Stable-Dreamfusion

Yufan Liu, Peizhi Li

CIS Summer Program

Abstract. Single image to 3d reconstruction is quite an exciting field. We in this report, have explored and implemented a single view image to fast and high-resolution 3d reconstruction pipeline based off an open-source stable-dreamfusion baseline. We managed to achieve resolution reconstructions with satisfactory visual effects, fine detail and realistic texture. The code can be found here. githublinkinserthere

1 Introduction

The realm of 3D reconstruction has long been an area of interest for computer scientists and researchers, aiming to bridge the gap between the physical world and digital representations. The single image input 3D reconstruction problem is a challenging task in computer vision and computer graphics. It involves taking a 2D image as input and generating a 3D representation of the scene or object depicted in the image. The goal is to recover the 3D structure, shape, and appearance of the scene or object solely from the information present in the 2D image. This problem is inherently ill-posed because a single 2D image lacks the depth information necessary for an unambiguous 3D reconstruction.

Traditional approaches in 3D reconstruction often rely on explicit geometric models, which require intricate manual modeling and suffer from limitations in capturing fine-grained details and complex scenes. However, advancements in deep learning techniques have opened up new avenues for tackling these challenges. NeRF introduces a paradigm shift by treating the 3D reconstruction problem as a volumetric rendering task. By leveraging the expressive power of neural networks, NeRF can implicitly model the 3D scene's radiance (color and opacity) as a continuous function, enabling the synthesis of novel viewpoints and interactive exploration within the reconstructed space.

In this report, our focus is on tackling the challenging task of 3D Reconstruction from single image input by integrating well-established frameworks, namely Stable-DreamFusion, Zero-1-to-3, and ThreeStudio. By thoroughly studying the pipelines of each framework, our objective is to achieve high-resolution results from a carefully selected set of images, considering the hardware constraints. Our report's primary contributions lie in two aspects: firstly, showcasing a substantial enhancement in Image Preprocessing within the Stable-DreamFusion framework, and secondly, exploring a diverse range of Pipeline Parameters to elevate the overall quality of the 3D Reconstruction results. Given the time constraints of two weeks, we will also use the Discussion section to thoroughly discuss major drawbacks observed in the pipeline and provide constructive insights for future research directions. Our ultimate goal is to make meaningful advancements in the field of single-image 3D reconstruction and lay the groundwork for further improvements beyond the scope of this report.

2 Methods

2.1 Framework Overview

The framework is designed to produce high-quality object reconstructions using Neural Radiance Fields (NeRF) by incorporating an innovative guidance model, a kind of pre-trained model that's able to compute meaningful loss terms from multi-perspective sampling from the NeRF [1]. Meanwhile, the NeRF is trained on high quality labels i.e., the depth and normal estimations of the raw image.

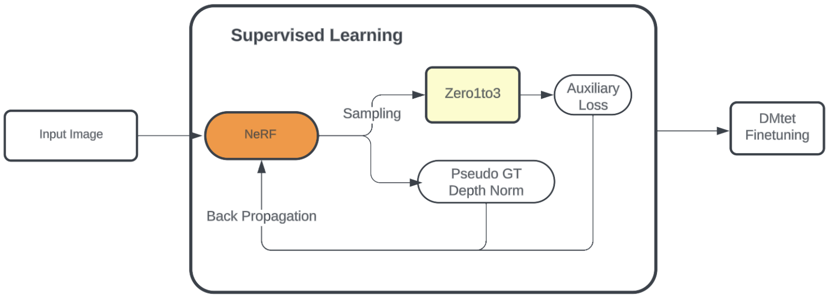

Fig. 1. The Framework of Zero1to3-Dreamfusion

2.2 Supervised Loss Explained

Inside of the supervision process, the loss is calculated based on the difference between the model's predictions and ground truth for RGB values, mask values, normal vectors (cosine similarity between predicted and ground truth normals), and depth(Pearson correlation between predicted and ground truth depths). In DMtet fintuning mode, additional mesh-related losses (normal and laplacian) would be added.

