Invisible Stitch: Generating Smooth 3D Scenes with Depth Inpainting (Supplementary Materials)

Paul Engstler, Andrea Vedaldi, Iro Laina, and Christian Rupprecht

Visual Geometry Group, University of Oxford
{paule,vedaldi,iro,chrisr}@robots.ox.ac.uk
https://research.paulengstler.com/invisible-stitch

1 Overview

This supplementary document is organized as follows: First, we provide more detailed implementation details with our specific choices of hyperparameters for the fine-tuning process of our model and the scene generation pipeline (Sec. 2). Second, we provide further qualitative results produced by our scene generation pipeline, including a new set of scenes for different real-world images (Sec. 3). Third, we show that our depth completion model can be plugged into an existing scene generation method to improve the structural quality of the scenes it generates (Sec. 4). Finally, we discuss the limitations of our as well as related approaches in 3D scene generation (Sec. 5). We hope that this discussion will inspire future research to further this field.

We invite the reader to consider the accompanying videos that show renderings of 3D scenes generated with our method. Please note that the videos have a higher resolution (1080p) than the frames used to generated the scenes (720p).

We will publish the code to train our depth completion model, the trained checkpoint, as well as our pipeline to generate 3D scenes.

2 Further Implementation Details

2.1 Fine-Tuning

We base our model on ZoeDepth, which uses a dense prediction transformer (DPT) [8] with a BeiT (Bidirectional Encoder representation from Image Transformers) [1] backbone at a resolution of 512×384. We fine-tune the model for 5 epochs with batch size 8, using a low learning rate of 0.00025 with a weight decay of 0.01. We train on four NVIDIA Tesla P40 GPUs.

2.2 Scene Generation

For the first two phases of our scene generation pipeline described in Section 5.2, we use PyTorch3D [9] to render our point cloud representation. We use alpha-composite rendering with 16 points per camera ray at an image size of 720×480. The point cloud generated by the first two phases provides the starting point

for the Gaussian splat optimization [5], alongside images of the generated and supporting views. We use the default set of parameters for the optimization but limit the number of iterations to 2,990.

2.3 Ablations

In Section 5.4, we present ablations of the design choices in our training pipeline. To demonstrate the benefit of providing a sparse depth input to our model, we compare it to a scenario where the model is only given an image as input, reverting to the original depth estimation task. While the sparse depth inherently provides information about the overall depth of the scene, this information is missing if only an image is given as input. To ensure a fair comparison, we thus run a global scale-and-shift optimization on the predicted depth to align it with the existing scene depth. This optimization is run for at most 100 steps, stopping early if no improvement is made for 10 steps. It is based on the absolute error between both depth maps (where available) and uses the Adam optimizer with a learning rate of 0.01.

3 Additional Qualitative Results

We present an additional set of generated scenes from real-world images in Figure S1. As in Figure 4, we provide individual images of the hallucinated views, a rendering of the entire generated 3D scene, as well as a cut-away that provides more detail.

In Figure S2, we provide further examples where we generate a scene multiple times from the same input image, using different seeds for Stable Diffusion.

4 Improving an Existing Method with Depth Completion

We demonstrate that our depth completion model is able to improve the geometric consistency of existing scene generation methods by plugging it into one such method, namely LucidDreamer [3]. We use the same input image, prompt, and fixed seed, while swapping the depth prediction backbone. The original approach uses the depth estimation network ZoeDepth, which does not utilize a sparse depth input. We consider the scenes of Figure 4, showing the results for *Prague* (Figure S3), *Kyoto* (Figure S4), and *North Carolina* (Figure S5). Having plugged our model into their method, we see that its generated scenes have an overall better structural quality than those produced by the original method.

5 Discussion & Limitations

In this section, we split our analysis of limitations into two distinct parts. First, we consider the limitations of our depth inpainting approach, specifically focusing on the trained network and the task of depth inpainting itself. Second, we provide

