Single-View Reconstruction: Optimizing One-2345 for Architectural Structures

Yikai Du 522030910121 | Yuan Gao 522030910129 | Yu Huang 522030910153 | Jiayang Li 522030910113 | Jiarui Li 522030910119

Ye Tao 522030910126 | Yuqing Tu 522030910152 | Huayi Wang 522030910116 | Shaojie Yin 522030910151 | Junqi You 522030910204

Zihao Zheng 522030910112 | Yihang Zhou 522030910219 | Zheli Zhou 522030910150

Abstract

With the rapid advancements in generative foundation models and neural 3D reconstruction techniques, the previously daunting challenge of single-view reconstruction has become a focal point of research and development in both academia and industry. Leveraging the rich prior knowledge embedded in diffusion models, One-2345 integrates multi-view synthesis, elevation estimation, and 3D reconstruction into a unified pipeline. Its departure from test-time optimization allows for rapid inference but introduces limitations when reconstructing objects from specific genres. To tackle these challenges, we compile a dataset of 5,000 3D models centered on architectural structures, using a web crawler and a user-friendly data-filtering program. Extensive experiments are carried out to evaluate the performance improvements resulting from fine-tuning the model on a genre-specific dataset. Based on detailed analysis and observations, we also propose several promising directions for future improvements.

1. Introduction

In recent years, the field of computer vision has undergone significant advancements, especially in the area of generating three-dimensional (3D) models from single-view images. This capability is vital for a wide range of applications, including virtual and augmented reality, robotics, autonomous navigation systems, 3D printing, and Digital heritage preservation. However, creating accurate 3D reconstructions from two-dimensional (2D) images remains a substantial challenge, as it requires the extraction and interpretation of complex spatial information embedded within flat visuals.

Traditionally, precise 3D object generation has relied heavily on multi-view datasets or specialized depth-sensing hardware. While these methods have proven effective, they are often limited by the requirement for additional data or equipment, which may not always be practical in real-world scenarios. In contrast, single-view 3D reconstruction must overcome the inherent difficulties of inferring depth, structural details, and surface textures from a single perspective. This necessitates the development of more sophisticated computational techniques that can accurately interpret limited visual information.

The emergence of deep learning, particularly convolutional neural networks (CNNs), has revolutionized this domain by enabling more advanced feature extraction and representation learning. Among the most notable advancements are Generative Adversarial Networks (GANs)[5], which consist of two neural networks—the generator and the discriminator—that compete against each other to produce increasingly realistic data. Models like 3D-GAN[29] extend this concept to generate 3D shapes from 2D images, demonstrating the potential to create plausible and diverse 3D structures. Similarly, Pix2Vox[30] leverages a voxel-based approach to convert 2D images into detailed 3D voxel grids, enhancing the fidelity of the generated models.

Despite these advancements, numerous challenges remain, including handling occlusions, resolving viewpoint ambiguities, ensuring generalization across various object categories and preserving fine-grained details. Moreover, enhancing the interpretability of generated 3D structures and incorporating semantic information continue to be critical areas of ongoing research. One-2345 [9] circumvents these problem by integrating multi-view synthesis, eleva-

tion estimation, and 3D reconstruction in a holistic pipeline. Without costly test-time optimization, it is able to reconstruct 3D meshes from single-view image in much less time.

In this study, we thoroughly explore the capabilities of One-2345 for single-view reconstruction. To facilitate a comprehensive evaluation of its performance within a specific domain, we have curated a dataset of 5,000 3D models focused on architectural structures. We examine the zero-shot reconstruction capabilities of One-2345 and fine-tune its pretrained weights on our custom dataset. Additionally, we conduct extensive experiments to assess the impact of fine-tuning on reconstruction performance and propose potential avenues for further improvement. The key contributions of this work are summarized as follows:

- Design a crawler with convenient data filtering logic and collect a clean and robust 3D model dataset which contains 5k models of buildings and architectures.
- Explore rendering techniques and obtain multi-view images of three-dimensional objects along with their corresponding camera poses.
- Fintune the popular single-view reconstruction model One-2345 and analyze performance difference between zero-shot evaluation and finetuned model.
- Analyze drawbacks of existing pipeline and propose several promising directions for future improvements.

2. Related Works

3D Generative Models. Recent advancements in generative image architectures combined with large scale image-text datasets [21] have made it possible to synthesize high-fidelity of diverse scenes and objects [13, 19, 20]. In particular, diffusion models have shown to be very effective at learning scalable image generators using a denoising objective [3, 24]. However, scaling them to the 3D domain would require large amounts of expensive annotated 3D data. Instead, recent approaches rely on transferring pre-trained large-scale 2D diffusion models to 3D without using any ground truth 3D data. Neural Radiance Fields or NeRFs [12] have emerged as a powerful representation, thanks to their ability to encode scenes with high fidelity. Typically, NeRF is used for single-scene reconstruction, where many posed images covering the entire scene are provided. The task is then to predict novel views from unobserved angles. DreamFields [6] has shown that NeRF is a more versatile tool that can also be used as the main component in a 3D generative system. Various follow-up works [8, 18, 26] substitute CLIP for a distillation loss from a 2D diffusion model that is repurposed to generate high-fidelity 3D objects and scenes from text inputs.

Novel View Synthesis. Novel view synthesis from a single RGB image requires imagining an object's unobserved geometry and textures, which is a precursor to 3D generation despite its ill-posed nature. Regression-based methods enhance NVS quality with geometry priors, but often with blurry novel view results because of the mean-seeking nature [7, 16, 17, 32]. Generation-based methods leverage generative capabilities to produce high-fidelity novel view predictions. GAN-based approaches [2, 15] can generate high-resolution novel view images, but suffer from unstable training. [1, 10, 27] leverage diffusion models' stronger modelling capability and training stability for higher-quality NVS results. Nevertheless, their efficacy remains confined to specific categories. Recently, Zero [10] pioneers in zero-shot open-set NVS utilizing pre-trained diffusion models and diverse 3D data [4].

