

# Off The Grid: Detection of Primitives for Feed-Forward 3D Gaussian Splatting

## Supplementary Material

### 7. Supplementary video

We present novel view synthesis comparison with AnySplat [18] in the [supplementary video](#). Trajectories include both interpolated and extrapolated views. We show models that use 6 views and generate 30 views interpolated between each. In the middle of the generated video, we extrapolate views with a spiral trajectory. Please note that despite using the same code to generate trajectories for both methods, views are not exactly aligned due to the difference in camera poses. Our method exhibits more accurate geometry and sharper rendering in most scenes.

### 8. Implementation details on our method

For the 3D reconstruction backbone, we use VGGT-1B [56], starting from official checkpoints released by the authors. We remove the pointmap head that we don't use. We tried to use it instead of depth map but observed significantly inferior accuracy, especially for large number of images. Regarding the decoder, the UNet module uses the implementation of Pytorch-UNet [50] with 13 input channels (3 for RGB, 2 for depth and depth confidence, and 8 for unpatchified latent features and 32 output channels. The  $h_{det}$  and  $h_{desc}$  heads process features with input image resolution through 3 convolutional layers with ReLU intermediate activations and 32 channels. Detection features are transformed into heatmaps through a softmax with temperature 0.2. Importantly, we remind that softmax is not applied over channels dimension but over spatial dimension at a patch level. Bilinear interpolation is done with Pytorch `grid.sample` function with `padding_mode` set to border.

### 9. Depth estimation

We present a qualitative evaluation of depth, shown in Fig. 9. The first two rows show examples from Charge, whereas next rows are from DL3DV where ground truth depth is not available.

First, on Charge, we observe accurate and highly similar depth maps between pre-trained VGGT and our fine-tuned version on this synthetic dataset. The main difference is observed on background areas (see second row, where our method is better aligned with GT for foreground areas but background is predicted too close, degrading metrics). On DL3DV, we observe more insights and failure cases from the pre-trained VGGT model. First, this model is quite sensitive to specularities and create holes on flat but highly reflective surfaces (see rows 3, 5 and 6). Then, during the supervised training of this model, sky pixels were masked out, resulting in close depth estimation for these pixels (row 4), which is not compatible with our rendering task. We also sometimes observe inaccuracy on some flat surfaces (e.g. the ceiling in row 7) without clear reasons. All these failures create geometrically inaccurate models with floating artefacts when we start to train our method. We observe that self-supervised fine-tuning through rendering is able to address these issues and obtain more accurate depth maps without holes, for both our method and AnySplat [18],

to a lesser degree. We observe that AnySplat depth maps are less accurate than ours, one recurrent artefact being edges appearing where depth is continuous (see row 4). DepthSplat [65] also claims to learn a depth estimation module from rendering but its accuracy is not comparable with VGGT-based models.

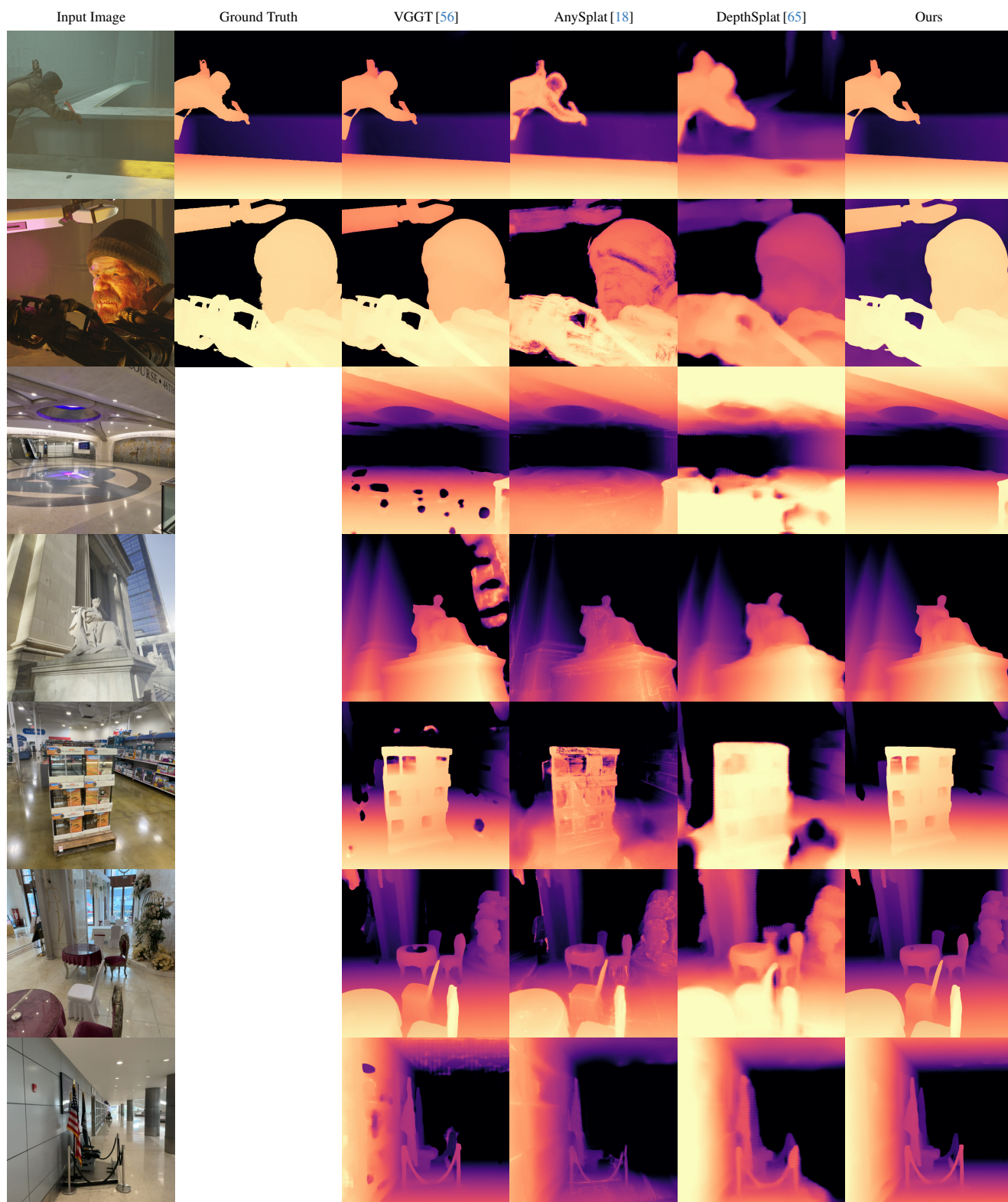


Figure 9. Qualitative results of our method compared with several SoTA on depth estimation.