2.3 Guidance Model

Zero1to3 serves as a guidance mechanism during the training of NeRF. It essentially provides additional cues or hints to the NeRF model, ensuring that the rendered outputs adhere multiple novel views rather than only the front view. In essence, zero1to3 is a pre-trained model that can take samples from NeRF from different camera poses, calculate the loss between sampled views and corresponding prediction based on the reference view. Specifically, zero1to3 is capable of working in the latent space. It doesn't predict what the current view should look like directly. Instead, it predicts adjustments (in the form of noise) to the latent representations, which is later used to guide the NeRF's training. This guidance is conditioned on the difference between the current view and multiple reference views.
The Guidance Model

```
1. COMPUTE ANGULAR DIFFERENCE BETWEEN CURRENT AND REFERENCE VIEWS:
   - the adjusted parameters of the current view v1

   - the spherical coordinates of all reference views v2

   -  angles between the current view and each reference view

2. SETTING WEIGHTS BASED ON CLOSENESS IN ANGLE

3. GUIDANCE PREDICTIN CALCULATION
    - Apply the model on the noisy latent representations (x_in)
      with time step (t_in) and conditioning info (cond)
      noise_pred = model(x_in, t_in, cond)

    - Refine noise_pred using a combination of its unconditional
       and conditioned parts, scaled by guidance_scale
       noise_pred = noise_pred_uncond + guidance_scale * \
       (noise_pred_cond - noise_pred_uncond)

    - Weight the refined noise_pred with its angular closeness
      weight (zero123_w) and add to noise_preds list

    - Aggregate all predictions in noise_preds into a single
      guidance prediction using a weighted sum based on zero123_ws
```

3 Preliminaries

3.1 Setup

Hardware and Network Setup(special need) During the first week, we spent a lot of time on setting up the server and configuring the intricate network proxy settings, which was essential to have connections to online codebase/library resources, given that direct internet access was restricted due to governmental regulations. We ended up using Clash, a VPN software, and a personal SSR server.

Cuda Setup In addressing software dependencies, we faced intricate incompatibility issues among cuda, pytorch, and cudnn — essential components for the renderer to function properly. We also spent time on consulting official documentation for cuda and cudnn. We ended up using cuda 11.8, cudnn 8.0, pytorch 11.6.

3.2 Codebase Digest and Bottleneck Identification

Codebase Digest We spent time on digesting the codebase, since a solid understanding of the hierarchal code structure will reduce our debugging work and give us a better place to start on modification and optimization. We've identified the following core components, where we spent most of the time on understanding the mechanism and looking for possible flaws and problems.

```
└─ nerf/
│  └─ network/
│  │  └─ Contains the NeRF model and various backbones
│  └─ utils/
│  │  └─ Manages the training, loss computations, and optimization
│  └─ renderer/
└─ guidance/
   └─ Houses the ’zero123’ guidance model and its utilities
```

Bottleneck Identification

– **Inaccurate Supervision Labels**: The (estimated) ground truth depth and normal used for computing supervision loss, may contain inaccuracies. This led to results that lack detail and even caused deformations and structural losses.
– **Improper Hyperparameter Settings** The improperly set hyperparameters especially the learning rate, leading to unstable training, overshooting, poor convergence and poor results.
– **Model Capacity and Architectural Limitations** The current NeRF (I-NGP) model might be limited in its ability to generate complex and unusual novel views. We thought of switching to more powerful models but realized we didn't have the computation resources.

4 Our Work

4.1 Upgrade from Cravekit to Clipdop (SOTA)

We have made an important upgrade here that transitioned from the CarveKit library to the cutting-edge Clipdrop by Stability AI for background removal. This shift reduced the unwanted truncations of the target object and more importantly the residual background. At the beginning, we did the fix due to the evident issues of truncation and residual. But the fix turned out to be important to the final result since the guidance model sees the residues as part of the object, greatly jeopardizing the training of NeRF.

4.2 Improved The Supervision Labels(depth and norm estimations)

The core idea of this framework somewhat resembles weakly supervised learning, a paradigm where the training process is guided by partial and noisy labels, and the objective is to learn robustly despite label imperfections. However, there's a distinctive difference in our task, even though our scenario is highly constrained: one front-view image, which serves as our partial label. The thing is we care most about the quality of output. Given the identified bottleneck from previous section, we know the labels from preprocessing process are defective, the refinement and perfection of them become pivotal, as it directly influences the quality of our 3D reconstructions.