Single-View Reconstruction. A 3D object has details of geometry with texture and can be projected to 2D image at any views. Dense view images with camera poses or depth information are always required to rebuild 3D objects. Therefore, reconstructing an arbitrary 3D object from a single-view image is inherently an ill-posed optimization problem, making it extremely challenging and largely unexplored. Recent work has begun incorporating general visual knowledge into 3D reconstruction models, enabling the generation of 3D models from single-view images. NeuraLift360 [31] employ pre-trained depth-estimator and 2D diffusion priors to recover coarse geometry and textures of 3D objects from single image. 3DFuse [22] initialize geometry with estimated point cloud [14], then fine-grained geometry and textures are learned from single image with a 2D diffusion prior with score distillation [25]. Another work MCC [28] learns from object-centric videos to generate 3D objects from single RGB-D image. These works show promising results in this direction but still rely heavily on human-specified text or captured depth information.

3. Method

3.1. Zero-123

Zero-123[11], also known as Zero-1-to-3, is an advanced method designed to perform novel view synthesis and 3D reconstruction from a single RGB image. The primary challenge addressed by Zero-123 is the severe under-constrained nature of these tasks when only a single view of an object is available. Unlike traditional methods that rely on multiple views or depth data to generate new perspectives and 3D structures, Zero-123 utilizes the capabilities of large pre-trained diffusion models, such as Stable Diffusion, which have been trained on extensive internet-scale data. This allows Zero-123 to generate accurate 3D shapes and novel views even from a single image, by leveraging learned priors about object geometry and appearance. The diffusion model, trained on vast amounts of natural images, captures high-level semantic and low-level geometric features that facilitate generalization to unseen objects and complex artistic styles, making the method highly versatile

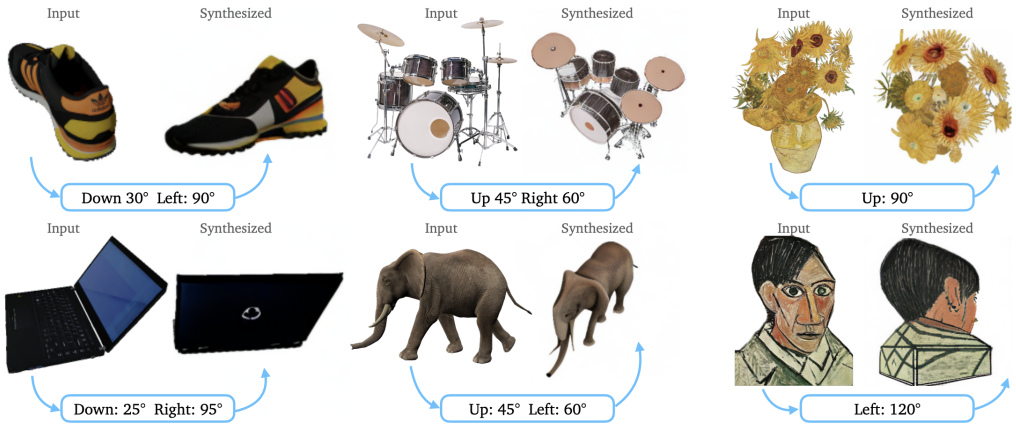

Figure 1. **Structure for Zero-123**. Given a reference view and relative transition angle, Zero-123 is able to synthesize high-fidelity image with strong spacial consistency.

in diverse scenarios.

Zero-123 integrates two core tasks: novel view synthesis and 3D shape reconstruction. Novel view synthesis involves creating images from different camera angles, while 3D reconstruction aims to infer the 3D geometry of the object. These tasks are performed in sequence, starting from generating multiple views of the object and then using those views to estimate the 3D structure. The model can generate realistic, high-quality images that preserve the geometric and textural details of the object as seen from various viewpoints, making it particularly useful for applications requiring detailed 3D reconstructions.

3.1.1. Viewpoint Control and Image Synthesis

The key innovation of Zero-123 lies in its ability to control the camera viewpoint through the fine-tuning of a pre-trained diffusion model. This control is crucial because most diffusion models are not initially designed to generate images from arbitrary viewpoints. Zero-123 solves this by introducing a mechanism to condition the model on the desired relative camera transformation, which includes both rotation and translation parameters. During training, Zero-123 is provided with pairs of images taken from different viewpoints along with their relative transformations, allowing the model to learn how to generate images based on these camera adjustments.

The input image and the relative camera transformation (rotation and translation) are fed into the diffusion model as conditional inputs, enabling the model to generate a novel view of the object from the desired perspective. The model uses these inputs to guide the generation process, ensuring that the new image remains consistent with the original input in terms of both geometry and appearance. By learning how to generate images from various viewpoints, Zero-123 can produce highly consistent novel views even when the transformations involve large relative changes, such as rotating the object by 90 degrees or more.

This viewpoint-conditioned generation allows Zero-123 to synthesize diverse images that capture the object's geometry and texture from different angles. As a result, the model is able to address the challenges posed by single-view 3D reconstruction, where the lack of multiple perspectives often leads to incomplete or inaccurate reconstructions.

3.1.2. 3D Reconstruction

Once multiple views are synthesized using the viewpoint control mechanism, Zero-123 transitions to the task of 3D reconstruction. This phase involves using the generated multi-view images to infer the 3D shape of the object. Zero-123 adopts a hybrid approach that combines diffusion models with neural field-based techniques to optimize the 3D geometry. Specifically, the model employs Score Jacobian Chaining (SJC), a method that optimizes a 3D representation based on the priors learned from the viewpoint-conditioned diffusion model.

