images, we propose a two-stages, automatic filtration technique based on DINO-V2 embeddings [45]. Remarkably, our model can synthesize defective images that look closely aligned to real ones, as also vouched by Fig. 3.1 (fourth column). As a result, our proposed anomaly generation approach achieves the new state-of-the-art for anomaly classification and detection when training task-specific models only with images provided obtained by the generator, i.e., the standard setting adopted in literature to pinpoint the impact of the quality of generated images. Our main contributions can be summarized as follows.

- We propose a few-shot anomaly generation framework that trains ControlNet using stable diffusion inpainting as the backbone to generate realistic anomalies.
- We propose a filtering method based on DINO-V2 embeddings to automatically select the highest quality images out of multiple generations.
- By extensive experiments on MVTec-AD [61] dataset, we show that our approach sets the new state-of-art in defective image generation for anomaly classification.

3.1 Method

In this section, we introduce our proposed framework, which consists of the anomaly generation methodology and a two-stage image filtering procedure. Our framework is based on the intuition that inpainting of abundant nominal images

¹Available at <https://github.com/sjtuplayer/anomalydiffusion>

²Available at <https://huggingface.co/stable-diffusion-v1-5/stable-diffusion-inpainting>

in areas indicated by defective masks can be an effective way to generate defective images in a data scarcity regime. Indeed, inpainting does not require the model to learn the variance of nominal objects and let it focus on the appearance of the defects, a simpler task that can be easier to learn from limited data. Hence, we use Stable Diffusion Inpainting as our generator.

Our intuition on such model being a good tool to synthesize defective images could have been realized seamlessly had the available frozen model been trained with images and text prompts with the specific kind of defects found in industrial scenarios. Unfortunately, this is not the case. Stable Diffusion Inpainting has been trained on LAION-400M [62] and does not include industrial anomalies. So, to follow up on our intuition we would need to fine-tune Stable Diffusion Inpainting on defective images, which may cause overfitting or catastrophic forgetting due to the data scarcity issue. Therefore, we propose to use the ControlNet architecture to deploy Stable Diffusion Inpainting as a frozen backbone while effectively fine-tuning the overall model with limited data [8]. Thus, key to our approach is the idea of training ControlNet to learn the representations of anomalies using a pre-trained Stable Diffusion Inpainting model and we dub it AnomalyControl.

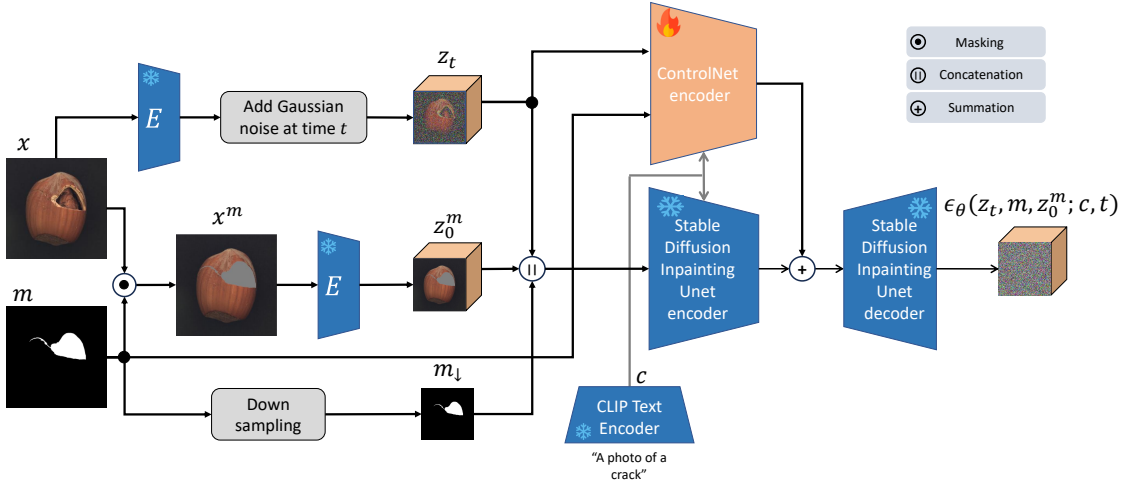


Figure 3.2: **An overview of the proposed training procedure.** At training time, a defective masked image x^m , noisy image latent z_t , its downsampled mask $m_{\downarrow}$, and text prompt c are fed to Stable Diffusion Inpainting, while ControlNet is given the mask m , noisy image latent z_t , and text prompt c . During training Stable Diffusion Inpainting, the CLIP encoder and the Stable Diffusion encoder are frozen.

3.1.1 Training

We now describe the actual training procedure with the help of Fig. 3.2. We randomly sample a defective image x with size $W \times H \times 3$, together with its associated anomaly binary mask $m \in \{0, 1\}^{W \times H}$, and hide the anomaly in x using m , so as to get x^m , i.e.,

$$x^m = x \odot m \quad (3.1)$$

where $\odot$ denotes the Hadamard product. Subsequently, we encode x^m and x using the Stable Diffusion encoder to obtain $z_0^m = E(x^m)$ and $z_0 = E(x)$ referred to as masked image latent and image latent, respectively. The encoder reduces the spatial size of the input images by a factor of 8 and creates 4 channels, i.e., $z_0^m, z_0 \in \mathbb{R}^{\frac{W}{8} \times \frac{H}{8} \times 4}$. As input to the pre-trained Stable Diffusion Inpainting U-Net [3] we concatenate three components: the noisy image latent z_t , obtained by using Eq. (3.2) on latents

$$z_t = \sqrt{\alpha_t} z_0 + \sqrt{1 - \alpha_t} \epsilon \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (3.2)$$

the masked image latent z_0^m , and the mask $m_{\downarrow} \in \{0, 1\}^{\frac{W}{8} \times \frac{H}{8}}$ obtained by downsampling m to the spatial resolution of the latent space. As input to the ControlNet, we provide the mask m and the noisy latent z_t . Both ControlNet and Stable Diffusion Inpainting are conditioned on t and the same anomaly-specific text prompt, such as “a photo of a [*]”, where $*$ represents the anomaly type, e.g. crack, scratch. The text prompt is encoded by CLIP [63] (see c in Fig. 3.2). The training objective to learn the parameters of the ControlNet, θ , is the standard MSE loss of diffusion models:

$$\mathcal{L} = \mathbb{E}_{(c, x, m), t} \left[\|\epsilon - \epsilon_{\theta}(z_t, m, z_0^m; t, c)\|_2^2 \right] \quad (3.3)$$

3.1.2 Image Generation

Here, we describe the generation process of defective images. We randomly pick a nominal image and a defect mask associated with one of the defective images used at training time. We augment the mask with a randomly selected set of transformations to enhance variability. Then, we pick the corresponding prompt from a prompt dictionary, e.g. “a photo of a crack” for the hazelnut-crack anomaly from the MVTec AD dataset [61]. Subsequently, we obtain $\hat{z}_T$ by randomly sampling noise in the latent space and feed it to our model together with the augmented mask and the prompt while running the denoising process in the latent space by DDIM [64]. Finally, the denoised latent representation, $\hat{z}_0$,

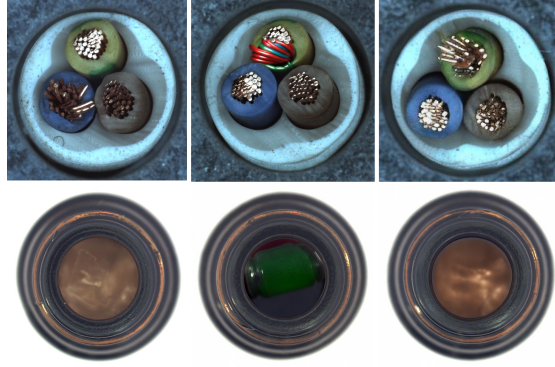


Figure 3.3: **From left to right:** real defective images, synthetic images generated with prompts “a photo of a cable with bent wire” (first row) and “a photo of a bottle with contamination” (second row), synthetic images generated with prompts “a photo of a bent wire” (first row) and “a photo of a contamination” (second row).

goes through the Stable Diffusion decoder to generate the final defective image $\hat{x}_0 = D(\hat{z}_0)$.

3.1.3 Prompt Engineering

Good quality prompt engineering is a crucial factor in text-to-image generative models. We first experimented with a basic template in the form “a photo of a {0} with 1” where {0} is replaced by the object class and {1} by the anomaly type, e.g., “a photo of a cable with bent wire”. Adopting this prompting scheme with our inpainting-based approach yields artifacts in the generated images, such as whole objects drawn within the defect mask. Therefore, we removed any reference to the object class from the prompt template and directly ask for a specific defect, e.g., “a photo of a bent wire”. Fig. 3.3 shows two examples in which our modified prompting scheme enables the generation of more realistic defective images. The quantitative impact on classification results of the two prompts is discussed in Section 3.3.

Also, we revised the anomaly names to be more descriptive. For instance, in MvTec-AD [61] when multiple defects are found on a *pill*, the anomaly class is set to “combined”. Besides, when the fabric is damaged in *zipper*, the anomaly classes are set to “fabric_border” and “fabric_interior”. which do not describe how the defect should appear. Hence, we engineered a set of better anomaly names to be included within the prompts, such as “multiple defects” instead of “combined” and “damaged fabric” in place of both “fabric_border” and

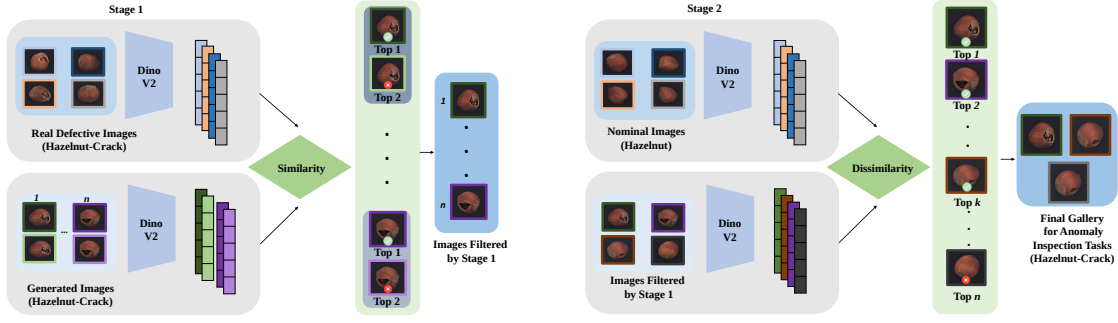


Figure 3.4: Two-Stage Filtration Strategy. **A) Stage 1:** in this stage we compute DINO [45] patch embeddings for both the original training set and generated images. For each pair of images generated with the same conditioning image, we retain the image with the closest average distance to the original training set. **B) Stage 2:** Here, we compute DINO [45] patch embeddings for good images and those filtered in stage A. We retain k images with the largest average distances to the good images.

“fabric_interior”. anomaly classes. The Appendix A reports the complete list of our prompts.

3.1.4 Image Filtering

The majority of images generated by our framework exhibit good quality, i.e., the defect is clearly visible and is well characterized with respect to its class. Yet, due to the randomness of the reverse diffusion process, some of the generated defective images may either (i) lack realism or (ii) look akin to nominal ones. Besides, the defect mask might be applied onto nominal images in a wrong position, e.g. in the background (iii). These issues motivated us to design a two-stage filtering approach, where the first addresses (i) and the second (ii) and (iii).

As illustrated in Fig. 3.4, *Stage 1* aims to remove images that deviate from the training set. We compute Dino-V2 patch embeddings for the generated images and the real defective ones used to train our model. In particular, we deploy the representation known as the *penultimate feature map*, which precedes pooling and consider only the patch embeddings, excluding the CLS token. Given a generated image, we score it by computing the average Euclidean distance of its patches in the embedding space with respect to the corresponding patches of the real defective images. Hence, to generate n images we generate n pairs under the same conditioning inputs (nominal image, augmented mask and text-prompt) and keep the best image in each pair.

Stage 2 focuses on removing images where defects are not properly inpainted, either because they are not generated or because the mask is in the wrong place. Here, we rank the n images obtained after filtration by *Stage 1* so as to keep only the best k ones ($k < n$). To rank images we deploy the same strategy based on Dino-V2 patch embeddings as in *Stage 1*, except that here we compare the n generated images to the nominal images and keep the k generated images with the largest average distance, i.e. those most dissimilar from nominal images.

3.1.5 Downstream tasks

We use the generated images to solve three downstream inspection tasks: image-level anomaly classification, image-level anomaly detection and pixel-level anomaly localization. For all these tasks, we follow the same methodology used in AnomalyDiffusion [52]. As far as classification is concerned, we train a linear classifier on top of a frozen ResNet-34 [65] backbone trained on ImageNet [66]. Conversely, a standard UNet [9] is trained from scratch to perform anomaly detection and localization.

3.2 Experiments

To assess the performance of our proposal, we carry out a series of experiments concerning anomaly classification, detection and localization. To this end, we adopt the MvTec-AD dataset [61], the main benchmark for industrial anomaly detection and localization. Finally, we present a qualitative comparison between the synthetic images generated by our approach and those obtained by AnomalyDiffusion [52].

3.2.1 Dataset

The MVTEC-AD dataset [61] used throughout our experiments comprises different objects captured at a resolution ranging from 700×700 to 1024×1024 pixels. As for defective images, the dataset provides pixel-level anomaly masks as well as image-level labels to denote the specific anomaly classes pertaining to each object. Following the protocol proposed in [52], we split the images belonging to each anomaly class into a train (1/3) and a test set (2/3), using the exact same splits as in AnomalyDiffusion [52], and resize all images and masks to 256×256 pixels. We point out that we can not use Visa [67], the other main anomaly detection and segmentation dataset, because it does not provide anomaly classes.

Parameter	Value
T	1000
(β_0, β_T)	(0.0001, 0.02)
β scheduler	linear
Batch Size	1
DDIM Steps	20
Learning Rate	0.00001
Training Steps	40000
Gradient Accumulation steps	8

Table 3.1: ControlNet Inpainting Hyperparameters

3.2.2 Implementation details

On a single NVIDIA 3090 GPU, our inpainting models takes 12-14 hours to train each object, when using the hyperparameter reported in Table 3.1. Generating an image requires about 2 seconds. Training the classifier on 500 generated images requires about 15 minutes per object.

3.2.3 Baseline Test Protocol

The experimental setup proposed in AnomalyDiffusion [52] deals with generating 1,000 synthetic images for each defect class and then, for each object, optimizing the downstream anomaly inspection models using only the generated images. In particular, to address the anomaly classification task, a linear classifier is optimized on top of a frozen ResNet-34 [65] backbone trained on ImageNet. Conversely, a UNet [9] is trained from scratch to perform anomaly detection and localization. However, to obtain the results reported in AnomalyDiffusion [52], the authors validated the models on the test set³. This model selection strategy may overestimates generalization performance significantly, to provide the reader with a thorough comparison we generated 1,000 images with our method and run similar experiments. In particular, we trained each model 10 times and report the average result across the trials. Hereinafter, we will refer to this experimental set-up as the *baseline test protocol*.

3.2.4 Proposed Test Protocol

In a real industrial setup one is often interested in assessing the performance attainable on unseen data gathered continuously from the field. To avoid overestimating performance, the test split of the available dataset should not be

³As discussed in <https://github.com/sjtuplayer/anomalydiffusion/issues/41>

Object	DiffAug	CDC	Crop&Paste	SDGAN	Defect-GAN	DFMGAN	Anomaly Diffusion	AnomalyControl (ours)
	[68]	[69]	[49]	[51]	[50]	[11]	[52]	
bottle	48.84	38.76	52.71	48.84	53.49	56.59	90.70	<u>87.44</u>
cable	21.36	39.06	32.81	21.88	21.36	45.31	<u>67.19</u>	73.75
capsule	34.67	28.89	32.89	30.22	32.00	37.23	<u>66.67</u>	69.73
carpet	35.48	25.27	27.96	21.50	29.03	47.31	<u>58.06</u>	61.21
grid	28.33	35.83	28.33	30.83	27.50	40.83	<u>42.50</u>	56.25
hazelnut	65.28	54.86	59.03	43.75	61.11	81.94	<u>85.42</u>	92.36
leather	40.74	43.38	34.39	38.10	42.33	49.73	<u>61.90</u>	75.87
metalnut	58.85	48.44	59.89	44.27	56.77	<u>64.58</u>	59.38	72.82
pill	29.86	21.88	26.74	20.49	28.47	<u>29.52</u>	<u>59.38</u>	68.13
screw	25.10	32.92	28.81	26.75	28.81	37.45	48.15	<u>39.38</u>
tile	59.65	48.54	68.42	42.69	26.90	74.85	84.21	<u>75.79</u>
transistor	38.09	29.76	41.67	32.14	35.72	52.38	<u>60.71</u>	84.64
wood	41.27	28.57	47.62	30.95	24.60	49.21	<u>71.43</u>	77.61
zipper	22.76	14.63	26.42	21.54	18.70	27.64	<u>69.51</u>	75.36
Average	39.31	35.06	40.55	32.43	34.77	49.61	<u>66.09</u>	72.16

Table 3.2: Anomaly classification accuracy - baseline test protocol. For each row, best result in bold, second best underlined.

used for model selection. Therefore, we propose an experimental protocol based on two different model selection criteria. In fact, while training models by synthetic images only, we propose to either pick the last checkpoint or the one that perform best on the training split of the real anomaly images (1/3 for each anomaly class). Since the authors of AnomalyDiffusion [52] provide only 504 generated images for each defect class through their project repository⁴, to perform a fair comparison we trained the downstream anomaly inspection models with their generated images and 500 randomly sampled images for each defect class from our 1,000 images. Finally, due to the high variance in performance of both methods, we trained 10 anomaly inspection models for each object of the MVTec-AD dataset and report the means. Hereinafter, we will refer to this experimental set-up as to the *proposed test protocol*.

3.2.5 Anomaly Classification

In Table 3.2, we report the classification accuracy following the *baseline test protocol*, i.e. we complement Table 4 of AnomalyDiffusion [52] with our results (last column). It is clear that, while AnomalyDiffusion had established the state-of-the-art by surpassing all previous proposal by a large margin (e.g. more than 16% in average accuracy), our method yields significantly better performance. For instance, our method outperforms AnomalyDiffusion on most objects by a 6% improvement in average accuracy. In Table 3.3 we rely on the *proposed test*

⁴as of 06-29-2024

Object	Validation on Anomaly Images (Train Split)			Last Checkpoint		
	Anomaly Diffusion	AnomalyControl (Ours)	Statistical Significance	Anomaly Diffusion	AnomalyControl (Ours)	Statistical Significance
bottle	76.97	73.95		80.92	79.76	
cable	71.87	65.46	✓	69.52	69.37	
capsule	39.60	61.46	✓	39.06	60.79	✓
carpet	56.61	51.12	✓	56.39	51.44	✓
grid	32.80	43.00	✓	39.75	38.75	
hazel nut	70.83	84.58	✓	73.22	80.20	✓
leather	48.09	65.55	✓	47.74	67.61	✓
metal nut	75.93	65.00	✓	75.76	66.36	✓
pill	49.79	57.39	✓	51.14	63.08	✓
screw	36.04	30.00	✓	39.74	29.25	✓
tile	85.08	71.22	✓	85.96	70.70	✓
transistor	58.92	75.00	✓	46.40	79.64	✓
wood	67.14	69.04		64.91	73.52	✓
zipper	61.70	63.29		58.51	66.92	✓
Average	59.39	62.60	✓	59.22	64.89	✓

Table 3.3: Anomaly classification accuracy - proposed test protocol. For each row, the best statistically significant comparisons are in bold.

protocol to compare our method to AnomalyDiffusion. To perform a statistically sound analysis, we assess whether the observed difference in means is significant or due to randomness. Purposedly, we performed a Welch’s t -test [70] to refute the null hypothesis that the two populations, i.e., the accuracies observed in each run for both methods, have equal means. Results of the test when the null hypothesis is refuted by comparing against a significance threshold of 0.05 are reported as check-marks in the table. Results where the difference is statistically significant are highlighted in bold.

Firstly, as expected, regardless of the adopted model selection strategy, the accuracy of both methods is lower compared to Table 3.2. Yet, the majority of the statistically significant comparisons show the superiority of AnomalyControl. Overall, our method yields higher average accuracy (by about 3-5 % depending on the model selection strategy) also the proposed set-up is more conducive to better assessment of the generalization performance. Finally, the few available real anomaly images seem insufficient to consistently provide a validation signal strong enough to outperform vanilla selection of the last checkpoint.

Although AnomalyControl produces on average better results than the baseline, it is important to understand if there is any benefit in training on generated images rather than on the original training set of real defects only. In Table 3.4, we compare the accuracy of the same classifier trained on real images and generated ones, respectively. The additional variance and the larger number of images that our framework can make available to the classifier produce notable improvements in classification accuracy, sometimes as large as 30 or 40 points of accuracy (wood and zipper) and on average it outperforms real images by

Object	Real Training Set	Generated Images	Statistical Significance
bottle	71.86	79.76	✓
cable	42.96	69.37	✓
capsule	37.06	60.79	✓
carpet	35.64	51.44	✓
grid	21.50	38.75	✓
hazelnut	63.95	80.20	✓
leather	57.61	67.61	✓
metalnug	60.15	66.36	✓
pill	36.45	63.08	✓
screw	27.65	29.25	✓
tile	81.05	70.70	✓
transistor	58.92	79.64	✓
wood	43.09	73.52	✓
zipper	28.08	66.92	✓
Average	47.56	64.89	✓

Table 3.4: Classification accuracy when training on real images versus generated ones. Here we use the proposed test protocol and the last checkpoint. For each row, statistically significant results are in bold.

Object	Crop & Paste [49]			DRAEM [34]			PRN [55]			DFMGAN [11]			AnomalyDiffusion [52]			AnomalyControl (Ours)		
	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max
bottle	85.4	95.1	90.9	99.3	99.8	<u>98.9</u>	94.9	98.4	94.1	99.3	99.8	97.7	99.8	<u>99.9</u>	<u>98.9</u>	100	100	99.8
cable	93.3	96.1	91.6	72.1	83.2	79.2	86.3	92.0	84.0	95.9	97.8	93.8	100	100	100	<u>98.1</u>	<u>98.7</u>	<u>95.8</u>
capsule	77.1	94.1	90.4	93.2	98.7	94.0	84.9	95.8	94.3	92.8	<u>98.5</u>	<u>94.5</u>	99.7	99.9	98.7	<u>94.2</u>	98.3	93.7
carpet	57.7	84.3	87.3	95.3	98.7	93.4	92.6	97.8	92.1	67.9	87.9	87.3	96.7	98.8	<u>94.3</u>	<u>96.0</u>	<u>98.5</u>	94.4
grid	83.0	94.1	87.6	99.8	99.9	<u>98.8</u>	96.6	98.9	95.0	73.0	90.4	85.4	<u>98.4</u>	<u>99.5</u>	98.7	99.8	99.9	99.0
hazelnut	68.8	85.0	78.0	100	100	100	93.6	96.0	94.1	99.9	100	99.0	<u>99.8</u>	<u>99.9</u>	98.9	<u>99.8</u>	<u>99.9</u>	<u>99.1</u>
leather	91.9	97.5	90.9	100	100	100	99.1	<u>99.7</u>	97.6	<u>99.9</u>	100	99.2	100	100	100	100	100	<u>99.8</u>
metal nut	92.2	98.1	93.3	97.8	99.6	97.6	97.8	99.5	96.9	99.3	<u>99.8</u>	99.2	100	100	100	<u>99.9</u>	100	<u>99.5</u>
pill	51.7	87.1	91.4	94.4	98.9	95.8	88.8	97.8	93.2	68.7	91.7	91.4	<u>98.0</u>	<u>99.6</u>	<u>97.0</u>	99.1	99.8	98.1
screw	59.3	81.9	86.0	88.5	96.3	89.3	84.1	94.7	87.2	22.3	64.7	85.3	96.8	97.9	95.5	<u>90.7</u>	<u>95.4</u>	<u>89.7</u>
tile	73.8	91.1	83.8	100	100	100	<u>91.1</u>	<u>96.9</u>	89.3	100	100	100	100	100	100	100	100	<u>99.7</u>
toothbrush	81.2	91.0	88.9	<u>99.4</u>	<u>99.8</u>	<u>97.6</u>	100	100	100	100	100	100	100	100	100	97.6	98.6	96.6
transistor	85.9	81.8	80.0	79.6	80.5	71.4	88.2	88.9	84.0	<u>90.8</u>	<u>92.5</u>	88.9	100	100	100	100	100	<u>99.8</u>
wood	49.5	81.2	86.6	100	100	100	77.5	92.7	86.7	98.4	99.4	98.8	98.4	99.4	98.8	<u>99.7</u>	<u>99.9</u>	<u>99.3</u>
zipper	59.4	82.8	88.9	100	100	100	98.7	99.7	97.6	99.7	99.9	99.4	99.9	100	99.4	100	100	<u>99.8</u>
Average	74.0	89.4	87.7	94.6	97.0	94.4	91.6	96.6	92.4	87.2	94.8	94.7	99.2	99.7	98.7	<u>98.3</u>	<u>99.3</u>	<u>97.6</u>

Table 3.5: Results on image-level anomaly detection - baseline test protocol

more than 17%. According to the Welch’s t -test, all comparisons are statistically significant.

3.2.6 Anomaly Detection and Localization

To assess anomaly detection and localization performance, we adopt the standard metrics introduced in [61] and, akin to [52], report the AUROC, AP, and F₁-max for the detection task, the previous metrics plus the PRO (Per Region Overlap)

Object	Crop & Paste [49]				DRAEM [34]				PRN [55]				DFMGAN [11]				AnomalyDiffusion [52]				AnomalyControl (Ours)			
	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO
bottle	94.5	67.4	63.5	77.8	96.7	80.2	74.0	91.2	97.5	76.4	71.3	88.5	98.9	90.2	83.9	91.7	99.4	94.1	87.3	94.3	98.8	88.6	80.9	92.4
cable	96.0	75.3	69.3	87.1	80.3	21.8	28.3	58.2	94.5	64.4	61.0	79.7	97.2	81.0	75.4	84.9	99.2	90.8	83.5	95.0	96.0	76.2	70.6	90.0
capsule	95.3	49.2	51.1	89.5	76.2	25.5	32.1	81.1	95.6	45.7	47.9	89.7	79.2	26.0	35.0	66.1	98.8	57.2	59.8	95.4	95.1	39.0	44.8	90.1
carpet	83.7	36.6	39.7	62.9	92.6	43.0	41.9	80.0	96.4	69.6	65.6	90.6	90.6	33.4	38.1	76.5	98.6	81.2	74.6	91.6	98.5	81.8	74.8	90.7
grid	84.7	13.1	22.4	70.2	99.1	59.3	58.7	95.8	98.9	58.6	58.9	95.8	75.2	14.3	20.5	52.3	98.3	52.9	54.6	92.3	98.6	49.5	53.5	95.3
hazelnut	88.5	38.0	42.8	74.1	98.8	73.6	68.5	95.9	98.0	73.9	68.2	92.7	99.7	95.2	89.5	96.4	99.8	96.5	90.6	97.1	99.6	93.2	86.0	96.3
leather	97.5	76.0	70.8	95.7	98.5	67.6	65.0	96.7	99.4	58.1	54.0	97.5	98.5	68.7	66.7	96.0	99.8	79.6	71.0	98.2	99.2	77.4	71.1	97.9
metalnut	96.3	84.2	74.0	67.2	96.9	84.2	74.5	90.4	97.9	93.0	87.1	85.0	99.3	98.1	94.5	88.0	99.8	98.7	94.0	94.8	99.6	97.8	92.6	92.8
pill	81.5	17.8	24.3	57.4	95.8	45.3	53.0	83.7	98.3	55.5	72.6	88.2	81.2	67.8	72.6	56.5	99.8	97.0	90.8	97.3	99.5	92.7	85.1	97.3
screw	93.4	31.2	36.0	83.9	91.0	30.1	35.7	78.1	94.0	47.7	49.8	83.8	58.8	2.2	5.3	41.8	97.0	51.8	50.9	80.3	89.9	13.5	23.2	71.2
tile	94.0	79.3	74.5	79.2	98.5	93.2	87.8	95.3	98.5	91.8	84.4	91.3	99.5	97.1	91.6	97.5	99.2	93.9	86.2	96.1	99.5	95.6	88.5	96.8
toothbrush	89.3	30.9	34.6	66.6	93.8	29.5	28.4	75.1	96.1	46.4	46.2	83.1	96.4	75.9	72.6	74.3	99.2	76.5	73.4	91.4	90.0	53.5	57.1	75.6
transistor	85.9	52.5	52.1	64.4	76.5	31.7	24.2	54.3	94.9	68.6	68.4	70.0	96.2	81.2	77.0	65.5	99.3	92.6	85.7	96.2	99.4	92.2	86.3	94.2
wood	84.0	45.7	48.0	57.9	98.8	87.8	80.9	94.7	96.2	74.2	67.4	82.1	95.3	70.7	65.8	89.9	98.9	84.6	74.5	94.3	99.1	86.6	77.7	96.4
zipper	94.8	47.6	51.4	83.4	93.4	65.4	64.7	84.6	98.4	79.0	73.7	93.7	92.9	65.6	64.9	83.0	99.4	86.0	79.2	96.3	99.0	84.0	77.2	95.8
Average	90.4	48.4	49.4	74.3	92.2	54.1	53.1	83.1	96.9	66.2	64.7	87.4	90.0	62.7	62.1	76.3	99.1	81.4	76.3	94.0	97.5	74.8	71.3	91.5

Table 3.6: Results on pixel-level anomaly localization - baseline test protocol

Object	Image-Level Anomaly Detection												Pixel-Level Anomaly Localization											
	Validation on Anomaly Images (Train Split)						Last Checkpoint						Validation on Anomaly Images (Train Split)						Last Checkpoint					
	AnomalyDiffusion			Ours			AnomalyDiffusion			Ours			AnomalyDiffusion			Ours			AnomalyDiffusion			Ours		
	AUC-I	AP-I	F ₁ -max-I	AUC-I	AP-I	F ₁ -max-I	AUC-I	AP-I	F ₁ -max-I	AUC-I	AP-I	F ₁ -max-I	AUC-P	AP-P	F ₁ -max-P	AUC-P	AP-P	F ₁ -max-P	AUC-P	AP-P	F ₁ -max-P	AUC-P	AP-P	F ₁ -max-P
bottle	99.5	99.9	99.3	99.9	99.9	99.3	99.8	99.9	99.0	99.0	99.6	98.5	98.8	84.5	78.6	92.4	98.6	88.3	81.5	91.6	98.8	83.3	77.6	92.3
cable	99.2	99.4	97.5	98.8	97.9	94.6	99.5	99.6	97.7	96.9	98.1	95.3	98.2	83.1	74.8	92.2	94.7	75.4	70.5	90.1	98.0	82.8	75.2	92.77
capsule	80.1	93.5	88.6	92.9	97.8	93.0	90.8	97.4	90.5	94.3	98.3	94.4	94.6	32.8	41.0	72.3	91.3	29.9	39.0	89.0	95.9	28.9	40.8	80.8
carpet	89.0	95.3	89.0	92.7	97.2	91.8	58.8	67.9	88.6	91.6	96.8	91.5	97.7	73.6	67.6	85.0	97.4	79.1	73.3	87.9	79.2	3.2	9.4	26.1
grid	93.9	97.5	93.2	98.5	99.4	98.5	93.8	97.2	91.9	98.3	99.4	98.6	97.3	46.4	51.5	92.6	95.4	19.0	27.3	70.7	96.2	47.2	52.3	91.4
hazelnut	99.7	99.8	99.0	100	100	99.9	98.0	98.7	95.5	100	100	100	98.9	81.9	76.2	92.7	98.0	82.9	76.2	95.9	98.6	75.8	70.1	91.2
leather	99.7	99.9	98.7	100	100	99.8	99.7	99.9	98.8	100	100	99.7	99.8	75.5	69.8	98.0	98.6	75.0	69.7	97.8	99.7	72.5	68.6	97.6
metal nut	99.7	99.9	99.1	99.5	99.9	99.2	98.2	99.5	98.2	100	100	99.8	99.5	97.8	92.7	91.7	99.4	97.7	92.9	91.3	99.2	96.7	91.5	99.1
pill	96.6	99.2	96.0	98.4	99.6	97.4	96.2	99.1	95.1	98.1	99.5	97.4	99.7	95.6	89.1	92.7	99.1	88.8	80.9	96.9	99.7	95.0	88.4	91.2
screw	82.5	91.8	83.5	72.4	85.8	80.9	79.0	91.0	83.4	65.7	82.1	79.9	91.8	45.0	49.1	80.4	86.5	14.0	24.2	61.6	80.2	43.9	49.3	65.8
tile	100	100	99.9	99.8	99.9	99.3	100	100	99.8	99.8	99.9	99.2	99.4	94.6	87.2	95.3	99.5	95.3	88.1	96.2	99.1	92.2	83.6	95.2
toothbrush	99.5	99.7	98.8	95.1	97.5	93.6	99.8	99.9	99.5	97.5	98.6	95.6	96.4	55.3	58.6	83.3	78.3	44.4	51.1	69.8	93.7	49.9	55.0	82.4
transistor	88.1	98.1	95.8	99.6	99.4	97.0	99.8	99.7	98.9	100	99.9	99.5	99.3	91.8	84.7	89.1	98.8	85.9	83.2	92.7	99.1	91.2	84.4	92.7
wood	98.2	99.2	96.2	99.9	99.9	99.1	97.8	99	95.6	99.8	99.9	99.0	97.6	70.5	67.0	89.5	97.4	81.6	74.9	95.2	96.0	58.5	59.9	82.6
zipper	100	100	99.9	100	100	99.9	99.9	100	99.2	99.9	100	99.6	99.2	84.7	78.2	96.0	99.3	85.7	79.0	96.3	99.0	83.6	76.3	95.8
Avg	95.7	98.2	95.6	96.4	98.3	96.3	94.1	96.6	95.4	96.1	98.1	96.5	97.1	72.4	69.5	88.2	95.6	71.4	69.1	89.7	94.8	65.1	63.8	83.1

Table 3.7: Results of image-level anomaly detection and pixel-level anomaly localization - proposed test protocol

for localization. For the *baseline test protocol* we complement the corresponding Tables from [52] with the results obtained by our methods.

Overall analysis show how with the *baseline test protocol*, whose results are reported in Table 3.5 and Table 3.6, AnomalyDiffusion and our method achieve good performance, outperforming by large margins all other baselines both at detection and localization. Comparing the two methods according to the *proposed test protocol* as done in Table 3.7, our method is preferable for detection with both validation strategies, while one performs better than the other and viceversa on localization depending on which validation strategy is deployed.

3.2.7 Anomaly Generation Quality

We present here a qualitative comparison between generated images. We randomly sample images from those provided by AnomalyDiffusion [52] and from those generated by our own model, and report them in Fig. 3.5. For several MVTec [61] objects, our model generates more realistic defects than AnomalyDiffusion. For instance, AnomalyDiffusion cannot generate realistic images for the anomaly categories **crack**, **squeeze** and **poke** of the “capsule” object, all categories of the “leather” object, and most categories of the “grid” object. In some objects, the defects are of similar quality, e.g. in the **bent wire** and **missing wire** of the “cable” object or the **bent anomaly** of the “metal nut” object. Finally, in a few

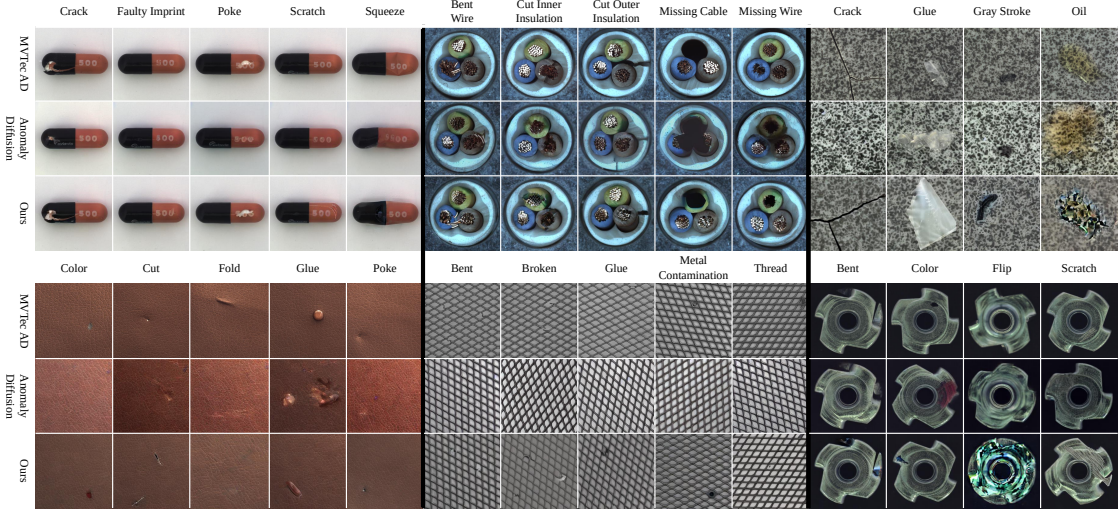


Figure 3.5: Qualitative comparison of the anomaly images from the MvTec-AD dataset [61], the images generated by AnomalyDiffusion [52] (middle row) and our method (third row).

cases the defects by AnomalyDiffusion are more aligned with the original defects, e.g., for `oil` of the “tile” class or the `flip` anomaly in the “metal nut” class. Indeed, these two objects are two of the objects where AnomalyDiffusion classification reports better classification results than ours, while “leather”, “grid” or “capsule” are some of those where we outperform AnomalyDiffusion. This confirms that there is strong correlation between the quality of the generated images and the classification performance, and that further improvements in the former will likely boost the latter.

Finally, to show that our framework is general and can be used in other domains, we present qualitative examples of generated images on retinal edema segmentation challenge [71]. We train AnomalyControl with the same procedure used for MVTEC AD and present qualitative images in Fig. 3.6. Interestingly, our framework effectively generalizes to this unrelated domain and is able to generate realistic defects with no changes to the pipeline.

3.3 Ablation Study

Ablation study on filtering stages.

We evaluated our proposed filtering stages under three different scenarios, i.e., “only the first stage”, “only the second stage”, “both stages”. Furthermore, we use average L1 distance, average Cosine distance and average Euclidean distance to

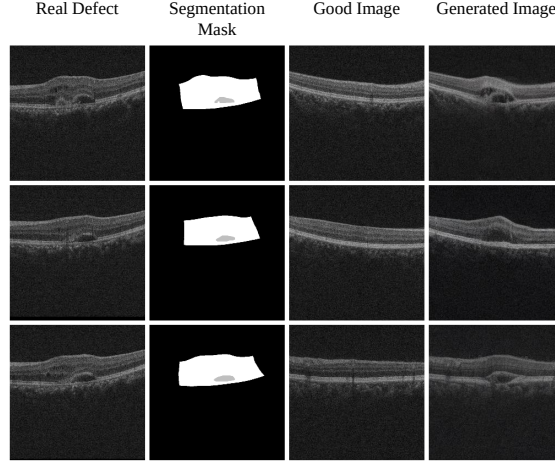


Figure 3.6: Qualitative comparison on Retinal Edema Segmentation Challenge [71]. Real defective images from (first column), segmentation masks (second column), good images (third column), and generated images by our method (fourth column).

evaluate each of the filtering strategies. For each scenario, we randomly sampled 500 images and trained a classifier 10 times. We use the last checkpoint and report the mean test accuracy in Table 3.8. The results indicate that both stages are important for achieving best classification accuracy, with the best performance obtained when using the average Euclidean distance.

Metric	None	Stage 1	Stage 2	Both
Euclidean distance	62.03	62.40	63.55	64.89
L1 distance	62.03	62.81	61.95	63.61
Cosine distance	62.03	63.14	63.18	63.56

Table 3.8: Ablation study on the distance used in the filtering stages. We report the mean classification accuracy across all objects. "None" train a classifier on the generated images without filtering them, "Stage 1" filters them applying only stage 1, "Stage 2" applies only stage 2 filtration, while "Both" is the proposed full pipeline.

Ablation study on prompt engineering.

We compare the classification accuracy on MVTEC classes “cable” and “bottle” using the two prompt engineering strategies discussed in Section 3.1.3. We achieve significantly better classification results with the proposed prompt templates in both classes as reported in Table 3.9, which confirms the importance of prompt

Object	Prompts w/ Objects	Proposed Prompts w/o Objects
bottle	66.27	79.76
cable	55.62	69.37

Table 3.9: Ablation study on prompt engineering.

engineering and validates the effectiveness of the proposed prompts.

Object	128 ²	256 ²	512 ²
hazelnut	76.60	85.62	91.87

Table 3.10: Ablation study on image resolution.

Ablation study on different resolutions. In the previous experiments, we used 256² as resolution when generating images to perform a fair comparison against the baselines. Here we perform an ablation study on MVTec class “hazelnut” using three different resolutions, i.e, 128², 256², and 512². We report the classification accuracy in Table 3.10. We observe that the classification accuracy increases as we move from lower to higher resolution. This suggests that our method holds to provide larger benefits in real-world deployments where resolution may be freely chosen.

Object	original	2-Shots	1-Shot
hazelnut	85.04	84.79	77.29

Table 3.11: Ablation study on varying number of training images per anomaly.

Ablation study on reducing number of training images. Here, we explore the robustness of our framework to limited numbers of training images per anomaly, a situation that is quite common in real deployments of inspection systems. We note that MVTec-AD already features a limited number of real defective images: for example, the transistor class contains only 3 images per defect type; the wood class includes just 2 or 3 images per defects; and the cable class has only 3 to 4 images per defect type. Yet, classification accuracy on these classes is in the range 69-79. To understand whether and how accuracy degrades when the number of training images is reduced significantly, we perform experiments on the “hazelnut” class, which has a relatively large number of images per anomaly (5,6 on average). We reduce the number of training images to 2 and 1 per anomaly type, and then train again our framework. We report the classification results in Table 3.11. The results show that images generated by our framework provide consistent classification results when as few as 2 images per anomaly type are used,

with accuracy degrading gracefully and turning out still acceptable when we use just 1 image per anomaly.

3.4 Conclusion

In this chapter, we have presented AnomalyControl [44], a framework to learn to generate large quantities of high-quality anomalous images from a few image-mask pairs. Moreover, the method includes a two-stage image filtering technique which further enhances the output quality. In the first stage, we filter out images that are not aligned with the real training set. In the second stage, we filter out images that do not contain proper defects. Extensive experiments demonstrate that our approach outperforms existing anomaly generation methods in defect classification, while it achieves similar performance to AnomalyDiffusion [52] in image-level anomaly detection and pixel-level anomaly localization.