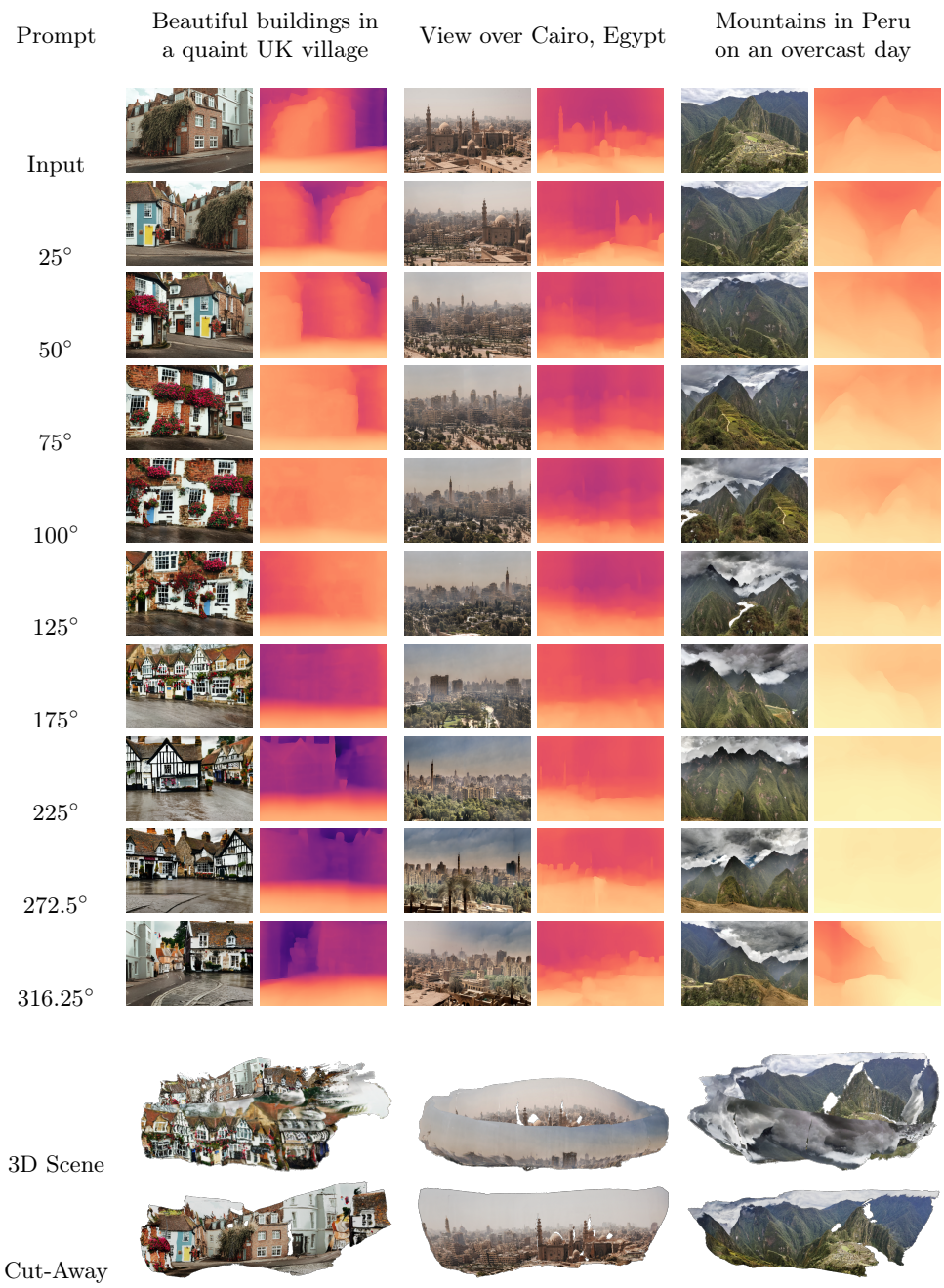

Fig. S1: Qualitative results of our method on additional real-world images. It is able to generate convincing, immersive scenes for a wide range of input images and associated prompts.

Fig. S2: Diversity of generated scenes. All scenes have been generated by our method, starting from the same input image and text prompt (consider the Kyoto scene in Figure 4). For visualization purposes, only two additional frames have been hallucinated.

(a) LucidDreamer (ZoeDepth [2])

(b) LucidDreamer (with our depth completion model)

Fig. S3: Qualitative comparison of depth prediction backbones for LucidDreamer (Prague). Consider the bridge discontinuities and ghosting artifacts, which disappear with our network.

(a) LucidDreamer (ZoeDepth [2])

(b) LucidDreamer (with our depth completion model)

Fig. S4: Qualitative comparison of depth prediction backbones for LucidDreamer (Kyoto). Our model leads to significantly less torn structures and provides an overall geometrically more sound scene. Please note that we manually removed visual obstructions from both scenes.

