Supplementary Material
BluNF: Blueprint Neural Field

Robin Courant[1]* Xi Wang[1]* Marc Christie[2] Vicky Kalogeiton[1]
[1]LIX, Ecole Polytechnique, IP Paris [2]Inria, IRISA, CNRS, Univ. Rennes

In this supplementary material, we first present the implementation details of BluNF (Section A). Next, we cover more details of datasets used in our work (Section B), and of BluNF pipeline (Section C). Finally, we discuss the broader impact of our proposed method (Section D).

In addition, a video '10_demo.mp4' accompanies this submission. In this material, we summarize our method, explain our results and illustrate the different applications.

A. Implementation details

Our implementation is based on Nerfstudio [8], on which we build our original BluNF pipeline. For the input encoder module of BluNF, we use 2D NeRF-like positional encoding [4], while the semantic field consists of two fully-connected layers. We use Adam optimizer [3] with a learning rate of 3.0×10^{-2}. The optimization for a single blueprint takes around 10k steps, i.e. approximately 10 minutes with a single NVIDIA Quadro RTX 6000.

B. Datasets

We apply our method on two different datasets: *Replica* [7] and *MatterPort3D* [1].

B.1. Dataset description

Replica comes with ceiling-related semantic classes; to avoid a complete occlusion of the blueprint, we mask these classes on semantic maps: i.e. we do not sample rays on these classes. Note that, *MatterPort3D* scenes do not have ceiling-related semantic classes.

***R1* (*Replica - room 0*).** We show in Figure 1.a that this scene corresponds to a classical living room layout, with, for instance, sofas, armchairs and a coffee table. This scene contains 17 **semantic** (set of entities) classes.

***R2* (*Replica - room 2*).** We show in Figure 1.b that this scene corresponds to a classical dining room layout, with, among others, chairs and a table. This scene contains 12 **semantic** (set of entities) classes.

***M1* (*MatterPort3d - gZ6f7yhEvPG*).** We show in Figure 1.c that this scene corresponds to a historical chapel layout, with, for example, stone benches and a reading desk. This scene contains 23 **instance** (single entity) classes.

***M2* (*MatterPort3D - pLe4wQe7qrG*).** We show in Figure 1.d that this scene corresponds to a historical chapel layout as well. This scene contains 27 **instance** (single entity) classes.

B.2. Extraction of ground-truth blueprint

Extracting the ground-truth blueprint is not a straightforward operation, and requires manual actions. In the Habitat-Sim 3D simulator [6], we load the 3D model of the scene. We look for the gravity direction, and we set the camera in this direction in the centre of the scene. For *Replica*, we need to adjust the height of the camera to avoid ceiling occlusion. Once we make sure that the camera is orthogonal to the ground and change the scaling factor to project all the pixels into a fixed resolution is correct, we take a shot. Next, we align this shot with a grid of coordinates to make sure that predictions and ground truth match at the pixel level.

B.3. Influence of dataset quality

In Section 4.1 of the main manuscript, we claim that NeRF-estimated depth from *MatterPort3D* is poorer than NeRF-estimated depth from *Replica*, because of the intrinsic quality of these datasets.

Quantitatively, we observe this phenomenon with the $l1$-error of depth maps over both datasets. As shown in Table 1, the mean and standard deviation over all depth maps of *R1* are respectively 0.063 and 0.091. In contrast, *M1* has $l1$-error mean of 0.377 and a standard deviation of 0.217. These numbers clearly show the difference in depth estimation quality between the two datasets.

Moreover, this error gap is visually verified in Figure 2. With these two examples, we observe that in *M1* scene: (i)

[*] Equal contribution.

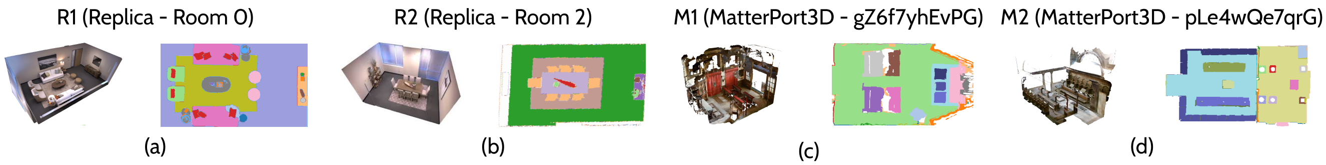

R1 (Replica - Room 0)
R2 (Replica - Room 2)
M1 (MatterPort3D - gZ6f7yhEvPG)
M2 (MatterPort3D - pLe4wQe7qrG)
(a)
(b)
(c)
(d)

Figure 1: **Overview of the rooms from Replica and MatterPort3D datasets.** For each room, we show a 3D view and the ground-truth blueprint.