The 3D reconstruction process begins by randomly sampling viewpoints, from which images are synthesized. These images, along with their corresponding camera poses, are used to generate a 3D representation of the object using volumetric rendering. The model introduces Gaussian noise to the rendered images, which is then iteratively denoised by the diffusion model, conditioned on the input image and the synthesized viewpoint. This process allows the model to progressively refine the 3D shape, using the geometric priors learned during training.

To ensure high-quality reconstructions, Zero-123 incorporates several regularization techniques, including depth smoothness loss and near-view consistency loss. These losses help to maintain the geometric integrity of the reconstructed object by reducing inconsistencies between nearby

views and ensuring that the 3D shape remains smooth and accurate across different perspectives. The final output of this phase is a 3D mesh that faithfully represents the shape and appearance of the object, enabling detailed 3D visualizations and applications such as 3D printing or virtual reality.

3.1.3. Dataset and Fine-Tuning

Zero-123's ability to generalize to various object types and scenarios comes from its fine-tuning on a large-scale synthetic dataset. The Objaverse dataset, a comprehensive collection of 3D models created by artists, serves as the basis for fine-tuning the model. This dataset contains diverse 3D objects with rich geometry and texture, which are rendered from multiple viewpoints using a ray-tracing engine to generate high-quality training images. Each object in the dataset is paired with its corresponding relative camera transformations, allowing Zero-123 to learn how to control the camera viewpoint during image synthesis.

During the fine-tuning process, the pre-trained diffusion model is adapted to learn the relationships between the input image and the camera transformations. The model is conditioned on the CLIP embeddings of the input images and the relative viewpoint information, which enables it to generate new views from arbitrary perspectives. The fine-tuning process ensures that the model can effectively handle a wide variety of objects and geometries, enabling it to generalize well to out-of-distribution data, such as in-the-wild images or artistic representations.

This ability to adapt to new object types and scenarios is crucial for the zero-shot performance of Zero-123. The model can generate high-quality novel views and 3D reconstructions even for objects it has never seen before, making it a powerful tool for applications in computer vision, 3D modeling, and augmented reality.

3.2. One-2345

3.2.1. Pipeline

One-2345 [9] is a single-view reconstruction method based on Zero123 [23]. One-2345 has three main steps as shown in figure 2.

- **Multi-view synthesis**:processed input image is applied to Zero123 to generate multi-view images in a two-stage manner;
- **Pose estimation**:the elevation angle θ of input view is estimated based on four nearby views generated by Zero123, then all the poses of generated multi-view images is calculated by combining the specified relative poses with the estimated pose of the input view;
- **3D reconstruction**:the multi-view posed images are given to an SDF-based generalizable neural surface reconstruction module for 3D mesh reconstruction.

3.2.2. Camera Pose Estimation

The reconstruction module requires camera poses for $4 \times n$ source view images. To use Zero123 for image synthesis, the camera parameters in the standard spherical coordinate system (θ, ϕ, r) are required, where θ, ϕ and r represent the elevation angle, azimuth angle, and radius. While it's possible to simultaneously adjust the zimuth angle and radius of all source view images, the knowledge of the absolute elevation angle θ is required to determine the relative poses of all cameras in a standard XYZ frame.

Therefore, One-2345 propose an elevation estimation module to infer the elevation angle of the input image. First, four nearby views of the input image are generated by Zero123; Then, the estimation module enumerates all possible elevation angles in a coarse-to-fine manner. For each elevation candidate angle,corresponding camera poses for the four images and reprojection error for the set of camera poses are calculated to measure the consistency between the images and the camera poses. The elevation angle with the smallest reprojection error is used to generate the estimation result, then used to infer the parameters of all the views.

3.2.3. 3D Reconstruction

The reconstruction module first uses a 2D feature network to extract m 2D feature maps. It then constructs a 3D cost volume by projecting each 3D voxel onto m 2D feature planes and computing the variance of features at the projected positions. A sparse 3D CNN processes the cost volume to generate a geometric volume encoding the input shape's geometry.

To predict the Signed Distance Function (SDF) for any 3D point, an MLP network takes the 3D coordinates and interpolated features from the geometric volume as input. Another MLP network predicts the color of a 3D point using the 2D features at the projected position, interpolated geometric features, and the viewing direction. The network predicts blending weights for each source view, and the final color is computed as a weighted sum of the projected colors. An SDF-based rendering technique is applied for RGB and depth rendering.

The model is trained on a 3D object dataset with Zero123 frozen. Zero123 normalizes the training shapes using a spherical camera model. For each shape, n ground-truth RGB and depth images are rendered from n camera poses on a sphere. During training, all $4 \times n$ predicted results with ground-truth poses are input into the reconstruction module, and one ground-truth RGB image is randomly selected as the target view. Ground-truth RGB and depth values supervise the training, enabling the module to handle inconsistent predictions and reconstruct a consistent 360° mesh.

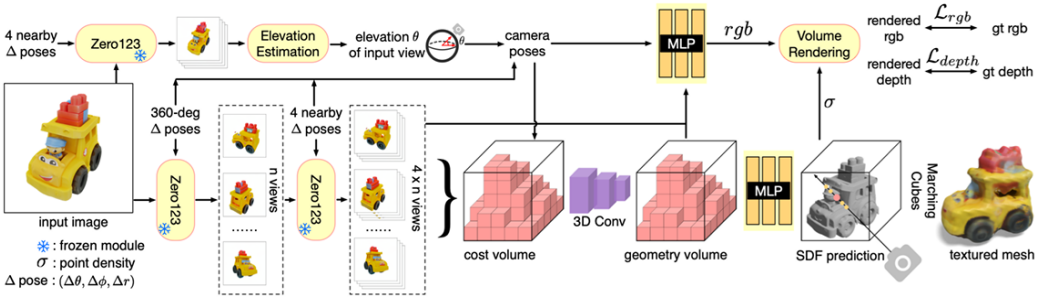

Figure 2. Three primary components of One-2345

3.3. CLIP

In traditional supervised learning, models are trained to distinguish between categories, such as cats and dogs, by providing labeled images and iteratively minimizing error until the model learns to differentiate between the classes. While this approach is effective and widely used across various tasks, it often results in highly specialized models that perform well only within the scope of their training data. For example, supervised models trained on ImageNet demonstrate excellent performance within this dataset but experience significant performance drops when exposed to similar datasets with variations in representation. In contrast, CLIP (Contrastive LanguageImage Pretraining) offers a more robust and generalized representation, making it less prone to over-specialization.