(a) LucidDreamer (ZoeDepth [2])

(b) LucidDreamer (with our depth completion model)

Fig. S5: Qualitative comparison of depth prediction backbones for LucidDreamer (North Carolina). With our depth completion network, the scene overall appears less distorted, in particular with regard to the foliage. Please note that we manually removed visual obstructions from both scenes.

a broader perspective on the limitations of the sequential scene generation task, which we present as an application of our depth completion model in line with other related works [3, 4, 6, 7, 10, 11].

Depth Inpainting The performance of our depth inpainting model may be limited by a shift in the data distribution between training and inference. The data the model is trained on (such indoor scenes from NYU Depth or the datasets used to pre-train DPT [8], which forms the core of ZoeDepth [2]), differ from the data it is typically applied to during 3D scene generation (*i.e.*, synthetic images generated by Stable Diffusion). The differences may be due to the type of imagery (*e.g.*, landscapes such as the *Mountains in Peru* in Figure S1), but also due to imperfections or artifacts caused by image generators that the depth network has never encountered during training.

Another challenge lies in the inherently limited resolution of all depth estimation networks, which constrains the model's ability to accurately predict the depth of fine structures, especially around object boundaries. As a consequence, these fine details might become detached from their corresponding objects during projection, which in turn affects the image inpainting step. An example is shown in Figure S6.

Scene Generation During scene generation, and similarly to prior work, we use supporting views to add further constraints to the Gaussian splat optimization process. How to optimally place the cameras for these views (to prevent "crashing" into the existing point cloud, see Figure S7) remains an open question. More sophisticated camera pose generation algorithms, a strictly enforced minimum depth for points, or view-dependent translucency of parts of the point cloud might yield a more stable procedure to produce supporting views.

Input image

Projection based on depth prediction, after depth snapping

Fig. S6: Loss of fine object details after projection. Due to the limited resolution of depth predictions, fine details might be detached from their corresponding objects and become part of the background. We present an example of this for the given image, showing the resulting projection after applying our depth snapping to remove floating points (as outlined in Section 5.2). The fine hairs of the dog at its boundary have become part of the background.

Fig. S7: Spurious points due to improperly placed supporting view cameras. In this example, a supporting view camera was placed inside the point cloud, with some points obstructing its view. These points are rendered into an image that is used to guide the Gaussian splat optimization process, which cause them to be baked into the final Gaussian representation. These artifacts solely stem from our simplified method to place supporting view camera, as the offending generated frame had unexpectedly low depth values.

References

1. Bao, H., Dong, L., Piao, S., Wei, F.: Beit: Bert pre-training of image transformers. arXiv preprint arXiv:2106.08254 (2021) 1
2. Bhat, S.F., Birkl, R., Wofk, D., Wonka, P., Müller, M.: Zoedepth: Zero-shot transfer by combining relative and metric depth. arXiv preprint arXiv:2302.12288 (2023) 5, 6, 7, 8
3. Chung, J., Lee, S., Nam, H., Lee, J., Lee, K.M.: Luciddreamer: Domain-free generation of 3d gaussian splatting scenes **abs/2311.13384** (2023) 2, 8
4. Höllein, L., Cao, A., Owens, A., Johnson, J., Nießner, M.: Text2Room: Extracting textured 3D meshes from 2D text-to-image models. In: ICCV (2023) 8
5. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023) 2
6. Liu, A., Tucker, R., Jampani, V., Makadia, A., Snavely, N., Kanazawa, A.: Infinite nature: Perpetual view generation of natural scenes from a single image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14458–14467 (2021) 8
7. Ouyang, H., Heal, K., Lombardi, S., Sun, T.: Text2immersion: Generative immersive scene with 3d gaussians **abs/2312.09242** (2023) 8
8. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 12179–12188 (2021) 1, 8
9. Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.Y., Johnson, J., Gkioxari, G.: Accelerating 3d deep learning with pytorch3d. arXiv preprint arXiv:2007.08501 (2020) 1
10. Yu, H., Duan, H., Hur, J., Sargent, K., Rubinstein, M., Freeman, W.T., Cole, F., Sun, D., Snavely, N., Wu, J., Herrmann, C.: Wonderjourney: Going from anywhere to everywhere **abs/2312.03884** (2023) 8
11. Zhang, J., Li, X., Wan, Z., Wang, C., Liao, J.: Text2nerf: Text-driven 3d scene generation with neural radiance fields **abs/2305.11588** (2023) 8