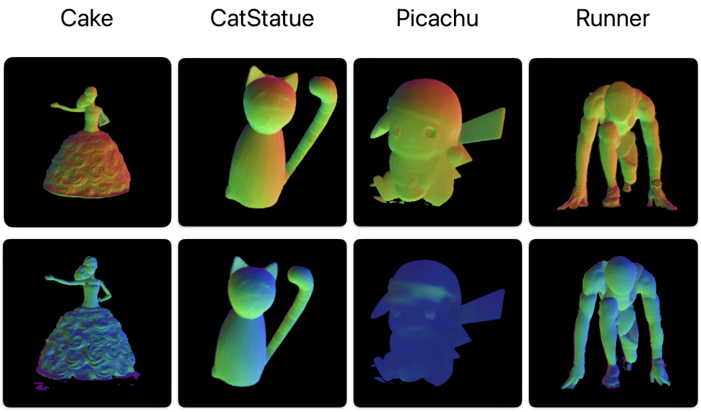

Fig. 2. Normal Estimation – After vs. Before

In this bottleneck, the crux hinges on two main operations: depth estimation and normal estimation. Both tasks used a Vision Transformer (ViT)-Resnet architecture, but with different pretrained weights. In the original approach [2, 3][4] , the preprocessing of input for the ViT-Resnet model and postprocessing of output were overlooked. Since ViT only accepts 384x384 inputs, the resizing led to a loss of image detail. Thus, we incorporated bilinear interpolation to retain image quality. Additionally, the lack of normalization in the original preprocessing hindered the model's performance.

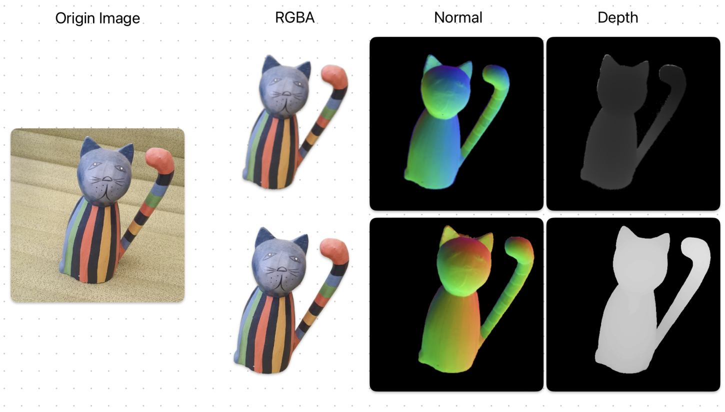

Fig. 3. Depth and Norm Estimations of CatSatue – After vs. Before. The difference is small here, but they lead to very different final outputs which we will see in section 5

When comparing our results to Stable DreamFusion's RGBA images (shown in Fig.3), noticeable differences become apparent, particularly in the loss of details, such as the cat-statue's eyeliners, and the presence of more aliasing and less smooth edges. These discrepancies can be attributed to the background removal process in Stable-DreamFusion, which utilize CarveKit and did not meet expectations due to constraints imposed by the use of params in the script. As a result, some critical details were lost during the preprocess, leading to a decrease in overall quality. In contrast, the Background removal function in ClipDrop demonstrates a superior level of performance and accuracy.

The Fix The preprocessing steps influence the model's performance. Different models have different specific preprocessing requirements to yield the best results. For ViT, which needs raw image resizing, we used interpolation to maintain detail when resizing to 384x384. For depth estimation, we normalized using transforms.Normalize(mean=0.5, std=0.5). In postprocessing, we clamped the output to the [0, 1] range for consistency, which will impact the subsequent training of the NeRF. In Fig.3, a notable difference can be observed between the Normal maps of Stable DreamFusion (SDF) and our results. SDF's Normal maps appear more blue-ish, while our Normal maps exhibit both red and blue channels without blue. Additionally, the object in SDF's output seems slightly smaller compared to ours. In contrast, our pipeline takes a different approach to address these challenges. We have identified and fixed the bugs in SDF. This decision ensures that we preserve important details and accurately represent the object's size ratio, resulting in more accurate and visually consistent Normal maps. By resolving these issues, our image preprocessing pipeline achieves more accurate depth and normal estimations, producing higher-quality outputs for 3D reconstruction tasks. The avoidance of unnecessary resizing and color-remapping during estimation ensures that we retain crucial image details and achieve better visual integrity in our results. In Fig.3, it is evident that the Depth maps generated by Stable DreamFusion (SDF) are remapped to a dimmer version, likely done to achieve a better visual representation. However, this approach inadvertently leads to an inaccurate estimation of depth, making objects appear shallower than they should be. Additionally, the observed ratio of the object's size is not accurately reflected in the remapped Depth maps. These discrepancies stem from the depth estimation process in SDF.