Chapter 4

Learning to Inpaint Nominal Images

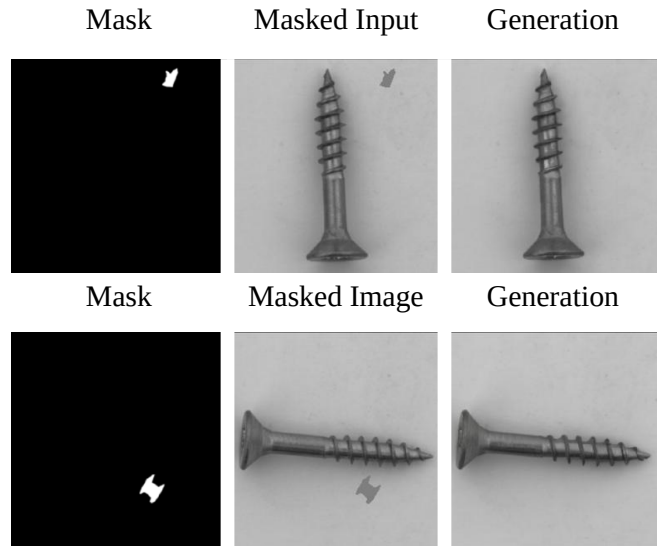


Figure 4.1: **Illustration of misalignment of mask and nominal Images.** Two examples showing that if the mask applied to a nominal image ends up in the background the model proposed in AnomalyControl [44] does not generate a defective image.

While developing AnomalyControl [44], we noted that it successfully produced realistic anomalous images; however, the random pairing of defective masks with the nominal samples results in masks being applied to the background or in regions where defects would not logically occur as illustrated in Fig. 4.1. We mitigate this limitation by generating high resolution defective images by inpainting multiple

times the few available real images of anomalies, thereby ensuring by construction that the mask and the image are properly aligned. To cope with the scarcity of anomalous images we create a VarianceGenerator [46] by fine-tuning another Stable Diffusion Inpainting model to generate non-defective parts of images.

Thereby we can increase the variance of the defective images generated. Purposely, we leverage the publicly available stable diffusion inpainting model trained on LAION-400M [62] as the backbone while training ControlNet [8] conditioned on masks created by a Perlin noise [72] generator and text prompts describing the object to synthesize. We show that this procedure generates better defective images for the objects in the MVTec AD dataset [61]. Where AnomalyControl [44] struggles due to the main limitation discussed above, this procedure results in higher classification accuracy for those objects. Since the images generated by our method have complementary properties with respect to those generated by AnomalyControl [44], training on a mixture of data generated by both models is shown to give the best results and attain the new state of the art in anomaly classification for the MVTec AD dataset. It is worth pointing out that anomaly classification, a key task in manufacturing scenarios where visual inspection is necessary for both quality control as well as process monitoring, cannot be tackled by an unsupervised paradigm. As such, it stands out as the anomaly inspection task most affected by the data scarcity problem.

4.1 Method

As discussed above, our framework mainly works on the idea inpainting-based strategy of AnomalyControl but differs in a key aspect by leveraging the few available real defects to guide the inpainting process. This ensures proper alignment between image foregrounds and defect masks, thereby overcoming the main limitation of AnomalyControl [44]. To further introduce the variance in nominal images, we fine-tune an additional Stable Diffusion inpainting model [3] using ControlNet [8] to replace nominal regions with newly synthesized counterparts, which we refer to as the VarianceGenerator [46]. For training this generator, multiple masks are randomly created using Perlin noise [72].

4.1.1 Model Training

Stable Diffusion Inpainting [3], the foundation model for various real-world inpainting tasks, is originally trained on LAION-400M [62], not on industrial data. To adapt it to the industrial domain, we fine-tune it using ControlNet [8] on the nominal samples of the MVTec AD dataset [61]. The training pipeline is illustrated in Fig. 4.2.

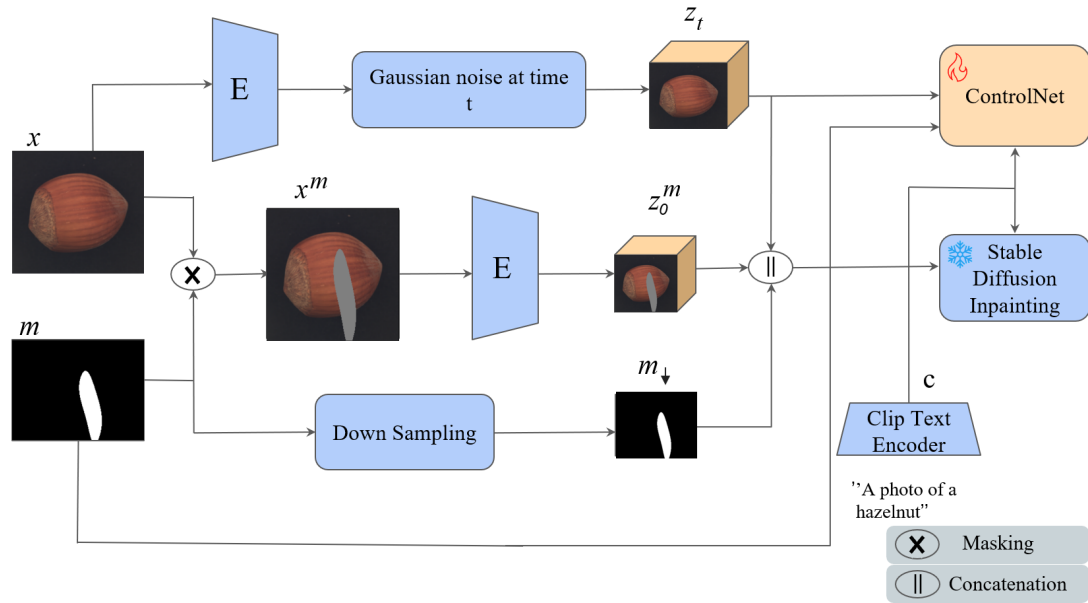


Figure 4.2: Training framework of the VarianceGenerator.

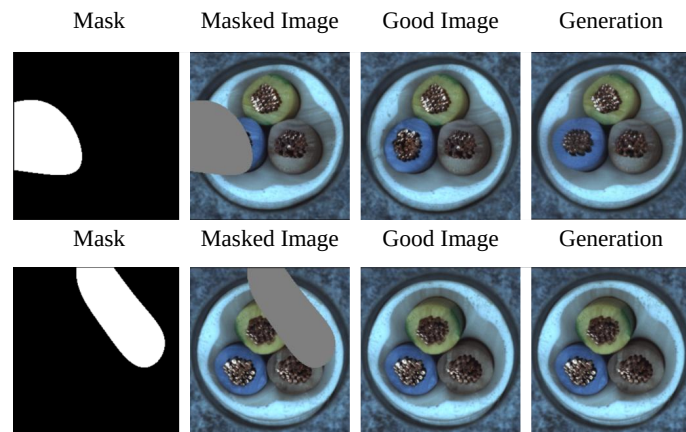


Figure 4.3: Two examples showing how the VarianceGenerator replaces good parts with other good parts.

For each nominal image x , a mask m is randomly generated using Perlin noise [72] to create a masked version x^m . Both the original and masked images are encoded into the latent space as $z_0 = E(x)$ and $z_0^m = E(x^m)$, respectively. Gaussian noise is then added to z_0 at timestep t to obtain z_t . The U-Net [9] takes as input the concatenation of z_t , z_0^m , and a downsampled mask $m_\downarrow$, while ControlNet is conditioned on z_t and the full-resolution mask m . Both the Stable Diffusion inpainting model and ControlNet are further conditioned via a text embedding c , obtained from a CLIP encoder [63] using prompts such as “A photo of a [*]”, where [*] represents an object from the MVTec AD dataset (e.g., hazelnut, bottle). The ControlNet is trained using the standard MSE loss for diffusion models:

$$L = \mathbb{E}_{(c,x,m),t} \left[\left\| \epsilon - \epsilon_\theta(z_t, t, c, m, z_0^m, m_\downarrow) \right\|_2^2 \right] \quad (4.1)$$

As shown in Fig. 4.3, the masked regions in the generated images (4th column) differ from those in the originals (3rd column), demonstrating the model’s ability to produce diverse and realistic variations of nominal object parts without anomalies.

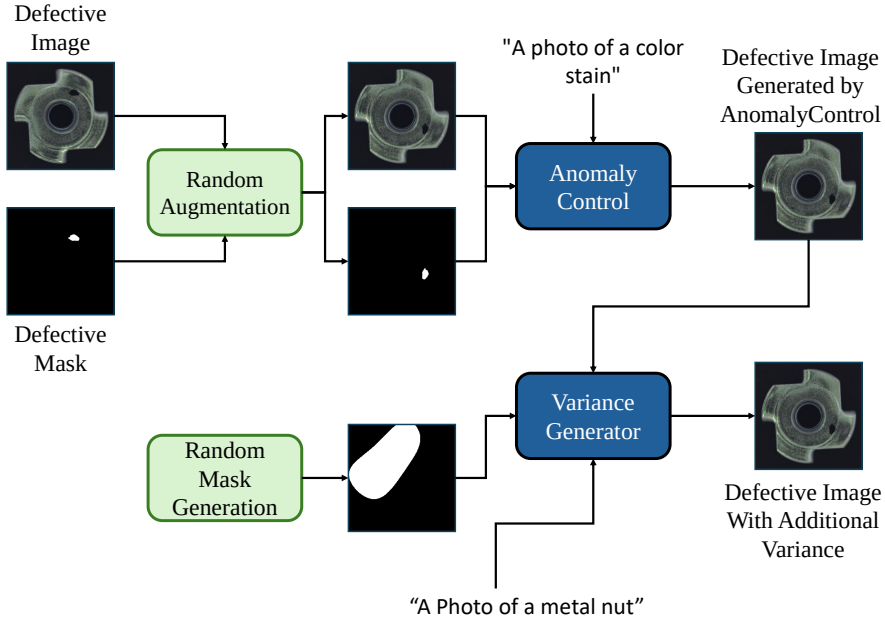


Figure 4.4: **Illustration of Image Generation Framework.**

4.1.2 Image Generation

Fig. 4.4 illustrates the proposed image generation framework, which comprises two models: AnomalyControl [44] used to synthesize initial defective images, and the

VarianceGenerator [46] designed to enhance diversity in the nominal regions. For each defective image-mask pair, random augmentations are applied to increase variability. The pair is first processed by AnomalyControl, which generates a defective image conditioned on a defect-specific prompt (e.g., “A *photo of a color stain*” for a pill anomaly in the MVTec-AD [61] dataset). Subsequently, a Perlin [72] noise mask is sampled, ensuring no overlap with the existing defective mask. If a non-overlapping mask cannot be found within 10 attempts, the AnomalyControl output is retained, and the process continues with the next pair. Otherwise, an object-specific prompt is sampled (e.g., “A *photo of a metal nut*”). Finally, the AnomalyControl output, the Perlin noise mask, and the object-specific prompt are passed to the VarianceGenerator, which produces the final defective image with enhanced variation.

4.2 Experiments

4.2.1 Dataset

The MVTec-AD dataset [61] used throughout our experiments, consists of 15 objects captured at resolutions ranging from 700×700 to 1024×1024 pixels. For defective samples, the dataset provides both pixel-level anomaly masks and image-level labels that indicate the specific anomaly class associated with each object. Following the protocol established in AnomalyControl [44], we split the images of each anomaly class into a training set (1/3) and a test set (2/3), using the exact same splits as reported in AnomalyControl [44].

4.2.2 Implementation Details

The generative models are trained on an NVIDIA GeForce RTX 2080 Ti GPU. For each object, training takes approximately 11 to 13 hours, and models are trained for a total of 20,000 steps. Training the classifier by the same GPU according to the evaluation protocol proposed in AnomalyControl [44], which deploys 500 generated images per defect class and requires running 10 trials, takes about 1 hour per object. The images using an NVIDIA TITAN V GPU, which requires about 2 seconds per image.

4.2.3 Evaluation Protocol

The evaluation protocol adopted in AnomalyControl deals with generating 500 synthetic images for each defect class and object. We conduct three separate experiments based on the size of the masks used within the VarianceGenerator [46].

In particular, we will denote as small, medium and large masks those covering between 5%–10%, 11%–18% and 11%–20% of the image area, respectively. Akin to [44], to address anomaly classification we utilize the generated defective images to train a linear classifier on top of a frozen ResNet-34 [65] backbone pre-trained on ImageNet. To perform a comparison with the results reported in AnomalyControl [44], we evaluate the models on the test set. Each model is trained 10 times, and we report the average performance across the 10 trials. For all experiments, we use the final training checkpoint. In Fig. 4.5 we show some examples of generated defective images.

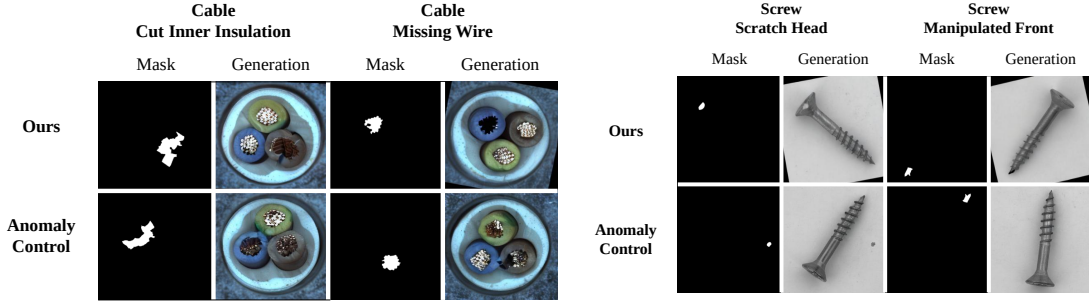


Figure 4.5: Qualitative comparison between AnomalyControl and our method using similar masks.

Object	Baseline	Large	Medium	Small
bottle	73.72 $\pm$ 2.76	78.84 $\pm$ 3.52	80.47 $\pm$ 3.15	78.14 $\pm$ 2.98
cable	73.75 $\pm$ 4.18	76.88 $\pm$ 5.27	75.78 $\pm$ 5.15	76.72 $\pm$ 2.92
capsule	35.07 $\pm$ 3.87	42.53 $\pm$ 3.12	41.07 $\pm$ 2.97	35.07 $\pm$ 4.22
carpet	56.45 $\pm$ 6.96	65.65 $\pm$ 4.79	62.26 $\pm$ 3.76	62.74 $\pm$ 6.90
grid	39.00 $\pm$ 6.54	29.50 $\pm$ 3.50	35.75 $\pm$ 3.72	37.00 $\pm$ 3.50
hazelnut	83.13 $\pm$ 4.70	83.75 $\pm$ 3.70	84.38 $\pm$ 4.39	84.79 $\pm$ 3.96
leather	64.29 $\pm$ 4.72	69.68 $\pm$ 8.20	70.95 $\pm$ 7.14	67.78 $\pm$ 5.90
metal nut	71.88 $\pm$ 2.10	71.88 $\pm$ 2.71	70.63 $\pm$ 1.68	69.84 $\pm$ 2.22
pill	37.40 $\pm$ 2.11	48.02 $\pm$ 5.11	47.19 $\pm$ 4.09	44.69 $\pm$ 3.62
screw	51.36 $\pm$ 3.99	53.70 $\pm$ 2.72	52.47 $\pm$ 3.63	49.63 $\pm$ 4.71
tile	79.12 $\pm$ 5.05	77.72 $\pm$ 6.18	75.97 $\pm$ 3.60	76.84 $\pm$ 3.40
transistor	68.21 $\pm$ 3.37	71.79 $\pm$ 6.28	72.86 $\pm$ 3.98	68.21 $\pm$ 5.64
wood	58.57 $\pm$ 5.24	56.43 $\pm$ 6.99	54.76 $\pm$ 5.22	53.10 $\pm$ 5.00
zipper	67.68 $\pm$ 3.50	62.81 $\pm$ 5.30	70.73 $\pm$ 4.60	68.78 $\pm$ 2.13
Average	61.40 $\pm$ 4.22	63.51 $\pm$ 4.81	63.95 $\pm$ 4.07	62.38 $\pm$ 4.07

Table 4.1: Ablation study of mask sizes.

Object	Ours	AnomalyControl	Ensemble
bottle	80.47 ± 3.52	79.76 ± 3.47	81.86 ± 2.28
cable	75.78 ± 5.27	69.37 ± 4.35	66.41 ± 2.90
capsule	41.07 ± 3.12	60.79 ± 2.82	57.73 ± 7.11
carpet	62.26 ± 4.79	51.44 ± 2.98	64.03 ± 2.61
grid	35.75 ± 3.50	38.75 ± 6.50	48.50 ± 4.50
hazelnut	84.38 ± 3.70	80.20 ± 4.25	80.42 ± 2.67
leather	70.95 ± 8.20	67.61 ± 4.62	85.24 ± 4.26
metal nut	70.63 ± 2.71	66.36 ± 1.61	68.28 ± 2.43
pill	47.19 ± 5.11	63.08 ± 4.31	61.46 ± 2.87
screw	52.47 ± 2.72	29.25 ± 4.93	36.17 ± 6.10
tile	75.97 ± 6.18	70.70 ± 2.74	78.25 ± 4.91
transistor	72.86 ± 6.28	79.64 ± 4.41	73.21 ± 8.03
wood	54.76 ± 6.99	73.52 ± 3.06	70.95 ± 3.66
zipper	70.73 ± 5.30	66.92 ± 4.79	69.27 ± 4.49
Average	63.95 ± 4.81	64.89 ± 3.91	67.27 ± 4.20

Table 4.2: Anomaly classification accuracy between AnomalyControl [44], Ensemble and our model.

4.2.4 Anomaly Classification

In Table 4.1, starting from real defective image, we compare classifier performance when trained on images generated by the VarianceGenerator [46] with varying mask sizes. As a reference, we also report the results of the pre-trained AnomalyControl model applied to the same images (denoted as *Baseline* in the table). The results indicate that medium-sized masks, covering approximately 11%–18% of the image area, yield the best performance. Furthermore, the table highlights that AnomalyControl struggles to generate effective images when applied directly to real defects, highlighting the importance of the proposed VarianceGenerator.

Table 4.2 reports a comparison between our model with medium size images and the original results from AnomalyControl (i.e. when applied on nominal images only). Our model performs particularly well on objects where AnomalyControl struggles. Since the images generated by our method exhibit complementary properties to those produced by AnomalyControl, training on a mixture of data generated by both models (i.e., 50% from each), referred to as the Ensemble approach in the Table, yields the best results. Indeed, this approach achieves the new state-of-the-art in anomaly classification on the MVTec AD [61] dataset.

4.3 Conclusion

In this chapter, we described framework to mitigate the weakness of AnomalyControl [44]. Specifically, we have discussed a novel method for generating defective images that synthesize anomalies on existing anomaly images, thereby addressing the common shortcomings of AnomalyControl [44]. However, due to the limited availability of anomaly images, the regenerated nominal regions tend to exhibit low diversity. To overcome this limitation, we proposed the VarianceGenerator, which replaces the nominal parts of the generated images and increases variability. Experimental results demonstrate that incorporating the VarianceGenerator into the image generation pipeline significantly improves performance on objects where AnomalyControl [44] struggles. Moreover, training the classifier on an ensemble of images produced by both AnomalyControl [44] and our method achieves state-of-the-art anomaly classification results.

Part II

Novel View Synthesis

Chapter 5

Related Work and Background

In this part of the thesis, we present our benchmark for novel view synthesis (NVS) of handheld objects. Chapter 6, begins by introducing the existing benchmarks that provide datasets of handheld objects. We critically examine their design choices and discuss their main limitations, such as their reliance on 3D object models, MANO-based hand models, and depth maps to annotate the ground-truth poses. After identifying these shortcomings, we introduce our proposed benchmark, which addresses several of these gaps. In particular, we emphasize the distinctive features of our dataset, the capture conditions under which it was collected, and its suitability for evaluating future NVS methods.

5.1 Datasets

This section reviews prior work on existing datasets including single-hand object datasets, hand-object interaction datasets, two-hand datasets, and full human interaction datasets. We also summarize common NVS and pose estimation techniques that build on these resources. As shown in Table 5.1, most existing hand object interaction datasets rely on MANO-based [73] hand models, pre-scanned 3D object models, or depth sensors to obtain ground-truth annotations, which limits their realism and scalability.

Importantly, our work introduces the first dataset that provides accurate object pose annotations without requiring complex hand or 3D object models. Instead, we use a ChArUco board, providing a simple, fast, and cost-effective way to enable reliable evaluation of novel view synthesis in unconstrained handheld settings.

Dataset	Venue & Year	Real Images	#Number of			Sensor	Hand 3D Model	Obj 3D Model	Task
			Img	Sbj	Obj				
GUN-71 [74]	ICCV 2015	✓	12k	8	1988	RGB	grasp ID	×	Grasp classification
Dexter+Object [75]	ECCV 2016	✓	3k	2	2	RGB-D	fingertips	×	Fingertip & hand pose
EgoDexter [76]	ICCV 2017	✓	3k	4	-	RGB-D	fingertips	×	Fingertip pose
FPHA [77]	CVPR 2018	✓	105k	6	4	RGB-D	keypoints	✓	Hand pose & action
ObMAN [78]	CVPR 2019	×	150k	20	2,772	RGB	MANO	✓	3D hand-object pose
FreiHand [79]	CVPR 2019	✓	37k	32	27	RGB	MANO	×	Hand pose
GRAB [80]	ECCV 2020	×	-	10	51	RGB	MANO	✓	3D hand-object pose
ContactPose [81]	ECCV 2020	✓	2,991k	50	25	RGB-D	MANO	✓	Hand-object contact pose
YCB-Affordance [82]	CVPR 2020	✓	133k	1	21	RGB	MANO	✓	Hand-object pose & affordance
HO3D [83]	CVPR 2020	✓	78k	10	10	RGB-D	MANO	✓	Hand-object pose
DexYCB [84]	CVPR 2021	✓	582k	10	20	RGB-D	MANO	✓	Hand-object pose
H2O [85]	ICCV 2021	✓	571k	4	8	RGB-D	MANO	✓	Hand-object pose
OakInk [86]	CVPR 2022	✓	230k	12	100	RGB-D	MANO	×	Hand pose
HOD [87]	SIGA 2022	✓	126k	1	35	RGB-D	×	✓	Hand-object pose
SHOWMe [88]	ICCVW 2023	✓	87k	15	42	RGB-D	MANO	✓	Hand-object pose
HANDAL [89]	IROS 2023	✓	150k	20	212	RGB	×	✓	Object pose
ARCTIC [90]	CVPR 2023	✓	1,200k	9	10	RGB-D	MANO	✓	Hand-object pose
HO-Nerf [91]	ICCV2023	✓	85k	4	4	RGB	×	×	Hand-object pose & NVS
HOT3D [92]	CVPR 2025	✓	3.7M	19	33	RGB/Mono	MANO	✓	Hand-object pose
Ours (NVS-HO)	2025	✓	1.3k	1	67	RGB	×	×	Object NVS

Table 5.1: **Comparison of hand-object interaction datasets.** NVS-HO is the first handheld object dataset specifically aimed at NVS of objects. It provides ground-truth object poses without relying on MANO hand models or 3D object models, using instead a ChArUco board as a simple, fast, and cost-effective solution.

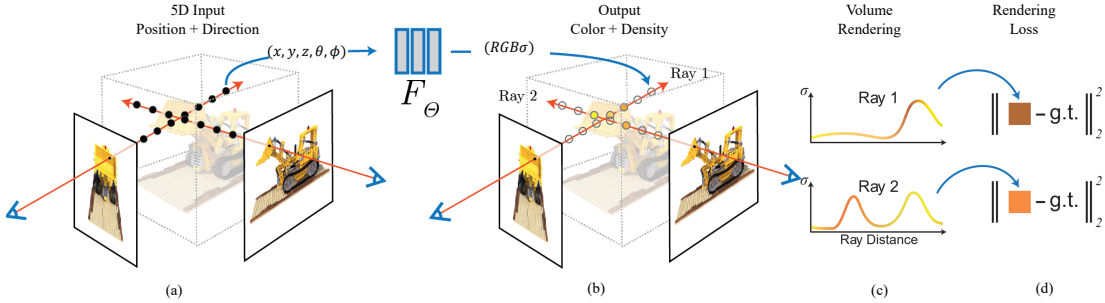


Figure 5.1: Illustration of neural radiance field scene representation and differentiable rendering procedure (adapted from Figure 2. in [4]) (a) NeRFs Feed the locations into an Multi Layer Perception to produce a color and volume density (b) using volume rendering techniques to composite these values into an image (c). This rendering function is differentiable, so it can be used to optimize the scene representation by minimizing the residual between synthesized and ground truth observed images (d).

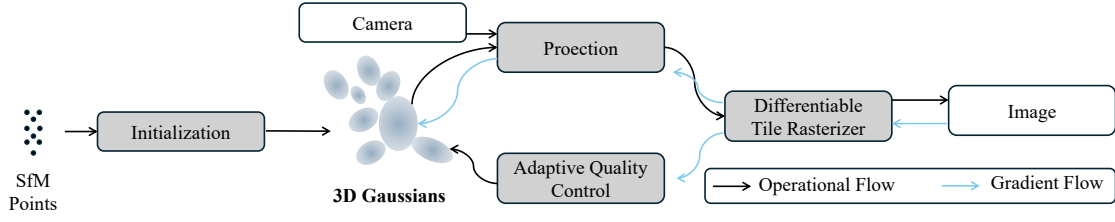


Figure 5.2: Illustration of 3D Gaussian Splatting (adapted from Figure 2. in [93]). The optimization starts with the sparse SfM point cloud and creates a set of 3D Gaussians. Subsequently, the density of this set is optimized using this set of Gaussians. During optimization fast tile-based renderer is used, which allows competitive training times compared to SOTA fast radiance field methods optimize and adaptively control the density of this set of Gaussians. Once trained, the renderer allows real-time navigation for a wide variety of scenes.

5.2 Related Works

5.2.1 Novel view synthesis

Novel view synthesis (NVS) aims to generate unseen viewpoints of a scene or object from a limited set of input images and has become a central problem in 3D vision. Neural Radiance Fields (NeRFs) [4] established the state of the art for NVS, while 3D Gaussian Splatting [5] introduced a novel rasterization-based approach. Both techniques achieve high-quality results, but their performance heavily depends on having accurate camera poses. Fig. 5.1 and Fig. 5.2 illustrate the functionality of NeRFs and Gaussian Splatting respectively.

More recently, Vision Transformer (ViT)-based approaches [94] have explored fully data-driven, feed-forward NVS, where explicit camera poses are not required. In particular, the Large View Synthesis Model (LVSM) [95] represents a significant step forward in pose-free NVS. Similarly, VGGT [96] extends this paradigm by leveraging four input views and representing target viewpoints using Plücker rays. In Fig. 5.3, we illustrate the functionality of VGGT. Nevertheless, these transformer-based approaches present limitations when scaling to large input sets (e.g., 100+ images). In such settings, traditional pose-based methods remain superior, as they can efficiently exploit dense multi-view data to train high-fidelity reconstructions and produce accurate novel views.

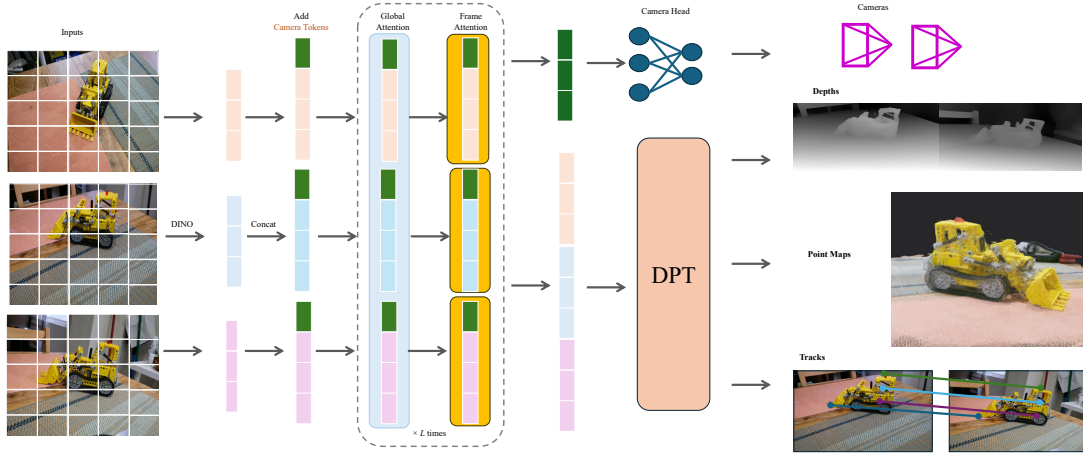


Figure 5.3: Illustration of Visual Geometry Grounded Transformer (adapted from Figure 2. in [96]). VGGT first patchifies the input images into tokens by DINO, and appends camera tokens for camera prediction. It then alternates between frame-wise and global self attention layers. A camera head makes the final prediction for camera extrinsics and intrinsics, and a DPT [97] head for any dense output.

5.2.2 Single-Hand objects

Recently, several hand-object datasets have been introduced for grasp classification and action recognition tasks [74, 98, 99, 100, 101]. However, despite the importance of hand action recognition, these datasets have rarely been used for object reconstruction, primarily due to the lack of ground-truth 6-DoF pose annotations.

Obtaining ground-truth pose information for hand-object interactions is a challenging task due to frequent hand occlusions and the typically low-texture appearance of objects. Moreover, in static scenes, it becomes nearly impossible to annotate object poses when the object is rotated around its own axis. These challenges also make it difficult to reliably integrate visual markers for pose annotation.

5.2.3 Two-hand 3D pose datasets

Annotating 3D hand poses becomes especially challenging when two hands interact or occlude each other. The Tzionas dataset [102] addressed this by combining RGB-D recordings with a realistic hand model that simulates skin movement [103], and by training a model to recognize fingertips from carefully labeled examples. Similarly, [104] introduced a dataset of two interacting hands captured with an

RGB-D camera, using a depth-based tracker [105] for annotation. Because the tracker was imperfect, they augmented the dataset with synthetic labels to improve accuracy. In a similar direction, ContactPose [81] relied on a richer setup of seven RGB cameras, three RGB-D sensors, and a thermal camera to capture detailed contact interactions between hands and objects.

5.2.4 6-DoF Object Pose Estimation and Tracking

6-DoF object pose estimation aims to recover the full position and orientation of an object in 3D space, expressed as its translation and rotation. Many state-of-the-art methods rely on category-agnostic CAD models for offline training and online template matching [106, 107, 108, 109]. While effective, this dependence restricts their ability to generalize to novel, previously unseen objects. More recent approaches [110, 111, 112] attempt to address this limitation by learning representations that extend beyond the training categories. However, they still require pre-captured reference images of the target objects at inference time, making them less practical in unconstrained real-world settings. Other works [113, 76, 114, 78, 82] explored synthetic data as an alternative, but the resulting domain gap limits their realism and robustness when deployed on natural imagery.

5.2.5 Object Reconstruction

Reconstructing 3D object geometry from images has been extensively explored through learning-based methods [115, 116, 4]. Recent advances in neural scene representations have enabled the recovery of high-quality 3D models from multi-view imagery [117, 118, 119, 120, 121, 5]. Most of these approaches rely on known and accurate camera poses, segmentation masks, and target scenes with rich textures and complex geometries. In particular [122], proposes a semi-automatic method with a similar objective, but it still required manual pose annotations to recover a textured model of the object.

Chapter 6

Novel View Synthesis of Handheld Objects

Hand-object interaction has received considerable attention in the computer vision field [123]. Vision-based systems capable of synthesizing realistic novel views from images of handheld objects enable a wide range of applications in robotics, augmented and virtual reality, digital content creation, and 3D reconstruction [124, 125, 126, 91]. Achieving high-quality novel view synthesis (NVS) in handheld settings requires accurate reasoning about object geometry and motion, often through intermediate 6-DoF pose estimation. While dedicated datasets have advanced handheld object pose estimation, the problem of NVS for handheld objects remains largely unexplored. Capturing handheld objects, where the object is rotated while the camera remains fixed, offers a particularly practical and accessible setup. This approach avoids camera drift, ensures stable framing, while guaranteeing complete 360° coverage with minimal effort. Compared to a moving camera, this setup is easier to reproduce with simple hardware and better reflects natural human interactions with objects, hence making it a strong foundation for benchmarking real-world NVS.

A wide range of datasets have been proposed for modeling and analyzing 3D hand-object interactions. Object-centric benchmarks such as LineMOD [127], YCB-Video [128], and T-LESS [129] paved the way for modern 6-DoF pose estimation by providing RGB-D images of texture-less objects together with accurate ground-truth poses derived from 3D models. Hand-object interaction datasets such as HO-3D [83] extends this direction by offering large-scale, multi-view data annotated with 3D hand pose and shape information. Other acquisition methods employ specialized equipment attached to the hand, including infrared markers [130], color-coded gloves [131], or electrical sensing devices [132], to facilitate pose estimation. While these datasets have driven progress in pose estimation, they were not designed for novel view synthesis and therefore lack

the ground-truth views and evaluation protocol to benchmark NVS methods. Furthermore, the techniques developed for existing datasets may fail to estimate poses from RGB images, due to their reliance on depth sensors, synthetic data, or 3D object models.

Recent NVS-oriented datasets have made important progress toward capturing objects in real-world environments. Though it primarily features static objects, CO3D [133] provides large-scale, in-the-wild captures across many categories. HO-NeRF [91] is specifically designed to model hand-object interactions, estimating hand-object poses as well as enabling joint NVS. WildRGB-D [134] brings objects into realistic contexts with background variation, albeit it relies on RGB-D hardware rather than RGB-only inputs. These considerations highlight the need for benchmarks that combine unconstrained handheld sequences with accurate reference viewpoints in RGB-only settings, so as to enable evaluation of NVS methods.

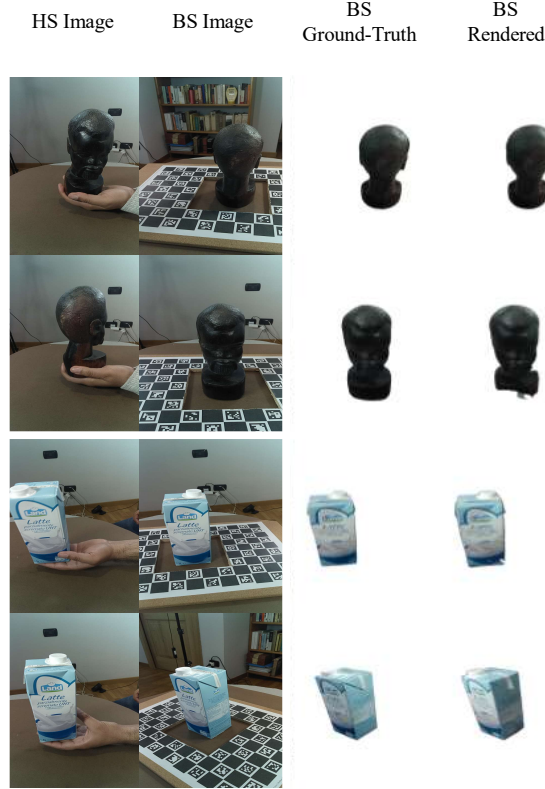


Figure 6.1: **Two exemplar objects from our dataset.** Our dataset addresses novel view synthesis from handheld images. The columns show: (1) images from the HS, (2) images from the BS, (3) masked ground-truth images and (4) masked rendered images.

Motivated by these limitations, we propose the first benchmark for NVS of handheld objects in unconstrained real-world conditions using only RGB images, without relying on MANO [73] hand models or pre-scanned 3D object models. As shown in Fig. 6.1, the dataset associated with our benchmark comprises two image sequences per object:

- **Handheld Sequence (HS)** – The object is freely manipulated by hand while the camera remains static, as in normal human interactions. This sequence is used to train a NVS model that can capture the full appearance of the object.
- **Board Sequence (BS)** – The object is placed on a ChArUco board while the camera is moved across a set of capture positions. The markers present on the board enable accurate pose annotations, referred to as ground-truth poses. This sequence is used to evaluate the trained NVS model.

To establish baselines, we use COLMAP [135], a state-of-the-art structure-from-motion (SfM) pipeline, to recover camera poses for the HS. We also adapt VGGT [96], a recent deep learning-based approach that leverages attention mechanisms to directly predict camera poses. Using the poses estimated by both methods, we train NVS models, including Neural Radiance Fields (NeRFs) [4] and Gaussian Splatting [5], on the HS of each of the object captured in our dataset.

To evaluate these methods, we render novel views from the same poses as the ground-truth ones and compare the rendered images to the corresponding BS images by standard metrics such as PSNR, SSIM, and LPIPS. As shown in Fig. 6.1, the comparison is carried out between masked image pairs where the background is represented with the same color (e.g. white). Our evaluation protocol ensures reliable reference viewpoints of the ground-truth poses of the BS, thereby highlighting any estimation errors derived from HS poses.

Key Contributions.

- We introduce the first benchmark specifically designed for NVS of handheld objects in real-world environments using only RGB images.
- We release a new dataset featuring RGB sequences that captures 67 common objects. Each object is recorded in two sequences: a HS featuring natural hand-object interaction and used for training and a BS equipped with ground-truth poses designed for evaluation.
- We propose a method-agnostic evaluation protocol to compare masked rendered images to masked ground-truth ones to separately evaluate object and background regions. Key to this protocol is the ability to bring the

ground-truth viewpoints into their own 3D reference frame that may be adopted by any NVS method.

- We benchmark state-of-the-art pipelines and reveal significant performance gaps under unconstrained handheld conditions, identifying directions for future research in pose estimation and NVS.

6.1 Dataset Creation

Material preparation. We collected 67 everyday objects spanning two broad categories: household items and toys. The objects vary in size, shape, and texture, introducing diverse challenges for 6-DoF pose estimation and novel view synthesis. As already highlighted in the introduction, two RGB sequences were acquired for each object both taken under controlled indoor lighting to ensure consistent image quality.

Camera. All sequences were captured with an OAK-D Lite camera¹, a compact RGB-D device with synchronized RGB and depth streams. In this work we only use the RGB modality. For every recording we captured approximately 100–200 RGB frames at a resolution of 1080×1920 pixels, starting and ending from the same viewpoint to guarantee complete 360° coverage of the object.

Calibration and Undistortion. The OAK-D Lite was calibrated using a standard chessboard pattern. The calibration images were captured at the same resolution as the dataset images. Calibration provides the intrinsic parameters matrix and the distortion coefficients, denoted here as $\delta = [k_1, k_2, k_3, p_1, p_2]$. All RGB frames are undistorted using these parameters, and the camera model is converted to a pure pinhole representation.

Image Acquisition. For each object we record two complementary monocular RGB sequences: HS and BS. The HS shows the object being freely manipulated by hand while the camera remains static. Although we capture image sequences rather than continuous video, some frames from HS exhibit motion blur, which we preserve to make the benchmark more realistic and challenging. These handheld sequences are used to estimate camera poses and train novel view synthesis methods.

The BS records the object fixed on a ChArUco board while the camera is moved around so as to capture a set of images that frame the object from different viewpoints. For these sequences, ground-truth poses are obtained using OpenCV [136]. The detected 2D ChArUco corners are matched to their known 3D board coordinates. Then, `solvePnP` computes the per-frame camera rotation R_i and translation T_i relative to the board’s coordinate frame. These board poses

¹<https://docs.luxonis.com/hardware/products/OAK-D Lite>

provide accurate ground truth annotations for evaluation, as well as reference viewpoints to quantify the impact of pose estimation errors in the handheld sequences on reconstruction and novel view synthesis quality.

6.2 Benchmark

6.2.1 Dataset, Task, and Metrics

Dataset. Using the procedures described in Section 6.1, we scanned 67 objects, capturing 100–200 views per object in both the HS and BS. The image sets densely cover 360° around each object, ensuring rich viewpoint diversity. One icon image per object taken from the HS is shown in Appendix B.1.

The key data released in our benchmark consist of undistorted RGB images from both kinds of sequences, alongside the estimated camera intrinsics. In particular, for each object, the images from the HS are used to train NVS models, while those pertaining to the BS are used for evaluation purposes. The number of training and testing (i.e. evaluation) images for each object is detailed in Appendix B.1.

Tasks. Our benchmark focuses on evaluating NVS of handheld objects in real-world conditions. Given a set of handheld images (i.e. the HS), a method must recover 6-DoF camera poses and use them to train a NVS model.

Evaluation is carried out by querying the trained NVS model at viewpoints corresponding to the BS, where accurate ground-truth poses are available thanks to the detection of ChArUco markers. The synthesized views are then compared to the corresponding masked RGB images from the BS frames. This setup directly measures both the quality of novel view synthesis and the robustness of the underlying pose estimation.

Evaluation protocol. Since external pose estimators may output trajectories in arbitrary scales and coordinate frames, we establish an evaluation protocol that can bring the test poses, defined in metric units within the coordinate frame of the BS, into the scale and coordinate frame estimated from the HS. To this end, we start by constructing a Combined Sequence (CS) that acts as a bridge between the coordinate frame of the BS and that estimated by the NVS from the HS.

As illustrated in Fig. 6.2, the camera poses of the BS are first estimated in metric units (mm) and with respect to a world reference frame associated with the ChArUco board via standard marker detection.