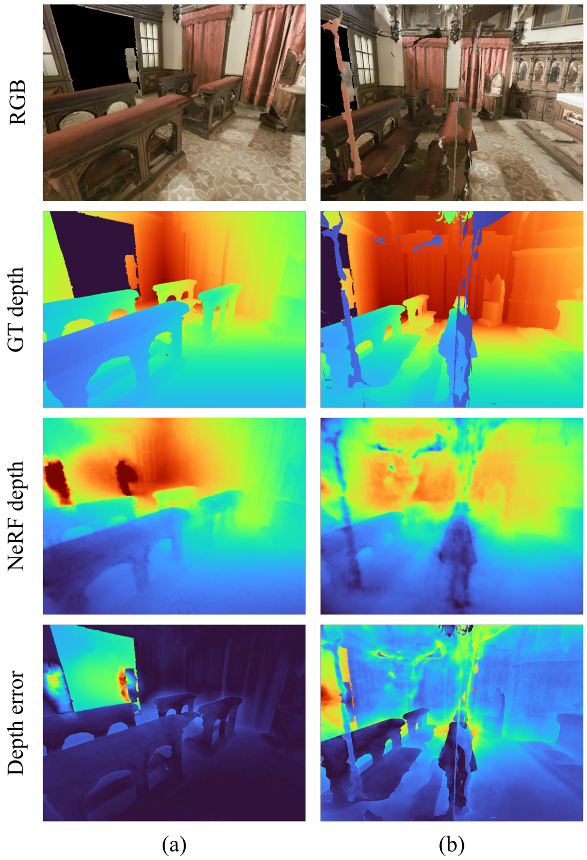
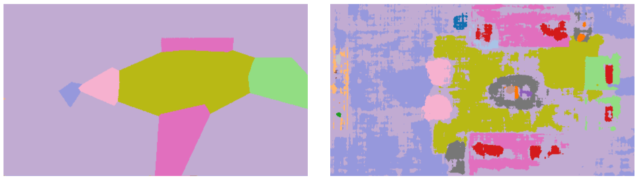


Figure 2: **Comparison between ground-truth depth (*row 2*) and NeRF-estimated depth (*row 3*) on the *M1* dataset**. Each column corresponds to a different frame within the dataset, and the last row shows the depth $l1$-error (dark for low error, light for high error).

Depth $l1$-error	**Mean**	**Std**
Replica - room 0	0.063	0.091
MatterPort3D - gZ6f7yhEvPG	0.377	0.217

Table 1: Mean and standard deviation (std) of depth $l1$-error over *Replica - room 0* and *MatterPort3D - gZ6f7yhEvPG*.



(a) w/o PE

(b) w/ PE

Figure 3: **Visual comparison of different input encodings under noisy conditions.** (a) Result with positional encoding (PE), and (b) result without positional encoding (w/o PE).

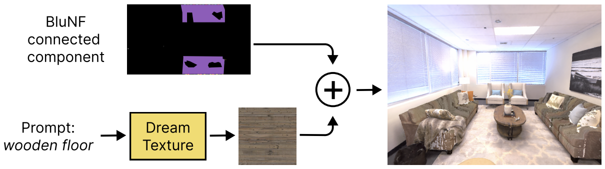


Figure 4:

black areas (undefined pixels) are sources of depth error (see Figure 2.a left part), and (ii) artefacts due to the real-world nature of this dataset are also sources of depth error (see Figure 2.b centre part).

C. BluNF

Positional encoding. In this paragraph, we demonstrate the efficacy of Positional Encoding (PE) in disentangling geometric ambiguities that arise when projecting pixel coordinates onto blueprint coordinates (refer to Section 3.1). To evaluate this, we conduct a toy experiment by replacing 40% of the BluNF training semantic views with uniform semantic views (only one semantic class).

Figure 3 showcases the results obtained with and without PE. We observe in Figure 3b, that PE achieves significantly superior results compared to Figure 3a. Indeed, without PE, BluNF tends to merge all shapes together. This observation highlights the effectiveness of PE in resolving geometric ambiguities and improving the overall disentanglement process.

Editing. In this paragraph, we present further details on the prompting editing method introduced in Section 4.3 of the main manuscript. Figure 4 shows the integration of BluNF with a text-to-image model. Users can edit the 3D scene effortlessly by writing a prompt and selecting a connected component on the blueprint by clicking. In this particular example, we utilize the DreamTextures [2] model, a DreamBooth [5] model fine-tuned to diffuse textures from text prompts.

D. Broader impact

Practical impact. There are various potential applications for BluNF. In this paragraph, we highlight two main application domains: (i) architecture, and (ii) cinematography.

First, our method could be useful for architectural purposes and especially for interior designers. For instance, we could imagine an interior designer using BluNF to show different layouts. The designer could directly manipulate the blueprint and show the results through the synthesized views of the NeRF.

Second, in cinematography, BluNF could be used as a strong post-processing tool. For example, a filmmaker could use it to remove a prop that disturbs the framing or to adjust the colour of the layout to better fit with the artistic direction.

Environmental impact. All experiments are done on a single NVIDIA Quadro RTX 6000 GPU, which requires 260W in power supply. Training a BluNF model on a scene requires around 10 GPU minutes. For this project, we use approximately 100 GPU hours, which amounts to 26kWh and 936g of CO2 emitted.

References

[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *3DV*, 2017. 1

[2] Carson Katri. Dreamtextures. https://github.com/carson-katri/dream-textures, 2023. 3

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 1

[4] 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. *ECCV*, 2021. 1

[5] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023. 3

[6] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *ICCV*, 2019. 1

[7] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 1

[8] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *SIGGRAPH*, 2023. 1