4.3 Hyperparamters Finetuning: The Balance between OOM and Quality

Fine-tuning hyperparameters has significantly increased the output quality. Here's a breakdown of the parameters we adjusted:

- Training Iterations (–iters):default: 10,000. After multiple trials, we discovered that in the original codebase, 20,000 iterations could improve the results. But after some of our modifications, the model seemed to reach a plateau after 10,000 iterations, meaning more training didn't make the model learn more and we supposed that we've maximized the NeRF(I-NGP)'s capabilities under the current configuration.
- Learning Rate Scheduler: The initial learning rate scheduler used a constant value of 0.01, leading to oscillations and overshooting during the training. We designed a decaying scheduler that reduces the learning rate over iterations, which stabilized the training process, resulting in a reduced loss.
- Floating Point Precision (–fp16): By default, the torch uses 32-bit floating point precision. Using 16-bit precision can speed up model training and reduce memory usage. However, it's essential to ensure that there's no significant loss in model accuracy or convergence.
- Different Rendering Resolution for NeRF and DMtet (–w and –h): These two parameters set the resolution where NeRF is trained. Higher values will increase the quality of the renderings but will also greatly increase the memory usage. We made the values to be 80 for NeRF training and 256 for DMtet finetuning.

5 Showcase

In the following visual cases, the distinction between the initial and the refined versions of our pipeline are quite evident. The top row presents the intermediate or original version, to give you a glimpse into our starting point. And the bottom

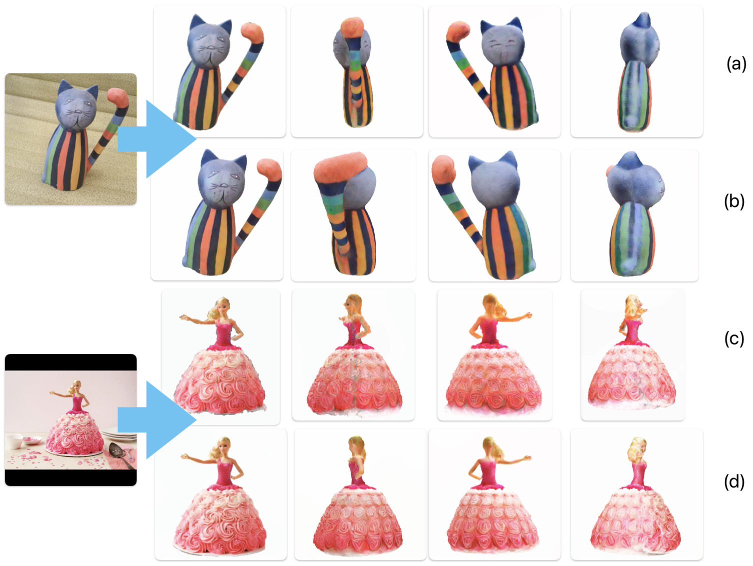

Fig. 4. After (second row) vs. Before (first row)

row showcases the final version, to prove the advancements and enhancements achieved through our refinements. These comparisons highlight on the improvements our modifications have brought about.

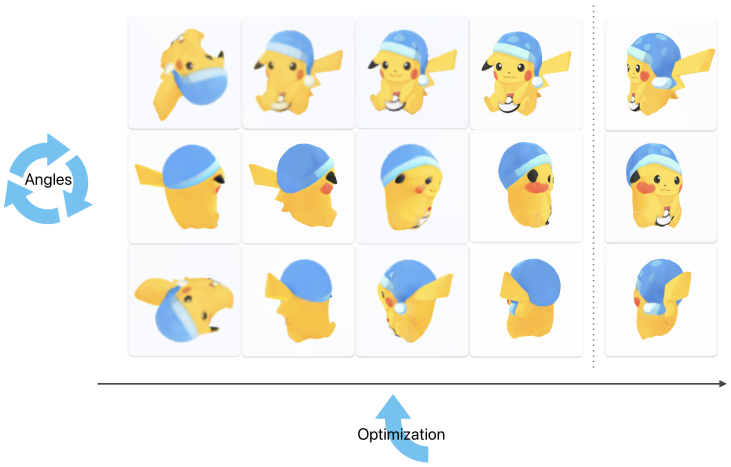

Fig. 5. The X axis is the refinement we've done along the way.

Recalling the revised normal and depth estimations provided from (shown in Fig.2), here in (shown in Fig.4), the final results show the distinctive difference. With our refinements, the cat has a rounder body, the aliasing is reduced, and the stripes are more defined, the fur texture appears more authentic. Overall, the result is now more visually appealing.