Then, we rely on Grounded SAM2 along with object specific prompts [137] to segment the foreground object in the images of the BS and HS, thereby getting rid of the background in both, as well as of the person’s hand in the latter. The foreground masks, thus are used to set to black all background pixels in both

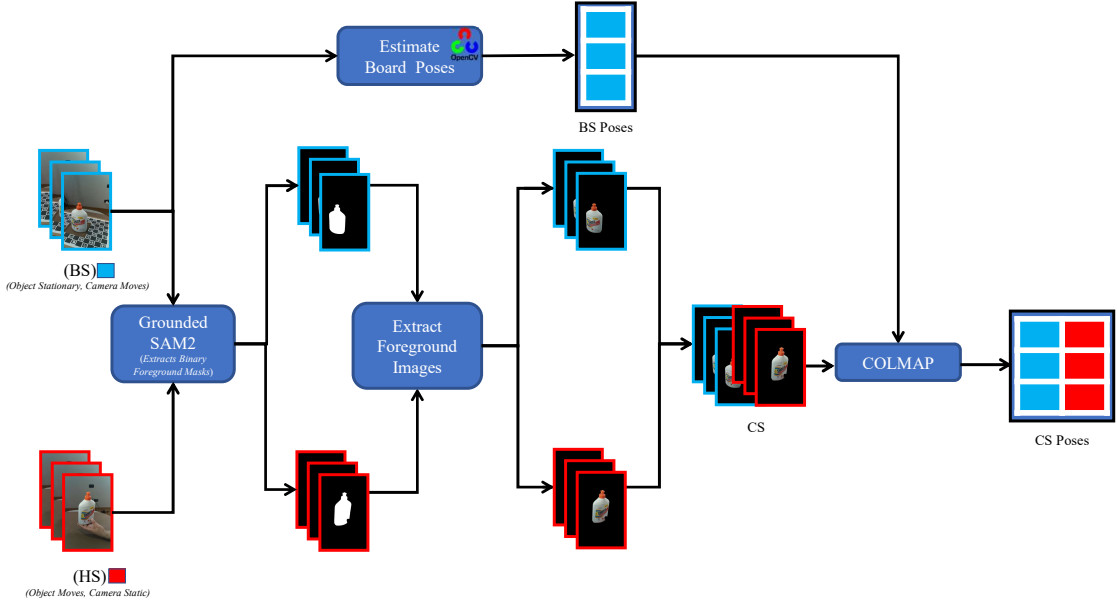


Figure 6.2: **Data processing pipeline.** For each object we record two monocular RGB sequences: an HS where the object is manipulated by a human while the camera is static, and a BS where the same object is fixed on a ChArUco board. BS poses are estimated with OpenCV via marker detection. Then, the Grounded SAM2 produces segmentation masks for BS and HS, enabling foreground extraction. HS and BS are combined to produce CS. The CS combined with the fixed BS poses and the estimated intrinsics are then fed into COLMAP that produces the resultant CS poses, where HS poses are estimated within coordinate system of the BS.

sequences. The masked images belonging to both the BS and HS are merged into a single sequence, previously introduced as CS. The next step of the procedure deals with running COLMAP [135] on the CS. In particular, we set the intrinsic parameters to the values estimated by camera calibration and inject as hard constraints the previously estimated poses of the BS. This forces COLMAP to recover the camera poses of the HS in the same metric units and world reference frame as those of the BS. Hence, the CS acts as a bridge between the two object captures and forms the foundation of our evaluation protocol.

As anticipated, a NVS method fed with the HS will estimate camera poses in its own units and coordinate frame. Thus, to synthesize novel views, it will have to be queried with poses defined according to such units and frame. As the poses of our test images are defined in the units and coordinate frame of the BS, to permit a quantitative evaluation between synthesized and test images, we need to bring the poses of the test images into the method’s own units and reference

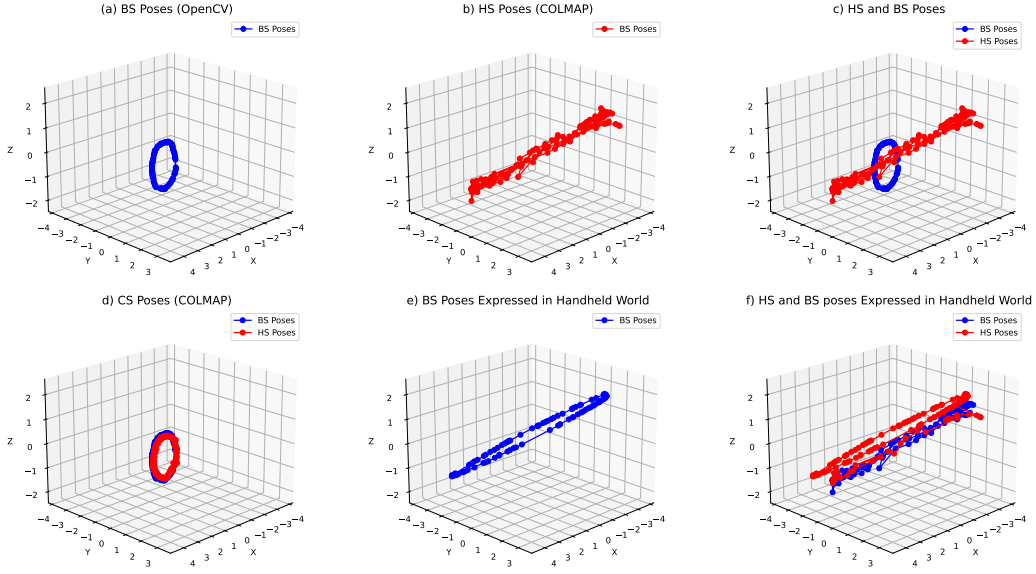


Figure 6.3: **Illustration of the pose alignment procedure.** (a) Camera poses estimated from the BS using a ChArUco board, expressed in metric scale with a world coordinate system defined by the calibration board. (b) Camera poses estimated from the HS using COLMAP, with an arbitrary scale and coordinate frame. (c) Overlay of (a) and (b), illustrating the misalignment due to differing coordinate systems and scales. (d) HS poses estimated by running COLMAP on the CS, with estimated intrinsics and BS poses injected, ensuring both sequences share a common coordinate system. This trajectory serves as a reference frame bridging (a) and (b). (e) The BS poses from (a) transformed with a Sim(3) alignment computed via the Kabsch–Umeyama [138] algorithm, and now fully expressed in the HS coordinate frame. (f) Overlay of (b) and the aligned BS poses from (e), confirming that the BS poses are now expressed in the HS world.

frame. Purposely, we deploy the Kabsch–Umeyama algorithm [138] to estimate a similarity transformation (SIM(3)) based on the 3D-3D correspondences provided by the camera centers estimated by the NVS and those estimated by COLMAP via the CS. Since this transformation (a 3D roto-translation along with a scale change) brings 3D coordinates from the reference frame of the CS into the method’s own frame, it can be applied to the camera poses of the BS to define the viewpoints to synthesize the test images in order to evaluate the NVS method.

An example reporting the different trajectories associated with the camera poses dealing with the BS, HS, CS and, finally, the aligned BS as reported in Fig. 6.3. In practice, given the camera poses estimated from the HS, the evaluation of a NVS method requires a simple two-step pose alignment procedure: i) compute

the SIM(3) transformation from the CS to the method’s own reference frame and ii) apply the transformation to the test (i.e. BS poses). This alignment procedure makes it straightforward for users to transform the BS poses into their own estimated coordinate systems. Indeed, as part of our evaluation suite, we provide a script that, given the estimated HS poses, returns the aligned test poses. Finally, we point out that along with the HS images, we also provide the associated segmentation masks obtained via Grounded SAM2 [137]. Hence, users may choose to rely on our pipeline to mask out the background or develop their own, or, perhaps, even to estimate poses on the original HS images, without performing any explicit foreground segmentation step.

Metrics. Following standard practice in NVS, we evaluate the methods applied on our dataset using Peak Signal-to-Noise Ratio (PSNR) [139], Structural Similarity Index Measure (SSIM) [140], and Learned Perceptual Image Patch Similarity (LPIPS) [141]. To assess the rendering quality in our object-centric setting, we leverage the segmentation masks generated with Grounding-SAM2 and category-specific prompts. In particular, we define two complementary evaluation modes:

- **Foreground Evaluation:** For PSNR, background pixels are masked out from the ground-truth images (i.e. those belonging to the BS) and the metric is computed solely within the object region. For SSIM and LPIPS, both the rendered and ground-truth images are cropped to tight bounding boxes defined by the foreground masks, focusing the evaluation on the object. Indeed, had the metrics been computed on whole images, the contribution of large background areas may have dominated the measured values, whereas the goal of our benchmark concerns the ability to accurately capture the full appearance of an object while it is being manipulated in front of a camera.
- **Background Evaluation:** The foreground masks are inverted to select background regions, and all metrics (PSNR, SSIM, LPIPS) are computed exclusively on those areas. This is motivated by the fact that methods may produce artifacts in the background, which would not be penalized by the Foreground Evaluation. To ensure fair comparison across methods, we normalize the background color used in the evaluation. Indeed, different NVS approaches may render backgrounds via different colors (e.g. white, black or some gray), which skews the metrics if a fixed background color is chosen for the ground-truth images. To address this, we explicitly set the background color of the ground-truth images to match that used by the specific NVS method under evaluation. This establishes a consistent evaluation protocol for future work. In particular, researchers willing to evaluate their method(s) will have to input the adopted background color into an evaluation script

provided with our benchmark, and the script will modify the background color of the ground-truth images accordingly. We also point out that PSNR goes to infinity when comparing two identical images. This might occur in our benchmark because, as discussed above, background pixels take the same color in rendered and ground-truth images. In particular, a method may render a background identical to the ground truth if pose estimation is nearly flawless or it fails to render any foreground at all, though in the latter circumstance the overall assessment will be penalized by the Foreground Evaluation. Due to the above eventualities in the Background Evaluation, we cap the PSNR to a very high value (i.e. 100) to enable score aggregation across images and objects.

6.2.2 Baseline Methods

Pose Estimation. For pose estimation from the HS, we employ two state-of-the-art approaches: a classical SfM pipeline (COLMAP [135]) and a recent ViT-based model (VGGT [96]). To estimate poses by these methods we input masked images containing only the foreground object extracted by Grounded SAM2 [137], as already discussed with reference to pose estimation from the CS.

For SfM-based pose estimation, we adopt COLMAP [135]. To estimate poses using COLMAP, all runs employ the sequential matcher, which is well-suited for video-style data. The sequential matcher relies on an overlap parameter, which determines how many neighboring frames are considered for feature matching. We tune this overlap individually for each object to account for variations in motion, frame density, and scene characteristics.

For ViT-based [94] pose estimation, we adapt VGGT [96], the current state-of-the-art in this field. Despite its general effectiveness, we found that VGGT struggles with sequences containing rather large uniformly colored backgrounds, which are common in our handheld-capture settings. To address this, we introduce several modifications to ensure that the model attends primarily to foreground object features. First, we compute the union of all foreground masks in the given HS and crop images accordingly, updating the camera’s principal point after cropping. The cropped images are then processed by VGGT’s standard pipeline, including square padding, resizing, and feature extraction via DINO-V2 [45]. Next, we modify the input to the first VGGT attention block by pruning the tokens corresponding to background regions. To ensure that only foreground is considered for computation of global and frame level attention. Additionally, to assure every frame provides the same number of tokens to the attention layers, we identify the minimum number of foreground tokens present in the sequence and use this count consistently for all frames. For each image, we retain only the first n tokens, while preserving their original positional encodings. These tokens are then passed

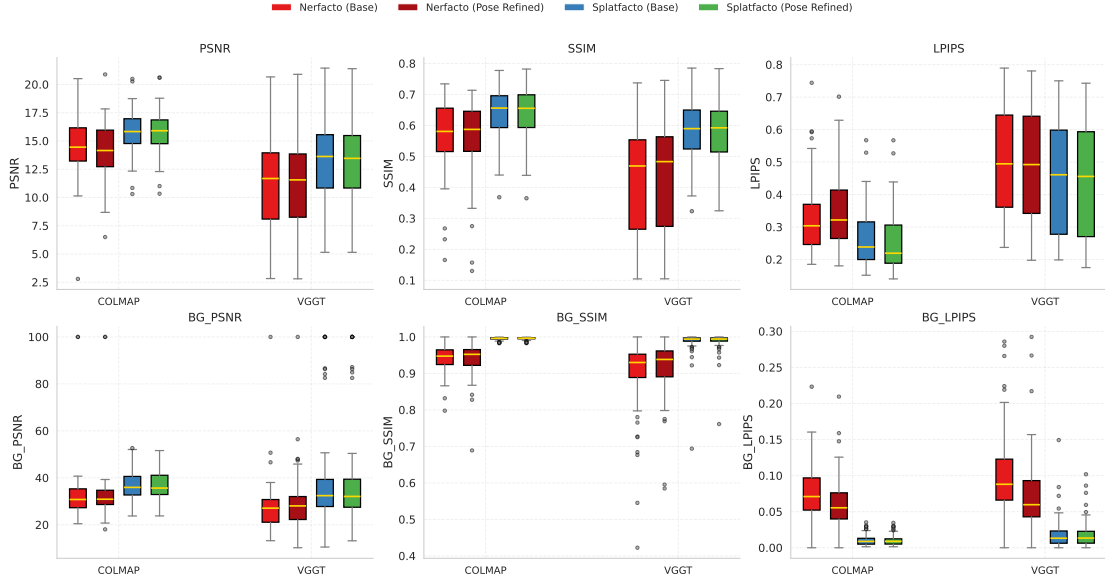


Figure 6.4: **NVS results.** Top and bottom rows report metrics dealing with Foreground and Background Evaluation, respectively.

through VGGT’s alternating attention layers. Finally, the pruned background tokens are reintroduced by scattering zeros into their original positions, ensuring compatibility with DPT [97] heads. For refinement, we leverage VGGT’s predicted point maps to perform bundle adjustment on the estimated poses, plugging in our calibrated (see Section 6.1), and later adjusted intrinsics in place of those estimated by the model.

Novel View Synthesis. For novel view synthesis, we establish the baselines considered in our benchmark by deploying the Nerfstudio [142] library, using Nerfacto [142], a method based on NeRF [4], and the Gaussian Splatting [93] method Splatfacto. For both Nerfacto and Splatfacto we train models based on poses estimated by COLMAP and VGGT. During training, we employ alpha transparency carving to better capture fine details and preserve clean object boundaries, which is critical for evaluating reconstruction quality on our object-centric dataset. In addition, we experiment with enabling pose optimization during training to assess its impact on performance.

The values of the key hyperparameters concerning the considered pose estimation and novel view synthesis methods are reported in Appendix B.1.

6.3 Benchmark Results

In Fig. 6.4, we present results on our dataset NVS-HO, where camera poses were estimated using COLMAP [135] and VGGT [96] and subsequently used to train Nerfacto [142] and Splatfacto [5], both with and without pose optimization. We report the results as box plots to capture the distribution of the metrics across objects, due to the large number of objects and the large variance in the results. The same results in tabular form are reported in the Appendix B.2. The most evident result is that COLMAP outperforms VGGT across all metrics, exhibiting both better median values as well as less variance in the metrics. Hence, for reliable pose estimation in handheld scenarios traditional SfM seems still preferable. Another interesting result is that Splatfacto consistently outperforms Nerfacto across all evaluation metrics. Yet, both the median PSNR and SSIM are low in absolute terms, e.g. the median PSNR of Splatfacto with COLMAP poses is about 16. Moreover, activating pose refinement at training time brings only marginal improvements. Since Nerfacto and Splatfacto usually deliver very good results in more controlled setups where poses are known or can be estimated reliably, these findings suggest that more accurate pose initialization is required to improve metrics in our benchmark. This trend is also evident in the qualitative examples of Fig. 6.5, where we can see that, although COLMAP yields renderings that better align with object boundaries compared to VGGT-based reconstructions, they both exhibit misalignments between rendered content and ground-truth mask edges. In Appendix B.2, we provide detailed results across few objects.

6.4 Conclusions

In this chapter, we introduced a benchmark collected during my PhD which is dedicated to NVS of handheld objects in unconstrained real-world conditions using only RGB inputs. This work has been documented in a paper that is currently under review for publication in a computer vision conference. Unlike prior datasets, our benchmark does not rely on pre-scanned 3D object models or MANO hand models. Our experiments with state-of-the-art methods reveal that current pipelines struggle under handheld scenarios, highlighting challenges such as hand-induced partial occlusions and texture-less surfaces.

We believe that NVS-HO establishes a valuable foundation for the community, offering both a practical benchmark and a standardized evaluation protocol to drive future research in handheld object reconstruction and novel view synthesis.



Figure 6.5: **Qualitative NVS results.** Red edges denote the ground-truth object masks. Renders based on VGGT poses show visible misalignment with the mask due to pose inaccuracies, whereas COLMAP poses produce better alignment and higher-quality novel views.

Chapter 7

Concluding Remarks

In this thesis, we have investigated neural image synthesis for industrial computer vision applications. The first part focuses on image synthesis for industrial anomaly inspection and the second on the novel view synthesis of handheld objects.

In Part I, we demonstrated that diffusion models can generate realistic synthetic anomaly images that significantly enhance downstream inspection performance. By synthesizing specific anomaly types, this approach adopts a *Closed World Assumption*, wherein the defect manifold is explicitly modeled. This stands in contrast to pure anomaly detection, which typically operates under an *Open World Assumption* by modeling only nominal conditions and flagging any deviation as a generic outlier. By explicitly defining these defect classes for training, our framework shifts the objective from simple novelty detection to a more robust classification task. To achieve this, we first introduce AnomalyControl [44], followed by VarianceGenerator [46], which addresses the former’s limitations.

The main issue found with AnomalyControl [44] was the misalignment between nominal samples and anomaly masks. VarianceGenerator [46] resolves this by generating anomalies directly on the training set, ensuring that masks are always aligned with foregrounds. Additionally, we adapt ControlNet inpainting [8] to learn how to inpaint nominal parts, which further improves downstream anomaly classification performance.

Despite these advances, these approaches continue to have limitations. First, diffusion model must be trained separately for each object. Second, both methods rely heavily on prompt engineering. A potential solution is to train a single diffusion model across all objects and learn prompts into a dictionary of diffusion models, similar to Textual Inversion [6]. Third, the DINO-V2 [45] based filtering strategy used to discard unrealistic images or those with missing defects could be further refined to select the most realistic images, which would likely improve the performance on downstream anomaly inspection tasks.

In Chapter 6, we introduced NVS-HO, a benchmark for novel view synthesis

of handheld objects. This includes a new dataset containing RGB sequences of 67 common objects. Each object is recorded in two sequences: a Handheld Sequence that captures natural hand-object interactions and is intended for training, and a Board Sequence equipped with ground-truth poses for evaluation. We also proposed a method-agnostic protocol to evaluate viewpoints rendered by novel view synthesis techniques. In this project, we adapt COLMAP [135] and VGGT [96] for pose estimation, and Nerfacto [142] and Splatfacto [5] for novel view synthesis. Our results highlight the challenges of pose estimation and novel view synthesis for handheld objects using RGB-only images.

A key limitation of current pose estimation pipelines is their poor performance on texture-less objects. Although VGGT [96] represents an improvement in recent state-of-the-art models, it struggles when objects have black or uniform backgrounds. To address this, we introduced token pruning in VGGT’s attention layers; however, results still underperform compared to COLMAP. We believe pose estimation accuracy can be further improved, especially since NeRFs and Gaussian Splats already achieve state-of-the-art performance in novel view synthesis.

Another promising direction is exploring feedforward novel view synthesis using LVSM, as adapted by VGGT [96]. Currently, this approach predicts only a fifth view from four input images, which makes it challenging to adapt to our dataset with 100+ images. Future work could investigate scaling feedforward NVS to handle larger image sequences while maintaining accurate novel view synthesis.

Part III

Appendices

Appendix A

FEW SHOT ANOMALY GENERATION

A.1 Introduction

In this Appendix we will provide more details about AnomalyControl as well as additional results. Appendix A.2 gives an in-depth description of our training procedure and chosen hyperparameters. Appendix A.3 shows examples of generated images to ease the comparison with similar techniques, e.g., AnomalyDiffusion [52]. Finally, Appendix A.4 reports additional results on downstream anomaly inspection tasks.

A.2 Implementation Details

Data Augmentation For ControlNet Training. When training ControlNet [8], to avoid overfitting we deploy a data augmentation pipeline that includes flipping, rotation and cropping of both the real defective images together and their associated defect masks. While cropping, we store the minimum and maximum locations of the ground truth mask to ensure that the defect does not fall outside the image during training.

More precisely, any training image is augmented with (50%) probability, and, if augmented, the following random transformations are applied :

- Vertical Flip (50% probability)
- Horizontal Flip (50% probability)
- Random Rotation: ± 20 degrees (50% probability)
- Crop

Anomaly Generation. When we generate new anomaly samples, we randomly select good images and defect masks using a uniform distribution. We also apply the transformations described in the previous paragraph, except for random cropping of the masks, to generate more varied synthetic anomalies. Additionally, we use 20 DDIM steps and a classifier-free guidance scale of 7.5 to generate the images.

Hyperparameters For ControlNet Training. We train a different ControlNet model for each MVTec-AD [61] object. We use a learning rate of 0.00001, a batch size of 1, gradient accumulation of 8, and train for 40,000 steps.

Classifier Hyperparameters. The linear classifier head is placed on top of a ResNet-34 [65] backbone and we train a different model for each MVTec-AD object. We use a learning rate of 0.00001, a batch size of 8 and train for 50 epochs. Furthermore, we define a learning rate scheduling where we decrease the learning rate by 0.1 at epochs 40 and 45.

UNet Hyperparameters. We train a UNet [9] from scratch for each MVTec-AD object for each MVTec-AD object to perform anomaly detection and localization. We use a learning rate of 0.0001, a batch size of 8, and train for 200 epochs. Also, we define a learning rate scheduling where we decrease the learning rate by 0.1 at epochs 160 and 180.

Prompt Engineering. In Table A.1, we sort the anomaly categories for each MVTec-Ad object in alphabetical order and show the prompts that we have designed and used for each anomaly class. It must be noted that the first part of any prompt follows the template `a photo of [*]`, where `[*]` is replaced by an anomaly-specific prompt.

Image Filtering. To detect low-quality generated images, we compute DINO-v2 [45] embeddings using the *small* variant. Since DINO-v2 requires an input size multiple of 14, we pad the images by 5 pixels in both dimensions, such that the resultant size becomes 266×266 .

A.3 Qualitative Comparison

Detailed Qualitative Results. For a fair qualitative comparison, we randomly sampled images from the MVTec-AD [61] training split as well as among the images generated by AnomalyDiffusion [52] and our own method. We present detailed qualitative results in Fig. A.1 through A.15.

Limitations. In Chapter 3 we discussed limitations of our methods. Here, some examples of failures are shown in Fig. A.16.

A.3.1 Anomaly Generation Quality

IC-LPIPS. As suggested by [11, 52], we compute the IC-LPIPS [69] on a filtered set of 1000 images for each anomaly type. We complement the Table from AnomalyDiffusion and add our column in Table A.2. However, we believe that IC-LPIPS is reliable metric for measuring the diversity rather the quality, i.e. realism, of the generated anomaly images. For instance, for the leather object, the images generated by AnomalyDiffusion appear unrealistic, as shown in Fig. A.7, despite the method having a large IC-LPIPS score on this object.

Kernel Inception Distance(KID). To compute the Kernel Inception Distance (KID) [144], we follow the approach of DFMGAN [11]. We compute the KID on the images provided by AnomalyDiffusion [52] and on our 500 randomly sampled images used for downstream anomaly inspection tasks. The results are reported in Table A.3. The results show that we achieve a better KID score across all the MVTec-AD [61] objects.

A.4 Additional Results on Anomaly Inspection

Anomaly Classification. Table A.4 reports downstream anomaly classification performance following the *proposed test protocol* or the *baseline test protocol* using images provided by AnomalyDiffusion and ours 500 randomly sampled images. It can be noted that images generated by our method consistently improve downstream anomaly classification with both the *proposed test protocol* or the *baseline test protocol*.

Anomaly Detection and Localization. Table A.6 and Table A.7 report the results on, respectively, downstream anomaly detection and localization following the *proposed test protocol* or the *baseline test protocol* using images provided by AnomalyDiffusion and ours 500 randomly sampled images. It can be the noted that both methods show similar performance on these tasks.

Additional Comparison on Anomaly Localization. Here, we provide additional experiments on pixel-level anomaly localization. We complement the tables form AnomalyDiffusion [52] with our results following the *baseline test protocol*. We report the AUC and AP in Table A.7 and F₁-max in Table A.9.

MVTec-Ad Object	Anomaly-specific prompts
'bottle'	'a large breakage', 'a with small breakage', 'a contamination'
'cable'	'a bent wire', 'a cable swap', 'multiple defects', 'a cut inner insulation', 'a cut outer insulation', 'a missing cable', 'a missing wire', 'a poke insulation'
'capsule'	'a crack', 'a faulty imprint', 'a scratch', 'a squeeze', 'a poke'
'carpet'	'a color stain', 'a cut', 'a hole', 'a metal contamination', 'a thread'
'grid'	'a bent part', 'a broken part', 'a glue', 'a metal contamination', 'a thread'
'hazelnut'	'a crack', 'a cut', 'a hole', 'a print'
'leather'	'a color stain', 'a cut', 'a fold', 'a glue', 'a poke'
'metal nut'	'a bent part', 'a color stain', 'a flipped metal nut', 'a scratch'
'pill'	'a color stain', 'a contamination', 'multiple detects', 'a crack', 'a faulty imprint', 'a different type of pill', 'a scratch'
'screw'	'a broken tip', 'a scratched head', 'a scratched neck', 'a damaged thread', 'a damaged thread'
'tile'	'a crack', 'a glue', 'a gray stain', 'a oiled surface', 'a rough surface'
'toothbrush'	'multiple defects'
'transistor'	'a bent lead', 'a cut lead', 'a damaged case', 'a misplaced case'
'wood'	'a color stain', 'multiple defects', 'a hole', 'a liquid', 'a scratch'
'zipper'	'broken teeth', 'multiple defects', 'a damaged fabric', 'a damaged fabric', 'a rough surface', 'split teeth', 'squeezed teeth'

Table A.1: Anomaly Specific prompts for MVTEC-Ad [61] dataset objects.

Object	DiffAug [68]	CDC [69]	Crop-Paste [49]	SDGAN [51]	Defect-GAN [143]	DFMGAN [11]	Anomaly Diffusion [52]	Ours
bottle	0.03	0.04	0.04	0.06	0.07	0.12	0.19	<u>0.15</u>
cable	0.07	0.19	0.25	0.19	0.22	0.25	0.41	<u>0.39</u>
capsule	0.03	0.06	0.05	0.03	0.04	0.11	0.21	<u>0.18</u>
carpet	0.06	0.03	0.11	0.11	0.12	0.13	0.24	<u>0.22</u>
grid	0.06	0.07	0.12	0.10	0.12	0.13	<u>0.44</u>	0.45
hazel	0.05	0.05	0.21	0.16	0.19	0.24	<u>0.31</u>	0.32
leather	0.06	0.07	0.14	0.12	0.14	0.17	0.41	<u>0.31</u>
metalnug	0.29	0.04	0.15	0.28	0.30	0.32	0.30	<u>0.28</u>
pill	0.05	0.06	0.11	0.07	0.10	0.16	0.26	<u>0.25</u>
screw	0.10	0.11	0.16	0.10	0.12	0.14	0.30	<u>0.23</u>
tile	0.09	0.12	0.20	0.21	0.22	0.22	0.55	<u>0.49</u>
toothbrush	0.06	0.06	0.08	0.03	0.03	0.18	0.21	<u>0.18</u>
transistor	0.05	0.13	0.15	0.13	0.13	0.25	0.34	<u>0.29</u>
wood	0.30	0.03	0.23	0.25	0.29	<u>0.35</u>	0.37	0.37
zipper	0.05	0.05	0.11	0.10	0.10	0.27	0.25	<u>0.23</u>
Avg	0.09	0.07	0.14	0.13	0.15	0.20	0.32	<u>0.29</u>

Table A.2: Comparison on IC-LPIPS [69] on MVTec-Ad dataset. For each row best in bold, second best is underlined.

Object	AnomalyDiffusion	Ours
bottle	139.66	49.220
cable	160.40	40.690
capsule	74.950	14.300
carpet	143.81	143.89
grid	66.310	44.090
hazelnut	40.090	13.380
leather	222.51	104.20
metal nut	179.67	75.070
pill	81.100	31.020
screw	20.450	0.9100
tile	339.77	210.77
toothbrush	78.400	43.990
transistor	122.02	3.6000
wood	184.53	32.970
zipper	142.11	98.850
Avg	133.02	61.260

Table A.3: Comparison on KID with AnomalyDiffusion[52]

Object	Validation on Anomaly Images (Test Split)		Validation on Anomaly Images (Train Split)		Last Checkpoint	
	AnomalyDiffusion	Ours	AnomalyDiffusion	Ours	AnomalyDiffusion	Ours
bottle	92.36	85.80	76.97	73.95	80.92	79.76
cable	73.28	72.81	71.87	65.46	69.52	69.37
capsule	45.47	66.93	39.60	61.46	39.06	60.79
carpet	64.35	58.22	56.61	51.12	56.39	51.44
grid	43.50	55.00	32.80	43.00	39.75	38.75
hazel nut	75.21	92.40	70.83	84.58	73.22	80.20
leather	62.22	74.44	48.09	65.55	47.74	67.61
metal nut	85.94	71.40	75.93	65.00	75.76	66.36
pill	54.37	64.89	49.79	57.39	51.14	63.08
screw	41.35	33.40	36.04	30.00	39.74	29.25
tile	91.05	74.21	85.08	71.22	85.96	70.70
transistor	66.07	85.71	58.92	75.00	46.40	79.64
wood	69.76	78.80	67.14	69.04	64.91	73.52
zipper	64.63	70.73	61.70	63.29	58.51	66.92
Average	66.40	70.30	59.39	62.60	59.22	64.89

Table A.4: Anomaly Classification accuracy - First Column: *baseline test protocol*, Second and Third column: *proposed test protocol*. For each row, best result in bold.

Object	Validation on Original Training Set								Anomaly Localization on Last Check Point							
	AnomalyDiffusion				Ours				AnomalyDiffusion				Ours			
	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO	AUC	AP	F ₁ -max	PRO
bottle	98.8	84.5	78.6	92.4	98.6	88.3	81.5	91.6	98.8	83.3	77.6	92.3	98.1	85.8	78.5	90.1
cable	98.2	83.1	74.8	92.2	94.7	75.4	70.5	90.1	98.0	82.8	75.2	92.77	92.9	74.4	70.6	88.9
capsule	94.6	32.8	41.0	72.3	91.3	29.9	39.0	89.0	95.9	28.9	40.8	80.8	88.9	30.5	39.7	88.6
carpet	97.7	73.6	67.6	85.0	97.4	79.1	73.3	87.9	79.2	3.2	9.4	26.1	95.4	74.5	71.3	84.2
grid	85.5	20.0	28.4	71.4	97.3	46.4	51.5	92.6	85.4	19.0	27.3	70.7	96.2	47.2	52.3	91.4
hazelnut	98.9	81.9	76.2	92.7	98.0	82.9	76.2	95.9	98.6	75.8	70.1	91.2	96.2	78.9	73.6	95.7
leather	99.8	75.5	69.8	98.0	98.6	75.0	69.7	97.8	99.7	72.5	68.6	97.6	98.0	73.4	69.1	97.5
metal nut	99.5	97.8	92.7	91.7	99.4	97.7	92.9	91.3	99.2	96.7	91.5	89.1	99.3	96.0	90.2	92.0
pill	99.7	95.6	89.1	92.7	99.1	88.8	80.9	96.9	99.7	95.0	88.4	91.2	99.2	91.5	83.9	97.4
screw	91.8	45.0	49.1	80.4	86.5	14.0	24.2	61.6	80.2	43.9	49.3	65.8	84.3	13.9	23.7	60.0
tile	99.4	94.6	87.2	95.3	99.5	95.3	88.1	96.2	99.1	92.2	83.6	95.2	99.4	94.7	87.5	96.4
toothbrush	96.4	55.3	58.6	83.3	78.3	44.4	51.1	69.8	93.7	49.9	55.0	82.4	76.8	45.7	52.9	66.7
transistor	99.3	91.8	84.7	89.3	98.8	86.9	83.2	92.7	99.1	91.2	84.4	92.7	99.1	88.9	84.3	93.6
wood	97.6	70.5	67.0	89.5	97.4	81.6	74.9	95.2	96.0	58.5	59.9	82.6	96.8	80.0	74.4	93.6
zipper	99.2	84.7	78.2	96.0	99.3	85.7	79.0	96.3	99.0	83.6	76.3	95.8	97.0	79.0	74.4	93.4
Avg	97.1	72.4	69.5	88.2	95.6	71.4	69.1	89.7	94.8	65.1	63.8	83.1	94.5	70.3	68.4	88.6

Table A.5: Results on pixel-level anomaly localization - First Column: *baseline test protocol*, Second and Third column: *proposed test protocol*.

Object	Validation on Anomaly Images (Test Split)						Validation on Anomaly Images (Train Split)						Last Check Point					
	AnomalyDiffusion			Ours			AnomalyDiffusion			Ours			AnomalyDiffusion			Ours		
	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max
bottle	99.9	99.9	99.4	99.9	99.9	99.4	99.6	99.9	99.3	99.9	99.9	99.3	99.8	99.9	99.0	99.0	99.6	98.5
cable	99.5	99.6	97.4	98.1	98.7	95.7	99.2	99.4	97.5	96.8	97.9	94.6	99.5	99.6	97.7	96.9	98.1	95.3
capsule	92.1	97.6	92.1	95.3	98.7	94.8	80.1	93.5	88.6	92.9	97.8	93.0	90.8	97.4	90.5	94.3	98.3	94.4
carpet	89.0	95.4	88.9	96.4	98.6	94.6	89.0	95.3	89.0	92.7	97.2	91.8	58.8	67.9	88.6	91.6	96.8	91.5
grid	97.3	98.8	96.1	99.6	99.8	98.7	93.9	97.5	93.2	98.5	99.4	98.5	93.8	97.2	91.9	98.3	99.4	98.6
hazelnut	99.8	99.8	98.4	99.8	99.8	98.7	99.7	99.8	99.0	100	100	99.9	98.0	98.7	95.5	100	100	100
leather	99.9	100	99.0	100	100	99.8	99.7	99.9	98.7	100	100	99.8	99.7	99.9	98.8	100	100	99.7
metal nut	100	100	99.7	100	100	99.8	99.7	99.9	99.1	99.5	99.9	99.2	98.2	99.5	98.2	100	100	99.8
pill	96.0	99.0	95.2	99.4	99.8	98.4	96.6	99.2	96.0	98.4	99.6	97.4	96.2	99.1	95.1	98.1	99.5	97.4
screw	93.9	97.1	91.3	91.9	95.8	91.5	82.5	91.8	83.5	72.4	85.8	80.9	79.0	91.0	83.4	65.7	82.1	79.9
tile	100	100	100	99.9	100	99.7	100	100	99.9	99.8	99.9	99.3	100	100	99.8	99.8	99.9	99.2
toothbrush	100	100	99.8	99.0	96.5	99.5	99.5	99.7	98.8	95.1	97.5	93.6	99.8	99.9	99.5	97.5	98.6	95.6
transistor	99.9	99.8	98.8	100	99.9	99.3	98.4	98.1	95.8	99.6	99.4	97.9	99.8	99.7	98.9	100	99.9	99.5
wood	99.4	99.7	98.5	99.8	99.9	98.9	98.2	99.2	96.2	99.9	99.9	99.1	97.8	99.9	95.6	99.8	99.9	99.0
zipper	100	100	99.9	100	100	99.9	100	100	99.9	100	100	99.9	99.9	100	99.2	99.9	100	99.6
Avg	97.8	99.1	97.0	98.6	99.3	97.7	95.7	98.2	95.6	96.4	98.3	96.3	94.1	96.6	95.4	96.1	98.1	96.5

Table A.6: Results of image-level anomaly detection - First Column: *baseline test protocol*, Second and Third column: *proposed test protocol*.

Object	Validation on Anomaly Images (Test Split)						Validation on Anomaly Images (Train Split)						Last Check Point					
	AnomalyDiffusion			Ours			AnomalyDiffusion			Ours			AnomalyDiffusion			Ours		
	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max	AUC	AP	F ₁ -max
bottle	99.1	86.9	80.0	92.8	98.6	88.2	80.5	92.1	98.8	84.5	78.6	92.4	98.6	88.3	81.5	91.6	98.8	83.3
cable	97.9	82.9	75.1	93.0	95.8	76.2	71.0	90.3	98.2	83.1	74.8	92.2	94.7	75.4	70.5	90.1	98.0	82.8
capsule	96.0	30.9	40.6	77.3	92.8	33.5	41.7	89.9	94.6	32.8	41.0	72.3	91.3	29.9	39.0	89.0	95.9	28.9
carpet	97.8	74.3	68.6	85.6	98.4	82.0	74.8	90.3	97.7	73.6	67.6	85.0	97.4	79.1	73.3	87.9	79.2	3.2
grid	91.6	27.5	33.3	81.5	98.7	50.6	53.9	95.4	85.5	20.0	28.4	71.4	97.3	46.4	51.5	92.6	85.4	19.0
hazelnut	99.2	86.4	80.2	92.9	99.6	93.9	86.8	96.5	98.9	81.9	76.2	92.7	98.0	82.9	76.2	95.9	98.6	75.8
leather	99.8	79.4	72.1	98.2	99.2	77.4	71.2	97.9	99.8	75.5	69.8	98.0	98.6	75.0	69.7	97.8	99.7	72.5
metal nut	99.7	98.4	93.6	93.2	99.6	98.0	92.9	93.1	99.5	97.8	92.7	91.7	99.4	97.7	92.9	91.3	99.2	96.7
pill	99.7	95.4	88.9	90.4	99.5	92.7	85.3	97.3	99.7	95.6	89.1	92.7	99.1	88.8	80.9	96.9	99.7	95.0
screw	91.0	44.5	48.5	79.1	90.4	15.6	24.8	70.9	91.8	45.0	49.1	80.4	86.5	14.0	24.2	61.6	80.2	43.9
tile	99.5	94.9	87.4	96.0	99.6	96.1	89.2	97.1	99.4	94.6	87.2	95.3	99.5	95.3	88.1	96.2	99.1	92.2
toothbrush	97.5	59.1	60.6	85.7	88.3	49.5	53.7	75.0	96.4	55.3	58.6	83.3	78.3	44.4	51.1	69.8	93.7	49.9
transistor	99.6	93.7	86.4	93.5	99.4	91.8	86.3	94.4	99.3	91.8	84.7	89.3	98.8	86.9	83.2	92.7	99.1	91.2
wood	97.9	78.4	70.7	92.1	98.8	86.0	77.4	95.6	97.6	70.5	67.0	89.5	97.4	81.6	74.9	95.2	96.0	58.5
zipper	99.4	86.5	79.2	96.6	99.1	84.0	77.1	95.9	99.2	84.7	78.2	96.0	99.3	85.7	79.0	96.3	99.0	83.6
Avg	97.7	74.6	71.0	89.9	97.2	74.4	71.1	91.4	97.1	72.4	69.5	88.2	95.6	71.4	69.1	89.7	94.8	65.1

Table A.7: Results on pixel-level anomaly localization - First Column: *baseline test protocol*, Second and Third column: *proposed test protocol*.

Object	Unsupervised								Supervised				
	KDAD [23]	CFLOW [17]	DRAEM [34]	SSPCAB [33]	CFA [145]	RD4AD [16]	PatchCore [47]		DevNet [146]	DRA [146]	PRN [55]	Anomaly Diffusion [52]	Ours
bottle	94.7/50.5	98.8/49.9	99.1/88.5	98.9/88.6	98.9/50.9	98.8/51.0	97.6/75.0		96.7/67.9	91.7/41.5	99.4/92.3	99.3/94.1	98.8/88.6
cable	79.2/11.6	98.9/72.6	94.8/61.4	93.1/52.1	98.4/79.8	98.8/77.0	96.8/65.9		97.9/67.6	86.1/34.8	98.8/78.9	99.2/90.8	96.0/76.2
capsule	96.3/ 9.9	99.5/64.0	97.6/47.9	90.4/48.7	98.9/71.1	99.0/60.5	98.6/46.6		91.1/46.6	88.5/11.0	98.5/62.2	98.8/57.2	95.1/39.0
carpet	91.5/45.8	99.7/67.0	96.3/62.5	92.3/49.1	99.1/47.7	99.4/46.0	98.7/65.0		94.6/19.6	98.2/54.0	99.0/82.0	98.6/81.2	98.5/81.8
grid	89.0/ 7.6	99.1/87.8	99.5/53.2	99.6/58.2	98.6/82.9	98.0/75.4	97.2/23.6		90.2/44.9	86.2/28.6	98.4/45.7	98.3/52.9	98.6/49.5
hazelnut	95.0/34.2	97.9/67.2	99.5/88.1	99.6/94.5	98.5/80.2	94.2/57.2	97.6/55.2		76.9/46.8	88.8/20.3	99.7/93.8	99.8/96.5	99.6/93.2
leather	98.2/26.7	99.2/91.1	98.8/68.5	97.2/60.3	96.2/60.9	96.6/53.5	98.9/43.4		94.3/66.2	97.2/ 5.1	99.7/69.7	99.8/79.6	99.2/77.4
metalnut	81.7/30.6	98.8/78.2	98.7/91.6	99.3/95.1	98.6/74.6	97.3/53.8	97.5/86.6		93.3/57.4	80.3/30.6	99.7/98.0	99.8/98.7	99.6/97.8
pill	90.1/23.1	98.9/60.3	97.7/44.8	96.5/48.1	98.8/67.9	98.4/58.1	97.0/75.9		98.9/79.9	79.6/22.1	99.5/91.3	99.8/97.0	99.5/92.7
screw	95.4/ 5.9	98.8/45.7	99.7/72.9	99.1/62.0	98.7/61.4	99.1/51.8	98.7/34.2		66.5/21.1	51.0/ 5.1	97.5/44.9	97.0/51.8	89.9/13.5
tile	78.6/26.7	98.0/86.7	99.4/96.4	99.2/96.3	98.6/92.6	97.4/78.2	94.9/56.0		88.7/63.9	91.0/54.4	99.6/96.5	99.2/93.9	99.5/95.6
toothbrush	95.6/20.0	99.1/56.9	97.3/49.2	97.5/38.9	98.4/61.7	99.0/63.1	97.6/37.1		96.3/52.4	74.5/ 4.8	99.6/78.1	99.1/76.5	90.0/53.5
transistor	76.0/25.9	98.8/40.6	92.2/56.0	85.3/36.5	98.6/82.9	99.6/50.3	91.8/66.7		55.2/ 4.4	79.3/11.2	98.4/85.6	99.3/92.6	99.4/92.2
wood	88.3/24.7	98.9/47.2	97.6/81.6	97.2/77.1	97.6/25.6	99.3/39.1	95.7/54.3		93.1/47.9	82.9/21.0	98.7/82.6	98.9/84.6	99.1/86.6
zipper	95.1/30.5	96.5/63.9	98.6/73.6	98.1/78.2	95.9/53.9	99.7/52.7	98.5/63.1		92.4/53.1	96.8/42.3	98.8/77.6	99.4/86.0	99.0/84.0
Average	89.6/24.9	98.7/65.3	97.7/69.0	96.2/65.5	98.3/66.3	98.3/57.8	97.1/56.6		86.4/49.3	84.8/25.7	99.0/78.6	99.1/81.4	97.4/74.8

Table A.8: Pixel-level anomaly localization results (AUROC/AP). "Ours" is a simple U-Net trained on our generated dataset using *best line test protocol*.