The core innovation of CLIP lies in its use of internet-sourced image-caption pairs to create a model capable of predicting whether a given text matches an image. CLIP achieves this by learning to embed both images and text into a shared embedding space. In this space, embeddings of matching image-text pairs are positioned closer together, while embeddings of non-matching pairs are placed farther apart. This process is a hallmark of contrastive learning, where the goal is to learn representations that distinguish between similar and dissimilar entities.

Specifically, CLIP employs two encoders: a text encoder and an image encoder. These encoders map input text and images into vector representations within the embedding space. During training, the model maximizes the distance between embeddings of non-matching pairs and minimizes the distance between embeddings of matching pairs. This approach allows CLIP to generalize beyond its training data and perform well across a variety of tasks.

The training strategy adopted by CLIP enables a wide range of applications:

- Image Classification: By querying the model with textual descriptions such as "a photo of a cat" or "a photo of a dog," CLIP can determine which text is most relevant to a given image, effectively functioning as an image classifier.
- Image Retrieval: CLIP can be used to build image search systems that retrieve images most relevant to a given textual query.
- Evaluation of 3D Reconstruction Models: CLIP's ability to associate text and images enables its use in evaluating the quality of 3D reconstructions, among other tasks.

4. Data Preparation

4.1. Data Acquisition

Sketchfab is a widely used 3D model showcase platform that offers a large number of 3D model files for download and use. For researchers or developers, acquiring these models on a large scale helps in training data for machine learning, computer vision, and other fields. To facilitate the bulk downloading of these models, we developed an automated script that uses Python's Selenium library to simulate browser operations and scrape a large number of 3D models from Sketchfab, saving them locally.

4.1.1. Crawler Design and Implementation

The main objective of this crawler script is to batch download 3D model data from Sketchfab while ensuring the downloading process is automated, stable, and efficient.

Data Source The data is sourced from publicly shared 3D model pages on Sketchfab. These pages contain download links for various 3D models, and the script parses each page to extract the corresponding download buttons, completing the file download. The list of URLs to be downloaded is stored in a local file, with one URL per line.

Data Scale The goal of this crawl is to retrieve data for approximately 5000 3D architecture models from Sketchfab. Each model's data includes a .glb file, with sizes typically ranging from 5MB to 150MB depending on the model's complexity and detail. Each model download takes about half a minute, the total download time will be about 40 hours, though the actual time will depend on network bandwidth and server response speed.

Crawler Workflow The main workflow of the crawler is as follows:

1. Read the file containing the list of URLs to be downloaded (`urls_file`), where each URL corresponds to a publicly shared 3D model page on Sketchfab.
2. Visit each URL and check if the model has already been downloaded to avoid duplicate downloads.
3. If the model has not been downloaded, the script will simulate a login to Sketchfab, click the "Download 3D Model" button, and initiate the download.
4. The download process is monitored by checking for .crdownload files in the download directory to ensure the file is fully downloaded.
5. Once the download is complete, the script records the URL in the `downloaded_urls_file` to skip it in future runs.

Technical Implementation The core technical implementation of the script includes:

- Using `Selenium WebDriver` to control the Chrome browser for automating web operations.
- Simulating the login process using pre-stored account information for automatic login.
- Automating the clicking of the download button using the browser and monitoring the download progress to ensure successful file download.
- Using the `tqdm` progress bar library to display download progress and enhance the user experience.

Performance and Optimization Given that we need to download about 5000 models of architecture, download efficiency and stability are critical factors in the crawler's design. To improve performance, the script incorporates the following optimization measures:

- Delay control during the download process: After clicking the download button, the script pauses for a certain period to ensure that the file starts downloading before proceeding with the next operation.
- Determining if the download is complete: The script checks for the presence of .crdownload files in the download directory to verify that the download has finished, preventing failures due to browser caching or network issues.
- Error handling: The script includes basic exception handling to manage common network errors or situations where page elements cannot be found, ensuring the stability of the download task.

4.1.2. Data Statistics and Results

As of now, the crawler has successfully downloaded approximately 5000 3D models from Sketchfab. These models are primarily in .glb format and are stored locally in a specified download directory. The total size of the downloaded files is about 100GB, with both the number of models and their sizes meeting expectations.

Data Storage and Management The downloaded model files are stored in a specified local directory, and the URL of each model is recorded in the `downloaded_urls_file` to avoid redundant downloads. Each time the crawler runs, it checks the `downloaded_urls_file` for previously downloaded URLs and skips models that have already been downloaded.

4.1.3. Data Cleaning Process

We divided the data cleaning task into thirteen parts, with each group member responsible for screening. This tedious work took about 40 hours in total.

Screening Criteria We aimed to screen for 3D models of external perspectives of buildings. Since the standard data volume was insufficient, we slightly relaxed the screening criteria to include statues and similar items in the dataset. It should be noted that there were many internal perspectives of buildings and flat views of only the front of buildings in the data, so it was necessary to actually rotate the preview images for inspection.

Screening Process The script's screening process is as follows:

1. Sequentially read URLs from the list of URLs to be screened, automatically open the web pages, and load the model rendering images.
2. Screeners use the arrow keys to control the rotation of the model view. Pressing the "y" key indicates selection, while pressing the "n" key indicates rejection.
3. The program automatically maintains two txt files to record the selected and rejected URLs, which can be used for resuming after interruption.
4. When opening the list of URLs to be screened, it is necessary to check the previously recorded URLs. If they have been recorded, they will be skipped to avoid duplication.