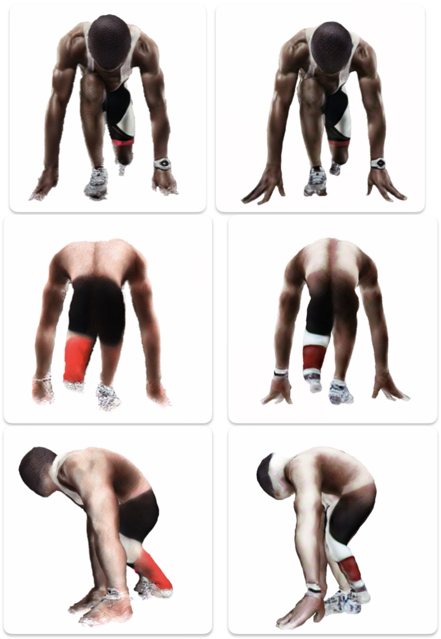

Fig. 6. The second column is the final version, while the first column is an intermediate version

6 Discussion

6.1 2D Hand-drawn Image Limitations in the Task

Single Image to 3D Reconstruction poses significant challenges, especially when dealing with hand-drawn cartoon images, such as the iconic Pikachu. Although hand-drawn images of 3D models present an artistic representation, the results from trained models often exhibit flattened features and lack vital information, particularly in the back of the body. This deficiency arises because hand-drawn images simulate 3D effects through shading, causing depth and normal maps to lose essential depth information and appear flattened compared to true 3D models. This limitation highlights the inherent challenge in converting 2D hand-drawn images to accurate 3D representations.

To address this issue, we conducted research on the sequencing of background removal and discovered that training the model in a modified order results in thicker representations. As depicted in Fig.5, this approach significantly improves the overall form of Pikachu. However, as shown in the last column of Fig.5, we observed that the facial region may still appear abruptly flattened due to the Front Image saturating during training, aligning entirely with the input's flat information. Consequently, achieving accurate facial depth in hand-drawn images remains a potential direction for future exploration. This study sheds light on the challenges and potential solutions in Single Image to 3D Reconstruction, offering valuable insights for improving the accuracy and fidelity of 3D representations from hand-drawn images. By further investigating the impact of background removal sequences and refining the training process, we aim to enhance the quality and depth perception of 3D reconstructions from hand-drawn inputs, bridging the gap between 2D artistic expression and accurate 3D representations.

7 Conclusion

In summary, over the past two weeks, our understanding of Nerf and Single Image 3D Reconstruction has significantly deepened. We began by setting up the environment and successfully running the basic pipeline. Through critical observation and analysis of the initial results, we gradually delved into understanding each step of the process, identifying areas for improvement. We focused on enhancing the preprocessing stage, fine-tuning the models, and exploring ways to increase the resolution from different perspectives. This iterative approach allowed us to address various challenges and optimize the overall performance of the pipeline.

Our journey involved a comprehensive exploration of the intricacies involved in Nerf and Single Image 3D Reconstruction, enhancing our expertise in the field. By critically analyzing results and making iterative improvements, we have strengthened our grasp of the underlying concepts and techniques. The progress made during these two weeks serves as a solid foundation for future research and development in the exciting and rapidly evolving field of 3D reconstruction from single images. Through continuous learning and exploration, we aim to further advance the capabilities of these techniques, ultimately contributing to cutting-edge advancements in computer vision, machine learning, and computer graphics.

Author Biographies

Yufan Liu(lanceliu@berkeley.edu) student at Cal EECS. Research Interest: Digital Twin in Supply Chain, Robotics, Computer Vision.

Peizhi Li(paige.li@richmond.edu)
Research Interest : Computer Graphics, Interactive Entertainment, XR

References

1. Liu, R., Wu, R., Hoorick, B.V., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object (2023)
2. Eftekhar, A., Sax, A., Malik, J., Zamir, A.: Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. (2021) 10786–10796
3. Kar, O.F., Yeo, T., Atanov, A., Zamir, A.: 3d common corruptions and data augmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2022) 18963–18974
4. Guo, Y.C., Liu, Y.T., Shao, R., Laforte, C., Voleti, V., Luo, G., Chen, C.H., Zou, Z.X., Wang, C., Cao, Y.P., Zhang, S.H.: threestudio: A unified framework for 3d content generation. https://github.com/threestudio-project/threestudio (2023)