Object	Unsupervised							Supervised				
	KDAD [23]	CFLOW [17]	DREAM [34]	SSPCAB [33]	CFA [145]	RD4AD [16]	PatchCore [47]	DevNet [147]	DRA [146]	PRN [55]	Anomaly Diffusion [52]	Ours
bottle	50.9	9.5	83.0	80.3	75.9	82.1	78.6	64.6	53.5	71.3	87.3	80.9
cable	18.3	10.5	58.5	51.2	76.3	65.2	68.5	54.9	55.3	61.0	83.5	70.6
capsule	15.1	6.9	48.9	49.5	57.0	60.4	56.7	38.7	47.7	47.9	60.8	44.8
carpet	54.2	3.5	60.0	47.1	48.3	67.8	67.9	52.3	42.3	65.6	74.6	74.8
grid	10.9	3.2	56.3	58.4	32.2	59.9	49.1	42.9	50.1	58.9	54.6	53.5
hazelnut	37.5	3.9	80.6	88.9	61.4	70.0	68.1	22.4	47.2	68.2	90.6	86.0
leather	30.4	3.9	63.2	58.1	53.8	67.2	54.7	32.1	19.8	54.0	71.0	71.1
metanlut	34.2	30.7	84.4	87.8	87.1	77.0	86.0	65.2	64.6	87.1	94.0	92.6
pill	29.9	17.6	62.6	46.5	79.5	63.7	73.5	22.8	45.5	72.6	90.8	85.1
screw	8.3	0.9	66.9	63.8	37.8	58.7	47.2	14.8	0.7	49.8	50.9	23.2
tile	27.8	26.6	90.8	88.5	77.8	71.8	69.4	69.9	61.4	84.4	86.2	88.5
toothbursh	25.1	4.7	47.5	37.2	62.1	58.7	63.8	35.1	22.6	46.2	73.4	57.1
transistor	26.7	19.9	55.2	34.8	76.3	59.8	64.2	28.9	33.2	68.4	85.7	86.3
wood	25.2	10.0	75.1	68.7	48.6	61.3	60.3	51.9	49.9	67.4	74.5	77.7
zipper	26.8	4.5	68.2	73.7	65.8	69.4	70.0	45.6	56.9	73.7	79.2	77.2
average	28.1	10.4	66.7	62.3	61.4	66.2	65.2	42.8	43.4	64.7	76.4	71.3

Table A.9: Pixel-level anomaly localization results (F_1 -max). "Ours" is a simple U-Net trained on our generated dataset using *baseline test protocol*.

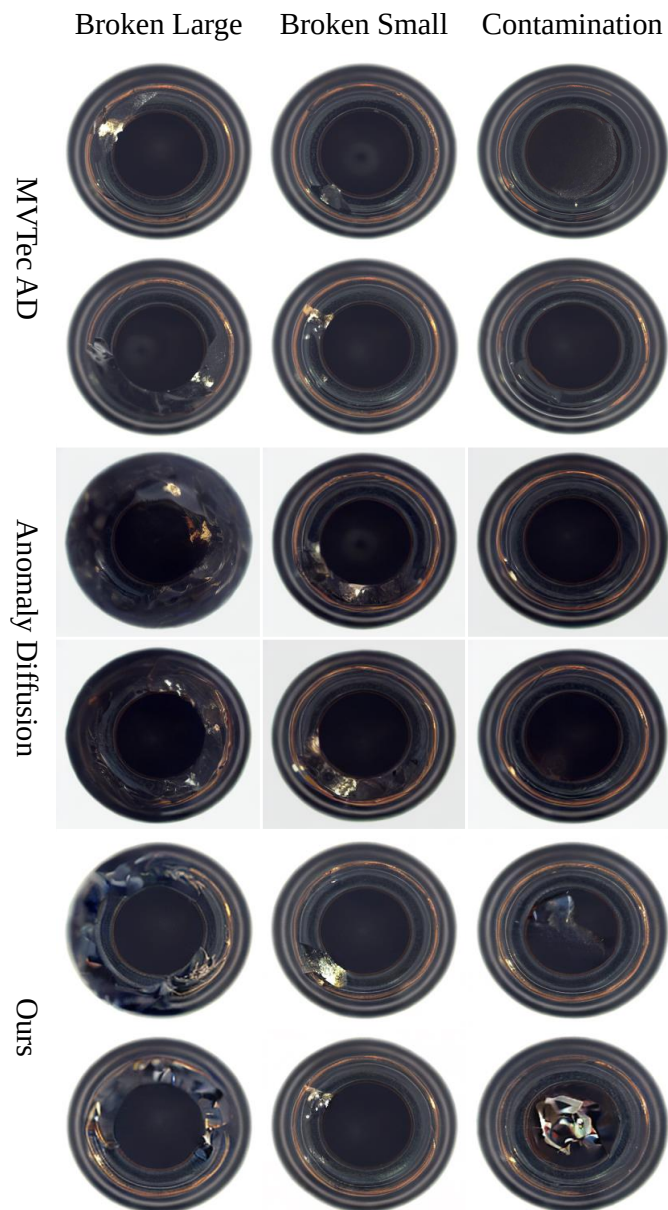


Figure A.1: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on bottle.

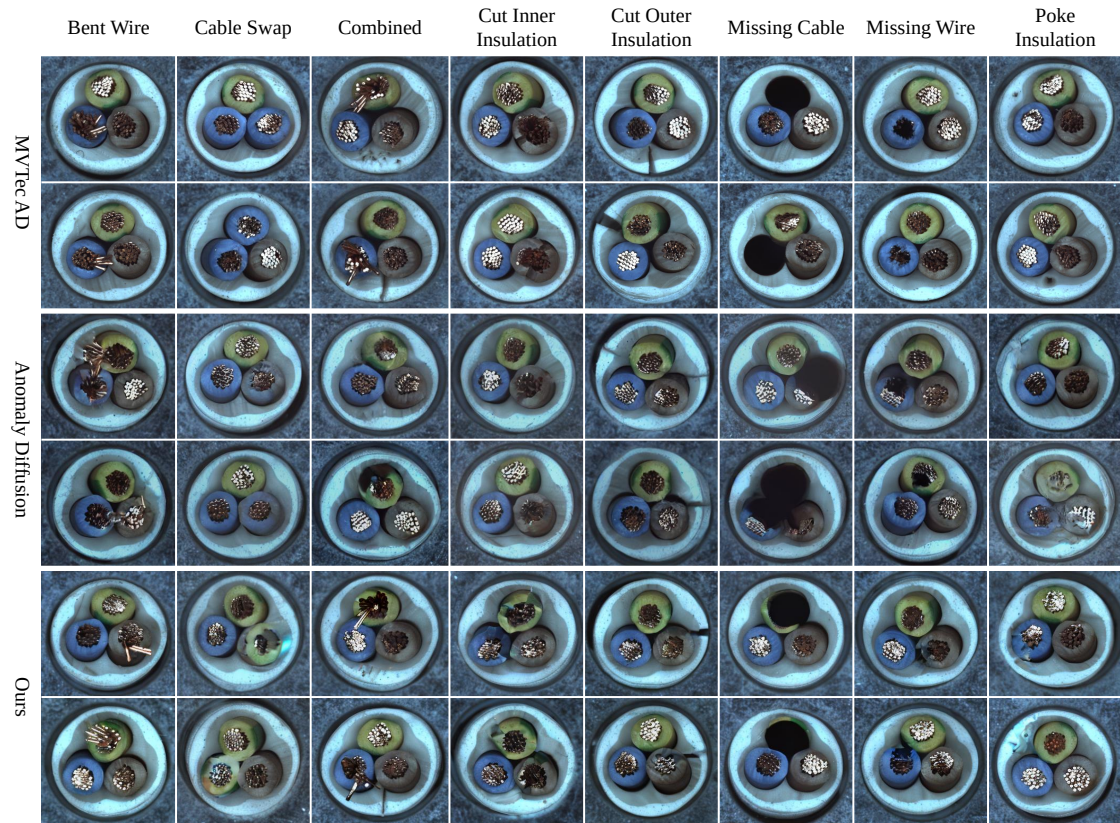


Figure A.2: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on cable.



Figure A.3: Examples of MVTec-Ad original training images and randomly sampled generated defect images by current AnomalyDiffusion [52] and our method on capsule.

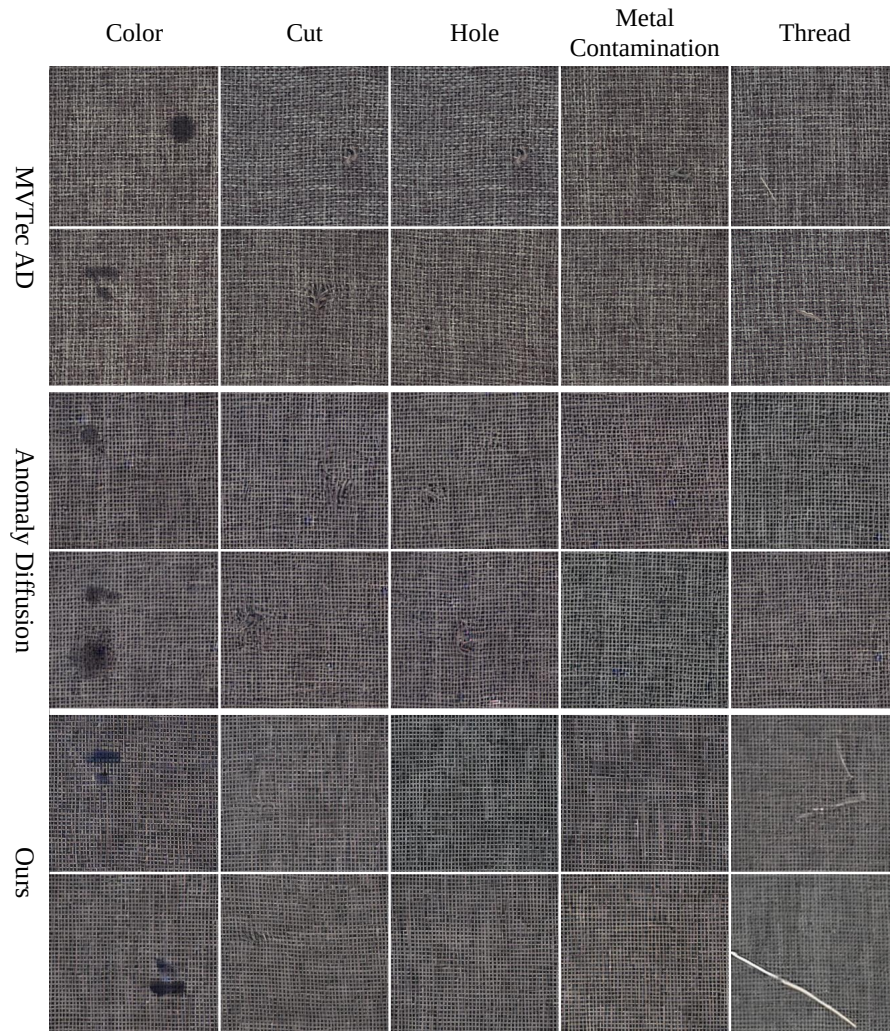


Figure A.4: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on carpet.

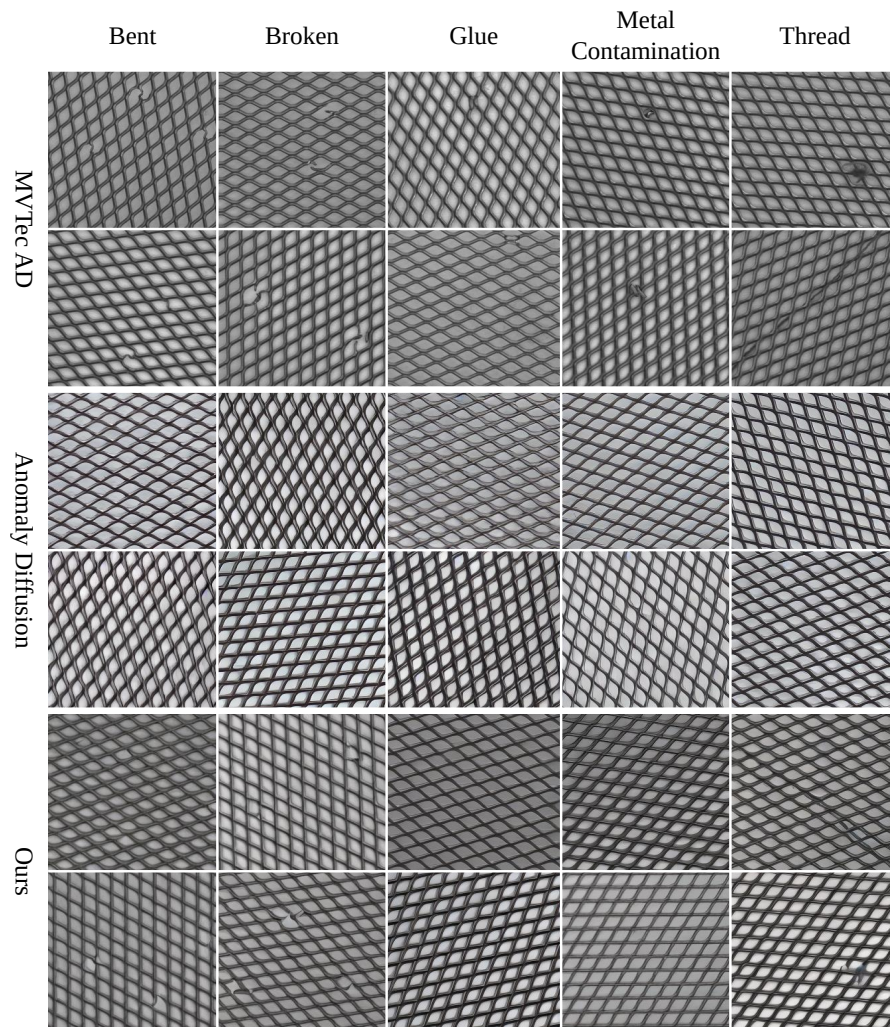


Figure A.5: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on grid.

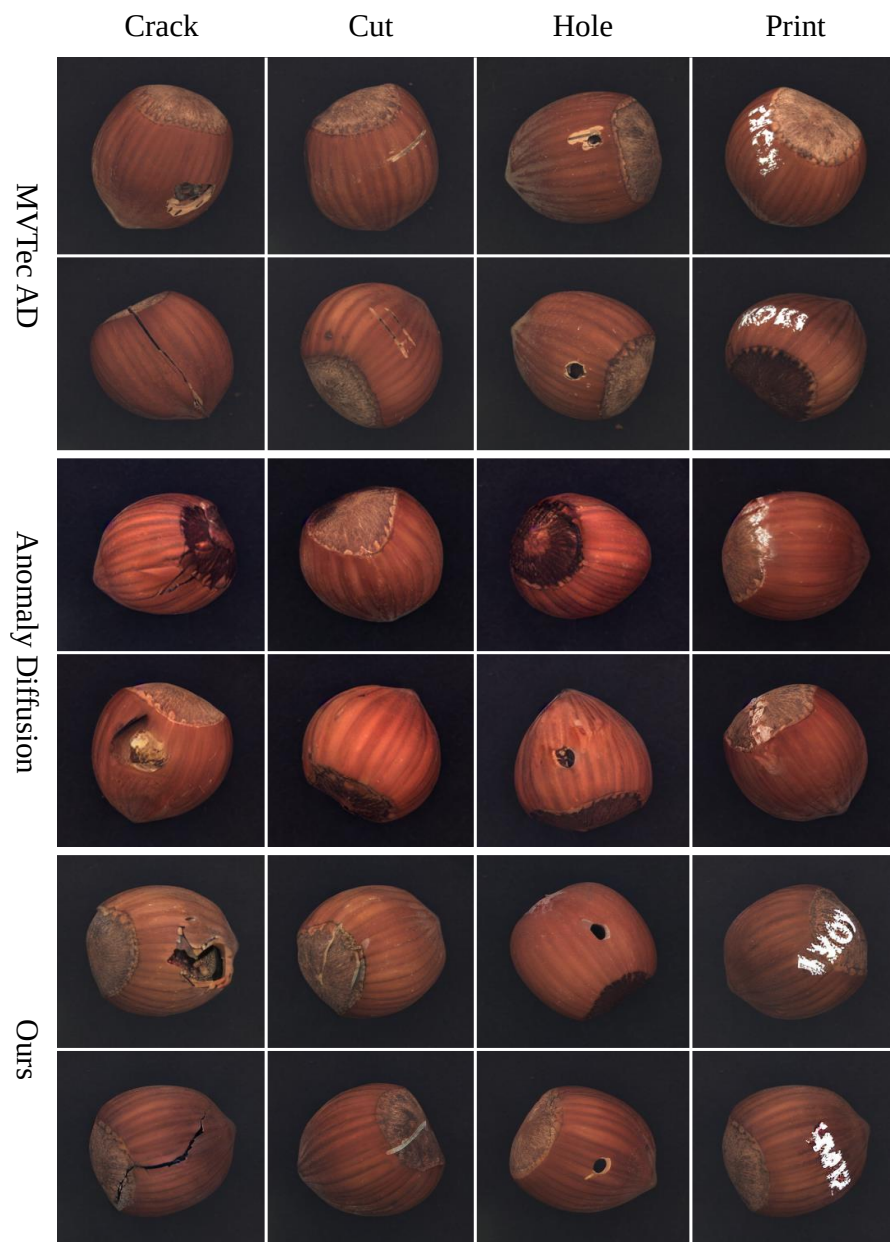


Figure A.6: Examples of MVTec-Ad original training images and randomly sampled generated defect images by current AnomalyDiffusion [52] and our method on hazel nut.

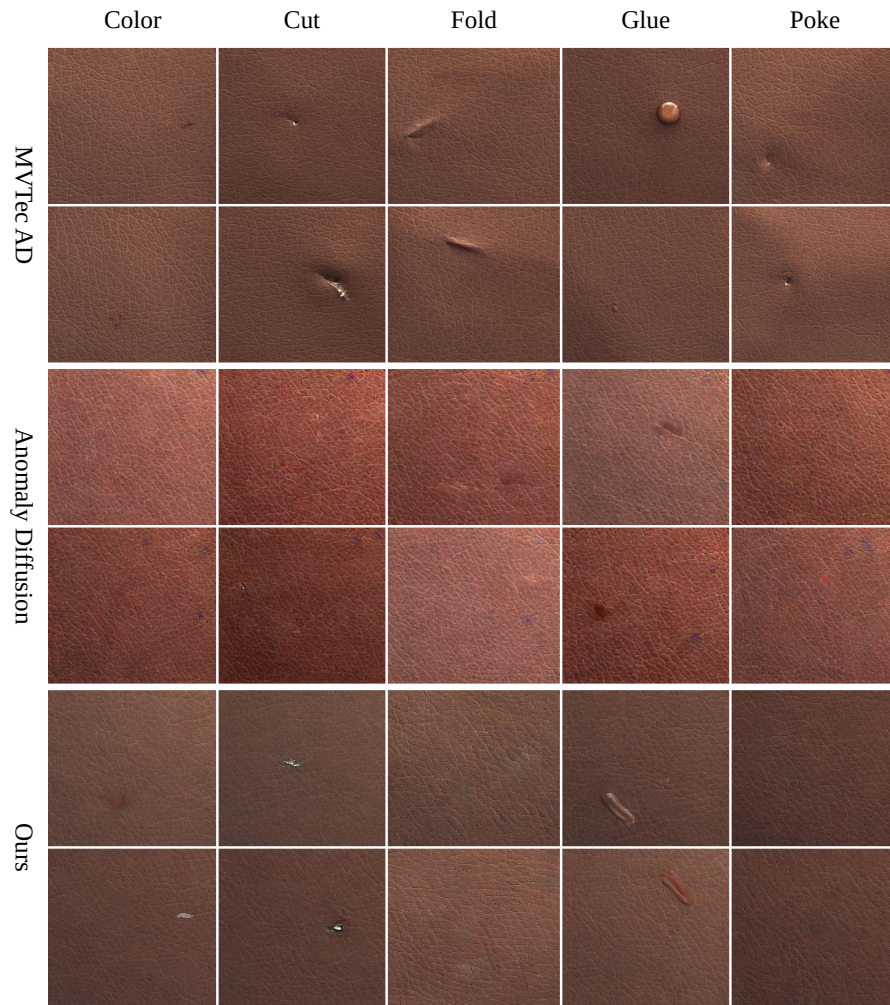


Figure A.7: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on leather.

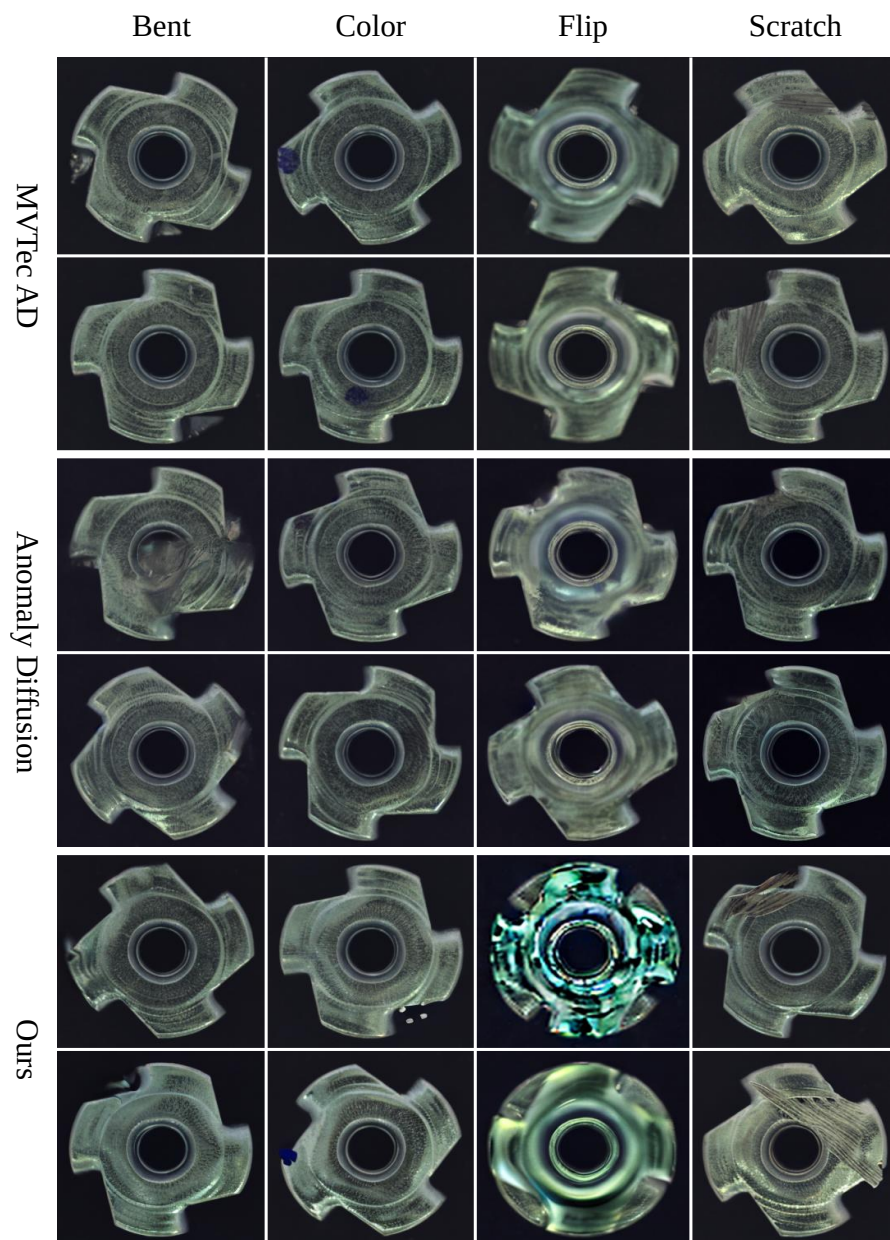


Figure A.8: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on metal nut.



Figure A.9: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on pill.

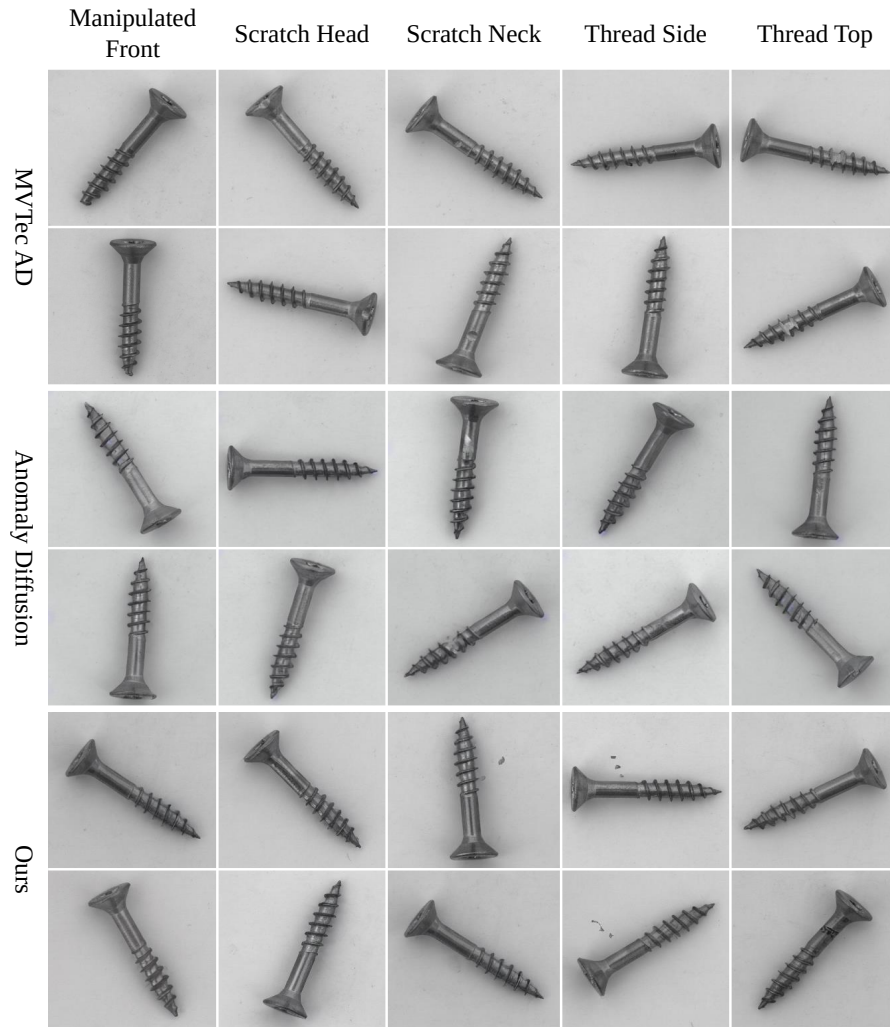


Figure A.10: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on screw.

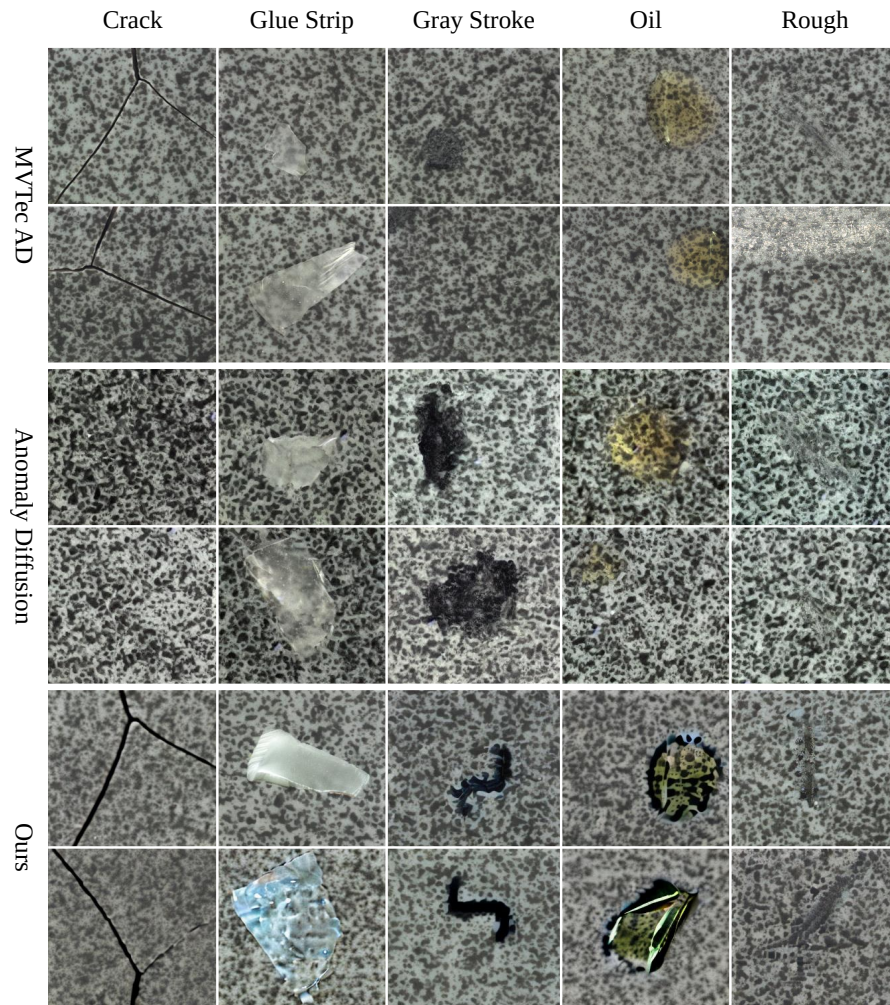


Figure A.11: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on tile.

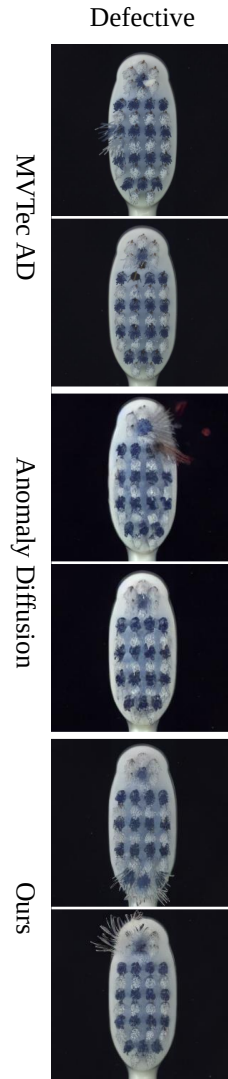


Figure A.12: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on toothbrush.

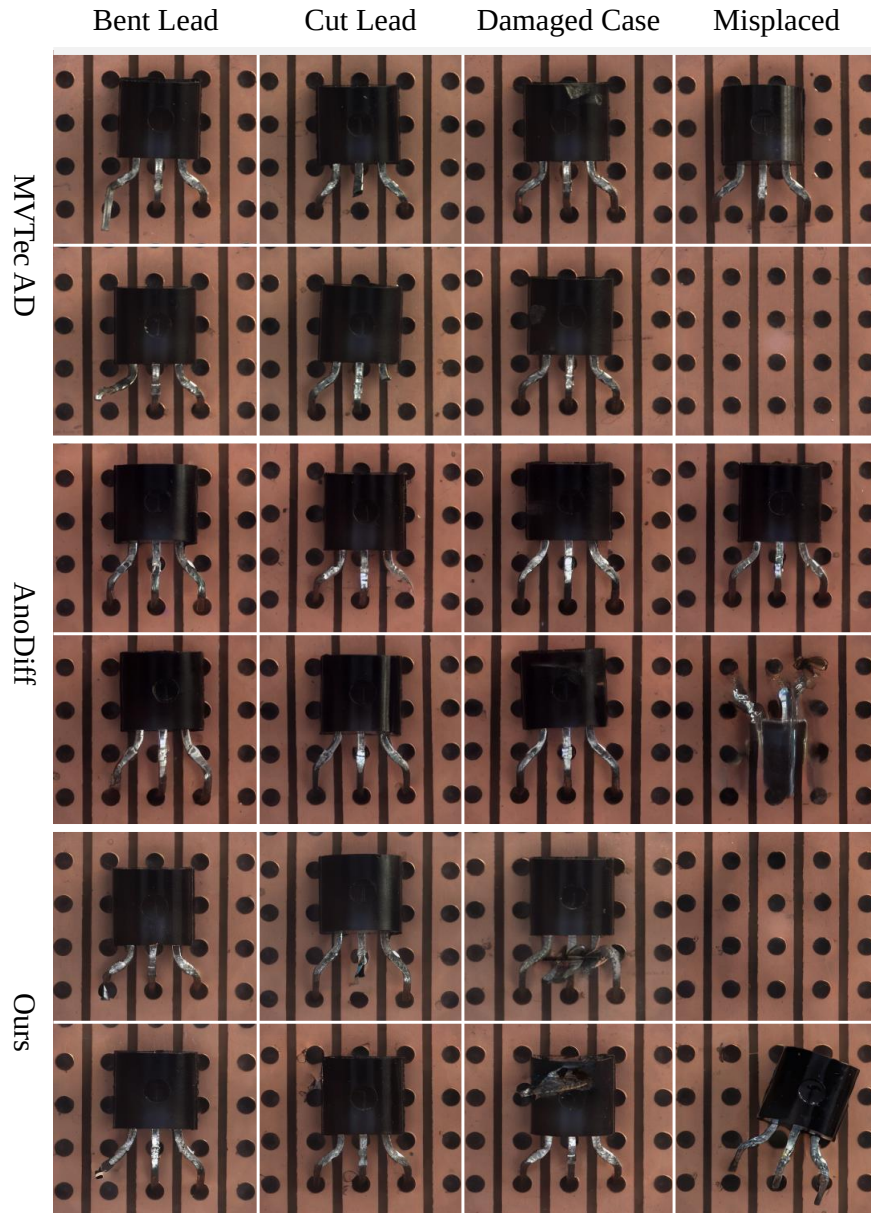


Figure A.13: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on transistor.

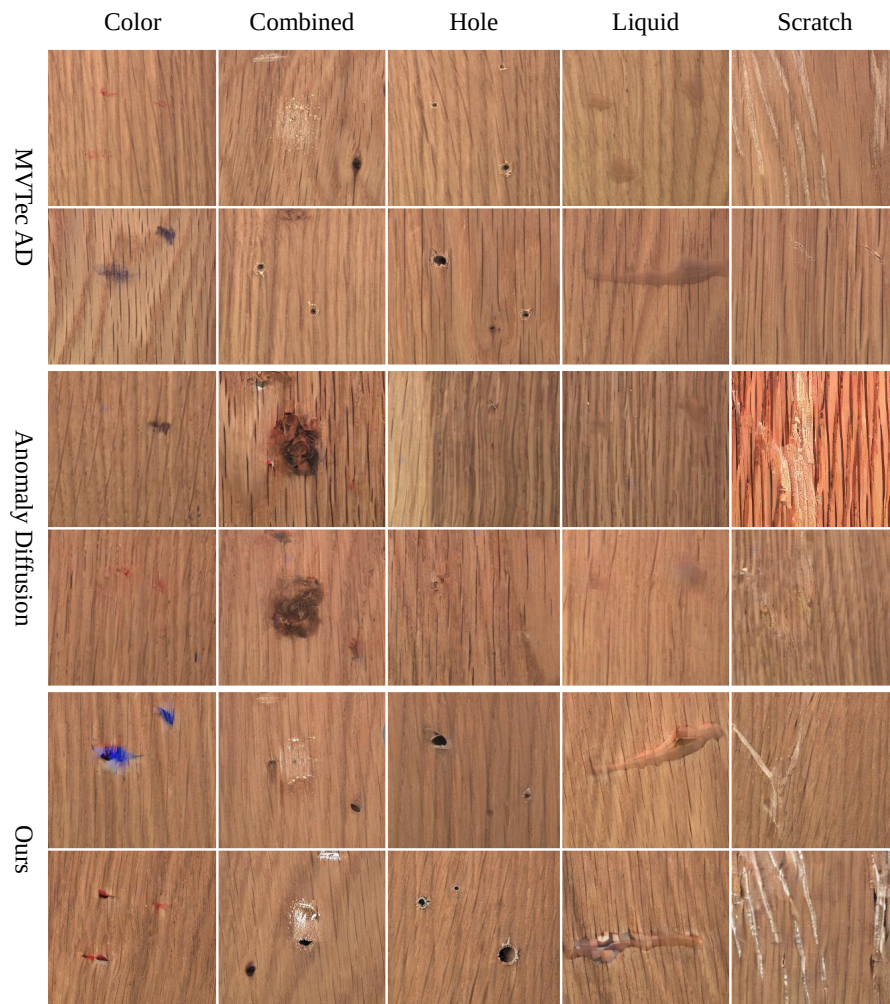


Figure A.14: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on wood.

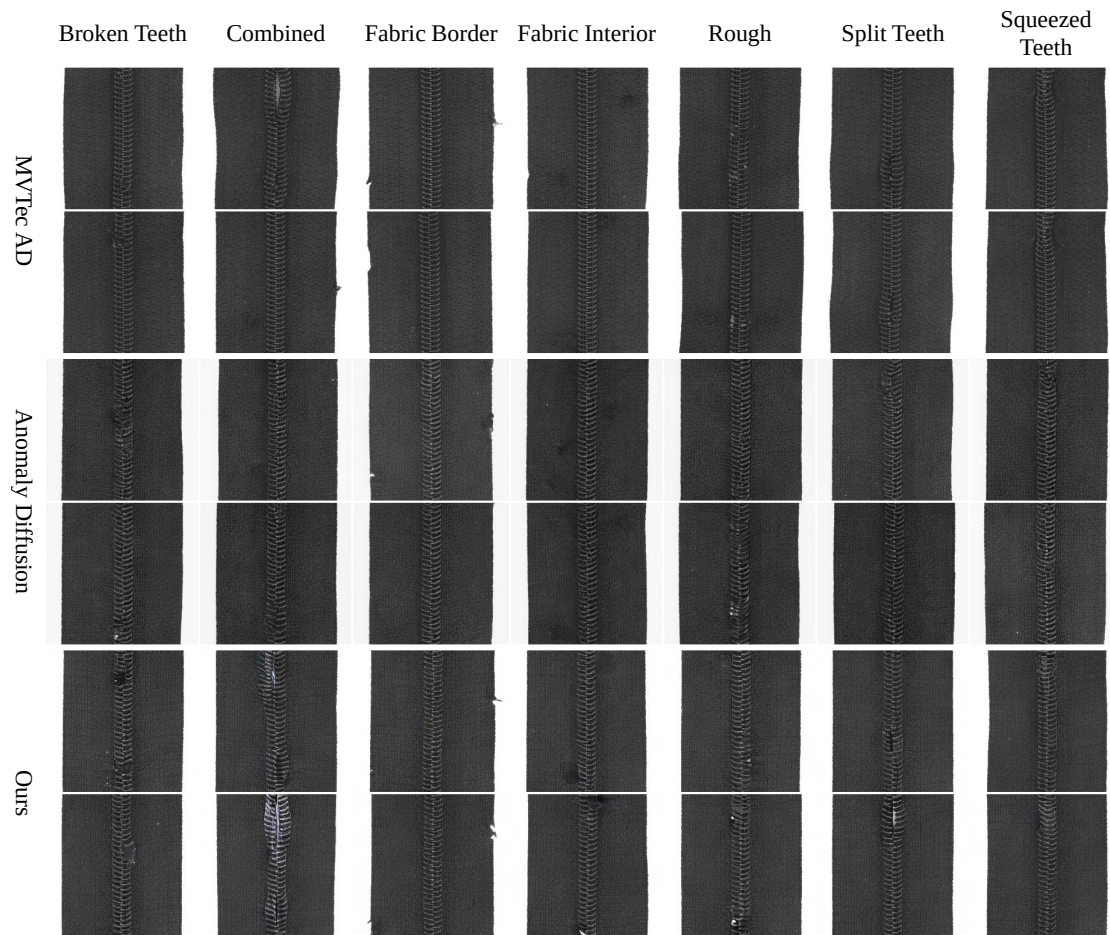


Figure A.15: Examples of MVTec-Ad original training images and randomly sampled generated defect images by AnomalyDiffusion [52] and our method on zipper.

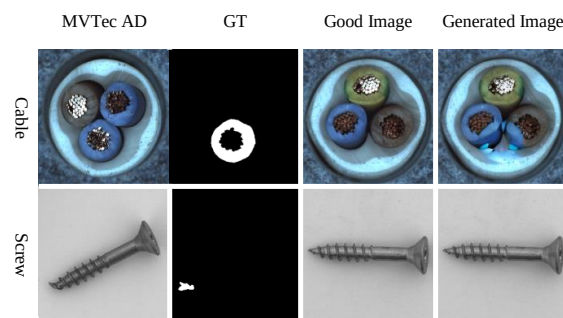


Figure A.16: Illustration of generated images, where masks fall out of appropriate region.

Appendix B

Novel View Syhthesis of Handheld Objects

In this appendix, we provide additional details and results dealing with the creation of our benchmark. We would like to highlight that if NVS-HO is accepted, the benchmark will be publicly released along with the codebase created to implement the baselines and compute the performance evaluation metrics, so as to permit reproduction of our results and stimulate further research on the open challenges set forth by our experimental investigation.

B.1 Additional details on the benchmark

In Table B.1, we present an icon image for each object from the HS images of our benchmark. In Table B.2, we summarize detailed information for all 67 objects, including HS and BS image counts, overlaps used for COLMAP’s sequential matcher, and the prompts applied for Grounded SAM2 [137].

As described in Section 6.2.2 of Chapter 6, our benchmark baselines are built upon two pose estimation methods (COLMAP [135] and VGGT [96]) and two NVS methods (Nerfacto and Splatfacto). In this section, we provide additional details on the hyperparameters chosen for each method. These details are provided to ensure reproducibility of our results.

Pose estimation. For COLMAP, we respect the sequential nature of the captured images when building the database for feature extraction. In the sequential matcher, the overlap parameter—which determines how many neighboring frames are matched—is tuned individually for each object, as reported in Table B.2. We enable guided feature matching to improve robustness, while quadratic overlap is disabled to avoid unnecessary attempts to match frames that are temporally distant. Camera intrinsics are set to the calibrated values obtained during

dataset creation (see Section 6.1 of the Chapter 6) and are kept fixed during pose estimation. All other hyperparameters in the feature extraction, sequential matcher, and mapper stages remain at the default values provided by the COLMAP repository.

Given that transformer-based attention layers impose high memory requirements, the VGGT model is configured with parameters chosen to balance performance and GPU memory constraints. We use NVIDIA RTX-3090 GPU to perform inference on VGGT and to avoid out-of-memory issues, we cap the maximum number of input images per sequence at 120. In cases where a sequence exceeds this limit, we subsample images uniformly to preserve full viewpoint coverage while respecting memory limits. Each image is resized to a resolution of 812×812 pixels, which provides a good trade-off between preserving detail and controlling computational cost. Given the high number of images in each sequence, the number of query frames used in the VGGSfM tracker [96] (`query_frame_num`) is set to 24 instead of 8, while the maximum number of query points per frame (`max_query_pts`) is kept to 4096. Bundle adjustment is enabled and performed using the 2D-3D correspondences predicted by the VGGSfM tracker. We discard the intrinsics predicted by the model, and instead inject our calibrated intrinsics (adjusted for cropping, padding and resizing transformations) for bundle adjustment.