4.2. Rendering

After obtaining the .glb models, we need to render them as 2D images to serve as input for fine-tuning models. Therefore, we wrote a Python script to load 3D models from a specified path, rotate the camera to capture multiple viewpoints of the model, and finally render each viewpoint as a .png image.

Input and Output - Input: Specify a file path containing the 3D model. - Output: Rendered images generated from

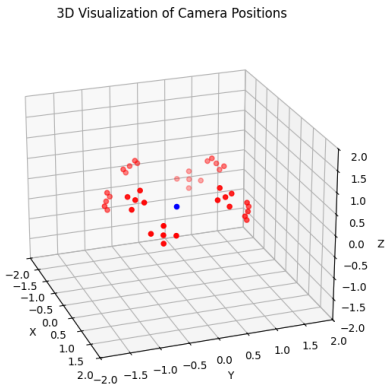

Figure 3. Camera poses used for rendering

multiple viewpoints of the model, saved to the specified output directory.

Rendering Process The rendering process includes the following steps:

1. Load the model from a supported file format and add it to the Blender scene.
2. To ensure consistent rendering, the script normalizes the loaded model to fit within the scene's range.
3. Add light sources to the scene to ensure proper illumination of the model.
4. Configure the camera's position and angle based on the given range of viewpoints to capture multiple different perspectives.
5. Render the image using Blender's rendering engine (such as `CYCLES` or `BLENDER_EEVEE`) and save it as a .png file.

Viewpoint Configuration The script samples the camera's position in a spherical coordinate system and uses different azimuthal (azimuth) and polar angles (polar angle) to achieve multi-viewpoint rendering of the model. Specifically, the camera position is calculated using the following formulas:

$$x = r \cdot \sin(\phi) \cdot \cos(\theta)$$

$$y = r \cdot \sin(\phi) \cdot \sin(\theta)$$

$$z = r \cdot \cos(\phi)$$

where r is the distance from the camera to the model, ϕ is the polar angle, and θ is the azimuthal angle. The camera poses used for rendering are shown in Figure 3.

Configuration Item	Setting Value
Rendering Engine	`CYCLES` / `BLENDER_EEVEE`
Image Format	.png
Color Mode	RGBA
Resolution	512x512
Render Percentage	100%
Device	CPU
Samples	32
Diffuse Bounces	1
Glossy Bounces	1
Transparent Bounces	3
Transmission Bounces	3
Filter Width	0.01
Denoising Enabled	Enabled
Transparent Background	Enabled
Camera Lens	35mm
Sensor Width	32mm

Table 1. Summary of Rendering Configuration

5. Experiments

5.1. Rendering Configuration

The rendering parameters are summarized in Table 1. Our script saves each rendered image from different viewpoints to the specified output directory and names the image files based on the viewpoint number. For each viewpoint, the image is saved in a background-free `.png` format for further processing.

5.2. Depth Map Rendering

The depth map rendering process consists of two main steps: depth calculation and depth map generation. After determining the camera's position, we need to compute the distance from each pixel to the camera, known as the depth value. This calculation involves the camera's intrinsic and extrinsic parameters, as well as projection information, resulting in a two-dimensional array representing the distance from each pixel on the camera image to the 3D scene. For each 3D scene, we generate depth maps from eight principal viewpoints, without creating depth maps for shifted viewpoints. Once this step is complete, we can map the calculated depth values to the image pixels to generate the depth map. Depending on the requirements, we can encode the depth values as grayscale or color values for storage and processing, providing foundational data for subsequent 3D reconstruction, object recognition, and other applications.

The presented Figure 5 shows the depth map rendering result. We inverse the value of the 2D depth matrix before generating the visualizing result to ensure the background is always black.