Novel View Synthesis. For NVS, we train both Nerfacto [142] and Splatfacto [93] for each object for 30,000 steps. The background color is set to random. We train the models with and without pose optimization. For Nerfacto, we disable scene contraction, as our data is object-centric and does not require the contraction used for large unbounded scenes. We also disable the appearance embedding, since it leads to overfitting to the lighting conditions of the training images and significantly degrades generalization of novel viewpoints. All other hyperparameters for both Nerfacto and Splatfacto are kept at their defaults, as provided by the Nerfstudio [142] implementation.

B.2 Additional quantitative and qualitative results

Quantitative Results. We present the complete novel view synthesis (NVS) results for all experiments, using both COLMAP and VGGT poses, and training with Nerfacto as well as Splatfacto. Results are summarized in tables, where we also report the variance across all frames for each object. The average performance across all objects for each method is shown in Table B.3, where a $\checkmark$ indicates that poses were optimized. As shown in Table B.3, pose optimization does not yield

a significant improvement in the evaluation metrics, likely because most of the objects are texture-less.

We organize the tables by method and by whether pose optimization was applied, as detailed below:

- **COLMAP + Nerfacto:** results without pose optimization are shown in Table B.4, and with pose optimization in Table B.5.
- **COLMAP + Splatfacto:** results without pose optimization are shown in Table B.6, and with pose optimization in Table B.7.
- **VGGT + Nerfacto:** results without pose optimization are shown in Table B.8, and with pose optimization in Table B.9.
- **VGGT + Splatfacto:** results without pose optimization are shown in Table B.10, and with pose optimization in Table B.11.

Qualitative Results. In Table B.12 we provide additional qualitative NVS results for five objects. These examples complement the quantitative results provided in the Chapter 6, by illustrating common failure cases.

In particular, models trained with VGGT-initialized poses produce noticeably noisier and less consistent renderings, often showing misalignment around object boundaries. In contrast, COLMAP-initialized poses yield cleaner reconstructions that align more closely with ground-truth viewpoints.

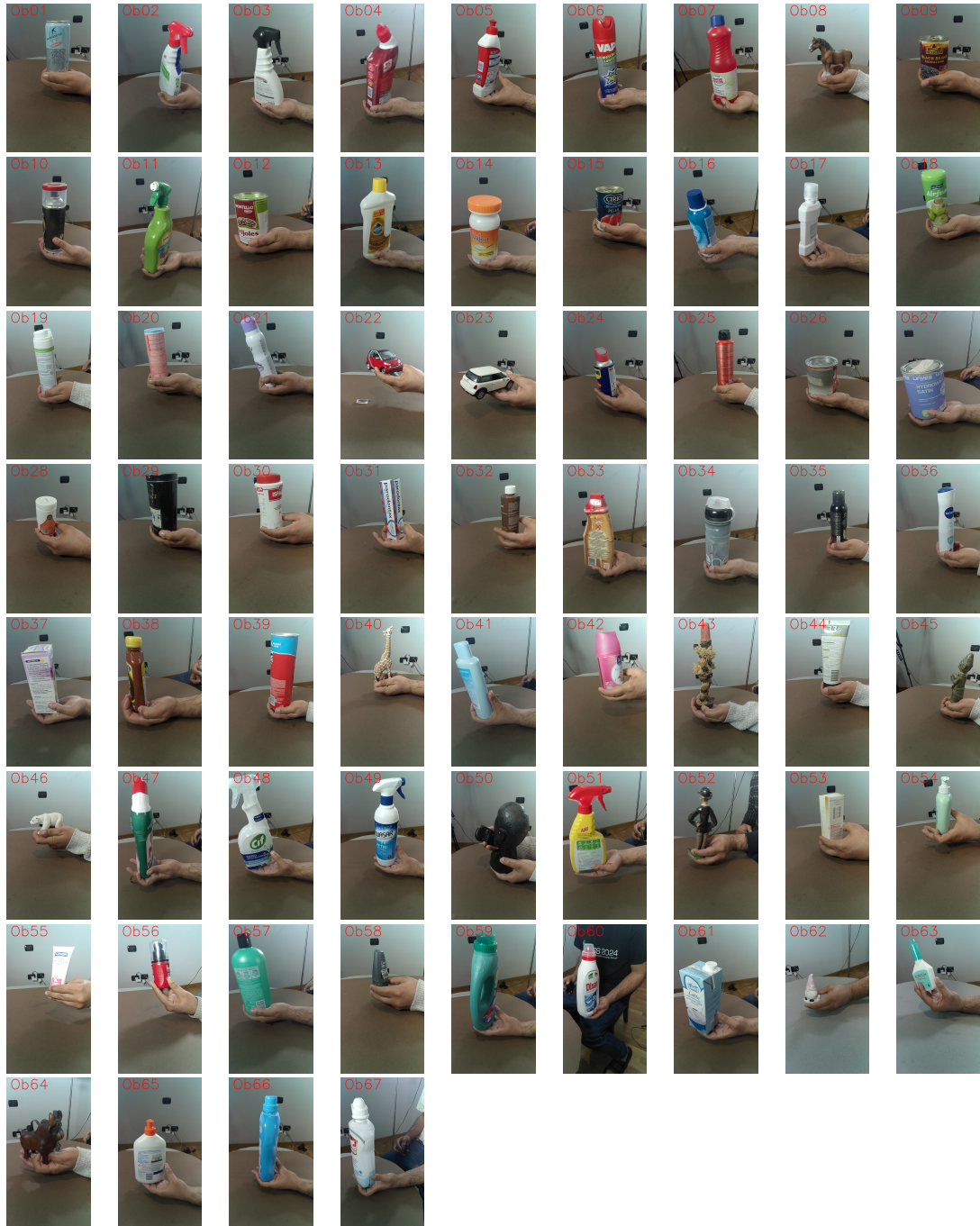


Table B.1: NVS-HO: Icon image for each object using HS.

Object	BS Images Count	HS Images Count	Overlap HS	Overlap CS	Grounded SAM2 Prompt
Ob01	96	128	2	2	Can.
Ob02	88	115	7	7	Spray bottle.
Ob03	81	136	10	10	Spray bottle.
Ob04	72	107	7	7	Toilet cleaner bottle.
Ob05	54	95	7	7	Cloths Detergent bottle.
Ob06	53	98	7	7	Vape spray bottle.
Ob07	86	130	7	7	bottle.
Ob08	93	90	1	3	horse.
Ob09	101	104	1	1	can.
Ob10	110	128	1	1	bottle.
Ob11	87	160	4	5	Spray bottle.
Ob12	72	104	3	3	can.
Ob13	86	106	7	7	Detergent bottle.
Ob14	97	154	1	1	jar.
Ob15	84	90	1	1	can.
Ob16	80	99	7	7	bottle.
Ob17	66	126	2	2	bottle.
Ob18	85	105	7	7	bottle.
Ob19	83	125	2	2	spray bottle.
Ob20	75	134	2	2	bottle.
Ob21	70	98	2	2	bottle.
Ob22	108	147	7	7	car.
Ob23	113	168	15	15	car.
Ob24	93	88	7	7	Spray bottle.
Ob25	63	102	1	1	Spray bottle.
Ob26	101	102	1	1	can.
Ob27	93	151	7	8	can.
Ob28	75	84	7	7	jar.
Ob29	101	105	10	15	jar.
Ob30	63	121	1	1	jar.
Ob31	96	108	1	1	packet.
Ob32	65	111	5	5	bottle.
Ob33	72	100	7	7	bottle.
Ob34	57	130	3	3	bottle.
Ob35	77	92	7	5	Spray bottle.
Ob36	80	119	7	7	Spray bottle.
Ob37	62	93	1	1	Packet.
Ob38	85	138	2	2	bottle.
Ob39	95	111	2	2	pringles bottle.
Ob40	197	116	1	2	giraffe.
Ob41	80	112	2	2	bottle.
Ob42	138	156	4	4	bottle.
Ob43	67	94	5	5	statue.
Ob44	66	82	7	7	tube.
Ob45	65	95	7	7	statue.
Ob46	83	132	7	7	animal.
Ob47	85	123	7	7	Toilet cleaner bottle.
Ob48	70	159	1	2	Spray bottle.
Ob49	76	144	7	7	Spray bottle.
Ob50	73	113	7	7	Statue.
Ob51	89	110	7	7	Spray botte.
Ob52	90	116	7	7	Statue.
Ob53	77	113	1	1	Packet.
Ob54	78	113	2	2	bottle.
Ob55	164	118	2	2	bottle.
Ob56	145	157	7	7	bottle.
Ob57	70	114	1	5	bottle.
Ob58	71	107	1	2	bottle.
Ob59	80	106	2	2	bottle.
Ob60	77	129	2	2	bottle.
Ob61	82	97	1	1	packet.
Ob62	121	124	15	25	toy.
Ob63	154	101	10	50	bottle.
Ob64	70	82	50	50	animal.
Ob65	79	113	2	2	bottle.
Ob66	57	95	4	3	bottle.
Ob67	85	107	2	3	bottle.

Table B.2: Benchmark statistics.

Method	Pose Refined	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	BG PSNR $\uparrow$	BG SSIM $\uparrow$	BG LPIPS $\downarrow$
Nerfacto Colmap		14.4226	0.5685	0.3288	32.7239	0.9415	0.0782
Nerfacto COLMAP	✓	14.1311	0.5633	0.3506	32.9721	0.9404	0.0621
Splatfacto COLMAP		15.8101	0.6376	0.2654	36.8489	0.9951	0.0107
Splatfacto COLMAP	✓	15.7914	0.6369	0.2566	36.9005	0.9953	0.0106
Nerfacto VGGT		11.0757	0.4288	0.4937	27.6944	0.8949	0.1017
Nerfacto VGGT	✓	11.1785	0.4380	0.4873	28.8060	0.9142	0.0748
Splatfacto VGGT		13.0393	0.5809	0.4450	40.2241	0.9852	0.0197
Splatfacto VGGT	✓	12.9919	0.5796	0.4392	40.1761	0.9864	0.0190

Table B.3: **NVS results.** Each row reports the average performance across all objects for the corresponding methods.

	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	BG PSNR $\uparrow$	BG SSIM $\uparrow$	BG LPIPS $\downarrow$
Ob01	16.5956 $\pm$ 0.9007	0.6131 $\pm$ 0.0572	0.2323 $\pm$ 0.0253	37.0735 $\pm$ 2.6114	0.9629 $\pm$ 0.0082	0.0522 $\pm$ 0.0092
Ob02	13.7039 $\pm$ 1.2434	0.5972 $\pm$ 0.0620	0.3770 $\pm$ 0.0594	22.9463 $\pm$ 2.2015	0.8840 $\pm$ 0.0498	0.1472 $\pm$ 0.0384
Ob03	13.2482 $\pm$ 0.5131	0.6578 $\pm$ 0.0367	0.3654 $\pm$ 0.0246	27.4496 $\pm$ 2.3859	0.9183 $\pm$ 0.0167	0.1047 $\pm$ 0.0147
Ob04	14.1864 $\pm$ 1.1923	0.5689 $\pm$ 0.0573	0.3684 $\pm$ 0.0604	25.3442 $\pm$ 1.7675	0.9127 $\pm$ 0.0175	0.1211 $\pm$ 0.0205
Ob05	15.4129 $\pm$ 1.3772	0.6748 $\pm$ 0.0505	0.1851 $\pm$ 0.0238	35.4430 $\pm$ 3.1535	0.9772 $\pm$ 0.0085	0.0463 $\pm$ 0.0064
Ob06	16.8467 $\pm$ 1.0122	0.6589 $\pm$ 0.0529	0.2319 $\pm$ 0.0332	35.6917 $\pm$ 2.7049	0.9350 $\pm$ 0.0245	0.0747 $\pm$ 0.0137
Ob07	18.1826 $\pm$ 1.0237	0.7343 $\pm$ 0.0259	0.2487 $\pm$ 0.0212	33.9074 $\pm$ 2.5909	0.9172 $\pm$ 0.0223	0.1002 $\pm$ 0.0198
Ob08	13.1973 $\pm$ 1.3214	0.6803 $\pm$ 0.1011	0.3216 $\pm$ 0.0894	30.6394 $\pm$ 2.1149	0.9438 $\pm$ 0.0134	0.0840 $\pm$ 0.0106
Ob09	17.2557 $\pm$ 2.0808	0.5244 $\pm$ 0.0409	0.2724 $\pm$ 0.0170	37.4428 $\pm$ 2.5796	0.9494 $\pm$ 0.0128	0.0573 $\pm$ 0.0081
Ob10	15.9220 $\pm$ 1.4249	0.4050 $\pm$ 0.0674	0.3494 $\pm$ 0.0686	30.6546 $\pm$ 1.3082	0.9097 $\pm$ 0.0180	0.0841 $\pm$ 0.0108
Ob11	14.5084 $\pm$ 1.5428	0.5938 $\pm$ 0.0493	0.4253 $\pm$ 0.0726	26.0034 $\pm$ 2.1750	0.9312 $\pm$ 0.0148	0.0961 $\pm$ 0.0170
Ob12	10.1370 $\pm$ 0.9889	0.2676 $\pm$ 0.0311	0.5923 $\pm$ 0.0370	23.7632 $\pm$ 2.4200	0.9265 $\pm$ 0.0120	0.0976 $\pm$ 0.0154
Ob13	12.4361 $\pm$ 1.5567	0.6665 $\pm$ 0.0564	0.3715 $\pm$ 0.0937	23.6163 $\pm$ 3.6898	0.9728 $\pm$ 0.0125	0.0681 $\pm$ 0.0195
Ob14	16.2819 $\pm$ 1.7575	0.6578 $\pm$ 0.0372	0.2314 $\pm$ 0.0338	33.4573 $\pm$ 1.8294	0.9470 $\pm$ 0.0109	0.0609 $\pm$ 0.0093
Ob15	14.2370 $\pm$ 1.2820	0.4335 $\pm$ 0.0491	0.3200 $\pm$ 0.0263	35.1781 $\pm$ 3.0849	0.9241 $\pm$ 0.0191	0.0691 $\pm$ 0.0118
Ob16	13.3855 $\pm$ 0.7356	0.5485 $\pm$ 0.0458	0.3216 $\pm$ 0.0312	27.1230 $\pm$ 2.3720	0.9182 $\pm$ 0.0186	0.1325 $\pm$ 0.0200
Ob17	14.0846 $\pm$ 0.9827	0.6721 $\pm$ 0.0849	0.2746 $\pm$ 0.0608	24.5655 $\pm$ 2.2897	0.9508 $\pm$ 0.0127	0.0953 $\pm$ 0.0177
Ob18	16.5754 $\pm$ 0.9189	0.6589 $\pm$ 0.0527	0.2501 $\pm$ 0.0304	32.6490 $\pm$ 4.6867	0.9622 $\pm$ 0.0158	0.0652 $\pm$ 0.0157
Ob19	12.9425 $\pm$ 0.5185	0.5108 $\pm$ 0.0690	0.2419 $\pm$ 0.0225	28.7351 $\pm$ 2.0432	0.9474 $\pm$ 0.0172	0.0706 $\pm$ 0.0111
Ob20	13.9817 $\pm$ 0.7075	0.4389 $\pm$ 0.0421	0.3882 $\pm$ 0.0410	31.8566 $\pm$ 3.2596	0.9558 $\pm$ 0.0124	0.0351 $\pm$ 0.0040
Ob21	12.5937 $\pm$ 0.6360	0.5796 $\pm$ 0.0702	0.2380 $\pm$ 0.0256	32.4147 $\pm$ 3.9260	0.9631 $\pm$ 0.0165	0.0676 $\pm$ 0.0165
Ob22	13.2484 $\pm$ 1.0731	0.5027 $\pm$ 0.0660	0.3303 $\pm$ 0.0687	39.3285 $\pm$ 3.9861	0.9847 $\pm$ 0.0048	0.0291 $\pm$ 0.0057
Ob23	13.5656 $\pm$ 0.4139	0.5723 $\pm$ 0.0458	0.3472 $\pm$ 0.0558	35.6182 $\pm$ 2.0748	0.9106 $\pm$ 0.0261	0.0801 $\pm$ 0.0152
Ob24	15.3227 $\pm$ 0.7083	0.4893 $\pm$ 0.0536	0.2606 $\pm$ 0.0444	33.3320 $\pm$ 3.9150	0.9535 $\pm$ 0.0163	0.0674 $\pm$ 0.0163
Ob25	13.9865 $\pm$ 0.7734	0.4338 $\pm$ 0.0724	0.4656 $\pm$ 0.0647	30.4175 $\pm$ 1.5199	0.9792 $\pm$ 0.0087	0.0489 $\pm$ 0.0109
Ob26	16.0637 $\pm$ 1.1602	0.5497 $\pm$ 0.0508	0.2817 $\pm$ 0.0262	38.1287 $\pm$ 3.1845	0.9426 $\pm$ 0.0120	0.0479 $\pm$ 0.0082
Ob27	16.6788 $\pm$ 1.2893	0.5292 $\pm$ 0.0668	0.3206 $\pm$ 0.0325	36.6111 $\pm$ 2.4121	0.9526 $\pm$ 0.0052	0.0590 $\pm$ 0.0062
Ob28	15.1572 $\pm$ 0.6345	0.5835 $\pm$ 0.0654	0.3056 $\pm$ 0.0464	30.5975 $\pm$ 4.1497	0.9600 $\pm$ 0.0243	0.0646 $\pm$ 0.0229
Ob29	14.3975 $\pm$ 1.8761	0.1659 $\pm$ 0.0392	0.4964 $\pm$ 0.0752	100.0000 $\pm$ 0.0000	1.0000 $\pm$ 0.0000	0.0000 $\pm$ 0.0000
Ob30	14.0067 $\pm$ 0.8169	0.6969 $\pm$ 0.0380	0.2128 $\pm$ 0.0301	32.2761 $\pm$ 1.7951	0.9293 $\pm$ 0.0074	0.0866 $\pm$ 0.0114
Ob31	13.5396 $\pm$ 1.0634	0.4653 $\pm$ 0.1071	0.2245 $\pm$ 0.0501	35.7919 $\pm$ 3.4711	0.9700 $\pm$ 0.0205	0.0492 $\pm$ 0.0149
Ob32	16.9768 $\pm$ 0.8458	0.5190 $\pm$ 0.0591	0.3161 $\pm$ 0.0352	32.6667 $\pm$ 3.8966	0.9340 $\pm$ 0.0331	0.0710 $\pm$ 0.0233
Ob33	16.2956 $\pm$ 1.3330	0.6077 $\pm$ 0.0979	0.1942 $\pm$ 0.0386	34.8231 $\pm$ 3.4159	0.9764 $\pm$ 0.0095	0.0516 $\pm$ 0.0094
Ob34	17.4435 $\pm$ 0.8581	0.6620 $\pm$ 0.0351	0.2874 $\pm$ 0.0232	31.3440 $\pm$ 1.4941	0.9395 $\pm$ 0.0134	0.0896 $\pm$ 0.0102
Ob35	17.8795 $\pm$ 1.4366	0.5657 $\pm$ 0.0978	0.3037 $\pm$ 0.0184	37.4521 $\pm$ 2.5555	0.9263 $\pm$ 0.0194	0.0811 $\pm$ 0.0158
Ob36	14.2281 $\pm$ 0.7180	0.5594 $\pm$ 0.1352	0.2442 $\pm$ 0.0409	31.9264 $\pm$ 2.1497	0.9651 $\pm$ 0.0121	0.0627 $\pm$ 0.0132
Ob37	15.3021 $\pm$ 1.3211	0.5125 $\pm$ 0.1278	0.2791 $\pm$ 0.0537	28.5956 $\pm$ 2.8034	0.9399 $\pm$ 0.0175	0.1040 $\pm$ 0.0173
Ob38	15.7495 $\pm$ 1.1850	0.5526 $\pm$ 0.0805	0.3637 $\pm$ 0.0551	28.4417 $\pm$ 2.6826	0.9096 $\pm$ 0.0223	0.1405 $\pm$ 0.0305
Ob39	15.7619 $\pm$ 2.1574	0.6449 $\pm$ 0.0935	0.1946 $\pm$ 0.0574	30.5416 $\pm$ 1.8728	0.9674 $\pm$ 0.0093	0.0611 $\pm$ 0.0060
Ob40	10.2096 $\pm$ 0.8615	0.6866 $\pm$ 0.0464	0.1993 $\pm$ 0.0306	30.7762 $\pm$ 2.1925	0.9540 $\pm$ 0.0095	0.0588 $\pm$ 0.0097
Ob41	13.8768 $\pm$ 1.0236	0.5800 $\pm$ 0.0904	0.4868 $\pm$ 0.1494	24.5374 $\pm$ 1.8048	0.9617 $\pm$ 0.0136	0.0768 $\pm$ 0.0153
Ob42	14.6277 $\pm$ 0.7471	0.6138 $\pm$ 0.0347	0.3168 $\pm$ 0.0404	40.7178 $\pm$ 2.0842	0.9747 $\pm$ 0.0061	0.0301 $\pm$ 0.0084
Ob43	16.0358 $\pm$ 0.7746	0.5219 $\pm$ 0.0347	0.2597 $\pm$ 0.0256	30.8072 $\pm$ 1.6595	0.9615 $\pm$ 0.0153	0.0775 $\pm$ 0.0136
Ob44	14.6057 $\pm$ 1.2601	0.5808 $\pm$ 0.0462	0.2616 $\pm$ 0.0618	29.7173 $\pm$ 4.2494	0.9597 $\pm$ 0.0308	0.0792 $\pm$ 0.0231
Ob45	17.7646 $\pm$ 1.6918	0.6029 $\pm$ 0.0550	0.2289 $\pm$ 0.0322	37.6802 $\pm$ 1.8626	0.9719 $\pm$ 0.0097	0.0452 $\pm$ 0.0056
Ob46	11.0398 $\pm$ 0.4202	0.5546 $\pm$ 0.0503	0.3551 $\pm$ 0.0503	30.7100 $\pm$ 3.4584	0.9663 $\pm$ 0.0101	0.0416 $\pm$ 0.0092
Ob47	14.4533 $\pm$ 1.3248	0.5521 $\pm$ 0.0826	0.2175 $\pm$ 0.0593	30.3006 $\pm$ 2.7471	0.9606 $\pm$ 0.0257	0.0897 $\pm$ 0.0434
Ob48	10.3821 $\pm$ 0.6771	0.5621 $\pm$ 0.0569	0.5143 $\pm$ 0.0663	21.6239 $\pm$ 3.4137	0.7979 $\pm$ 0.0901	0.2233 $\pm$ 0.1233
Ob49	16.2635 $\pm$ 1.4786	0.7017 $\pm$ 0.0529	0.2181 $\pm$ 0.0397	25.1616 $\pm$ 3.9327	0.9216 $\pm$ 0.0381	0.1243 $\pm$ 0.0330
Ob50	20.5167 $\pm$ 1.8335	0.6400 $\pm$ 0.0373	0.2481 $\pm$ 0.0222	35.5328 $\pm$ 2.2535	0.9209 $\pm$ 0.0190	0.0751 $\pm$ 0.0130
Ob51	15.4773 $\pm$ 0.7790	0.6946 $\pm$ 0.0511	0.2270 $\pm$ 0.0413	25.7808 $\pm$ 3.5258	0.9246 $\pm$ 0.0166	0.1388 $\pm$ 0.0231
Ob52	16.6015 $\pm$ 1.0765	0.6893 $\pm$ 0.0386	0.3038 $\pm$ 0.0477	31.0831 $\pm$ 1.0391	0.9120 $\pm$ 0.0219	0.1048 $\pm$ 0.0198
Ob53	13.4366 $\pm$ 1.0278	0.6034 $\pm$ 0.0376	0.2912 $\pm$ 0.0321	35.7023 $\pm$ 3.6809	0.9641 $\pm$ 0.0170	0.0522 $\pm$ 0.0106
Ob54	12.8277 $\pm$ 1.1685	0.6200 $\pm$ 0.0526	0.3043 $\pm$ 0.0457	26.4563 $\pm$ 2.4672	0.9570 $\pm$ 0.0183	0.0793 $\pm$ 0.0219
Ob55	2.7938 $\pm$ 1.0605	0.2326 $\pm$ 0.0704	0.7441 $\pm$ 0.0284	100.0000 $\pm$ 0.0000	1.0000 $\pm$ 0.0000	0.0000 $\pm$ 0.0000
Ob56	13.0775 $\pm$ 1.4291	0.3954 $\pm$ 0.0667	0.4229 $\pm$ 0.0337	35.9861 $\pm$ 3.7726	0.9896 $\pm$ 0.0016	0.0237 $\pm$ 0.0037
Ob57	15.7637 $\pm$ 0.5085	0.6846 $\pm$ 0.0562	0.3035 $\pm$ 0.0459	27.4237 $\pm$ 1.8226	0.8981 $\pm$ 0.0203	0.1327 $\pm$ 0.0232
Ob58	16.4396 $\pm$ 0.4212	0.5121 $\pm$ 0.0278	0.2854 $\pm$ 0.0272	35.0303 $\pm$ 2.2821	0.9335 $\pm$ 0.0243	0.0556 $\pm$ 0.0111
Ob59	12.3608 $\pm$ 1.2451	0.5024 $\pm$ 0.0829	0.5733 $\pm$ 0.0755	20.4637 $\pm$ 3.1947	0.8719 $\pm$ 0.0361	0.1604 $\pm$ 0.0381
Ob60	11.6420 $\pm$ 0.4864	0.6299 $\pm$ 0.0377	0.4413 $\pm$ 0.0341	20.9947 $\pm$ 1.3908	0.9388 $\pm$ 0.0238	0.0670 $\pm$ 0.0134
Ob61	15.0726 $\pm$ 1.1297	0.5849 $\pm$ 0.0465	0.3211 $\pm$ 0.0414	24.5630 $\pm$ 2.6278	0.9284 $\pm$ 0.0187	0.1144 $\pm$ 0.0202
Ob62	11.5780 $\pm$ 1.3322	0.6214 $\pm$ 0.0358	0.3874 $\pm$ 0.0482	29.8169 $\pm$ 2.2624	0.9760 $\pm$ 0.0096	0.0327 $\pm$ 0.0096
Ob63	11.1363 $\pm$ 0.8045	0.5841 $\pm$ 0.0348	0.4360 $\pm$ 0.0459	29.8183 $\pm$ 2.4555	0.9769 $\pm$ 0.0040	0.0393 $\pm$ 0.0121
Ob64	20.1110 $\pm$ 0.8882	0.7235 $\pm$ 0.0419	0.2979 $\pm$ 0.0312	32.9267 $\pm$ 1.9446	0.9341 $\pm$ 0.0109	0.0933 $\pm$ 0.0093
Ob65	15.3896 $\pm$ 0.9469	0.6539 $\pm$ 0.0390	0.2126 $\pm$ 0.0413	30.5124 $\pm$ 2.2110	0.9766 $\pm$ 0.0175	0.0507 $\pm$ 0.0248
Ob66	11.2471 $\pm$ 0.8741	0.4934 $\pm$ 0.0738	0.5419 $\pm$ 0.0633	21.3039 $\pm$ 2.5495	0.8661 $\pm$ 0.0312	0.1125 $\pm$ 0.0308
Ob67	10.2641 $\pm$ 1.0791	0.5110 $\pm$ 0.0306	0.5947 $\pm$ 0.0196	23.2279 $\pm$ 3.5608	0.8320 $\pm$ 0.0588	0.1370 $\pm$ 0.0432
Average	14.4226	0.5685	0.3288	32.7239	0.9415	0.0782

Table B.4: NVS Results for Nerfacto with COLMAP-initialized poses and no pose optimization.

	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	BG PSNR $\uparrow$	BG SSIM $\uparrow$	BG LPIPS $\downarrow$
Ob01	16.5992 $\pm$ 1.0305	0.6172 $\pm$ 0.0630	0.2265 $\pm$ 0.0189	38.8860 $\pm$ 2.6261	0.9670 $\pm$ 0.0092	0.0363 $\pm$ 0.0054
Ob02	14.0804 $\pm$ 1.0347	0.6138 $\pm$ 0.0536	0.3590 $\pm$ 0.0525	23.9900 $\pm$ 1.6161	0.8676 $\pm$ 0.0453	0.1256 $\pm$ 0.0235
Ob03	13.3263 $\pm$ 0.6028	0.6691 $\pm$ 0.0361	0.3738 $\pm$ 0.0249	26.9364 $\pm$ 2.5012	0.9086 $\pm$ 0.0212	0.0960 $\pm$ 0.0150
Ob04	14.9418 $\pm$ 0.7448	0.6018 $\pm$ 0.0545	0.2764 $\pm$ 0.0553	27.5388 $\pm$ 1.7984	0.9212 $\pm$ 0.0233	0.0884 $\pm$ 0.0175
Ob05	15.5098 $\pm$ 1.2067	0.6805 $\pm$ 0.0495	0.1804 $\pm$ 0.0203	35.1326 $\pm$ 3.3313	0.9779 $\pm$ 0.0075	0.0404 $\pm$ 0.0061
Ob06	16.8899 $\pm$ 0.8757	0.6470 $\pm$ 0.0579	0.2476 $\pm$ 0.0233	35.9691 $\pm$ 2.5898	0.9430 $\pm$ 0.0242	0.0538 $\pm$ 0.0099
Ob07	17.2910 $\pm$ 0.6866	0.7054 $\pm$ 0.0234	0.2667 $\pm$ 0.0261	31.6000 $\pm$ 4.4115	0.9229 $\pm$ 0.0451	0.0616 $\pm$ 0.0279
Ob08	13.3891 $\pm$ 1.4420	0.7121 $\pm$ 0.0968	0.3500 $\pm$ 0.0925	30.7727 $\pm$ 2.0981	0.9196 $\pm$ 0.0159	0.0765 $\pm$ 0.0104
Ob09	9.9297 $\pm$ 0.9666	0.1304 $\pm$ 0.0370	0.7015 $\pm$ 0.0279	100.0000 $\pm$ 0.0000	1.0000 $\pm$ 0.0000	0.0000 $\pm$ 0.0000
Ob10	16.2259 $\pm$ 0.8114	0.4232 $\pm$ 0.0512	0.3575 $\pm$ 0.0523	30.0709 $\pm$ 1.5912	0.9063 $\pm$ 0.0188	0.0726 $\pm$ 0.0110
Ob11	14.4980 $\pm$ 1.9732	0.6133 $\pm$ 0.0507	0.4229 $\pm$ 0.0963	25.7735 $\pm$ 2.6843	0.9526 $\pm$ 0.0149	0.0631 $\pm$ 0.0085
Ob12	10.1370 $\pm$ 0.9742	0.2750 $\pm$ 0.0311	0.5913 $\pm$ 0.0393	24.1850 $\pm$ 2.4232	0.9337 $\pm$ 0.0128	0.0758 $\pm$ 0.0099
Ob13	12.1981 $\pm$ 1.6971	0.6633 $\pm$ 0.0650	0.3764 $\pm$ 0.0885	24.0259 $\pm$ 3.6501	0.9683 $\pm$ 0.0102	0.0605 $\pm$ 0.0119
Ob14	16.1411 $\pm$ 1.6909	0.6658 $\pm$ 0.0344	0.2184 $\pm$ 0.0385	33.6878 $\pm$ 1.4050	0.9545 $\pm$ 0.0075	0.0486 $\pm$ 0.0042
Ob15	12.5917 $\pm$ 0.9509	0.3325 $\pm$ 0.0249	0.4190 $\pm$ 0.0593	31.9218 $\pm$ 2.3230	0.9410 $\pm$ 0.0163	0.0556 $\pm$ 0.0107
Ob16	13.7014 $\pm$ 0.7738	0.5722 $\pm$ 0.0469	0.2816 $\pm$ 0.0293	29.6153 $\pm$ 1.4177	0.9236 $\pm$ 0.0137	0.0973 $\pm$ 0.0111
Ob17	14.0811 $\pm$ 0.8821	0.6897 $\pm$ 0.0722	0.3200 $\pm$ 0.0505	25.2360 $\pm$ 3.3936	0.9630 $\pm$ 0.0177	0.0586 $\pm$ 0.0145
Ob18	16.2348 $\pm$ 0.8397	0.6147 $\pm$ 0.0708	0.2610 $\pm$ 0.0294	34.6577 $\pm$ 4.1706	0.9635 $\pm$ 0.0150	0.0408 $\pm$ 0.0116
Ob19	13.1904 $\pm$ 0.6088	0.5229 $\pm$ 0.0626	0.2597 $\pm$ 0.0305	31.3949 $\pm$ 1.6612	0.9460 $\pm$ 0.0141	0.0512 $\pm$ 0.0049
Ob20	13.9932 $\pm$ 0.7946	0.4544 $\pm$ 0.0415	0.4110 $\pm$ 0.0396	30.6398 $\pm$ 4.9759	0.9542 $\pm$ 0.0350	0.0394 $\pm$ 0.0391
Ob21	12.6491 $\pm$ 0.5584	0.5873 $\pm$ 0.0690	0.2626 $\pm$ 0.0324	32.9154 $\pm$ 3.0687	0.9590 $\pm$ 0.0115	0.0451 $\pm$ 0.0071
Ob22	13.3032 $\pm$ 0.9234	0.5053 $\pm$ 0.0557	0.3455 $\pm$ 0.0624	39.2900 $\pm$ 3.5040	0.9812 $\pm$ 0.0060	0.0299 $\pm$ 0.0061
Ob23	13.9897 $\pm$ 1.1323	0.6290 $\pm$ 0.0444	0.3201 $\pm$ 0.0521	37.3841 $\pm$ 3.8089	0.9567 $\pm$ 0.0165	0.0368 $\pm$ 0.0053
Ob24	14.3700 $\pm$ 0.6926	0.4524 $\pm$ 0.0583	0.3298 $\pm$ 0.0213	34.7694 $\pm$ 2.2264	0.9467 $\pm$ 0.0145	0.0501 $\pm$ 0.0091
Ob25	13.8398 $\pm$ 1.1170	0.4392 $\pm$ 0.0715	0.4731 $\pm$ 0.0550	30.2745 $\pm$ 1.5246	0.9792 $\pm$ 0.0090	0.0358 $\pm$ 0.0056
Ob26	15.9632 $\pm$ 1.1169	0.5324 $\pm$ 0.0535	0.2864 $\pm$ 0.0327	37.8714 $\pm$ 2.4973	0.9427 $\pm$ 0.0088	0.0385 $\pm$ 0.0054
Ob27	16.6488 $\pm$ 1.1955	0.5352 $\pm$ 0.0478	0.2712 $\pm$ 0.0273	38.1080 $\pm$ 3.2844	0.9585 $\pm$ 0.0054	0.0419 $\pm$ 0.0065
Ob28	17.1165 $\pm$ 1.1424	0.6197 $\pm$ 0.0781	0.3217 $\pm$ 0.0519	31.0637 $\pm$ 2.8052	0.9523 $\pm$ 0.0302	0.0489 $\pm$ 0.0164
Ob29	14.9736 $\pm$ 2.2294	0.3748 $\pm$ 0.1343	0.5064 $\pm$ 0.0506	27.5663 $\pm$ 2.6947	0.6890 $\pm$ 0.1626	0.1476 $\pm$ 0.0538
Ob30	14.0965 $\pm$ 0.9700	0.6549 $\pm$ 0.0714	0.2266 $\pm$ 0.0237	33.5902 $\pm$ 2.1942	0.9396 $\pm$ 0.0219	0.0624 $\pm$ 0.0080
Ob31	13.7428 $\pm$ 1.1419	0.4564 $\pm$ 0.0913	0.2378 $\pm$ 0.0522	36.9301 $\pm$ 2.7703	0.9522 $\pm$ 0.0314	0.0457 $\pm$ 0.0064
Ob32	16.4980 $\pm$ 1.2905	0.4759 $\pm$ 0.0504	0.4167 $\pm$ 0.0319	28.7020 $\pm$ 2.5986	0.9211 $\pm$ 0.0340	0.0567 $\pm$ 0.0129
Ob33	16.3041 $\pm$ 1.2414	0.6235 $\pm$ 0.0848	0.2056 $\pm$ 0.0394	36.1408 $\pm$ 3.8712	0.9800 $\pm$ 0.0096	0.0441 $\pm$ 0.0088
Ob34	17.8366 $\pm$ 1.1773	0.7028 $\pm$ 0.0370	0.2464 $\pm$ 0.0202	33.6116 $\pm$ 1.8318	0.9648 $\pm$ 0.0081	0.0433 $\pm$ 0.0045
Ob35	16.7985 $\pm$ 1.0185	0.5308 $\pm$ 0.0975	0.3443 $\pm$ 0.0282	39.0971 $\pm$ 1.8969	0.9429 $\pm$ 0.0169	0.0482 $\pm$ 0.0094
Ob36	13.9137 $\pm$ 0.7800	0.5653 $\pm$ 0.1138	0.2750 $\pm$ 0.0385	30.8857 $\pm$ 2.6466	0.9719 $\pm$ 0.0115	0.0393 $\pm$ 0.0053
Ob37	15.6660 $\pm$ 1.8159	0.5523 $\pm$ 0.0947	0.2332 $\pm$ 0.0429	30.1649 $\pm$ 2.2542	0.9432 $\pm$ 0.0133	0.0707 $\pm$ 0.0092
Ob38	15.4324 $\pm$ 0.9227	0.5646 $\pm$ 0.0780	0.3806 $\pm$ 0.0581	30.4754 $\pm$ 2.7354	0.9184 $\pm$ 0.0150	0.0821 $\pm$ 0.0159
Ob39	14.7184 $\pm$ 1.2922	0.6167 $\pm$ 0.0583	0.2009 $\pm$ 0.0424	31.9758 $\pm$ 3.6016	0.9636 $\pm$ 0.0086	0.0513 $\pm$ 0.0064
Ob40	10.0547 $\pm$ 1.0238	0.6963 $\pm$ 0.0603	0.2099 $\pm$ 0.0352	31.1163 $\pm$ 2.6277	0.9641 $\pm$ 0.0109	0.0488 $\pm$ 0.0071
Ob41	13.9500 $\pm$ 0.7784	0.6155 $\pm$ 0.0721	0.4413 $\pm$ 0.1077	25.3200 $\pm$ 2.0933	0.9653 $\pm$ 0.0109	0.0632 $\pm$ 0.0085
Ob42	14.7964 $\pm$ 0.8260	0.5927 $\pm$ 0.0261	0.3394 $\pm$ 0.0523	37.3282 $\pm$ 2.6731	0.9729 $\pm$ 0.0071	0.0316 $\pm$ 0.0069
Ob43	15.6741 $\pm$ 0.6737	0.5099 $\pm$ 0.0264	0.2683 $\pm$ 0.0252	31.2297 $\pm$ 1.9697	0.9601 $\pm$ 0.0167	0.0627 $\pm$ 0.0088
Ob44	15.0523 $\pm$ 1.0831	0.5952 $\pm$ 0.0501	0.2760 $\pm$ 0.0610	27.1351 $\pm$ 4.0963	0.9346 $\pm$ 0.0213	0.0825 $\pm$ 0.0163
Ob45	16.2111 $\pm$ 1.3331	0.5553 $\pm$ 0.0464	0.3155 $\pm$ 0.0180	34.6268 $\pm$ 2.2769	0.9619 $\pm$ 0.0125	0.0375 $\pm$ 0.0050
Ob46	11.1796 $\pm$ 0.7163	0.5868 $\pm$ 0.0588	0.3870 $\pm$ 0.0521	30.8595 $\pm$ 1.3612	0.9637 $\pm$ 0.0114	0.0410 $\pm$ 0.0089
Ob47	14.1547 $\pm$ 2.0328	0.5638 $\pm$ 0.0785	0.2730 $\pm$ 0.0974	29.1757 $\pm$ 2.5799	0.9688 $\pm$ 0.0121	0.0572 $\pm$ 0.0122
Ob48	9.8272 $\pm$ 0.6365	0.5695 $\pm$ 0.0589	0.5270 $\pm$ 0.0508	20.7469 $\pm$ 2.4337	0.8415 $\pm$ 0.0699	0.2096 $\pm$ 0.0904
Ob49	15.9557 $\pm$ 1.6481	0.6957 $\pm$ 0.0641	0.1985 $\pm$ 0.0247	28.1446 $\pm$ 2.4702	0.9421 $\pm$ 0.0297	0.0718 $\pm$ 0.0118
Ob50	20.8876 $\pm$ 1.9520	0.6606 $\pm$ 0.0436	0.2738 $\pm$ 0.0292	34.9999 $\pm$ 2.0979	0.8937 $\pm$ 0.0249	0.0866 $\pm$ 0.0123
Ob51	15.7411 $\pm$ 1.0936	0.7031 $\pm$ 0.0729	0.2117 $\pm$ 0.0474	28.5702 $\pm$ 2.0610	0.9426 $\pm$ 0.0137	0.0998 $\pm$ 0.0192
Ob52	15.9858 $\pm$ 0.8460	0.6714 $\pm$ 0.0401	0.3584 $\pm$ 0.0561	29.6734 $\pm$ 2.3318	0.8893 $\pm$ 0.0277	0.0877 $\pm$ 0.0201
Ob53	12.7983 $\pm$ 0.8879	0.5855 $\pm$ 0.0633	0.3721 $\pm$ 0.0318	38.1437 $\pm$ 4.5153	0.9655 $\pm$ 0.0123	0.0292 $\pm$ 0.0082
Ob54	13.4389 $\pm$ 1.1735	0.6446 $\pm$ 0.0528	0.3155 $\pm$ 0.0396	26.9014 $\pm$ 2.2075	0.9526 $\pm$ 0.0152	0.0554 $\pm$ 0.0122
Ob55	8.6810 $\pm$ 0.8368	0.4707 $\pm$ 0.0899	0.5540 $\pm$ 0.0364	32.0861 $\pm$ 4.6148	0.9687 $\pm$ 0.0071	0.0373 $\pm$ 0.0074
Ob56	12.3733 $\pm$ 0.6699	0.3751 $\pm$ 0.0486	0.4692 $\pm$ 0.0251	35.7062 $\pm$ 4.0166	0.9912 $\pm$ 0.0022	0.0200 $\pm$ 0.0035
Ob57	16.5078 $\pm$ 0.5755	0.7137 $\pm$ 0.0575	0.2823 $\pm$ 0.0442	30.3623 $\pm$ 1.8662	0.9164 $\pm$ 0.0151	0.0716 $\pm$ 0.0083
Ob58	10.4022 $\pm$ 0.9032	0.1574 $\pm$ 0.0309	0.6065 $\pm$ 0.0369	100.0000 $\pm$ 0.0000	1.0000 $\pm$ 0.0000	0.0000 $\pm$ 0.0000
Ob59	12.4620 $\pm$ 1.2338	0.5258 $\pm$ 0.0866	0.5517 $\pm$ 0.0773	22.1999 $\pm$ 3.0984	0.9173 $\pm$ 0.0231	0.0984 $\pm$ 0.0242
Ob60	10.4857 $\pm$ 1.3116	0.6126 $\pm$ 0.0475	0.5178 $\pm$ 0.0621	18.0625 $\pm$ 3.4305	0.8908 $\pm$ 0.0763	0.0935 $\pm$ 0.0575
Ob61	14.9765 $\pm$ 1.0903	0.5813 $\pm$ 0.0406	0.2695 $\pm$ 0.0322	29.4410 $\pm$ 1.9612	0.9443 $\pm$ 0.0127	0.0654 $\pm$ 0.0074
Ob62	11.8853 $\pm$ 1.2600	0.6447 $\pm$ 0.0390	0.4003 $\pm$ 0.0381	30.3483 $\pm$ 2.2787	0.9755 $\pm$ 0.0127	0.0244 $\pm$ 0.0065
Ob63	12.1424 $\pm$ 0.9430	0.6345 $\pm$ 0.0290	0.3997 $\pm$ 0.0327	30.0790 $\pm$ 2.4098	0.9723 $\pm$ 0.0031	0.0347 $\pm$ 0.0103
Ob64	15.0763 $\pm$ 0.8425	0.4963 $\pm$ 0.0688	0.4925 $\pm$ 0.0327	33.0838 $\pm$ 1.0676	0.8280 $\pm$ 0.0240	0.1589 $\pm$ 0.0166
Ob65	15.0364 $\pm$ 0.9995	0.6496 $\pm$ 0.0401	0.2179 $\pm$ 0.0376	31.3611 $\pm$ 1.5654	0.9818 $\pm$ 0.0098	0.0352 $\pm$ 0.0067
Ob66	11.7368 $\pm$ 1.1874	0.5384 $\pm$ 0.0391	0.5506 $\pm$ 0.0495	20.8369 $\pm$ 1.9264	0.8985 $\pm$ 0.0200	0.1092 $\pm$ 0.0201
Ob67	6.4969 $\pm$ 0.3899	0.4735 $\pm$ 0.0170	0.6288 $\pm$ 0.0392	28.7444 $\pm$ 2.8747	0.9062 $\pm$ 0.0327	0.0867 $\pm$ 0.0195
Average	14.1311	0.5633	0.3506	32.9721	0.9404	0.0621

Table B.5: NVS Results for Nerfacto with COLMAP-initialized poses and pose optimization.

	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	BG PSNR $\uparrow$	BG SSIM $\uparrow$	BG LPIPS $\downarrow$
Ob01	18.2889 $\pm$ 1.3123	0.6581 $\pm$ 0.0699	0.1884 $\pm$ 0.0247	44.2930 $\pm$ 2.9987	0.9985 $\pm$ 0.0004	0.0040 $\pm$ 0.0008
Ob02	15.7618 $\pm$ 1.7773	0.6732 $\pm$ 0.0649	0.2919 $\pm$ 0.0607	28.4662 $\pm$ 1.3422	0.9903 $\pm$ 0.0024	0.0238 $\pm$ 0.0059
Ob03	14.8321 $\pm$ 0.5932	0.7183 $\pm$ 0.0343	0.3342 $\pm$ 0.0346	30.5217 $\pm$ 2.5332	0.9927 $\pm$ 0.0024	0.0152 $\pm$ 0.0042
Ob04	14.5645 $\pm$ 1.0926	0.6138 $\pm$ 0.0651	0.3181 $\pm$ 0.0725	28.7304 $\pm$ 2.6089	0.9928 $\pm$ 0.0019	0.0169 $\pm$ 0.0028
Ob05	15.9571 $\pm$ 1.1825	0.7001 $\pm$ 0.0465	0.1533 $\pm$ 0.0234	38.7232 $\pm$ 3.9928	0.9971 $\pm$ 0.0011	0.0081 $\pm$ 0.0027
Ob06	16.9219 $\pm$ 1.0109	0.6996 $\pm$ 0.0547	0.1994 $\pm$ 0.0289	43.6600 $\pm$ 2.8276	0.9982 $\pm$ 0.0005	0.0052 $\pm$ 0.0014
Ob07	18.6613 $\pm$ 1.4258	0.7719 $\pm$ 0.0345	0.1901 $\pm$ 0.0289	41.2731 $\pm$ 2.5438	0.9979 $\pm$ 0.0007	0.0060 $\pm$ 0.0021
Ob08	17.2134 $\pm$ 1.8402	0.7535 $\pm$ 0.0791	0.2757 $\pm$ 0.0846	35.9280 $\pm$ 2.6649	0.9950 $\pm$ 0.0013	0.0092 $\pm$ 0.0022
Ob09	18.3966 $\pm$ 2.2720	0.5983 $\pm$ 0.0471	0.2000 $\pm$ 0.0179	52.0694 $\pm$ 4.0406	0.9996 $\pm$ 0.0003	0.0014 $\pm$ 0.0008
Ob10	16.9118 $\pm$ 1.1390	0.4556 $\pm$ 0.0687	0.2834 $\pm$ 0.0582	35.9407 $\pm$ 3.0096	0.9957 $\pm$ 0.0013	0.0088 $\pm$ 0.0023
Ob11	15.9357 $\pm$ 1.5702	0.6610 $\pm$ 0.0533	0.3928 $\pm$ 0.0834	29.4342 $\pm$ 2.1579	0.9893 $\pm$ 0.0033	0.0264 $\pm$ 0.0076
Ob12	12.3338 $\pm$ 0.9530	0.3683 $\pm$ 0.0348	0.5673 $\pm$ 0.0345	27.8906 $\pm$ 2.4910	0.9873 $\pm$ 0.0039	0.0204 $\pm$ 0.0048
Ob13	14.1948 $\pm$ 1.4689	0.7217 $\pm$ 0.0553	0.3436 $\pm$ 0.0915	26.7737 $\pm$ 3.5670	0.9886 $\pm$ 0.0046	0.0227 $\pm$ 0.0075
Ob14	17.0737 $\pm$ 2.0061	0.6728 $\pm$ 0.0380	0.1906 $\pm$ 0.0343	37.3496 $\pm$ 1.6046	0.9968 $\pm$ 0.0006	0.0084 $\pm$ 0.0011
Ob15	15.5694 $\pm$ 1.3086	0.5016 $\pm$ 0.0562	0.2329 $\pm$ 0.0215	52.6519 $\pm$ 8.0470	0.9994 $\pm$ 0.0006	0.0019 $\pm$ 0.0015
Ob16	15.0016 $\pm$ 0.9038	0.6041 $\pm$ 0.0516	0.2652 $\pm$ 0.0246	38.7998 $\pm$ 1.8836	0.9966 $\pm$ 0.0008	0.0081 $\pm$ 0.0016
Ob17	15.3802 $\pm$ 0.9394	0.7126 $\pm$ 0.0741	0.2385 $\pm$ 0.0521	30.7341 $\pm$ 2.6895	0.9939 $\pm$ 0.0009	0.0123 $\pm$ 0.0015
Ob18	18.5930 $\pm$ 1.7149	0.7516 $\pm$ 0.0611	0.1996 $\pm$ 0.0233	44.6277 $\pm$ 2.6558	0.9987 $\pm$ 0.0004	0.0040 $\pm$ 0.0011
Ob19	14.4490 $\pm$ 0.5886	0.5534 $\pm$ 0.0770	0.2069 $\pm$ 0.0167	33.3966 $\pm$ 1.2790	0.9968 $\pm$ 0.0005	0.0093 $\pm$ 0.0010
Ob20	16.1019 $\pm$ 1.0000	0.5042 $\pm$ 0.0507	0.3208 $\pm$ 0.0347	35.1582 $\pm$ 3.5517	0.9965 $\pm$ 0.0011	0.0071 $\pm$ 0.0019
Ob21	14.8235 $\pm$ 1.0192	0.6551 $\pm$ 0.0856	0.2149 $\pm$ 0.0143	39.1036 $\pm$ 3.1878	0.9974 $\pm$ 0.0008	0.0067 $\pm$ 0.0014
Ob22	13.5990 $\pm$ 1.2906	0.5435 $\pm$ 0.0668	0.2803 $\pm$ 0.0660	44.5855 $\pm$ 5.3312	0.9986 $\pm$ 0.0007	0.0035 $\pm$ 0.0017
Ob23	14.4731 $\pm$ 1.1161	0.6475 $\pm$ 0.0493	0.2177 $\pm$ 0.0361	39.6708 $\pm$ 3.4198	0.9977 $\pm$ 0.0005	0.0056 $\pm$ 0.0014
Ob24	16.1826 $\pm$ 0.6621	0.5282 $\pm$ 0.0521	0.2222 $\pm$ 0.0344	45.8961 $\pm$ 2.6527	0.9989 $\pm$ 0.0004	0.0034 $\pm$ 0.0012
Ob25	15.1102 $\pm$ 0.9136	0.5215 $\pm$ 0.0744	0.4403 $\pm$ 0.0497	34.7708 $\pm$ 1.1267	0.9955 $\pm$ 0.0009	0.0114 $\pm$ 0.0026
Ob26	17.4101 $\pm$ 1.3996	0.5918 $\pm$ 0.0530	0.2114 $\pm$ 0.0196	48.4479 $\pm$ 4.0208	0.9992 $\pm$ 0.0004	0.0024 $\pm$ 0.0012
Ob27	17.8139 $\pm$ 1.6154	0.5737 $\pm$ 0.0745	0.2570 $\pm$ 0.0300	44.5000 $\pm$ 4.7640	0.9986 $\pm$ 0.0008	0.0035 $\pm$ 0.0016
Ob28	16.6636 $\pm$ 1.0656	0.6251 $\pm$ 0.0686	0.2500 $\pm$ 0.0365	36.3163 $\pm$ 1.4982	0.9970 $\pm$ 0.0007	0.0065 $\pm$ 0.0013
Ob29	15.4171 $\pm$ 3.1964	0.5392 $\pm$ 0.1099	0.3751 $\pm$ 0.1379	33.7524 $\pm$ 4.8705	0.9887 $\pm$ 0.0083	0.0194 $\pm$ 0.0117
Ob30	15.5200 $\pm$ 0.9324	0.7440 $\pm$ 0.0399	0.1856 $\pm$ 0.0208	38.8248 $\pm$ 2.7239	0.9969 $\pm$ 0.0008	0.0086 $\pm$ 0.0018
Ob31	14.7297 $\pm$ 1.4474	0.5023 $\pm$ 0.1139	0.1825 $\pm$ 0.0388	43.4611 $\pm$ 7.3334	0.9983 $\pm$ 0.0011	0.0048 $\pm$ 0.0024
Ob32	17.7169 $\pm$ 0.9059	0.5976 $\pm$ 0.0628	0.2153 $\pm$ 0.0243	39.8898 $\pm$ 3.9008	0.9983 $\pm$ 0.0005	0.0045 $\pm$ 0.0011
Ob33	17.2793 $\pm$ 2.2407	0.6466 $\pm$ 0.1058	0.1668 $\pm$ 0.0389	39.9387 $\pm$ 4.8379	0.9975 $\pm$ 0.0012	0.0068 $\pm$ 0.0026
Ob34	18.6274 $\pm$ 1.0021	0.7087 $\pm$ 0.0387	0.2084 $\pm$ 0.0209	33.1124 $\pm$ 1.8714	0.9954 $\pm$ 0.0007	0.0104 $\pm$ 0.0012
Ob35	18.0349 $\pm$ 1.2443	0.6048 $\pm$ 0.0886	0.2711 $\pm$ 0.0256	51.6789 $\pm$ 3.6535	0.9993 $\pm$ 0.0004	0.0014 $\pm$ 0.0006
Ob36	14.6288 $\pm$ 0.8893	0.5945 $\pm$ 0.1402	0.1947 $\pm$ 0.0327	35.4236 $\pm$ 2.2282	0.9963 $\pm$ 0.0008	0.0088 $\pm$ 0.0011
Ob37	16.2226 $\pm$ 1.5377	0.5681 $\pm$ 0.1399	0.2099 $\pm$ 0.0658	36.1195 $\pm$ 4.5679	0.9960 $\pm$ 0.0021	0.0095 $\pm$ 0.0036
Ob38	16.5842 $\pm$ 1.0331	0.6341 $\pm$ 0.0867	0.3135 $\pm$ 0.0616	37.1973 $\pm$ 3.3644	0.9968 $\pm$ 0.0013	0.0080 $\pm$ 0.0028
Ob39	15.9157 $\pm$ 2.5212	0.6671 $\pm$ 0.0984	0.1795 $\pm$ 0.0585	33.1081 $\pm$ 2.2007	0.9959 $\pm$ 0.0010	0.0120 $\pm$ 0.0022
Ob40	10.8444 $\pm$ 0.9596	0.7082 $\pm$ 0.0433	0.1776 $\pm$ 0.0292	33.6304 $\pm$ 2.6771	0.9951 $\pm$ 0.0010	0.0127 $\pm$ 0.0022
Ob41	14.8359 $\pm$ 1.6517	0.6683 $\pm$ 0.0677	0.4311 $\pm$ 0.1256	25.0509 $\pm$ 2.9590	0.9859 $\pm$ 0.0081	0.0258 $\pm$ 0.0098
Ob42	15.3901 $\pm$ 1.2877	0.6762 $\pm$ 0.0583	0.2087 $\pm$ 0.0266	45.2177 $\pm$ 4.1273	0.9989 $\pm$ 0.0005	0.0036 $\pm$ 0.0014
Ob43	16.3691 $\pm$ 0.7819	0.5565 $\pm$ 0.0334	0.2132 $\pm$ 0.0227	34.9357 $\pm$ 2.0664	0.9945 $\pm$ 0.0011	0.0127 $\pm$ 0.0022
Ob44	15.6955 $\pm$ 1.2593	0.6813 $\pm$ 0.0541	0.1641 $\pm$ 0.0206	31.9713 $\pm$ 2.8175	0.9941 $\pm$ 0.0061	0.0135 $\pm$ 0.0081
Ob45	18.7475 $\pm$ 2.1625	0.6418 $\pm$ 0.0473	0.2089 $\pm$ 0.0336	44.6144 $\pm$ 2.4346	0.9982 $\pm$ 0.0005	0.0040 $\pm$ 0.0008
Ob46	13.1379 $\pm$ 0.7889	0.5949 $\pm$ 0.0497	0.3183 $\pm$ 0.0442	38.0201 $\pm$ 3.4235	0.9980 $\pm$ 0.0006	0.0043 $\pm$ 0.0010
Ob47	14.8164 $\pm$ 1.5044	0.5905 $\pm$ 0.0857	0.1902 $\pm$ 0.0462	36.1440 $\pm$ 2.2869	0.9962 $\pm$ 0.0008	0.0096 $\pm$ 0.0013
Ob48	15.8326 $\pm$ 1.0092	0.7360 $\pm$ 0.0502	0.2172 $\pm$ 0.0367	27.3225 $\pm$ 1.5949	0.9904 $\pm$ 0.0019	0.0202 $\pm$ 0.0039
Ob49	16.4267 $\pm$ 1.4571	0.7274 $\pm$ 0.0521	0.1515 $\pm$ 0.0209	32.5193 $\pm$ 2.0624	0.9937 $\pm$ 0.0010	0.0157 $\pm$ 0.0020
Ob50	20.4905 $\pm$ 1.5925	0.6870 $\pm$ 0.0362	0.2435 $\pm$ 0.0225	48.7294 $\pm$ 4.8102	0.9989 $\pm$ 0.0006	0.0021 $\pm$ 0.0010
Ob51	16.2963 $\pm$ 0.9809	0.7338 $\pm$ 0.0548	0.1754 $\pm$ 0.0395	35.1289 $\pm$ 1.7159	0.9953 $\pm$ 0.0010	0.0141 $\pm$ 0.0038
Ob52	16.7977 $\pm$ 1.1159	0.7318 $\pm$ 0.0390	0.2569 $\pm$ 0.0489	37.9337 $\pm$ 1.4097	0.9955 $\pm$ 0.0011	0.0101 $\pm$ 0.0021
Ob53	15.4455 $\pm$ 1.4647	0.6685 $\pm$ 0.0461	0.2371 $\pm$ 0.0251	43.0745 $\pm$ 6.3377	0.9984 $\pm$ 0.0011	0.0048 $\pm$ 0.0027
Ob54	15.5677 $\pm$ 1.8778	0.6923 $\pm$ 0.0571	0.2667 $\pm$ 0.0581	29.9789 $\pm$ 2.9103	0.9930 $\pm$ 0.0021	0.0137 $\pm$ 0.0038
Ob55	10.3052 $\pm$ 1.3808	0.5436 $\pm$ 0.1248	0.2996 $\pm$ 0.0848	34.2933 $\pm$ 5.2904	0.9967 $\pm$ 0.0018	0.0090 $\pm$ 0.0032
Ob56	12.9184 $\pm$ 0.9337	0.4400 $\pm$ 0.0383	0.3199 $\pm$ 0.0234	37.9158 $\pm$ 4.1317	0.9977 $\pm$ 0.0010	0.0054 $\pm$ 0.0019
Ob57	17.0121 $\pm$ 0.8124	0.7406 $\pm$ 0.0529	0.2524 $\pm$ 0.0446	34.3122 $\pm$ 2.9763	0.9954 $\pm$ 0.0012	0.0115 $\pm$ 0.0027
Ob58	17.7000 $\pm$ 0.6544	0.6561 $\pm$ 0.0264	0.2440 $\pm$ 0.0209	48.8794 $\pm$ 4.6583	0.9992 $\pm$ 0.0003	0.0015 $\pm$ 0.0006
Ob59	14.5945 $\pm$ 1.0350	0.6565 $\pm$ 0.0435	0.5291 $\pm$ 0.0800	27.0836 $\pm$ 3.6819	0.9837 $\pm$ 0.0076	0.0352 $\pm$ 0.0139
Ob60	12.6551 $\pm$ 0.4206	0.6621 $\pm$ 0.0415	0.4186 $\pm$ 0.0209	23.7629 $\pm$ 1.2198	0.9827 $\pm$ 0.0020	0.0294 $\pm$ 0.0034
Ob61	16.4072 $\pm$ 1.3674	0.6427 $\pm$ 0.0446	0.2313 $\pm$ 0.0304	35.2833 $\pm$ 3.6860	0.9957 $\pm$ 0.0013	0.0092 $\pm$ 0.0021
Ob62	13.3733 $\pm$ 1.1567	0.6810 $\pm$ 0.0263	0.3598 $\pm$ 0.0487	32.8746 $\pm$ 2.3083	0.9964 $\pm$ 0.0007	0.0069 $\pm$ 0.0011
Ob63	12.8376 $\pm$ 0.7928	0.6325 $\pm$ 0.0331	0.4158 $\pm$ 0.0287	32.3159 $\pm$ 2.4411	0.9956 $\pm$ 0.0013	0.0098 $\pm$ 0.0025
Ob64	20.2673 $\pm$ 1.5728	0.7776 $\pm$ 0.0376	0.2536 $\pm$ 0.0392	39.8519 $\pm$ 3.3824	0.9961 $\pm$ 0.0015	0.0074 $\pm$ 0.0029
Ob65	16.4662 $\pm$ 0.9081	0.6866 $\pm$ 0.0411	0.1806 $\pm$ 0.0317	34.9664 $\pm$ 1.8712	0.9960 $\pm$ 0.0006	0.0092 $\pm$ 0.0012
Ob66	15.0905 $\pm$ 0.8704	0.6546 $\pm$ 0.0531	0.4253 $\pm$ 0.0523	25.9045 $\pm$ 2.0547	0.9854 $\pm$ 0.0042	0.0307 $\pm$ 0.0064
Ob67	14.0585 $\pm$ 1.1671	0.6652 $\pm$ 0.0505	0.4059 $\pm$ 0.0368	24.9427 $\pm$ 3.2712	0.9838 $\pm$ 0.0056	0.0293 $\pm$ 0.0093
Average	15.8101	0.6376	0.2654	36.8489	0.9951	0.0107

Table B.6: NVS results for Splatfacto with COLMAP-initialized poses and no pose optimization.

	PSNR↑	SSIM↑	LPIPS↓	BG PSNR↑	BG SSIM↑	BG LPIPS↓
Ob01	18.1675 ± 1.4447	0.6525 ± 0.0778	0.1775 ± 0.0280	44.0161 ± 2.7611	0.9986 ± 0.0004	0.0042 ± 0.0010
Ob02	15.5458 ± 1.4987	0.6663 ± 0.0610	0.2857 ± 0.0624	28.3758 ± 1.3375	0.9903 ± 0.0024	0.0242 ± 0.0062
Ob03	14.7817 ± 0.5867	0.7185 ± 0.0373	0.3255 ± 0.0366	30.4751 ± 2.8492	0.9931 ± 0.0020	0.0145 ± 0.0042
Ob04	14.5628 ± 0.9811	0.6146 ± 0.0678	0.2930 ± 0.0749	28.7140 ± 2.3425	0.9930 ± 0.0015	0.0169 ± 0.0024
Ob05	16.1960 ± 1.1884	0.7024 ± 0.0648	0.1483 ± 0.0259	38.6194 ± 3.6773	0.9971 ± 0.0010	0.0081 ± 0.0024
Ob06	16.7341 ± 0.8520	0.6883 ± 0.0494	0.1861 ± 0.0253	43.7627 ± 3.1245	0.9982 ± 0.0007	0.0053 ± 0.0018
Ob07	18.3536 ± 1.2498	0.7387 ± 0.0377	0.1861 ± 0.0274	41.0164 ± 3.2751	0.9980 ± 0.0007	0.0060 ± 0.0021
Ob08	17.2011 ± 1.8212	0.7532 ± 0.0788	0.2679 ± 0.0787	36.4620 ± 2.7211	0.9954 ± 0.0012	0.0087 ± 0.0022
Ob09	18.2555 ± 2.2535	0.5895 ± 0.0447	0.1908 ± 0.0219	49.4674 ± 3.6071	0.9994 ± 0.0003	0.0021 ± 0.0010
Ob10	16.8628 ± 1.0893	0.4541 ± 0.0672	0.2834 ± 0.0543	35.9189 ± 3.2747	0.9960 ± 0.0012	0.0083 ± 0.0025
Ob11	15.9128 ± 1.6490	0.6612 ± 0.0518	0.3929 ± 0.0913	29.5342 ± 2.3969	0.9893 ± 0.0031	0.0255 ± 0.0060
Ob12	12.2940 ± 0.9594	0.3650 ± 0.0355	0.5671 ± 0.0350	27.7846 ± 2.4442	0.9869 ± 0.0040	0.0210 ± 0.0049
Ob13	14.0929 ± 1.4755	0.7219 ± 0.0546	0.3471 ± 0.0927	26.6642 ± 3.5378	0.9885 ± 0.0046	0.0229 ± 0.0075
Ob14	16.8178 ± 1.8887	0.6684 ± 0.0368	0.1824 ± 0.0373	37.0923 ± 1.5449	0.9967 ± 0.0006	0.0091 ± 0.0012
Ob15	15.6290 ± 1.3251	0.5090 ± 0.0631	0.2076 ± 0.0263	51.6111 ± 6.7977	0.9994 ± 0.0005	0.0020 ± 0.0016
Ob16	14.9834 ± 0.8515	0.5985 ± 0.0596	0.2454 ± 0.0244	37.9661 ± 1.1936	0.9963 ± 0.0004	0.0090 ± 0.0010
Ob17	15.4430 ± 0.9302	0.7117 ± 0.0732	0.2219 ± 0.0378	31.2204 ± 2.5855	0.9946 ± 0.0010	0.0117 ± 0.0018
Ob18	18.5567 ± 1.7196	0.7385 ± 0.0742	0.1932 ± 0.0245	45.1866 ± 3.2146	0.9988 ± 0.0004	0.0038 ± 0.0013
Ob19	14.4489 ± 0.6068	0.5485 ± 0.0756	0.2046 ± 0.0176	33.4662 ± 1.3191	0.9968 ± 0.0005	0.0094 ± 0.0011
Ob20	15.9045 ± 1.0125	0.4907 ± 0.0599	0.3158 ± 0.0325	34.8749 ± 3.5810	0.9965 ± 0.0010	0.0072 ± 0.0016
Ob21	14.7375 ± 0.9862	0.6709 ± 0.0766	0.2089 ± 0.0159	40.3661 ± 5.8145	0.9976 ± 0.0011	0.0058 ± 0.0020
Ob22	13.9812 ± 1.4394	0.5626 ± 0.0744	0.2731 ± 0.0648	44.9400 ± 5.3226	0.9987 ± 0.0007	0.0034 ± 0.0016
Ob23	15.0208 ± 1.5355	0.6800 ± 0.0525	0.2058 ± 0.0374	41.1740 ± 3.5046	0.9980 ± 0.0005	0.0050 ± 0.0017
Ob24	16.1288 ± 0.6531	0.5291 ± 0.0521	0.2193 ± 0.0304	46.0544 ± 2.5092	0.9989 ± 0.0003	0.0033 ± 0.0011
Ob25	15.1739 ± 0.9338	0.5340 ± 0.0753	0.4372 ± 0.0443	34.9678 ± 1.2496	0.9957 ± 0.0009	0.0114 ± 0.0027
Ob26	17.2369 ± 1.4341	0.5792 ± 0.0464	0.1965 ± 0.0154	48.0521 ± 3.5059	0.9993 ± 0.0004	0.0024 ± 0.0010
Ob27	17.7440 ± 1.4433	0.5752 ± 0.0523	0.2071 ± 0.0225	44.5386 ± 5.9196	0.9986 ± 0.0008	0.0037 ± 0.0018
Ob28	16.8643 ± 1.2991	0.6178 ± 0.0696	0.2497 ± 0.0367	35.3622 ± 1.4021	0.9966 ± 0.0007	0.0073 ± 0.0013
Ob29	15.1300 ± 3.3129	0.5318 ± 0.1104	0.3707 ± 0.1272	35.4220 ± 3.8615	0.9911 ± 0.0057	0.0161 ± 0.0091
Ob30	15.5224 ± 1.0270	0.7370 ± 0.0576	0.1760 ± 0.0149	39.0245 ± 2.6303	0.9971 ± 0.0008	0.0084 ± 0.0020
Ob31	14.1965 ± 0.9361	0.4556 ± 0.0679	0.1862 ± 0.0359	42.2728 ± 7.4079	0.9982 ± 0.0012	0.0052 ± 0.0026
Ob32	18.1175 ± 1.0584	0.6352 ± 0.0828	0.2086 ± 0.0250	39.1204 ± 3.6752	0.9982 ± 0.0005	0.0050 ± 0.0011
Ob33	17.3891 ± 2.5010	0.6554 ± 0.1145	0.1631 ± 0.0470	40.2514 ± 5.4398	0.9975 ± 0.0013	0.0068 ± 0.0028
Ob34	18.7831 ± 1.1415	0.7262 ± 0.0371	0.1786 ± 0.0283	34.8272 ± 2.4216	0.9963 ± 0.0006	0.0090 ± 0.0013
Ob35	17.9810 ± 1.1653	0.6108 ± 0.0840	0.2624 ± 0.0269	51.2424 ± 3.7667	0.9993 ± 0.0004	0.0015 ± 0.0007
Ob36	14.7282 ± 0.9301	0.5972 ± 0.1375	0.1650 ± 0.0266	35.4876 ± 2.5048	0.9966 ± 0.0008	0.0084 ± 0.0009
Ob37	16.3621 ± 1.6653	0.5757 ± 0.1173	0.1726 ± 0.0452	35.0773 ± 3.9235	0.9956 ± 0.0019	0.0107 ± 0.0034
Ob38	16.3160 ± 1.1139	0.6252 ± 0.0892	0.3203 ± 0.0705	36.1080 ± 3.2062	0.9965 ± 0.0012	0.0087 ± 0.0026
Ob39	16.0432 ± 2.5567	0.6813 ± 0.1057	0.1672 ± 0.0531	33.3895 ± 2.1457	0.9960 ± 0.0009	0.0119 ± 0.0021
Ob40	11.0092 ± 1.0614	0.7194 ± 0.0522	0.1675 ± 0.0330	34.1998 ± 2.6960	0.9956 ± 0.0010	0.0121 ± 0.0022
Ob41	15.0680 ± 1.5807	0.6701 ± 0.0696	0.4388 ± 0.1265	24.3987 ± 3.2247	0.9835 ± 0.0100	0.0277 ± 0.0114
Ob42	15.5606 ± 1.4137	0.6839 ± 0.0502	0.1965 ± 0.0197	44.7564 ± 4.7160	0.9989 ± 0.0005	0.0039 ± 0.0015
Ob43	16.1181 ± 0.5397	0.5445 ± 0.0248	0.2084 ± 0.0234	34.8318 ± 2.0846	0.9944 ± 0.0011	0.0130 ± 0.0020
Ob44	15.3603 ± 1.3204	0.6339 ± 0.0696	0.1682 ± 0.0215	31.9410 ± 1.9879	0.9950 ± 0.0024	0.0126 ± 0.0042
Ob45	18.5924 ± 1.8989	0.6344 ± 0.0414	0.2018 ± 0.0368	44.7109 ± 2.4112	0.9982 ± 0.0005	0.0039 ± 0.0008
Ob46	13.2792 ± 0.7944	0.5986 ± 0.0504	0.3186 ± 0.0442	38.1387 ± 3.2988	0.9980 ± 0.0006	0.0043 ± 0.0010
Ob47	14.8212 ± 1.5780	0.5972 ± 0.0877	0.1913 ± 0.0482	36.4298 ± 2.2926	0.9963 ± 0.0009	0.0095 ± 0.0014
Ob48	16.0492 ± 1.0304	0.7500 ± 0.0525	0.2018 ± 0.0406	27.7366 ± 1.6935	0.9912 ± 0.0016	0.0193 ± 0.0035
Ob49	16.2327 ± 1.6436	0.7060 ± 0.0581	0.1402 ± 0.0240	32.8751 ± 2.0868	0.9937 ± 0.0010	0.0157 ± 0.0025
Ob50	20.5944 ± 1.7276	0.7032 ± 0.0447	0.2438 ± 0.0216	49.4655 ± 4.7383	0.9990 ± 0.0006	0.0019 ± 0.0009
Ob51	16.4043 ± 1.1792	0.7306 ± 0.0690	0.1687 ± 0.0434	35.1274 ± 1.3338	0.9956 ± 0.0007	0.0135 ± 0.0023
Ob52	16.7962 ± 1.1756	0.7316 ± 0.0455	0.2500 ± 0.0528	37.6559 ± 1.1792	0.9952 ± 0.0011	0.0107 ± 0.0021
Ob53	15.5303 ± 1.5141	0.6956 ± 0.0684	0.2167 ± 0.0311	44.6147 ± 6.7467	0.9987 ± 0.0009	0.0042 ± 0.0023
Ob54	15.2004 ± 1.2365	0.6875 ± 0.0515	0.2505 ± 0.0426	30.5184 ± 2.8702	0.9942 ± 0.0019	0.0119 ± 0.0034
Ob55	10.3312 ± 1.4245	0.5459 ± 0.1327	0.2963 ± 0.0914	34.3431 ± 5.8527	0.9969 ± 0.0019	0.0092 ± 0.0034
Ob56	12.9672 ± 0.8520	0.4390 ± 0.0395	0.3187 ± 0.0225	37.9037 ± 3.8655	0.9978 ± 0.0009	0.0054 ± 0.0018
Ob57	17.1984 ± 0.8761	0.7443 ± 0.0604	0.2253 ± 0.0342	34.2544 ± 2.0601	0.9954 ± 0.0008	0.0116 ± 0.0019
Ob58	17.5984 ± 0.8008	0.6518 ± 0.0319	0.2403 ± 0.0206	49.9692 ± 4.5785	0.9992 ± 0.0004	0.0014 ± 0.0005
Ob59	14.5310 ± 0.9796	0.6565 ± 0.0415	0.5269 ± 0.0760	27.0730 ± 3.5689	0.9841 ± 0.0071	0.0345 ± 0.0128
Ob60	12.6604 ± 0.4390	0.6641 ± 0.0411	0.4109 ± 0.0205	23.7853 ± 1.3690	0.9830 ± 0.0020	0.0294 ± 0.0035
Ob61	16.0800 ± 1.1982	0.6159 ± 0.0326	0.1948 ± 0.0303	35.6395 ± 3.1369	0.9961 ± 0.0010	0.0089 ± 0.0017
Ob62	13.3695 ± 1.0870	0.6822 ± 0.0259	0.3569 ± 0.0492	32.9096 ± 2.2773	0.9965 ± 0.0007	0.0068 ± 0.0011
Ob63	12.8400 ± 0.6982	0.6401 ± 0.0287	0.4136 ± 0.0293	32.2459 ± 2.8164	0.9956 ± 0.0014	0.0099 ± 0.0026
Ob64	20.6094 ± 1.7193	0.7821 ± 0.0450	0.2381 ± 0.0429	40.9637 ± 4.1246	0.9967 ± 0.0014	0.0063 ± 0.0026
Ob65	16.2744 ± 1.0077	0.6807 ± 0.0405	0.1777 ± 0.0324	35.0115 ± 1.9339	0.9962 ± 0.0005	0.0089 ± 0.0010
Ob66	14.8357 ± 0.9015	0.6497 ± 0.0552	0.4358 ± 0.0559	25.7764 ± 2.2485	0.9855 ± 0.0047	0.0304 ± 0.0071
Ob67	13.9144 ± 1.2840	0.6653 ± 0.0528	0.4029 ± 0.0420	25.1533 ± 3.2760	0.9842 ± 0.0054	0.0294 ± 0.0092
Average	15.7914	0.6369	0.2566	36.9005	0.9953	0.0106

Table B.7: NVS results for Splatfacto with COLMAP-initialized poses and pose optimization.