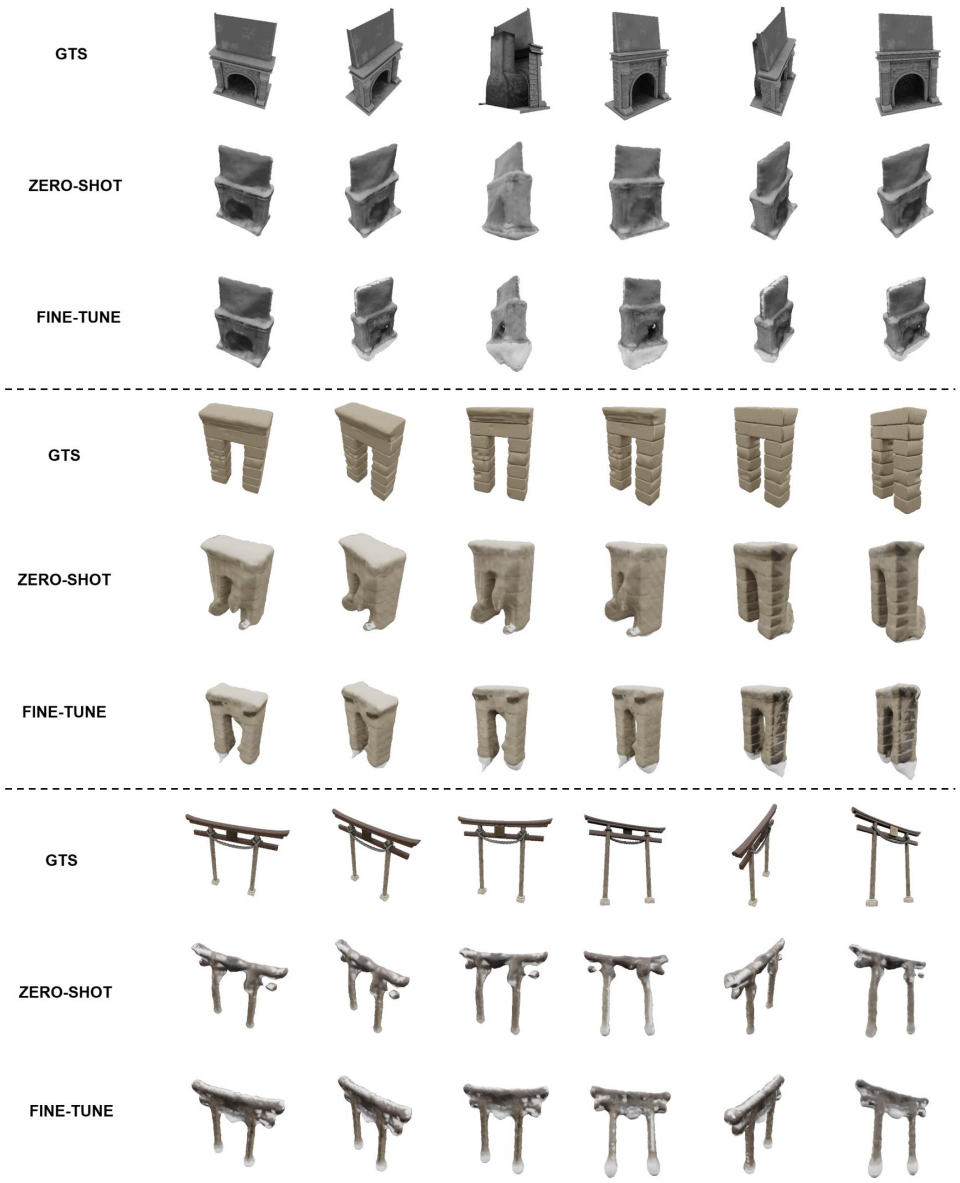

Figure 4. Visualization of Zero-shot and Finetuned Reconstruction Performance

5.3. Fintuning Settings

For each input image, we generate a total of 32 images by selecting 8 camera positions distributed evenly on the surface of a sphere. For each of these 8 viewpoints, we create 4 additional local images spaced 10° apart, yielding 32 images in total for the reconstruction process.

During the experiment, we find out that introducing depth loss often results in model collapsing. This finding is further supported by following work that depth loss will leads to white noise. Therefore, we obtain binary mask from extracted depth mask and optimize pretrained weight on mask loss instead.

We randomly pick 50 models for validation and use the rest for finetuning. After processing the 3D model into perspective images, we cast ray on each pixel and construct dataloader based on colored rays. We optimize the pretrained weight for 3 epochs with a batch-size of 512 and a learning rate of 2e-4. We keep the rest of configuration same as the original paper, including loss weight, variance weight schedualing, etc.

5.4. Metric

To evaluate the 3D reconstruction capabilities of the model before and after fine-tuning, we employ the CLIP similar-

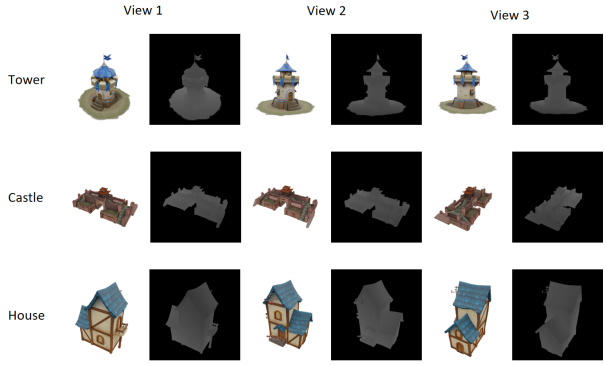

Figure 5. Depth map of 3D images from different perspectives. Left is the 2D rendering result and right is the 2D depth map result.

ity metric. As previously mentioned, CLIP can measure not only the similarity between images and text but also the similarity between two images.

Specifically, we select 98 models as the validation set. For both zero-shot and fine-tuned models, we generate a rendered result (OBJ file). For each OBJ file, we capture images from 8 different viewpoints, with 4 slightly varied orientations per viewpoint using a virtual camera, resulting in a total of 32 images per model. These images are then compared with the corresponding ground truth model images using CLIP similarity, and the final metric is obtained by averaging these similarity scores.

CLIP Similarity Calculation The CLIP similarity between two matching images I_1 and I_2 is calculated as follows:

$$\text{similarity}(I_1, I_2) = \cos(\theta) = \hat{\mathbf{v}}_1 \cdot \hat{\mathbf{v}}_2 \tag{1}$$

$$= \left(\frac{\mathbf{v}_1}{\|\mathbf{v}_1\|}\right) \cdot \left(\frac{\mathbf{v}_2}{\|\mathbf{v}_2\|}\right) \tag{2}$$

$$= \frac{f(I_1)}{\|f(I_1)\|} \cdot \frac{f(I_2)}{\|f(I_2)\|} \tag{3}$$

$$= \frac{\sum_{i=1}^{d} f(I_1)_i \cdot f(I_2)_i}{\sqrt{\sum_{i=1}^{d} f(I_1)_i^2} \cdot \sqrt{\sum_{i=1}^{d} f(I_2)_i^2}} \tag{4}$$

where f represents the CLIP image encoder that projects images into a vector space.

5.5. Main Results

5.5.1. Quantitative Results

The presented Figure 6 provides a comparative analysis of the performance of the "Zeroshot" and "Finetune" models across 98 samples, indexed from 0 to 97. Overall, the "Finetune" model demonstrates a clear advantage over the "Zeroshot" approach, as reflected in its higher peak scores and more consistent performance across the majority of samples. The scores for "Finetune" frequently surpass those of "Zeroshot," particularly in regions where challenging samples appear to create significant variability. This suggests that the "Finetune" model benefits from its ability to adapt to specific data distributions, leading to improved outcomes in scenarios that demand fine-grained optimization.