	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	BG_PSNR $\uparrow$	BG_SSIM $\uparrow$	BG_LPIPS $\downarrow$
Ob01	13.3907 $\pm$ 0.7790	0.4283 $\pm$ 0.0654	0.4912 $\pm$ 0.0602	25.2960 $\pm$ 1.0714	0.9339 $\pm$ 0.0210	0.0676 $\pm$ 0.0145
Ob02	13.1115 $\pm$ 1.2768	0.5777 $\pm$ 0.0636	0.3902 $\pm$ 0.0624	23.2196 $\pm$ 1.9034	0.8849 $\pm$ 0.0640	0.1471 $\pm$ 0.0294
Ob03	9.2294 $\pm$ 1.2746	0.5543 $\pm$ 0.0336	0.5197 $\pm$ 0.0586	20.9229 $\pm$ 1.9535	0.9126 $\pm$ 0.0149	0.1153 $\pm$ 0.0145
Ob04	13.1898 $\pm$ 0.4934	0.5483 $\pm$ 0.0388	0.3846 $\pm$ 0.0493	24.5132 $\pm$ 1.1566	0.9211 $\pm$ 0.0190	0.1151 $\pm$ 0.0227
Ob05	12.9433 $\pm$ 0.9608	0.5903 $\pm$ 0.0412	0.3396 $\pm$ 0.0366	28.4805 $\pm$ 0.8510	0.9629 $\pm$ 0.0087	0.0746 $\pm$ 0.0085
Ob06	13.8001 $\pm$ 1.0115	0.5167 $\pm$ 0.0366	0.4414 $\pm$ 0.0520	26.5204 $\pm$ 0.7208	0.9270 $\pm$ 0.0208	0.0939 $\pm$ 0.0181
Ob07	7.7348 $\pm$ 0.7650	0.2054 $\pm$ 0.0659	0.5773 $\pm$ 0.0479	38.0153 $\pm$ 9.8928	0.9530 $\pm$ 0.0532	0.2015 $\pm$ 0.0806
Ob08	14.5899 $\pm$ 1.0890	0.7144 $\pm$ 0.0693	0.2786 $\pm$ 0.0553	34.0969 $\pm$ 2.4198	0.9251 $\pm$ 0.0243	0.0820 $\pm$ 0.0117
Ob09	15.8209 $\pm$ 1.4823	0.3979 $\pm$ 0.0392	0.3294 $\pm$ 0.0395	33.5910 $\pm$ 1.2222	0.9502 $\pm$ 0.0131	0.0626 $\pm$ 0.0094
Ob10	9.1019 $\pm$ 0.2470	0.1047 $\pm$ 0.0339	0.6904 $\pm$ 0.0179	50.6827 $\pm$ 16.4922	0.9785 $\pm$ 0.0287	0.0426 $\pm$ 0.0577
Ob11	13.7033 $\pm$ 1.5386	0.5793 $\pm$ 0.0441	0.4607 $\pm$ 0.0739	24.3812 $\pm$ 1.5816	0.9458 $\pm$ 0.0095	0.0863 $\pm$ 0.0070
Ob12	11.6781 $\pm$ 1.3445	0.3263 $\pm$ 0.0750	0.5519 $\pm$ 0.1089	20.9661 $\pm$ 7.0545	0.7803 $\pm$ 0.3161	0.1337 $\pm$ 0.0994
Ob13	5.2956 $\pm$ 0.8997	0.2783 $\pm$ 0.0546	0.5795 $\pm$ 0.0383	24.7986 $\pm$ 6.2008	0.8260 $\pm$ 0.0888	0.1127 $\pm$ 0.0524
Ob14	15.8863 $\pm$ 1.4723	0.6560 $\pm$ 0.0389	0.2409 $\pm$ 0.0327	31.8170 $\pm$ 1.9024	0.9488 $\pm$ 0.0082	0.0688 $\pm$ 0.0095
Ob15	13.8764 $\pm$ 1.5273	0.3919 $\pm$ 0.0876	0.2967 $\pm$ 0.0506	33.3144 $\pm$ 1.5746	0.9275 $\pm$ 0.0154	0.0699 $\pm$ 0.0099
Ob16	11.1333 $\pm$ 0.5659	0.4690 $\pm$ 0.0575	0.4888 $\pm$ 0.0422	27.4274 $\pm$ 0.7804	0.9275 $\pm$ 0.0177	0.1063 $\pm$ 0.0146
Ob17	3.7822 $\pm$ 0.3932	0.2608 $\pm$ 0.0545	0.6621 $\pm$ 0.0465	20.5772 $\pm$ 2.1017	0.8722 $\pm$ 0.0836	0.0944 $\pm$ 0.0374
Ob18	15.3449 $\pm$ 0.8347	0.5454 $\pm$ 0.0569	0.3474 $\pm$ 0.0385	27.2919 $\pm$ 2.6579	0.9524 $\pm$ 0.0162	0.0925 $\pm$ 0.0156
Ob19	10.8066 $\pm$ 0.8044	0.4110 $\pm$ 0.0511	0.5055 $\pm$ 0.0568	21.9574 $\pm$ 0.7729	0.9300 $\pm$ 0.0194	0.0951 $\pm$ 0.0202
Ob20	13.5576 $\pm$ 0.8153	0.4164 $\pm$ 0.0314	0.3893 $\pm$ 0.0289	28.7908 $\pm$ 0.9855	0.9486 $\pm$ 0.0127	0.0663 $\pm$ 0.0083
Ob21	5.6755 $\pm$ 0.9884	0.2677 $\pm$ 0.0835	0.6962 $\pm$ 0.0335	20.1883 $\pm$ 1.7375	0.6766 $\pm$ 0.0767	0.0658 $\pm$ 0.0218
Ob22	11.9254 $\pm$ 1.1504	0.4630 $\pm$ 0.0497	0.4733 $\pm$ 0.0509	29.8007 $\pm$ 1.7863	0.9758 $\pm$ 0.0060	0.0451 $\pm$ 0.0060
Ob23	13.4466 $\pm$ 1.1422	0.6005 $\pm$ 0.0456	0.3051 $\pm$ 0.0487	31.0118 $\pm$ 1.4505	0.9729 $\pm$ 0.0101	0.0446 $\pm$ 0.0063
Ob24	15.6826 $\pm$ 1.1804	0.4941 $\pm$ 0.0560	0.2809 $\pm$ 0.0417	33.2634 $\pm$ 2.5517	0.9516 $\pm$ 0.0145	0.0667 $\pm$ 0.0108
Ob25	12.9032 $\pm$ 1.5566	0.4391 $\pm$ 0.0944	0.5340 $\pm$ 0.0561	27.7804 $\pm$ 1.6059	0.9594 $\pm$ 0.0129	0.0489 $\pm$ 0.0092
Ob26	15.4987 $\pm$ 1.2737	0.4919 $\pm$ 0.0677	0.2501 $\pm$ 0.0263	34.3010 $\pm$ 2.5851	0.9439 $\pm$ 0.0108	0.0559 $\pm$ 0.0071
Ob27	5.8468 $\pm$ 0.5009	0.1307 $\pm$ 0.0169	0.7287 $\pm$ 0.0270	46.6284 $\pm$ 9.1214	0.9622 $\pm$ 0.0184	0.0250 $\pm$ 0.0315
Ob28	13.3032 $\pm$ 1.2383	0.5367 $\pm$ 0.0824	0.3940 $\pm$ 0.0623	28.8698 $\pm$ 2.4489	0.9656 $\pm$ 0.0150	0.0444 $\pm$ 0.0091
Ob29	16.0678 $\pm$ 1.5770	0.5545 $\pm$ 0.0803	0.4104 $\pm$ 0.0547	26.4539 $\pm$ 1.0746	0.8926 $\pm$ 0.0287	0.1046 $\pm$ 0.0160
Ob30	4.8579 $\pm$ 0.6715	0.1620 $\pm$ 0.0168	0.7282 $\pm$ 0.0511	21.4250 $\pm$ 1.5146	0.9013 $\pm$ 0.0505	0.1604 $\pm$ 0.0634
Ob31	8.3332 $\pm$ 1.1868	0.2463 $\pm$ 0.0552	0.6511 $\pm$ 0.0667	20.6753 $\pm$ 1.7520	0.9003 $\pm$ 0.0234	0.0892 $\pm$ 0.0133
Ob32	9.3317 $\pm$ 0.5644	0.1329 $\pm$ 0.0256	0.6562 $\pm$ 0.0194	30.8249 $\pm$ 4.7844	0.9436 $\pm$ 0.0608	0.0765 $\pm$ 0.0639
Ob33	15.6454 $\pm$ 1.5343	0.5443 $\pm$ 0.0818	0.3141 $\pm$ 0.0674	29.7056 $\pm$ 2.0160	0.9479 $\pm$ 0.0185	0.0829 $\pm$ 0.0222
Ob34	7.1448 $\pm$ 0.5709	0.1805 $\pm$ 0.0352	0.6609 $\pm$ 0.0295	21.2796 $\pm$ 1.9826	0.7974 $\pm$ 0.0759	0.2659 $\pm$ 0.0905
Ob35	15.5269 $\pm$ 0.9125	0.4393 $\pm$ 0.0832	0.4946 $\pm$ 0.0414	30.6967 $\pm$ 1.3788	0.9085 $\pm$ 0.0174	0.0886 $\pm$ 0.0151
Ob36	10.9100 $\pm$ 0.7430	0.4693 $\pm$ 0.0948	0.4465 $\pm$ 0.0609	26.6147 $\pm$ 1.0718	0.9477 $\pm$ 0.0171	0.0747 $\pm$ 0.0141
Ob37	6.2627 $\pm$ 2.2077	0.2315 $\pm$ 0.1096	0.7894 $\pm$ 0.0352	13.2315 $\pm$ 3.2407	0.5453 $\pm$ 0.2132	0.2801 $\pm$ 0.1155
Ob38	14.8896 $\pm$ 1.0169	0.5394 $\pm$ 0.0806	0.4077 $\pm$ 0.0918	27.0991 $\pm$ 1.4326	0.9327 $\pm$ 0.0131	0.1082 $\pm$ 0.0140
Ob39	15.2014 $\pm$ 1.9557	0.6265 $\pm$ 0.0792	0.2372 $\pm$ 0.0641	29.2155 $\pm$ 1.4416	0.9555 $\pm$ 0.0149	0.0691 $\pm$ 0.0107
Ob40	9.8968 $\pm$ 1.0177	0.6782 $\pm$ 0.0470	0.2501 $\pm$ 0.0469	29.5034 $\pm$ 1.6483	0.9574 $\pm$ 0.0110	0.0549 $\pm$ 0.0084
Ob41	10.4849 $\pm$ 2.4914	0.5443 $\pm$ 0.0806	0.5611 $\pm$ 0.0630	18.0350 $\pm$ 1.4194	0.8467 $\pm$ 0.0562	0.1342 $\pm$ 0.0330
Ob42	14.0230 $\pm$ 0.5162	0.5671 $\pm$ 0.0414	0.4070 $\pm$ 0.0313	34.8313 $\pm$ 2.9730	0.9663 $\pm$ 0.0074	0.0453 $\pm$ 0.0061
Ob43	14.9204 $\pm$ 0.8153	0.5002 $\pm$ 0.0275	0.2952 $\pm$ 0.0301	31.6731 $\pm$ 1.1837	0.9592 $\pm$ 0.0113	0.0816 $\pm$ 0.0117
Ob44	6.0758 $\pm$ 1.7211	0.2625 $\pm$ 0.1452	0.6230 $\pm$ 0.0615	17.8532 $\pm$ 2.9917	0.9249 $\pm$ 0.0210	0.1166 $\pm$ 0.0155
Ob45	16.8146 $\pm$ 1.0919	0.5531 $\pm$ 0.0558	0.3252 $\pm$ 0.0270	32.6264 $\pm$ 1.6496	0.9647 $\pm$ 0.0164	0.0607 $\pm$ 0.0099
Ob46	8.9837 $\pm$ 1.1493	0.5331 $\pm$ 0.0644	0.4978 $\pm$ 0.0486	27.0845 $\pm$ 1.5839	0.9608 $\pm$ 0.0107	0.0587 $\pm$ 0.0125
Ob47	13.7037 $\pm$ 1.4281	0.5371 $\pm$ 0.0452	0.3003 $\pm$ 0.0486	28.1294 $\pm$ 2.7300	0.9158 $\pm$ 0.0464	0.1437 $\pm$ 0.0645
Ob48	11.0240 $\pm$ 0.5975	0.6063 $\pm$ 0.0472	0.5130 $\pm$ 0.0434	20.6040 $\pm$ 1.3175	0.7653 $\pm$ 0.0481	0.2241 $\pm$ 0.0438
Ob49	13.2021 $\pm$ 1.2700	0.6021 $\pm$ 0.0444	0.3101 $\pm$ 0.0335	21.3700 $\pm$ 2.3160	0.9070 $\pm$ 0.0323	0.1300 $\pm$ 0.0258
Ob50	14.3558 $\pm$ 0.9355	0.2049 $\pm$ 0.0419	0.4685 $\pm$ 0.0549	100.0000 $\pm$ 0.0000	1.0000 $\pm$ 0.0000	0.0000 $\pm$ 0.0000
Ob51	9.2677 $\pm$ 1.3254	0.5888 $\pm$ 0.0541	0.6716 $\pm$ 0.0495	18.7372 $\pm$ 1.5653	0.4224 $\pm$ 0.0256	0.1420 $\pm$ 0.0332
Ob52	14.8268 $\pm$ 0.4993	0.6336 $\pm$ 0.0241	0.3920 $\pm$ 0.0226	29.5665 $\pm$ 2.4498	0.9154 $\pm$ 0.0236	0.0982 $\pm$ 0.0221
Ob53	12.2294 $\pm$ 0.8320	0.5215 $\pm$ 0.0879	0.3745 $\pm$ 0.0673	29.3403 $\pm$ 2.8616	0.9587 $\pm$ 0.0178	0.0730 $\pm$ 0.0151
Ob54	5.9990 $\pm$ 1.9297	0.2944 $\pm$ 0.1116	0.6525 $\pm$ 0.0481	17.1006 $\pm$ 3.1002	0.8037 $\pm$ 0.0585	0.1291 $\pm$ 0.0321
Ob55	2.8262 $\pm$ 1.0793	0.2337 $\pm$ 0.0712	0.7429 $\pm$ 0.0258	25.7994 $\pm$ 4.6722	0.9392 $\pm$ 0.0438	0.0762 $\pm$ 0.0494
Ob56	9.1814 $\pm$ 1.2024	0.1843 $\pm$ 0.0905	0.6680 $\pm$ 0.0376	20.9933 $\pm$ 3.2537	0.7271 $\pm$ 0.1583	0.1671 $\pm$ 0.0677
Ob57	7.8436 $\pm$ 0.5050	0.2405 $\pm$ 0.0506	0.7156 $\pm$ 0.0571	20.4132 $\pm$ 3.1097	0.8099 $\pm$ 0.1054	0.2859 $\pm$ 0.1428
Ob58	10.5210 $\pm$ 1.0324	0.1650 $\pm$ 0.0441	0.6141 $\pm$ 0.0428	35.6928 $\pm$ 8.1763	0.9457 $\pm$ 0.0567	0.0872 $\pm$ 0.0829
Ob59	12.1177 $\pm$ 1.5430	0.5000 $\pm$ 0.0934	0.5910 $\pm$ 0.0693	21.2086 $\pm$ 2.8004	0.8815 $\pm$ 0.0249	0.1468 $\pm$ 0.0306
Ob60	5.6781 $\pm$ 2.5424	0.3644 $\pm$ 0.1218	0.6128 $\pm$ 0.0499	13.2446 $\pm$ 4.6837	0.8043 $\pm$ 0.0848	0.1108 $\pm$ 0.0454
Ob61	4.9584 $\pm$ 0.7123	0.2208 $\pm$ 0.0692	0.7363 $\pm$ 0.0401	16.8515 $\pm$ 1.7904	0.6844 $\pm$ 0.0749	0.1895 $\pm$ 0.0231
Ob62	8.5265 $\pm$ 1.2103	0.5201 $\pm$ 0.0552	0.5656 $\pm$ 0.0670	25.6148 $\pm$ 2.5441	0.9748 $\pm$ 0.0096	0.0249 $\pm$ 0.0044
Ob63	4.9887 $\pm$ 0.4122	0.3374 $\pm$ 0.0589	0.6570 $\pm$ 0.0254	25.1478 $\pm$ 8.8056	0.9124 $\pm$ 0.0393	0.0521 $\pm$ 0.0231
Ob64	20.6685 $\pm$ 1.1112	0.7374 $\pm$ 0.0410	0.2631 $\pm$ 0.0245	33.7425 $\pm$ 1.6553	0.9372 $\pm$ 0.0130	0.0880 $\pm$ 0.0136
Ob65	5.1792 $\pm$ 0.4541	0.2226 $\pm$ 0.0326	0.6384 $\pm$ 0.0233	28.8365 $\pm$ 1.2300	0.9222 $\pm$ 0.0336	0.0338 $\pm$ 0.0262
Ob66	6.6946 $\pm$ 0.5318	0.2688 $\pm$ 0.0366	0.6912 $\pm$ 0.0361	17.8507 $\pm$ 1.1927	0.7255 $\pm$ 0.0541	0.2191 $\pm$ 0.0397
Ob67	10.6506 $\pm$ 1.5622	0.5841 $\pm$ 0.0460	0.5158 $\pm$ 0.0623	17.9489 $\pm$ 0.9004	0.9346 $\pm$ 0.0189	0.1125 $\pm$ 0.0251
Average	11.0757	0.4288	0.4937	27.6944	0.8949	0.1017

Table B.8: NVS results for Nerfacto with VGGT-initialized poses and no pose optimization.

	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	BG PSNR $\uparrow$	BG SSIM $\uparrow$	BG LPIPS $\downarrow$
Ob01	13.3328 $\pm$ 0.9608	0.4338 $\pm$ 0.0624	0.4988 $\pm$ 0.0690	25.1140 $\pm$ 1.2429	0.9575 $\pm$ 0.0093	0.0431 $\pm$ 0.0055
Ob02	13.2844 $\pm$ 1.3277	0.5900 $\pm$ 0.0630	0.3745 $\pm$ 0.0687	24.3444 $\pm$ 1.2947	0.8949 $\pm$ 0.0590	0.1035 $\pm$ 0.0224
Ob03	9.4396 $\pm$ 1.4139	0.5783 $\pm$ 0.0375	0.5173 $\pm$ 0.0634	20.6895 $\pm$ 1.7558	0.9090 $\pm$ 0.0135	0.0904 $\pm$ 0.0092
Ob04	12.6723 $\pm$ 0.8604	0.5395 $\pm$ 0.0460	0.4234 $\pm$ 0.0306	23.8612 $\pm$ 1.3745	0.9249 $\pm$ 0.0212	0.0831 $\pm$ 0.0174
Ob05	13.2641 $\pm$ 0.8083	0.6031 $\pm$ 0.0399	0.3063 $\pm$ 0.0467	28.1994 $\pm$ 1.1604	0.9757 $\pm$ 0.0087	0.0472 $\pm$ 0.0041
Ob06	13.7876 $\pm$ 0.9406	0.5310 $\pm$ 0.0343	0.4260 $\pm$ 0.0471	26.3262 $\pm$ 0.7930	0.9375 $\pm$ 0.0202	0.0560 $\pm$ 0.0093
Ob07	7.7287 $\pm$ 0.7562	0.2032 $\pm$ 0.0646	0.5767 $\pm$ 0.0412	47.9037 $\pm$ 16.7024	0.9782 $\pm$ 0.0389	0.0400 $\pm$ 0.0695
Ob08	15.2519 $\pm$ 1.2010	0.7320 $\pm$ 0.0688	0.2553 $\pm$ 0.0446	35.6457 $\pm$ 1.7829	0.9303 $\pm$ 0.0195	0.0692 $\pm$ 0.0098
Ob09	15.4237 $\pm$ 1.3176	0.3894 $\pm$ 0.0471	0.3366 $\pm$ 0.0419	35.6909 $\pm$ 1.2143	0.9570 $\pm$ 0.0104	0.0426 $\pm$ 0.0052
Ob10	9.1099 $\pm$ 0.2295	0.1050 $\pm$ 0.0337	0.6818 $\pm$ 0.0139	56.4395 $\pm$ 24.2383	0.9853 $\pm$ 0.0200	0.0298 $\pm$ 0.0409
Ob11	13.6817 $\pm$ 1.6120	0.5900 $\pm$ 0.0419	0.4615 $\pm$ 0.0798	24.4269 $\pm$ 1.6838	0.9561 $\pm$ 0.0072	0.0651 $\pm$ 0.0069
Ob12	11.5565 $\pm$ 1.4026	0.3065 $\pm$ 0.0503	0.5204 $\pm$ 0.0718	25.2824 $\pm$ 1.6279	0.9430 $\pm$ 0.0089	0.0712 $\pm$ 0.0079
Ob13	5.2896 $\pm$ 0.8607	0.2778 $\pm$ 0.0489	0.5775 $\pm$ 0.0384	23.2381 $\pm$ 6.0369	0.8152 $\pm$ 0.0854	0.1053 $\pm$ 0.0396
Ob14	15.8060 $\pm$ 1.7802	0.6636 $\pm$ 0.0310	0.2143 $\pm$ 0.0523	30.7762 $\pm$ 3.5569	0.9438 $\pm$ 0.0109	0.0544 $\pm$ 0.0110
Ob15	13.3033 $\pm$ 1.1576	0.3721 $\pm$ 0.0442	0.3049 $\pm$ 0.0402	33.1393 $\pm$ 0.8386	0.9408 $\pm$ 0.0132	0.0531 $\pm$ 0.0079
Ob16	11.3132 $\pm$ 0.6195	0.4806 $\pm$ 0.0582	0.4639 $\pm$ 0.0440	28.0469 $\pm$ 0.6691	0.9322 $\pm$ 0.0151	0.0769 $\pm$ 0.0082
Ob17	3.8113 $\pm$ 0.4327	0.2620 $\pm$ 0.0554	0.6616 $\pm$ 0.0476	19.6287 $\pm$ 1.4764	0.8673 $\pm$ 0.0733	0.1069 $\pm$ 0.0440
Ob18	15.5035 $\pm$ 0.8895	0.5728 $\pm$ 0.0512	0.3472 $\pm$ 0.0477	28.8000 $\pm$ 3.1115	0.9575 $\pm$ 0.0170	0.0584 $\pm$ 0.0122
Ob19	10.6481 $\pm$ 1.0201	0.4286 $\pm$ 0.0387	0.5191 $\pm$ 0.0466	21.4654 $\pm$ 0.8769	0.9304 $\pm$ 0.0227	0.0582 $\pm$ 0.0088
Ob20	13.8632 $\pm$ 0.8605	0.4286 $\pm$ 0.0333	0.3911 $\pm$ 0.0336	28.7201 $\pm$ 0.9405	0.9579 $\pm$ 0.0131	0.0347 $\pm$ 0.0037
Ob21	6.1337 $\pm$ 1.8683	0.2543 $\pm$ 0.1541	0.6821 $\pm$ 0.0351	14.0143 $\pm$ 2.0827	0.8373 $\pm$ 0.0760	0.1044 $\pm$ 0.0447
Ob22	12.5281 $\pm$ 1.3877	0.4833 $\pm$ 0.0512	0.4446 $\pm$ 0.0504	30.2512 $\pm$ 2.0072	0.9728 $\pm$ 0.0085	0.0398 $\pm$ 0.0060
Ob23	13.2506 $\pm$ 1.4719	0.5947 $\pm$ 0.0418	0.3151 $\pm$ 0.0488	30.0607 $\pm$ 1.8393	0.9606 $\pm$ 0.0118	0.0438 $\pm$ 0.0052
Ob24	15.7246 $\pm$ 1.6063	0.4871 $\pm$ 0.0692	0.2728 $\pm$ 0.0434	33.3389 $\pm$ 1.5466	0.9383 $\pm$ 0.0172	0.0505 $\pm$ 0.0092
Ob25	12.9802 $\pm$ 1.5267	0.4454 $\pm$ 0.0895	0.5378 $\pm$ 0.0552	27.3894 $\pm$ 1.5861	0.9549 $\pm$ 0.0156	0.0401 $\pm$ 0.0048
Ob26	15.4358 $\pm$ 1.1151	0.4776 $\pm$ 0.0491	0.2448 $\pm$ 0.0186	35.9656 $\pm$ 1.8294	0.9492 $\pm$ 0.0076	0.0405 $\pm$ 0.0038
Ob27	5.8431 $\pm$ 0.4999	0.1304 $\pm$ 0.0167	0.7371 $\pm$ 0.0246	47.3562 $\pm$ 9.3753	0.9715 $\pm$ 0.0099	0.0093 $\pm$ 0.0110
Ob28	13.7375 $\pm$ 1.2858	0.5608 $\pm$ 0.0810	0.3845 $\pm$ 0.0604	29.0170 $\pm$ 2.0728	0.9668 $\pm$ 0.0120	0.0352 $\pm$ 0.0057
Ob29	16.1214 $\pm$ 1.6366	0.5587 $\pm$ 0.0828	0.4081 $\pm$ 0.0507	27.0831 $\pm$ 0.9297	0.8854 $\pm$ 0.0292	0.0859 $\pm$ 0.0107
Ob30	4.8264 $\pm$ 0.6099	0.1590 $\pm$ 0.0166	0.7272 $\pm$ 0.0295	32.6887 $\pm$ 1.9413	0.9197 $\pm$ 0.0347	0.0717 $\pm$ 0.0235
Ob31	8.5034 $\pm$ 1.3101	0.2622 $\pm$ 0.0617	0.6534 $\pm$ 0.0680	20.4866 $\pm$ 1.7540	0.9294 $\pm$ 0.0178	0.0666 $\pm$ 0.0125
Ob32	9.3047 $\pm$ 0.5933	0.1273 $\pm$ 0.0161	0.6617 $\pm$ 0.0222	100.0000 $\pm$ 0.0000	1.0000 $\pm$ 0.0000	0.0000 $\pm$ 0.0000
Ob33	15.8107 $\pm$ 1.5038	0.5522 $\pm$ 0.0829	0.2813 $\pm$ 0.0675	29.9907 $\pm$ 1.9950	0.9729 $\pm$ 0.0117	0.0475 $\pm$ 0.0083
Ob34	7.0133 $\pm$ 0.5065	0.1701 $\pm$ 0.0323	0.6408 $\pm$ 0.0246	48.0701 $\pm$ 2.7572	0.9841 $\pm$ 0.0115	0.0122 $\pm$ 0.0071
Ob35	15.7351 $\pm$ 0.9382	0.4591 $\pm$ 0.0826	0.4921 $\pm$ 0.0396	30.7274 $\pm$ 1.0668	0.9289 $\pm$ 0.0159	0.0597 $\pm$ 0.0106
Ob36	10.9256 $\pm$ 0.8079	0.4708 $\pm$ 0.1128	0.4404 $\pm$ 0.0631	26.2716 $\pm$ 1.3084	0.9665 $\pm$ 0.0104	0.0436 $\pm$ 0.0057
Ob37	6.2223 $\pm$ 1.7298	0.2307 $\pm$ 0.0911	0.7806 $\pm$ 0.0288	11.8048 $\pm$ 3.3607	0.5955 $\pm$ 0.1932	0.2923 $\pm$ 0.1400
Ob38	15.0528 $\pm$ 0.9421	0.5473 $\pm$ 0.0841	0.3900 $\pm$ 0.0923	28.2866 $\pm$ 1.5921	0.9365 $\pm$ 0.0127	0.0645 $\pm$ 0.0094
Ob39	14.8101 $\pm$ 1.1358	0.6223 $\pm$ 0.0486	0.1976 $\pm$ 0.0338	31.9109 $\pm$ 2.9985	0.9601 $\pm$ 0.0115	0.0492 $\pm$ 0.0053
Ob40	10.1389 $\pm$ 1.1064	0.6793 $\pm$ 0.0508	0.2327 $\pm$ 0.0429	29.6910 $\pm$ 1.7212	0.9616 $\pm$ 0.0181	0.0484 $\pm$ 0.0071
Ob41	9.9770 $\pm$ 2.5777	0.5428 $\pm$ 0.0802	0.5705 $\pm$ 0.0509	17.4122 $\pm$ 1.3854	0.8627 $\pm$ 0.0501	0.1128 $\pm$ 0.0305
Ob42	13.5804 $\pm$ 0.8056	0.5837 $\pm$ 0.0458	0.4349 $\pm$ 0.0512	34.3431 $\pm$ 3.2318	0.9684 $\pm$ 0.0063	0.0321 $\pm$ 0.0068
Ob43	15.1696 $\pm$ 1.4151	0.5197 $\pm$ 0.0460	0.2856 $\pm$ 0.0374	31.7653 $\pm$ 1.9535	0.9617 $\pm$ 0.0129	0.0579 $\pm$ 0.0068
Ob44	6.2461 $\pm$ 1.9087	0.2724 $\pm$ 0.1501	0.6474 $\pm$ 0.0396	17.6971 $\pm$ 2.8430	0.8702 $\pm$ 0.0467	0.1539 $\pm$ 0.0263
Ob45	17.1361 $\pm$ 1.1211	0.5662 $\pm$ 0.0503	0.3188 $\pm$ 0.0264	32.8465 $\pm$ 0.9624	0.9618 $\pm$ 0.0160	0.0413 $\pm$ 0.0033
Ob46	9.2709 $\pm$ 1.3593	0.5566 $\pm$ 0.0696	0.5277 $\pm$ 0.0487	27.5117 $\pm$ 1.7859	0.9594 $\pm$ 0.0068	0.0419 $\pm$ 0.0077
Ob47	12.8865 $\pm$ 1.3721	0.5276 $\pm$ 0.0476	0.3023 $\pm$ 0.0442	29.9381 $\pm$ 2.5816	0.9422 $\pm$ 0.0216	0.0761 $\pm$ 0.0164
Ob48	11.0376 $\pm$ 0.6056	0.6020 $\pm$ 0.0512	0.4923 $\pm$ 0.0359	20.5710 $\pm$ 1.4279	0.7691 $\pm$ 0.0535	0.2171 $\pm$ 0.0478
Ob49	13.8377 $\pm$ 1.7150	0.6273 $\pm$ 0.0572	0.2535 $\pm$ 0.0377	23.7410 $\pm$ 1.5114	0.9472 $\pm$ 0.0247	0.0727 $\pm$ 0.0136
Ob50	20.6179 $\pm$ 1.9631	0.6586 $\pm$ 0.0392	0.3068 $\pm$ 0.0258	32.1612 $\pm$ 2.1697	0.8745 $\pm$ 0.0367	0.0720 $\pm$ 0.0109
Ob51	9.4320 $\pm$ 2.2909	0.5469 $\pm$ 0.1433	0.6692 $\pm$ 0.0510	10.2306 $\pm$ 2.9441	0.5848 $\pm$ 0.2053	0.2665 $\pm$ 0.0875
Ob52	14.8963 $\pm$ 0.6049	0.6453 $\pm$ 0.0267	0.3762 $\pm$ 0.0249	30.2735 $\pm$ 2.5033	0.9202 $\pm$ 0.0222	0.0776 $\pm$ 0.0159
Ob53	12.4095 $\pm$ 0.8980	0.5389 $\pm$ 0.0898	0.3961 $\pm$ 0.0756	28.6853 $\pm$ 3.3349	0.9628 $\pm$ 0.0163	0.0441 $\pm$ 0.0104
Ob54	5.9704 $\pm$ 2.0557	0.2918 $\pm$ 0.1206	0.6485 $\pm$ 0.0450	17.7503 $\pm$ 3.7282	0.8590 $\pm$ 0.0416	0.0988 $\pm$ 0.0266
Ob55	2.8015 $\pm$ 1.0650	0.2331 $\pm$ 0.0707	0.7438 $\pm$ 0.0283	45.8635 $\pm$ 9.1769	0.9818 $\pm$ 0.0166	0.0128 $\pm$ 0.0082
Ob56	8.5351 $\pm$ 0.5048	0.1187 $\pm$ 0.0497	0.6654 $\pm$ 0.0320	23.0280 $\pm$ 4.4261	0.8594 $\pm$ 0.0650	0.1105 $\pm$ 0.0529
Ob57	8.0116 $\pm$ 0.8525	0.2578 $\pm$ 0.0798	0.6416 $\pm$ 0.0581	19.5067 $\pm$ 2.2531	0.8865 $\pm$ 0.0658	0.1187 $\pm$ 0.0496
Ob58	10.4755 $\pm$ 0.9608	0.1633 $\pm$ 0.0420	0.6053 $\pm$ 0.0372	35.4474 $\pm$ 10.7454	0.9731 $\pm$ 0.0268	0.0379 $\pm$ 0.0347
Ob59	12.0231 $\pm$ 1.6318	0.5010 $\pm$ 0.0968	0.5883 $\pm$ 0.0760	21.2800 $\pm$ 2.8725	0.9121 $\pm$ 0.0276	0.1007 $\pm$ 0.0254
Ob60	5.6923 $\pm$ 2.5698	0.3777 $\pm$ 0.1354	0.6127 $\pm$ 0.0458	13.4825 $\pm$ 4.7254	0.7983 $\pm$ 0.0968	0.0958 $\pm$ 0.0356
Ob61	4.9190 $\pm$ 0.7193	0.2162 $\pm$ 0.0739	0.7327 $\pm$ 0.0436	17.4658 $\pm$ 3.2613	0.7750 $\pm$ 0.0566	0.1569 $\pm$ 0.0219
Ob62	8.4948 $\pm$ 1.2231	0.5215 $\pm$ 0.0569	0.5830 $\pm$ 0.0624	25.6152 $\pm$ 2.5869	0.9760 $\pm$ 0.0092	0.0235 $\pm$ 0.0039
Ob63	4.9833 $\pm$ 0.3957	0.3388 $\pm$ 0.0560	0.6542 $\pm$ 0.0237	24.4214 $\pm$ 8.8193	0.9120 $\pm$ 0.0351	0.0608 $\pm$ 0.0192
Ob64	20.8978 $\pm$ 0.8811	0.7455 $\pm$ 0.0324	0.2622 $\pm$ 0.0208	34.0652 $\pm$ 2.2946	0.9289 $\pm$ 0.0140	0.0792 $\pm$ 0.0105
Ob65	5.1858 $\pm$ 0.4461	0.2238 $\pm$ 0.0314	0.6387 $\pm$ 0.0227	26.9446 $\pm$ 1.2282	0.8964 $\pm$ 0.0524	0.0581 $\pm$ 0.0283
Ob66	6.8380 $\pm$ 0.5350	0.2760 $\pm$ 0.0385	0.6767 $\pm$ 0.0325	17.1519 $\pm$ 1.3108	0.8072 $\pm$ 0.0254	0.1560 $\pm$ 0.0272
Ob67	9.4669 $\pm$ 1.3632	0.5298 $\pm$ 0.0351	0.6333 $\pm$ 0.0351	18.6621 $\pm$ 1.1197	0.8134 $\pm$ 0.0409	0.1445 $\pm$ 0.0302
Average	11.1785	0.4380	0.4873	28.8060	0.9142	0.0748

Table B.9: NVS results for Nerfacto with VGTT-initialized poses and pose optimization.

	PSNR↑	SSIM↑	LPIPS↓	BG PSNR↑	BG SSIM↑	BG LPIPS↓
Ob01	15.5654 ± 0.6413	0.5046 ± 0.0715	0.4609 ± 0.0597	27.8891 ± 0.9728	0.9904 ± 0.0019	0.0157 ± 0.0026
Ob02	15.2544 ± 1.3689	0.6571 ± 0.0579	0.3100 ± 0.0597	28.6144 ± 1.2103	0.9891 ± 0.0019	0.0250 ± 0.0052
Ob03	11.3083 ± 1.5303	0.6724 ± 0.0381	0.4927 ± 0.0576	23.6596 ± 2.0057	0.9836 ± 0.0042	0.0286 ± 0.0046
Ob04	14.1630 ± 1.5372	0.6029 ± 0.0635	0.3468 ± 0.0568	27.5938 ± 1.2703	0.9908 ± 0.0012	0.0199 ± 0.0020
Ob05	13.6550 ± 0.9808	0.6312 ± 0.0447	0.3058 ± 0.0410	30.6709 ± 1.1033	0.9927 ± 0.0007	0.0170 ± 0.0021
Ob06	14.2696 ± 0.9756	0.5676 ± 0.0370	0.4332 ± 0.0546	29.5690 ± 0.8367	0.9922 ± 0.0012	0.0178 ± 0.0024
Ob07	11.0110 ± 0.9034	0.6652 ± 0.0279	0.5351 ± 0.0447	86.6449 ± 28.1156	0.9978 ± 0.0055	0.0038 ± 0.0096
Ob08	17.7559 ± 1.3190	0.7594 ± 0.0739	0.2158 ± 0.0539	43.3830 ± 1.9669	0.9978 ± 0.0005	0.0041 ± 0.0010
Ob09	16.2338 ± 1.3908	0.4392 ± 0.0352	0.2733 ± 0.0306	40.0258 ± 0.7389	0.9975 ± 0.0004	0.0048 ± 0.0009
Ob10	12.7190 ± 0.4083	0.4799 ± 0.0545	0.6539 ± 0.0168	84.1713 ± 28.5661	0.9982 ± 0.0037	0.0030 ± 0.0062
Ob11	15.2875 ± 1.2912	0.6539 ± 0.0441	0.4421 ± 0.0797	27.7608 ± 1.4803	0.9870 ± 0.0034	0.0285 ± 0.0058
Ob12	13.5166 ± 1.3069	0.3726 ± 0.0377	0.5009 ± 0.0785	28.9472 ± 1.9135	0.9899 ± 0.0027	0.0172 ± 0.0038
Ob13	8.1807 ± 0.9180	0.6053 ± 0.0213	0.5453 ± 0.0380	50.6656 ± 35.2363	0.9876 ± 0.0185	0.0190 ± 0.0246
Ob14	17.0156 ± 1.7948	0.6896 ± 0.0408	0.2012 ± 0.0349	33.8747 ± 1.7524	0.9952 ± 0.0006	0.0125 ± 0.0011
Ob15	15.1356 ± 1.6011	0.4552 ± 0.0940	0.2444 ± 0.0537	38.9287 ± 1.1321	0.9976 ± 0.0004	0.0057 ± 0.0008
Ob16	12.4638 ± 0.7073	0.5206 ± 0.0618	0.4704 ± 0.0446	33.6046 ± 0.5487	0.9942 ± 0.0007	0.0145 ± 0.0013
Ob17	6.4082 ± 0.4844	0.5337 ± 0.0437	0.6172 ± 0.0477	23.8781 ± 2.4945	0.9868 ± 0.0042	0.0236 ± 0.0058
Ob18	17.2849 ± 1.1646	0.6329 ± 0.0528	0.3030 ± 0.0447	33.4426 ± 1.1966	0.9957 ± 0.0009	0.0102 ± 0.0018
Ob19	12.0649 ± 1.0074	0.4687 ± 0.0524	0.4947 ± 0.0552	24.2443 ± 0.5349	0.9904 ± 0.0010	0.0180 ± 0.0013
Ob20	15.5271 ± 0.9217	0.4826 ± 0.0322	0.3530 ± 0.0348	31.3724 ± 1.1389	0.9952 ± 0.0005	0.0098 ± 0.0008
Ob21	7.6062 ± 0.9695	0.4566 ± 0.0454	0.6490 ± 0.0331	82.5493 ± 21.8332	0.9999 ± 0.0002	0.0001 ± 0.0003
Ob22	12.3771 ± 1.1053	0.5117 ± 0.0450	0.3888 ± 0.0657	32.8813 ± 2.0325	0.9951 ± 0.0008	0.0093 ± 0.0012
Ob23	13.6273 ± 1.1455	0.6404 ± 0.0436	0.2323 ± 0.0477	32.4159 ± 1.4017	0.9962 ± 0.0006	0.0090 ± 0.0013
Ob24	16.4375 ± 1.1210	0.5385 ± 0.0572	0.2291 ± 0.0414	42.0775 ± 2.5198	0.9980 ± 0.0005	0.0051 ± 0.0016
Ob25	14.0868 ± 1.1532	0.5303 ± 0.0837	0.5024 ± 0.0578	31.0123 ± 1.5501	0.9906 ± 0.0019	0.0213 ± 0.0042
Ob26	16.9845 ± 1.4750	0.5361 ± 0.0714	0.1987 ± 0.0276	39.7313 ± 2.7505	0.9978 ± 0.0005	0.0055 ± 0.0010
Ob27	9.2367 ± 0.7113	0.4673 ± 0.0330	0.6693 ± 0.0278	86.2942 ± 27.0586	0.9988 ± 0.0024	0.0022 ± 0.0044
Ob28	14.9048 ± 1.4264	0.5758 ± 0.0827	0.3649 ± 0.0685	32.1474 ± 2.2872	0.9948 ± 0.0014	0.0108 ± 0.0025
Ob29	15.9653 ± 1.5606	0.5899 ± 0.0774	0.4011 ± 0.0530	32.5762 ± 0.6742	0.9913 ± 0.0011	0.0173 ± 0.0021
Ob30	7.5436 ± 0.8642	0.4794 ± 0.0298	0.6894 ± 0.0307	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob31	10.0956 ± 1.2069	0.3232 ± 0.0500	0.6198 ± 0.0654	22.7894 ± 1.6797	0.9776 ± 0.0071	0.0324 ± 0.0096
Ob32	12.8946 ± 0.7813	0.5478 ± 0.0364	0.6389 ± 0.0221	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob33	16.6515 ± 1.7707	0.5959 ± 0.0916	0.2648 ± 0.0731	33.2453 ± 2.2561	0.9940 ± 0.0014	0.0134 ± 0.0016
Ob34	10.6647 ± 0.6707	0.6062 ± 0.0296	0.6100 ± 0.0231	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob35	16.3072 ± 0.9979	0.5563 ± 0.0874	0.4833 ± 0.0382	35.4693 ± 0.9602	0.9944 ± 0.0019	0.0111 ± 0.0042
Ob36	12.0939 ± 0.6822	0.5275 ± 0.1080	0.4305 ± 0.0570	29.4797 ± 1.3241	0.9916 ± 0.0013	0.0165 ± 0.0020
Ob37	7.8856 ± 0.9616	0.3966 ± 0.0324	0.7502 ± 0.0231	27.8095 ± 25.8739	0.9446 ± 0.0385	0.0719 ± 0.0488
Ob38	15.5904 ± 1.0949	0.6099 ± 0.0826	0.3815 ± 0.0993	31.4186 ± 1.5462	0.9943 ± 0.0016	0.0132 ± 0.0031
Ob39	15.7425 ± 2.2861	0.6608 ± 0.0883	0.2018 ± 0.0623	31.7101 ± 1.5230	0.9939 ± 0.0012	0.0169 ± 0.0017
Ob40	10.3947 ± 0.9175	0.6964 ± 0.0428	0.2127 ± 0.0464	32.3213 ± 1.6949	0.9946 ± 0.0011	0.0128 ± 0.0022
Ob41	12.9229 ± 1.9766	0.6728 ± 0.0438	0.5117 ± 0.0551	20.4906 ± 1.5994	0.9727 ± 0.0051	0.0422 ± 0.0061
Ob42	14.3000 ± 0.9620	0.6121 ± 0.0367	0.2705 ± 0.0298	37.6020 ± 4.6711	0.9971 ± 0.0012	0.0068 ± 0.0024
Ob43	15.5474 ± 0.8317	0.5381 ± 0.0273	0.2430 ± 0.0363	35.9176 ± 1.5391	0.9943 ± 0.0011	0.0122 ± 0.0015
Ob44	8.6751 ± 1.5940	0.5075 ± 0.0642	0.5941 ± 0.0677	20.4474 ± 2.9700	0.9705 ± 0.0138	0.0426 ± 0.0166
Ob45	18.5048 ± 1.4324	0.6106 ± 0.0489	0.2821 ± 0.0209	38.0003 ± 1.6754	0.9954 ± 0.0009	0.0087 ± 0.0012
Ob46	11.1449 ± 1.3350	0.5896 ± 0.0553	0.4550 ± 0.0555	29.5159 ± 1.5942	0.9944 ± 0.0010	0.0099 ± 0.0010
Ob47	13.9768 ± 1.5734	0.5747 ± 0.0488	0.2368 ± 0.0337	36.6864 ± 2.3630	0.9951 ± 0.0012	0.0113 ± 0.0024
Ob48	15.6197 ± 1.0281	0.7406 ± 0.0358	0.2520 ± 0.0322	26.4755 ± 2.0876	0.9891 ± 0.0022	0.0229 ± 0.0042
Ob49	13.9178 ± 1.3915	0.6464 ± 0.0473	0.2614 ± 0.0338	26.0890 ± 1.0307	0.9891 ± 0.0015	0.0214 ± 0.0028
Ob50	21.4484 ± 2.3277	0.7161 ± 0.0421	0.2531 ± 0.0308	42.1310 ± 2.9489	0.9977 ± 0.0006	0.0039 ± 0.0011
Ob51	12.8359 ± 2.0735	0.7263 ± 0.0412	0.6089 ± 0.0394	10.5288 ± 3.5578	0.6939 ± 0.1141	0.1493 ± 0.0565
Ob52	15.4296 ± 0.5492	0.6982 ± 0.0256	0.3565 ± 0.0247	34.9925 ± 2.5839	0.9922 ± 0.0024	0.0160 ± 0.0037
Ob53	14.6308 ± 1.2620	0.5898 ± 0.0905	0.3390 ± 0.0678	32.4557 ± 2.8389	0.9953 ± 0.0013	0.0101 ± 0.0020
Ob54	8.8712 ± 1.9035	0.5684 ± 0.0635	0.6205 ± 0.0406	20.1871 ± 4.9475	0.9610 ± 0.0204	0.0544 ± 0.0277
Ob55	5.1466 ± 1.3950	0.4216 ± 0.0522	0.6929 ± 0.0283	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob56	11.9781 ± 0.6205	0.5369 ± 0.0288	0.6566 ± 0.0247	35.5978 ± 26.4776	0.9794 ± 0.0168	0.0294 ± 0.0244
Ob57	11.1425 ± 0.5977	0.6755 ± 0.0434	0.5782 ± 0.0572	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob58	15.7282 ± 1.4834	0.7048 ± 0.0458	0.5691 ± 0.0342	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob59	14.1779 ± 1.1470	0.6533 ± 0.0500	0.5595 ± 0.0732	25.8043 ± 3.1382	0.9801 ± 0.0070	0.0418 ± 0.0129
Ob60	8.0383 ± 2.1574	0.5702 ± 0.0413	0.5783 ± 0.0502	15.6162 ± 5.0937	0.9220 ± 0.0348	0.0841 ± 0.0346
Ob61	7.7367 ± 0.7706	0.4881 ± 0.0353	0.6852 ± 0.0450	22.2113 ± 12.6188	0.9666 ± 0.0125	0.0484 ± 0.0184
Ob62	11.0064 ± 1.0359	0.6435 ± 0.0345	0.5360 ± 0.0570	28.3164 ± 2.4866	0.9928 ± 0.0024	0.0125 ± 0.0032
Ob63	7.8298 ± 0.4876	0.6174 ± 0.0299	0.6178 ± 0.0291	36.1797 ± 28.7714	0.9829 ± 0.0140	0.0279 ± 0.0209
Ob64	20.7881 ± 1.6754	0.7852 ± 0.0400	0.2117 ± 0.0256	41.0601 ± 2.3882	0.9974 ± 0.0007	0.0047 ± 0.0015
Ob65	8.1941 ± 0.6646	0.5429 ± 0.0276	0.6027 ± 0.0243	31.1929 ± 1.1183	0.9959 ± 0.0007	0.0100 ± 0.0014
Ob66	9.8028 ± 0.7432	0.6104 ± 0.0461	0.6360 ± 0.0366	22.5442 ± 1.7680	0.9751 ± 0.0049	0.0463 ± 0.0102
Ob67	12.3295 ± 1.2467	0.6400 ± 0.0463	0.4911 ± 0.0719	20.1015 ± 0.8884	0.9786 ± 0.0037	0.0333 ± 0.0044
Average	13.0393	0.5809	0.4450	40.2241	0.9852	0.0197

Table B.10: NVS results for Splatfacto with VGGT-initialized poses and no pose optimization.