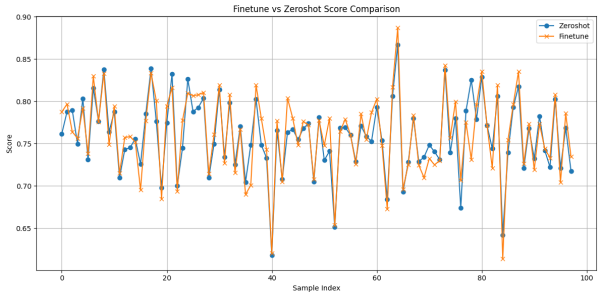

Figure 6. Clip Similarity of zero-shot and finetuned model

However, the analysis also highlights a few limitations of the "Finetune" model. For instance, while it achieves higher peak performance, it occasionally exhibits sharper declines in certain indices (e.g., around index 40), which may indicate a sensitivity to certain types of data. In contrast, the "Zeroshot" model, though generally less effective, demonstrates slightly more stable performance in some isolated cases, such as around index 60. Despite these minor drawbacks, the "Finetune" model's superior ability to maximize performance in critical scenarios makes it the preferred choice for applications requiring high accuracy and robustness. This analysis underscores the competitive nature of the two approaches while emphasizing the practical benefits of fine-tuning in enhancing model performance.

5.5.2. Qualitative Results

We provide qualitative results of zero-shot referance and the finetuned model in Fig. 4. We can find that reconstructed models clearly contains more high-frequency details after finetuning (e.g. the beam on the arch, more acute edges). Finetuned model also harbours more prior knowledge regarding architecture structures in that the reconstructed mesh demonstrate less abnormal projection.

5.6. Failure Cases

We present several common failure cases in Fig. 7. While One-2345 is able to capture the overall shape of the target structures, it struggles with high-frequency details. Although fine-tuning offers some improvement, the model still fails to accurately reconstruct complex features, such as the intricate mane of a stone lion' s head. Additionally, due to the relatively uniform color distribution in our dataset, the model encounters difficulties in faithfully reproducing colors with deeper hues.

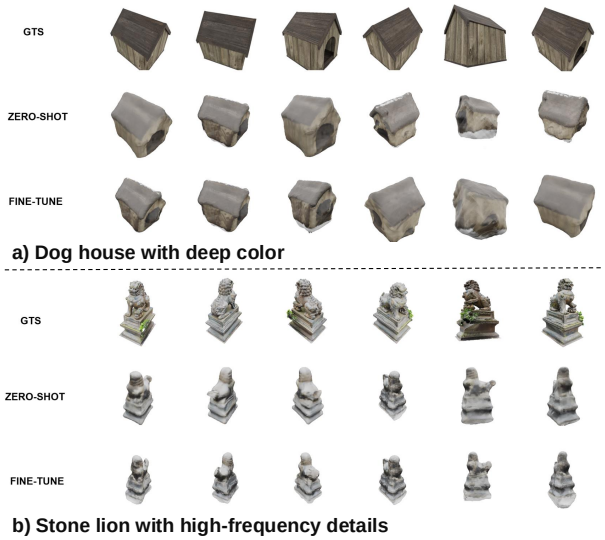

Figure 7. Failure Reconstruction Cases

5.7. Discussion and Improvement

While fine-tuning the One-2345 model has demonstrated promising results on our new dataset, several critical limitations remain within the overall pipeline that must be addressed to improve performance. One of the primary issues is the inconsistency between views generated by the Zero123 model. This arises because the Zero123 model independently estimates each view' s conditional marginal distribution without accounting for the interdependencies between different views during the multi-view generation process. Specifically, by neglecting the relational context between views, the model fails to enforce coherence across different perspectives of the same object, leading to noticeable discrepancies in shape, texture, and overall 3D structure. This problem becomes particularly pronounced in more complex scenes where subtle inconsistencies between views can significantly affect the final 3D reconstruction.

In addition to this, the process of constructing the cost volume using estimated elevation angles and inconsistent views exacerbates the issue by introducing further noise. When views are misaligned or inconsistent, the cost volume construction, which typically relies on depth information and spatial relations across different viewpoints, becomes prone to significant error. The propagation of these inconsistencies through the cost volume leads to an accumulation of noise, further degrading the quality of the 3D reconstruction and introducing artifacts such as blurred surfaces or incorrect object geometries. As a result, the overall accuracy and robustness of the 3D models produced by the pipeline suffer, making it essential to address both the inter-view consistency and the noise in the cost volume.

In order to tackle these limitations, future work should focus on reducing the inconsistencies in multi-view generation and improving the robustness of the 3D reconstruction process. One promising direction is the integration of inter-view communication during the diffusion process, which could enable the model to jointly optimize the generation of multiple views rather than treating them as independent. This could be achieved by designing a model that explicitly incorporates inter-view relationships into the diffusion mechanism, potentially through a shared latent space or attention-based strategies that allow each view to inform and refine others. Such an approach would help enforce consistency across views, particularly in the representation of shared object features, improving the overall coherence of the 3D reconstruction.

Furthermore, consistency can be enhanced by employing techniques such as multi-view tiling and conditioning on a reference image. In multi-view tiling, the model would generate views in a manner that respects the spatial and contextual relationships between different perspectives, ensuring that information is propagated coherently across the entire scene. Conditioning on a reference image could guide the generation of the remaining views, aligning them with a known, high-quality representation that serves as a consistent anchor. This method would reduce view-to-view discrepancies by leveraging a strong prior to guide the generation process, thus promoting higher quality reconstructions.

Another area for improvement is the model's pose estimation capabilities. The current approach to pose estimation may be too rigid or simplistic, limiting the model's ability to adapt to diverse camera angles and dynamic viewpoints. By adopting a more dynamic and flexible pose estimation strategy, the model could better account for variations in camera positioning and orientation, which would enhance its ability to reconstruct scenes from a wider range of viewpoints. This would be particularly valuable in real-world scenarios where camera positions are often unpredictable and may vary significantly across different datasets or use cases.

Addressing the issue of noise accumulation in the cost volume is also crucial for improving the overall 3D reconstruction process. Refining the elevation angle estimation, which directly influences the depth and alignment of views, is essential for reducing misalignments that contribute to noise. In addition, leveraging more advanced techniques in view consistency, such as iterative refinement or multi-scale processing, could help mitigate the impact of noisy inputs by gradually refining the cost volume over multiple iterations. These techniques could effectively reduce the propagation of errors through the pipeline, leading to cleaner and more accurate 3D models.

Moreover, the high computational demands of the two-stage diffusion process, especially when dealing with high-resolution volumes, necessitate improvements in memory

management and computational efficiency. Exploring more robust and scalable solutions to handle memory usage, such as using more efficient data structures or optimizing the diffusion process to operate in a lower-dimensional latent space, could significantly reduce the memory footprint of the model. Such optimizations would enable the model to handle larger datasets and generate higher-quality 3D reconstructions while maintaining reasonable computational efficiency. Balancing memory efficiency with the quality of generated 3D shapes will be key to ensuring that the model is both scalable and performant in practical applications.

6. Conclusions

Our study demonstrates the robust capabilities of One-2345 in single-view reconstruction. Through the creation of a diverse dataset of 5,000 3D models, we have shown the model' s strong zero-shot performance as well as the significant improvements achieved by fine-tuning on domain-specific data. Our extensive experimental analysis reveals valuable insights into the effect of fine-tuning on reconstruction accuracy, highlighting the importance of domain adaptation for enhanced model performance. Overall, our findings contribute to a deeper understanding of how pre-trained models can be effectively adapted for single-view reconstruction tasks, paving the way for future advancements in this field.

7. Contribution Division

Table 2. Contribution Division

Name	Contribution Content	Percentage
Yikai Du	Data collection, Section 2 4	7.69%
Yuan Gao	Data collection, Section 2 4	7.69%
Yu Huang	Data collection, Finetuning, Section 5	7.69%
Jiayang Li	Data collection, Data processing	7.69%
Jiarui Li	Data collection, Index calculation, Section 5	7.69%
Ye Tao	Data collection, Visualization	7.69%
Yuqing Tu	Data collection, Data processing, Section 3	7.69%
Huayi Wang	Data collection, Section 2 4	7.69%
Shaojie Yin	Data collection, Data processing, Section 3	7.69%
Junqi You	Data collection, Finetuning, Section 5	7.69%
Zihao Zheng	Data collection, Data processing	7.69%
Yihang Zhou	Data collection, Section 1	7.69%
Zheli Zhou	Data collection, Data processing, Section 3	7.69%

References

[1] EricR Chan, Koki Nagano, MatthewA Chan, AlexanderW Bergman, JeongJoon Park, Axel Levy, Miika Aittala, ShaliniDe Mello, Tero Karras, and Gordon Wetzstein. Generative Novel View Synthesis with 3D-Aware Diffusion Models. *arXiv preprint arXiv:2304.02602*, 2023. 2

[2] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient Geometry-aware 3D Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16102–16112, 2021. 2

[3] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. WaveGrad: Estimating gradients for waveform generation. In *ICLR*, 2021. 2

[4] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A Universe of Annotated 3D Objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 2

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014. 1

[6] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *CVPR*, 2022. 2

[7] Hoang-An Le, Thomas Mensink, Partha Das, and Theo Gevers. Novel View Synthesis from Single Images via Point Cloud Transformation . *Proceedings of the British Machine Vision Conference*, 2020. 2

[8] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3D: High-resolution text-to-3D content creation. *arXiv preprint arXiv:2211.10440*, 2022. 2

[9] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization, 2023. 1, 4

[10] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot One Image to 3D Object. *arXiv preprint arXiv:2303.11328*, 2023. 2

[11] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object, 2023. 2

[12] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[13] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: Towards photorealistic image generation and editing with text-guided diffusion models. In *ICML*, 2022. 2

[14] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 2

[15] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Unsupervised Learning of Efficient Geometry-Aware Neural Articulated Representations. In *European*

Conference on Computer Vision, pages 597–614. Springer, 2022. 2

[16] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. StyleSDF: High-Resolution 3D-Consistent Image and Geometry Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513, 2022. 2

[17] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2

[18] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. In *ICLR*, 2023. 2

[19] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021. 2

[20] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022. 2

[21] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 2

[22] Junyoung Seo, Wooseok Jang, Min-Seop Kwak, Jaehoon Ko, Hyeonsu Kim, Junho Kim, Jin-Hwa Kim, Jiyoung Lee, and Seungryong Kim. Let 2d diffusion model know 3d-consistency for robust text-to-3d generation. *arXiv preprint arXiv:2303.07937*, 2023. 2

[23] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model, 2023. 4

[24] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 2

[25] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. *arXiv preprint arXiv:2212.00774*, 2022. 2

[26] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2D diffusion models for 3D generation. In *CVPR*, 2023. 2

[27] Daniel Watson, William Chan, Ricardo Martin-Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel View Synthesis with Diffusion Models. *arXiv preprint arXiv:2210.04628*, 2022. 2

[28] Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3d reconstruction. *arXiv preprint arXiv:2301.08247*, 2023. 2

[29] Zirui Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1

[30] Wenhai Xie, Haozhe Yao, Jianheng Mao, Songyou Zhang, Sainan Zhou, Weiming Yu, Hong Zhou, and Jingyi Yu. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[31] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Yi Wang, and Zhangyang Wang. Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360 views. *arXiv preprint arXiv:2211.16431*, 2022. 2

[32] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural Radiance Fields from One or Few Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4576–4585, 2021. 2