	PSNR↑	SSIM↑	LPIPS↓	BG_PSNR↑	BG_SSIM↑	BG_LPIPS↓
Ob01	15.4335 ± 0.6729	0.5021 ± 0.0681	0.4658 ± 0.0671	27.7554 ± 1.1217	0.9906 ± 0.0018	0.0157 ± 0.0027
Ob02	14.9593 ± 1.4852	0.6480 ± 0.0581	0.2996 ± 0.0662	28.1133 ± 1.2213	0.9898 ± 0.0025	0.0250 ± 0.0059
Ob03	11.0229 ± 1.2699	0.6653 ± 0.0287	0.4979 ± 0.0573	23.4562 ± 1.9111	0.9836 ± 0.0041	0.0286 ± 0.0046
Ob04	13.7286 ± 0.5213	0.5924 ± 0.0482	0.3429 ± 0.0555	27.3318 ± 1.2232	0.9917 ± 0.0010	0.0191 ± 0.0016
Ob05	13.4974 ± 0.9509	0.6221 ± 0.0441	0.2898 ± 0.0280	30.2276 ± 1.2987	0.9938 ± 0.0008	0.0166 ± 0.0020
Ob06	14.2414 ± 0.9341	0.5655 ± 0.0337	0.4153 ± 0.0487	29.4926 ± 0.8908	0.9926 ± 0.0013	0.0170 ± 0.0028
Ob07	11.0073 ± 0.8990	0.6652 ± 0.0279	0.5347 ± 0.0451	87.1474 ± 27.7129	0.9980 ± 0.0050	0.0035 ± 0.0087
Ob08	18.0812 ± 1.2651	0.7729 ± 0.0688	0.1886 ± 0.0434	43.3205 ± 1.8360	0.9979 ± 0.0006	0.0044 ± 0.0013
Ob09	16.1017 ± 1.3453	0.4322 ± 0.0390	0.2698 ± 0.0313	40.1820 ± 0.7904	0.9976 ± 0.0004	0.0047 ± 0.0009
Ob10	12.7217 ± 0.4040	0.4803 ± 0.0542	0.6536 ± 0.0170	84.8946 ± 27.9107	0.9986 ± 0.0029	0.0024 ± 0.0051
Ob11	15.1982 ± 1.2858	0.6515 ± 0.0429	0.4442 ± 0.0773	27.5815 ± 1.3989	0.9872 ± 0.0033	0.0284 ± 0.0050
Ob12	13.4631 ± 1.2844	0.3719 ± 0.0377	0.5034 ± 0.0773	28.8779 ± 1.9602	0.9899 ± 0.0028	0.0177 ± 0.0040
Ob13	8.1782 ± 0.9083	0.6054 ± 0.0214	0.5454 ± 0.0381	50.4018 ± 35.1993	0.9873 ± 0.0192	0.0193 ± 0.0252
Ob14	17.0481 ± 1.6126	0.6977 ± 0.0392	0.1751 ± 0.0245	34.3684 ± 1.6979	0.9959 ± 0.0006	0.0115 ± 0.0015
Ob15	15.0045 ± 1.5519	0.4472 ± 0.0907	0.2368 ± 0.0535	39.2418 ± 1.1001	0.9980 ± 0.0004	0.0053 ± 0.0009
Ob16	12.4855 ± 0.7036	0.5161 ± 0.0648	0.4558 ± 0.0480	33.4218 ± 0.5578	0.9945 ± 0.0007	0.0146 ± 0.0012
Ob17	6.4101 ± 0.4812	0.5338 ± 0.0439	0.6173 ± 0.0477	23.7503 ± 2.3912	0.9866 ± 0.0042	0.0238 ± 0.0058
Ob18	17.1904 ± 1.1453	0.6319 ± 0.0550	0.2993 ± 0.0438	33.1727 ± 1.4051	0.9958 ± 0.0009	0.0102 ± 0.0021
Ob19	11.9942 ± 1.0585	0.4667 ± 0.0501	0.4834 ± 0.0590	24.1873 ± 0.5443	0.9912 ± 0.0010	0.0172 ± 0.0018
Ob20	15.5284 ± 0.9759	0.4853 ± 0.0337	0.3317 ± 0.0360	31.6368 ± 0.9453	0.9959 ± 0.0006	0.0091 ± 0.0009
Ob21	7.6062 ± 0.9695	0.4566 ± 0.0454	0.6490 ± 0.0331	82.5493 ± 21.8332	0.9999 ± 0.0002	0.0001 ± 0.0003
Ob22	12.2625 ± 1.0677	0.5103 ± 0.0452	0.3915 ± 0.0619	32.7149 ± 1.8940	0.9951 ± 0.0007	0.0095 ± 0.0012
Ob23	13.4088 ± 1.2236	0.6376 ± 0.0465	0.2345 ± 0.0584	31.8527 ± 1.5372	0.9963 ± 0.0006	0.0096 ± 0.0013
Ob24	16.2718 ± 1.0481	0.5298 ± 0.0564	0.2195 ± 0.0461	42.2455 ± 3.2196	0.9983 ± 0.0006	0.0048 ± 0.0017
Ob25	14.0513 ± 1.1521	0.5278 ± 0.0877	0.4950 ± 0.0627	30.9718 ± 1.6068	0.9907 ± 0.0021	0.0216 ± 0.0044
Ob26	16.8282 ± 1.3956	0.5323 ± 0.0720	0.2014 ± 0.0253	39.5909 ± 3.0153	0.9980 ± 0.0005	0.0057 ± 0.0011
Ob27	9.2367 ± 0.7113	0.4673 ± 0.0330	0.6693 ± 0.0278	86.2844 ± 27.0786	0.9988 ± 0.0024	0.0022 ± 0.0044
Ob28	14.8078 ± 1.2895	0.5763 ± 0.0806	0.3627 ± 0.0715	32.0800 ± 2.1842	0.9950 ± 0.0013	0.0107 ± 0.0025
Ob29	15.9534 ± 1.5625	0.5874 ± 0.0783	0.3985 ± 0.0524	32.4514 ± 0.6600	0.9914 ± 0.0013	0.0172 ± 0.0024
Ob30	7.5436 ± 0.8642	0.4794 ± 0.0298	0.6894 ± 0.0307	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob31	10.1280 ± 1.2415	0.3246 ± 0.0498	0.6135 ± 0.0682	22.8283 ± 1.6431	0.9778 ± 0.0070	0.0325 ± 0.0094
Ob32	12.8946 ± 0.7813	0.5478 ± 0.0364	0.6389 ± 0.0221	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob33	16.3784 ± 1.6002	0.5931 ± 0.0924	0.2462 ± 0.0721	33.1221 ± 2.7504	0.9948 ± 0.0014	0.0128 ± 0.0024
Ob34	10.6647 ± 0.6707	0.6062 ± 0.0296	0.6100 ± 0.0231	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob35	16.3912 ± 1.0418	0.5567 ± 0.0849	0.4809 ± 0.0454	35.0942 ± 1.0457	0.9943 ± 0.0021	0.0114 ± 0.0045
Ob36	11.9254 ± 0.7217	0.5128 ± 0.1269	0.4071 ± 0.0557	29.0582 ± 1.5028	0.9925 ± 0.0013	0.0158 ± 0.0022
Ob37	7.9392 ± 1.0126	0.4006 ± 0.0309	0.7429 ± 0.0232	27.3790 ± 26.0056	0.9433 ± 0.0396	0.0761 ± 0.0523
Ob38	15.5935 ± 1.0792	0.6118 ± 0.0805	0.3763 ± 0.0969	31.1758 ± 1.7691	0.9943 ± 0.0017	0.0134 ± 0.0034
Ob39	15.4132 ± 2.0633	0.6511 ± 0.0836	0.1920 ± 0.0571	31.6896 ± 1.6541	0.9950 ± 0.0009	0.0146 ± 0.0016
Ob40	10.2638 ± 0.9761	0.6933 ± 0.0460	0.2074 ± 0.0447	31.6162 ± 1.9681	0.9945 ± 0.0013	0.0135 ± 0.0024
Ob41	12.6267 ± 1.9423	0.6665 ± 0.0435	0.5160 ± 0.0555	20.2504 ± 1.5463	0.9722 ± 0.0050	0.0424 ± 0.0059
Ob42	14.3623 ± 0.9913	0.6140 ± 0.0304	0.2578 ± 0.0284	37.1682 ± 4.5117	0.9972 ± 0.0011	0.0070 ± 0.0022
Ob43	15.8508 ± 1.3203	0.5489 ± 0.0493	0.2276 ± 0.0415	35.5215 ± 1.8555	0.9944 ± 0.0013	0.0128 ± 0.0019
Ob44	8.6711 ± 1.5860	0.5082 ± 0.0645	0.5919 ± 0.0685	20.4723 ± 2.9632	0.9711 ± 0.0136	0.0426 ± 0.0166
Ob45	18.2946 ± 1.1889	0.6014 ± 0.0508	0.2712 ± 0.0281	37.8075 ± 1.3358	0.9957 ± 0.0007	0.0087 ± 0.0012
Ob46	11.0817 ± 1.5008	0.5928 ± 0.0564	0.4551 ± 0.0558	29.3801 ± 1.5363	0.9945 ± 0.0010	0.0098 ± 0.0010
Ob47	13.8763 ± 1.4048	0.5735 ± 0.0513	0.2271 ± 0.0335	36.7734 ± 2.8983	0.9955 ± 0.0014	0.0110 ± 0.0028
Ob48	15.6294 ± 0.8278	0.7419 ± 0.0415	0.2222 ± 0.0239	26.6162 ± 1.8538	0.9903 ± 0.0021	0.0218 ± 0.0042
Ob49	13.9286 ± 1.5018	0.6388 ± 0.0553	0.2398 ± 0.0483	25.8962 ± 1.1209	0.9907 ± 0.0016	0.0207 ± 0.0032
Ob50	21.3957 ± 2.4184	0.7183 ± 0.0501	0.2427 ± 0.0388	41.3820 ± 2.6207	0.9977 ± 0.0007	0.0041 ± 0.0011
Ob51	13.2996 ± 1.5602	0.7236 ± 0.0439	0.5950 ± 0.0411	13.1883 ± 4.2568	0.7613 ± 0.0884	0.1019 ± 0.0533
Ob52	15.4016 ± 0.6338	0.6980 ± 0.0274	0.3468 ± 0.0231	34.7918 ± 2.5539	0.9922 ± 0.0026	0.0166 ± 0.0043
Ob53	14.5527 ± 1.1684	0.5954 ± 0.0880	0.3313 ± 0.0737	32.4845 ± 3.3320	0.9958 ± 0.0013	0.0100 ± 0.0021
Ob54	8.9049 ± 1.9341	0.5690 ± 0.0647	0.6211 ± 0.0390	20.5518 ± 8.4452	0.9574 ± 0.0238	0.0594 ± 0.0320
Ob55	5.1466 ± 1.3950	0.4216 ± 0.0522	0.6929 ± 0.0283	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob56	11.9727 ± 0.6327	0.5372 ± 0.0296	0.6562 ± 0.0253	34.9658 ± 26.4606	0.9797 ± 0.0163	0.0291 ± 0.0239
Ob57	11.1425 ± 0.5977	0.6755 ± 0.0434	0.5782 ± 0.0572	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob58	15.7282 ± 1.4834	0.7048 ± 0.0458	0.5691 ± 0.0342	100.0000 ± 0.0000	1.0000 ± 0.0000	0.0000 ± 0.0000
Ob59	14.1141 ± 1.1677	0.6506 ± 0.0511	0.5620 ± 0.0759	25.7238 ± 3.2274	0.9802 ± 0.0074	0.0414 ± 0.0132
Ob60	8.0161 ± 2.1307	0.5714 ± 0.0429	0.5777 ± 0.0520	15.6502 ± 5.1845	0.9228 ± 0.0349	0.0861 ± 0.0349
Ob61	7.7410 ± 0.7767	0.4882 ± 0.0354	0.6846 ± 0.0458	22.1001 ± 12.6259	0.9664 ± 0.0125	0.0495 ± 0.0185
Ob62	11.0159 ± 1.0424	0.6441 ± 0.0350	0.5350 ± 0.0587	28.3420 ± 2.4997	0.9929 ± 0.0024	0.0127 ± 0.0033
Ob63	7.8332 ± 0.4848	0.6176 ± 0.0300	0.6179 ± 0.0289	36.3647 ± 29.2436	0.9832 ± 0.0138	0.0277 ± 0.0209
Ob64	20.7516 ± 1.6309	0.7835 ± 0.0318	0.2059 ± 0.0225	41.1769 ± 2.5536	0.9973 ± 0.0009	0.0050 ± 0.0017
Ob65	8.1941 ± 0.6646	0.5429 ± 0.0276	0.6027 ± 0.0243	31.1616 ± 1.1517	0.9958 ± 0.0007	0.0102 ± 0.0014
Ob66	9.7902 ± 0.7000	0.6102 ± 0.0460	0.6323 ± 0.0389	22.6680 ± 1.6586	0.9766 ± 0.0040	0.0455 ± 0.0088
Ob67	12.2784 ± 1.2154	0.6410 ± 0.0450	0.4869 ± 0.0721	20.0973 ± 0.8991	0.9794 ± 0.0034	0.0336 ± 0.0042
Average	12.9919	0.5796	0.4392	40.1761	0.9864	0.0190

Table B.11: NVS results for Splatfacto with VGGT-initialized poses and pose optimization.

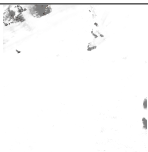
Object	Board	Nerfacto COLMAP Poses	Splatfacto COLMAP Poses	Nerfacto VGGT Poses	Splatfacto VGGT Poses
Ob01					
Ob07					
Ob13					
Ob28					
Ob32					

Table B.12: **Qualitative comparison of NVS-rendered viewpoints for selected objects.** Each row represents an object, and each column represents a method.

List of Publications

- [1] Musawar Ali, Nicola Fioraio, Samuele Salti, and Luigi Di Stefano. “AnomalyControl: Few-Shot Anomaly Generation by ControlNet Inpainting.” *IEEE Access*, 2024. IEEE.
- [2] Rubin Carkaxhia, Musawar Ali, Nicola Fioraio, Luigi Di Stefano, and Samuele Salti. “Few-Shot Anomaly Classification by Learning to Inpaint Nominal Images.” In: *Proceedings of the 23rd International Conference on Image Analysis and Processing (ICIAP 2025)*, Rome, Italy, Sep. 2025. Springer.
- [3] Musawar Ali, Manuel Carranza-García, Nicola Fioraio, Samuele Salti, and Luigi Di Stefano. “NVS-HO: A Benchmark for Novel View Synthesis of Handheld Objects.” (Under Review).

Bibliography

- [1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.
- [2] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.
- [3] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.
- [4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.
- [5] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3d gaussian splatting for real-time radiance field rendering,” *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.
- [6] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-or, “An image is worth one word: Personalizing text-to-image generation using textual inversion,” in *The Eleventh International Conference on Learning Representations*, 2023.
- [7] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman, “Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510, 2023.
- [8] L. Zhang, A. Rao, and M. Agrawala, “Adding conditional control to text-to-image diffusion models,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

- [9] O. Ronneberger, P. Fischer, and T. 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*, pp. 234–241, Springer, 2015.
- [10] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross, “Ewa volume splatting,” in *Proceedings Visualization, 2001. VIS’01.*, pp. 29–538, IEEE, 2001.
- [11] Y. Duan, Y. Hong, L. Niu, and L. Zhang, “Few-shot defect image generation via defect-aware feature manipulation,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 571–578, 2023.
- [12] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, “Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4183–4192, 2020.
- [13] Y. Cao, Q. Wan, W. Shen, and L. Gao, “Informative knowledge distillation for image anomaly segmentation,” *Knowledge-Based Systems*, vol. 248, p. 108846, 2022.
- [14] L.-L. Chiu and S.-H. Lai, “Self-supervised normalizing flows for image anomaly detection and localization,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2926–2935, 2023.
- [15] T. Defard, A. Setkov, A. Loesch, and R. Audigier, “Padim: a patch distribution modeling framework for anomaly detection and localization,” in *International Conference on Pattern Recognition*, pp. 475–489, Springer, 2021.
- [16] H. Deng and X. Li, “Anomaly detection via reverse distillation from one-class embedding,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9737–9746, 2022.
- [17] D. Gudovskiy, S. Ishizaka, and K. Kozuka, “Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows,” in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 98–107, 2022.
- [18] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, “Cutpaste: Self-supervised learning for anomaly detection and localization,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9664–9674, 2021.

- [19] O. Rippel, P. Mertens, and D. Merhof, “Modeling the distribution of normal data in pre-trained deep features for anomaly detection,” in *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 6726–6733, IEEE, 2021.
- [20] F. V. Massoli, F. Falchi, A. Kantarci, Ş. Akti, H. K. Ekenel, and G. Amato, “Mocca: Multilayer one-class classification for anomaly detection,” *IEEE transactions on neural networks and learning systems*, vol. 33, no. 6, pp. 2313–2323, 2021.
- [21] T. Reiss, N. Cohen, L. Bergman, and Y. Hoshen, “Panda: Adapting pretrained features for anomaly detection and segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2806–2814, 2021.
- [22] M. Rudolph, B. Wandt, and B. Rosenhahn, “Same same but different: Semi-supervised defect detection with normalizing flows,” in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 1907–1916, 2021.
- [23] M. Salehi, N. Sadjadi, S. Baselizadeh, M. H. Rohban, and H. R. Rabiee, “Multiresolution knowledge distillation for anomaly detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14902–14912, 2021.
- [24] K. Sohn, C.-L. Li, J. Yoon, M. Jin, and T. Pfister, “Learning and evaluating representations for deep one-class classification,” in *International Conference on Learning Representations*, 2021.
- [25] G. Wang, S. Han, E. Ding, and D. Huang, “Student-teacher feature pyramid matching for anomaly detection,” *arXiv preprint arXiv:2103.04257*, 2021.
- [26] M. Yang, P. Wu, and H. Feng, “Memseg: A semi-supervised method for image surface defect detection using differences and commonalities,” *Engineering Applications of Artificial Intelligence*, vol. 119, p. 105835, 2023.
- [27] J. Yi and S. Yoon, “Patch svdd: Patch-level svdd for anomaly detection and segmentation,” in *Proceedings of the Asian conference on computer vision*, 2020.
- [28] S. Yoa, S. Lee, C. Kim, and H. J. Kim, “Self-supervised learning for anomaly detection with dynamic local augmentation,” *IEEE Access*, vol. 9, pp. 147201–147211, 2021.

-
- [29] J. Yu, Y. Zheng, X. Wang, W. Li, Y. Wu, R. Zhao, and L. F. Wu, “Unsupervised anomaly detection and localization via 2d normalizing flows,” *arXiv preprint arXiv:2111.07677*, 2021.
 - [30] Z. Zhang and X. Deng, “Anomaly detection using improved deep svdd model with data structure preservation,” *Pattern Recognition Letters*, vol. 148, pp. 1–6, 2021.
 - [31] P. Bergmann, S. Löwe, M. Fauser, D. Sattlegger, and C. Steger, “Improving unsupervised defect segmentation by applying structural similarity to autoencoders,” *arXiv preprint arXiv:1807.02011*, 2018.
 - [32] J. Hou, Y. Zhang, Q. Zhong, D. Xie, S. Pu, and H. Zhou, “Divide-and-assemble: Learning block-wise memory for unsupervised anomaly detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8791–8800, 2021.
 - [33] N.-C. Ristea, N. Madan, R. T. Ionescu, K. Nasrollahi, F. S. Khan, T. B. Moeslund, and M. Shah, “Self-supervised predictive convolutional attentive block for anomaly detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13576–13586, 2022.
 - [34] V. Zavrtanik, M. Kristan, and D. Skočaj, “Draem-a discriminatively trained reconstruction embedding for surface anomaly detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8330–8339, 2021.
 - [35] J. Wyatt, A. Leach, S. M. Schmon, and C. G. Willcocks, “Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 650–656, 2022.
 - [36] J. Pirnay and K. Chai, “Inpainting transformer for anomaly detection,” in *International Conference on Image Analysis and Processing*, pp. 394–406, Springer, 2022.
 - [37] J. Jeong, Y. Zou, T. Kim, D. Zhang, A. Ravichandran, and O. Dabeer, “Winclip: Zero-/few-shot anomaly classification and segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19606–19616, 2023.
 - [38] H. He, J. Zhang, H. Chen, X. Chen, Z. Li, X. Chen, Y. Wang, C. Wang, and L. Xie, “A diffusion-based framework for multi-class anomaly detection,”

- in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 8472–8480, 2024.
- [39] S. Wei, X. Wei, Z. Ma, S. Dong, S. Zhang, and Y. Gong, “Few-shot online anomaly detection and segmentation,” *arXiv preprint arXiv:2403.18201*, 2024.
- [40] Y. Cao, X. Xu, C. Sun, Y. Cheng, Z. Du, L. Gao, and W. Shen, “Segment any anomaly without training via hybrid prompt regularization,” *arXiv preprint arXiv:2305.10724*, 2023.
- [41] X. Chen, J. Zhang, G. Tian, H. He, W. Zhang, Y. Wang, C. Wang, Y. Wu, and Y. Liu, “Clip-ad: A language-guided staged dual-path model for zero-shot anomaly detection,” *arXiv preprint arXiv:2311.00453*, 2023.
- [42] Q. Zhou, G. Pang, Y. Tian, S. He, and J. Chen, “AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection,” in *The Twelfth International Conference on Learning Representations*, 2024.
- [43] X. Chen, Y. Han, and J. Zhang, “A zero-/few-shot anomaly classification and segmentation method for cvpr 2023 vand workshop challenge tracks 1&2: 1st place on zero-shot ad and 4th place on few-shot ad,” *arXiv preprint arXiv:2305.17382*, 2023.
- [44] M. Ali, N. Fioraio, S. Salti, and L. Di Stefano, “Anomalycontrol: Few-shot anomaly generation by controlnet inpainting,” *IEEE Access*, 2024.
- [45] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, *et al.*, “Dinov2: Learning robust visual features without supervision,” *arXiv preprint arXiv:2304.07193*, 2023.
- [46] R. Carkaxhia, M. Ali, N. Fioraio, L. Di Stefano, and S. Salti, “Few-shot anomaly classification by learning to inpaint nominal images,” in *Image Analysis and Processing – ICIAP 2025* (E. Rodolà, F. Galasso, and I. Masi, eds.), (Cham), pp. 520–532, Springer Nature Switzerland, 2026.
- [47] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, “Towards total recall in industrial anomaly detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14318–14328, 2022.
- [48] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, “Unsupervised anomaly detection with generative adversarial networks

- to guide marker discovery,” in *International conference on information processing in medical imaging*, pp. 146–157, Springer, 2017.
- [49] D. Lin, Y. Cao, W. Zhu, and Y. Li, “Few-shot defect segmentation leveraging abundant defect-free training samples through normal background regularization and crop-and-paste operation,” in *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, IEEE, 2021.
- [50] G. Zhang, K. Cui, T.-Y. Hung, and S. Lu, “Defect-gan: High-fidelity defect synthesis for automated defect inspection,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2524–2534, 2021.
- [51] S. Niu, B. Li, X. Wang, and H. Lin, “Defect image sample generation with gan for improving defect recognition,” *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1611–1622, 2020.
- [52] T. Hu, J. Zhang, R. Yi, Y. Du, X. Chen, L. Liu, Y. Wang, and C. Wang, “Anomalydiffusion: Few-shot anomaly image generation with diffusion model,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 8526–8534, 2024.
- [53] A. Q. Nichol and P. Dhariwal, “Improved denoising diffusion probabilistic models,” in *International conference on machine learning*, pp. 8162–8171, PMLR, 2021.
- [54] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, *et al.*, “Laion-5b: An open large-scale dataset for training next generation image-text models,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 25278–25294, 2022.
- [55] H. Zhang, Z. Wu, Z. Wang, Z. Chen, and Y.-G. Jiang, “Prototypical residual networks for anomaly detection and localization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16281–16291, 2023.
- [56] T. Karras, S. Laine, and T. Aila, “A style-based generator architecture for generative adversarial networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.
- [57] X. Zhang, M. Xu, and X. Zhou, “Realnet: A feature selection network with realistic synthetic anomaly for anomaly detection,” in *Proceedings of*

- the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16699–16708, 2024.
- [58] O. Avrahami, D. Lischinski, and O. Fried, “Blended diffusion for text-driven editing of natural images,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18208–18218, 2022.
- [59] M. Lee and J. Choi, “Text-guided variational image generation for industrial anomaly detection and segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26519–26528, 2024.
- [60] A. Van Den Oord, O. Vinyals, *et al.*, “Neural discrete representation learning,” *Advances in neural information processing systems*, vol. 30, 2017.
- [61] P. Bergmann, M. Fauser, D. Sattlegger, and C. 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*, pp. 9592–9600, 2019.
- [62] C. Schuhmann, R. Vencu, R. Beaumont, R. Kaczmarczyk, C. Mullis, A. Katta, T. Coombes, J. Jitsev, and A. Komatsuzaki, “Laion-400m: Open dataset of clip-filtered 400 million image-text pairs,” *arXiv preprint arXiv:2111.02114*, 2021.
- [63] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable visual models from natural language supervision,” in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.
- [64] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” in *International Conference on Learning Representations*, 2021.
- [65] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [66] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.
- [67] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer, “Spot-the-difference self-supervised pre-training for anomaly detection and segmentation,” in *European Conference on Computer Vision*, pp. 392–408, Springer, 2022.

- [68] S. Zhao, Z. Liu, J. Lin, J.-Y. Zhu, and S. Han, “Differentiable augmentation for data-efficient gan training,” *Advances in neural information processing systems*, vol. 33, pp. 7559–7570, 2020.
- [69] U. Ojha, Y. Li, J. Lu, A. A. Efros, Y. J. Lee, E. Shechtman, and R. Zhang, “Few-shot image generation via cross-domain correspondence,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10743–10752, 2021.
- [70] B. L. WELCH, “The generalization of ‘student’s’ problem when several different population variances are involved,” *Biometrika*, vol. 34, pp. 28–35, 01 1947.
- [71] R. Rasti, A. Biglari, M. Rezapourian, Z. Yang, and S. Farsiu, “Retifluidnet: a self-adaptive and multi-attention deep convolutional network for retinal oct fluid segmentation,” *IEEE Transactions on Medical Imaging*, vol. 42, no. 5, pp. 1413–1423, 2022.
- [72] K. Perlin, “An image synthesizer,” *ACM Siggraph Computer Graphics*, vol. 19, no. 3, pp. 287–296, 1985.
- [73] J. Romero, D. Tzionas, and M. J. Black, “Embodied hands: Modeling and capturing hands and bodies together,” *arXiv preprint arXiv:2201.02610*, 2022.
- [74] G. Rogez, J. S. Supancic, and D. Ramanan, “Understanding everyday hands in action from rgb-d images,” in *Proceedings of the IEEE international conference on computer vision*, pp. 3889–3897, 2015.
- [75] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, “Real-time joint tracking of a hand manipulating an object from rgb-d input,” in *European conference on computer vision*, pp. 294–310, Springer, 2016.
- [76] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt, “Real-time hand tracking under occlusion from an egocentric rgb-d sensor,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1154–1163, 2017.
- [77] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim, “First-person hand action benchmark with rgb-d videos and 3d hand pose annotations,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 409–419, 2018.

- [78] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid, “Learning joint reconstruction of hands and manipulated objects,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11807–11816, 2019.
- [79] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox, “Freihand: A dataset for markerless capture of hand pose and shape from single rgb images,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 813–822, 2019.
- [80] O. Taheri, N. Ghorbani, M. J. Black, and D. Tzionas, “Grab: A dataset of whole-body human grasping of objects,” in *European conference on computer vision*, pp. 581–600, Springer, 2020.
- [81] S. Brahmabhatt, C. Tang, C. D. Twigg, C. C. Kemp, and J. Hays, “Contactpose: A dataset of grasps with object contact and hand pose,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pp. 361–378, Springer, 2020.
- [82] E. Corona, A. Pumarola, G. Alenya, F. Moreno-Noguer, and G. Rogez, “Ganhand: Predicting human grasp affordances in multi-object scenes,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5031–5041, 2020.
- [83] S. Hampali, M. Rad, M. Oberweger, and V. Lepetit, “Honnotate: A method for 3d annotation of hand and object poses,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3196–3206, 2020.
- [84] Y.-W. Chao, W. Yang, Y. Xiang, P. Molchanov, A. Handa, J. Tremblay, Y. S. Narang, K. Van Wyk, U. Iqbal, S. Birchfield, *et al.*, “Dexycb: A benchmark for capturing hand grasping of objects,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9044–9053, 2021.
- [85] T. Kwon, B. Tekin, J. Stühmer, F. Bogo, and M. Pollefeys, “H2o: Two hands manipulating objects for first person interaction recognition,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10138–10148, 2021.
- [86] L. Yang, K. Li, X. Zhan, F. Wu, A. Xu, L. Liu, and C. Lu, “Oakink: A large-scale knowledge repository for understanding hand-object interaction,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 20953–20962, 2022.

-
- [87] D. Huang, X. Ji, X. He, J. Sun, T. He, Q. Shuai, W. Ouyang, and X. Zhou, “Reconstructing hand-held objects from monocular video,” in *SIGGRAPH Asia 2022 Conference Papers*, pp. 1–9, 2022.
 - [88] A. Swamy, V. Leroy, P. Weinzaepfel, F. Baradel, S. Galaaoui, R. Brégier, M. Armando, J.-S. Franco, and G. Rogez, “Showme: Benchmarking object-agnostic hand-object 3d reconstruction,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1935–1944, 2023.
 - [89] A. Guo, B. Wen, J. Yuan, J. Tremblay, S. Tyree, J. Smith, and S. Birchfield, “Handal: A dataset of real-world manipulable object categories with pose annotations, affordances, and reconstructions,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11428–11435, IEEE, 2023.
 - [90] Z. Fan, O. Taheri, D. Tzionas, M. Kocabas, M. Kaufmann, M. J. Black, and O. Hilliges, “Arctic: A dataset for dexterous bimanual hand-object manipulation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12943–12954, 2023.
 - [91] W. Qu, Z. Cui, Y. Zhang, C. Meng, C. Ma, X. Deng, and H. Wang, “Novel-view synthesis and pose estimation for hand-object interaction from sparse views,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 15100–15111, 2023.
 - [92] P. Banerjee, S. Shkodrani, P. Moulon, S. Hampali, S. Han, F. Zhang, L. Zhang, J. Fountain, E. Miller, S. Basol, *et al.*, “Hot3d: Hand and object tracking in 3d from egocentric multi-view videos,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 7061–7071, 2025.
 - [93] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3d gaussian splatting for real-time radiance field rendering,” *ACM Transactions on Graphics*, vol. 42, July 2023.
 - [94] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.
 - [95] H. Jin, H. Jiang, H. Tan, K. Zhang, S. Bi, T. Zhang, F. Luan, N. Snavely, and Z. Xu, “Lvsm: A large view synthesis model with minimal 3d inductive bias,” *arXiv preprint arXiv:2410.17242*, 2024.

- [96] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, “Vggt: Visual geometry grounded transformer,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025.
- [97] R. Ranftl, A. Bochkovskiy, and V. Koltun, “Vision transformers for dense prediction,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12179–12188, 2021.
- [98] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, “Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1949–1957, 2015.
- [99] I. M. Bullock, T. Feix, and A. M. Dollar, “The yale human grasping dataset: Grasp, object, and task data in household and machine shop environments,” *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 251–255, 2015.
- [100] M. Cai, K. M. Kitani, and Y. Sato, “A scalable approach for understanding the visual structures of hand grasps,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1360–1366, IEEE, 2015.
- [101] A. Fathi, X. Ren, and J. M. Rehg, “Learning to recognize objects in egocentric activities,” in *CVPR 2011*, pp. 3281–3288, IEEE, 2011.
- [102] D. Tzionas, L. Ballan, A. Srikantha, P. Aponte, M. Pollefeys, and J. Gall, “Capturing hands in action using discriminative salient points and physics simulation,” *International Journal of Computer Vision*, vol. 118, pp. 172–193, 2016.
- [103] J. P. Lewis, M. Cordner, and N. Fong, “Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation,” in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '00, (USA)*, p. 165–172, ACM Press/Addison-Wesley Publishing Co., 2000.
- [104] J. Wang, F. Mueller, F. Bernard, S. Sorli, O. Sotnychenko, N. Qian, M. A. Otaduy, D. Casas, and C. Theobalt, “Rgb2hands: real-time tracking of 3d hand interactions from monocular rgb video,” *ACM Transactions on Graphics (ToG)*, vol. 39, no. 6, pp. 1–16, 2020.
- [105] F. Mueller, M. Davis, F. Bernard, O. Sotnychenko, M. Verschoor, M. A. Otaduy, D. Casas, and C. Theobalt, “Real-time pose and shape

- reconstruction of two interacting hands with a single depth camera,” *ACM Transactions on Graphics (ToG)*, vol. 38, no. 4, pp. 1–13, 2019.
- [106] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, “Cosypose: Consistent multi-view multi-object 6d pose estimation,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, pp. 574–591, Springer, 2020.
- [107] Y. Labbé, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic, “Megapose: 6d pose estimation of novel objects via render & compare,” *arXiv preprint arXiv:2212.06870*, 2022.
- [108] M. Tian, M. H. Ang Jr, and G. H. Lee, “Shape prior deformation for categorical 6d object pose and size estimation,” in *European Conference on Computer Vision*, pp. 530–546, Springer, 2020.
- [109] M. Slavcheva, W. Kehl, N. Navab, and S. Ilic, “Sdf-2-sdf registration for real-time 3d reconstruction from rgb-d data,” *International Journal of Computer Vision*, vol. 126, pp. 615–636, 2018.
- [110] Y. Liu, Y. Wen, S. Peng, C. Lin, X. Long, T. Komura, and W. Wang, “Gen6d: Generalizable model-free 6-dof object pose estimation from rgb images,” in *European Conference on Computer Vision*, pp. 298–315, Springer, 2022.
- [111] K. Park, A. Mousavian, Y. Xiang, and D. Fox, “Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10710–10719, 2020.
- [112] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou, “Onepose: One-shot object pose estimation without cad models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6825–6834, 2022.
- [113] G. Rogez, J. S. Supancic, and D. Ramanan, “First-person pose recognition using egocentric workspaces,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4325–4333, 2015.
- [114] C. Choi, S. Ho Yoon, C.-N. Chen, and K. Ramani, “Robust hand pose estimation during the interaction with an unknown object,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3123–3132, 2017.

-
- [115] J. Lei, S. Sridhar, P. Guerrero, M. Sung, N. Mitra, and L. J. Guibas, “Pix2surf: Learning parametric 3d surface models of objects from images,” in *European Conference on Computer Vision*, pp. 121–138, Springer, 2020.
 - [116] Z. Yang, Z. Ren, M. A. Bautista, Z. Zhang, Q. Shan, and Q. Huang, “Fvor: Robust joint shape and pose optimization for few-view object reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2497–2507, 2022.
 - [117] D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, “Neural rgb-d surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6290–6301, 2022.
 - [118] M. Oechsle, S. Peng, and A. Geiger, “Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5589–5599, 2021.
 - [119] J. Sun, Y. Xie, L. Chen, X. Zhou, and H. Bao, “Neuralrecon: Real-time coherent 3d reconstruction from monocular video,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15598–15607, 2021.
 - [120] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, “Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction,” *arXiv preprint arXiv:2106.10689*, 2021.
 - [121] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, “Volume rendering of neural implicit surfaces,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 4805–4815, 2021.
 - [122] T. Patten, K. Park, M. Leitner, K. Wolfram, and M. Vincze, “Object learning for 6d pose estimation and grasping from rgb-d videos of in-hand manipulation,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4831–4838, IEEE, 2021.
 - [123] I. Oikonomidis, G. Garcia-Hernando, A. Yao, A. Argyros, V. Lepetit, and T.-K. Kim, “Hands18: Methods, techniques and applications for hand observation,” in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pp. 0–0, 2018.

- [124] S. Jiang, Q. Ye, R. Xie, Y. Huo, X. Li, Y. Zhou, and J. Chen, “In-hand 3d object reconstruction from a monocular rgb video,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 2525–2533, 2024.
- [125] J. Wu, G. Pavlakos, G. Gkioxari, and J. Malik, “Reconstructing hand-held objects in 3d from images and videos,” *arXiv preprint arXiv:2404.06507*, 2024.
- [126] A. Prakash, M. Chang, M. Jin, and S. Gupta, “Learning hand-held object reconstruction from in-the-wild videos,” *arXiv preprint*, 2023.
- [127] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, “Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes,” in *Asian conference on computer vision*, pp. 548–562, Springer, 2012.
- [128] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, “Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes,” *arXiv preprint arXiv:1711.00199*, 2017.
- [129] T. Hodan, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis, “T-less: An rgb-d dataset for 6d pose estimation of texture-less objects,” in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 880–888, IEEE, 2017.
- [130] G. Hillebrand, M. Bauer, K. Achatz, G. Klinker, and A. Oferl, “Inverse kinematic infrared optical finger tracking,” in *Proceedings of the 9th International Conference on Humans and Computers (HC 2006)*, Aizu, Japan, pp. 6–9, 2006.
- [131] R. Y. Wang and J. Popović, “Real-time hand-tracking with a color glove,” *ACM transactions on graphics (TOG)*, vol. 28, no. 3, pp. 1–8, 2009.
- [132] T. G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill, “A hand gesture interface device,” *ACM Sigchi Bulletin*, vol. 18, no. 4, pp. 189–192, 1986.
- [133] J. Reizenstein, R. Shapovalov, P. Henzler, L. Sbordon, P. Labatut, and D. Novotny, “Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction,” in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10881–10891, 2021.
- [134] H. Xia, Y. Fu, S. Liu, and X. Wang, “RGBD Objects in the Wild: Scaling Real-World 3D Object Learning from RGB-D Videos,” in *2024 IEEE/CVF*

- Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 22378–22389, IEEE Computer Society, June 2024.
- [135] J. L. Schönberger and J.-M. Frahm, “Structure-from-motion revisited,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [136] G. Bradski, “The opencv library.,” *Dr. Dobb’s Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.
- [137] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang, “Grounded sam: Assembling open-world models for diverse visual tasks,” 2024.
- [138] S. Umeyama, “Least-squares estimation of transformation parameters between two point patterns,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 13, no. 4, pp. 376–380, 2002.
- [139] Q. Huynh-Thu and M. Ghanbari, “Scope of validity of psnr in image/video quality assessment,” *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.
- [140] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [141] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.
- [142] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, *et al.*, “Nerfstudio: A modular framework for neural radiance field development,” in *ACM SIGGRAPH 2023 conference proceedings*, pp. 1–12, 2023.
- [143] G. Zhang, K. Cui, T.-Y. Hung, and S. Lu, “Defect-gan: High-fidelity defect synthesis for automated defect inspection,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2524–2534, 2021.
- [144] J. Sutherland, M. Arbel, and A. Gretton, “Demystifying mmd gans,” in *International Conference for Learning Representations*, pp. 1–36, 2018.

- [145] S. Lee, S. Lee, and B. C. Song, “Cfa: Coupled-hypersphere-based feature adaptation for target-oriented anomaly localization,” *IEEE Access*, vol. 10, pp. 78446–78454, 2022.
- [146] C. Ding, G. Pang, and C. Shen, “Catching both gray and black swans: Open-set supervised anomaly detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [147] G. Pang, C. Ding, C. Shen, and A. v. d. Hengel, “Explainable deep few-shot anomaly detection with deviation networks,” *arXiv preprint arXiv:2108.00462*, 2021.
- [148] T. D. Tien, A. T. Nguyen, N. H. Tran, T. D. Huy, S. Duong, C. D. T. Nguyen, and S. Q. Truong, “Revisiting reverse distillation for anomaly detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24511–24520, 2023.
- [149] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.
- [150] Z. Liu, Y. Zhou, Y. Xu, and Z. Wang, “Simplenet: A simple network for image anomaly detection and localization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20402–20411, 2023.
- [151] N.-C. Ristea, N. Madan, R. T. Ionescu, K. Nasrollahi, F. S. Khan, T. B. Moeslund, and M. Shah, “Self-supervised predictive convolutional attentive block for anomaly detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13576–13586, 2022.
- [152] J. Yu, Y. Zheng, X. Wang, W. Li, Y. Wu, R. Zhao, and L. Wu, “Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows,” *arXiv preprint arXiv:2111.07677*, 2021.
- [153] N. Cohen and Y. Hoshen, “Sub-image anomaly detection with deep pyramid correspondences,” *arXiv preprint arXiv:2005.02357*, 2020.
- [154] P. Bergmann, K. Batzner, M. Fauser, D. Sattlegger, and C. Steger, “Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization,” *International Journal of Computer Vision*, vol. 130, no. 4, p. 947–969, 2022.

- [155] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *International conference on machine learning*, pp. 1597–1607, PMLR, 2020.
- [156] C. Bishop and M. Tipping, “Pattern analysis and machine intelligence,” *IEEE Transactions on*, vol. 20, no. 3, pp. 281–293, 1998.
- [157] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, “Emerging properties in self-supervised vision transformers,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.
- [158] Y. Li, Q. Lao, Q. Kang, Z. Jiang, S. Du, S. Zhang, and K. Li, “Self-supervised anomaly detection, staging and segmentation for retinal images,” *Medical Image Analysis*, vol. 87, p. 102805, 2023.
- [159] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.
- [160] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, “Deep autoencoding models for unsupervised anomaly segmentation in brain mr images,” in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part I 4*, pp. 161–169, Springer, 2019.
- [161] O. Avrahami, D. Lischinski, and O. Fried, “Blended diffusion for text-driven editing of natural images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18208–18218, 2022.
- [162] V. Zavrtanik, M. Kristan, and D. Skočaj, “Draem-a discriminatively trained reconstruction embedding for surface anomaly detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8330–8339, 2021.
- [163] S. Niu, B. Li, X. Wang, and H. Lin, “Defect image sample generation with gan for improving defect recognition,” *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1611–1622, 2020.
- [164] H. Li, Z. Zhang, H. Chen, L. Wu, B. Li, D. Liu, and M. Wang, “A novel approach to industrial defect generation through blended latent diffusion model with online adaptation,” *arXiv preprint arXiv:2402.19330*, 2024.

- [165] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, “Ganomaly: Semi-supervised anomaly detection via adversarial training,” in *Computer Vision—ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, pp. 622–637, Springer, 2019.
- [166] W. Liu, R. Li, M. Zheng, S. Karanam, Z. Wu, B. Bhanu, R. J. Radke, and O. Camps, “Towards visually explaining variational autoencoders,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8642–8651, 2020.
- [167] D. Dehaene, O. Frigo, S. Combrexelle, and P. Eline, “Iterative energy-based projection on a normal data manifold for anomaly localization,” in *International Conference on Learning Representations*, 2020.
- [168] Y. Zou, J. Jeong, L. Pemula, D. Zhang, and O. Dabeer, “Spot-the-difference self-supervised pre-training for anomaly detection and segmentation,” in *European Conference on Computer Vision*, pp. 392–408, Springer, 2022.
- [169] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, *et al.*, “Photorealistic text-to-image diffusion models with deep language understanding,” *Advances in neural information processing systems*, vol. 35, pp. 36479–36494, 2022.
- [170] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, “Deep learning for anomaly detection: A review,” *ACM computing surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.
- [171] J. Hu, Y. Chen, and Z. Yi, “Automated segmentation of macular edema in oct using deep neural networks,” *Medical image analysis*, vol. 55, pp. 216–227, 2019.
- [172] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [173] T. H. E. Tse, K. I. Kim, A. Leonardis, and H. J. Chang, “Collaborative learning for hand and object reconstruction with attention-guided graph convolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1664–1674, 2022.
- [174] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, “Real-time joint tracking of a hand manipulating an object

- from rgb-d input,” in *European conference on computer vision*, pp. 294–310, Springer, 2016.
- [175] A. Bardo, K. Town, T. L. Kivell, G. Donati, H. Ballieux, C. Stamate, T. Edginton, and G. S. Forrester, “The precision of the human hand: variability in pinch strength and manual dexterity,” *Symmetry*, vol. 14, no. 1, p. 71, 2022.
- [176] Z. Lin, C. Ding, H. Yao, Z. Kuang, and S. Huang, “Harmonious feature learning for interactive hand-object pose estimation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12989–12998, 2023.
- [177] L. Yang, K. Li, X. Zhan, J. Lv, W. Xu, J. Li, and C. Lu, “Artiboost: Boosting articulated 3d hand-object pose estimation via online exploration and synthesis,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2750–2760, 2022.
- [178] S. Hampali, S. D. Sarkar, M. Rad, and V. Lepetit, “Keypoint transformer: Solving joint identification in challenging hands and object interactions for accurate 3d pose estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11090–11100, 2022.
- [179] Y. Chen, Z. Tu, D. Kang, R. Chen, L. Bao, Z. Zhang, and J. Yuan, “Joint hand-object 3d reconstruction from a single image with cross-branch feature fusion,” *IEEE Transactions on Image Processing*, vol. 30, pp. 4008–4021, 2021.
- [180] Y. Liu, Y. Liu, C. Jiang, K. Lyu, W. Wan, H. Shen, B. Liang, Z. Fu, H. Wang, and L. Yi, “Hoi4d: A 4d egocentric dataset for category-level human-object interaction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21013–21022, 2022.
- [181] R. Carkaxhia, M. Ali, N. Fioraio, L. D. Stefano, and S. Salti, “Few-shot anomaly classification by learning to inpaint nominal images,” in *Proceedings of the 23rd International Conference on Image Analysis and Processing (ICIAP 2025)*, (Rome, Italy), Springer, Sept. 2025.
- [182] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities,” *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.