CaesarNeRF: Calibrated Semantic Representation for Few-Shot Generalizable Neural Rendering

Supplementary Material

In this supplementary material, we provide more details in addition to the main manuscript, including specifics of datasets, along with additional analysis and discussions with visualizations, where we also discuss the GIF examples attached in the supplementary material.

6. Dataset Details

In this section, we discuss the further details of the datasets used in our experiments. Our generalizable experimental settings consist of two different configurations, the details of which are presented as follows:

(a) LLFF, Shiny, and mip-NeRF 360. In this configuration, we conduct experiments on the LLFF [39], Shiny [63], and mip-NeRF 360 [4] datasets with the GNT training settings as specified in [56]. We adopt the GPNR settings [54] to sample every eighth frame in each category for testing. Specifically, for LLFF [39], we evaluate on eight categories: trex, fern, flower, leaves, room, fortress, horns, and orchids. For Shiny [63], we test on eight categories: CD, crest, food, giants, lab, pasta, seasoning, and tools. For mip-NeRF 360 [4], we test on seven categories that are available without external restrictions: bicycle, bonsai, counter, garden, kitchen, room, and stump.

(b) MVImgNet. The MVImgNet dataset [72] comprises 6.5 million frames from 219,188 videos across 238 classes, making it infeasible to train on all sequences or categories. We adhere to the official split and focus on the *container* category, which includes category ID 0, 1, 14, 26, 28, 37, 39, 43, 48, 49, 83, 119, 145, and 160. We randomly subsample 2,500 training examples from the training set and select 108 sequences for the test set. Testing is conducted on the first example of each sequence, using the remaining samples as reference views. We attach the list of scenes used for training and inference to the supplementary material.

7. Additional Analysis

In this section, we present further analysis of CaesarNeRF, as well as its comparison with other methods, including diffusion-based and NeRF-based methods, along with more experimental details.

Comparison with generative methods. Single-view scenarios are frequently encountered in existing generative methods [33, 34, 36] based on diffusion models [18, 50]. Although these models yield reasonable results for object-centric rendering (focusing on one object at the center of the image), they fall short in rendering scene-level examples from novel viewpoints. Different from NeRF-based approaches that reconstruct images based on observed pixels, generative models aim to synthesize an entire image using its semantic representation, often without preserving the style or intricate details of the original image.

(a) leaves (b) orchid

Figure 6. Synthetic results for two examples from LLFF [39], "leaves" and "orchid", using zero 1-to-3 [33] with one reference image as input and 2 degrees of vertical shift.

Method	PSNR (↑)	LPIPS (↓)	SSIM (↑)
PixelNeRF [71]	18.66	0.463	0.588
MVSNeRF [6]	21.18	0.301	0.691
IBRNet [60]	25.17	0.200	0.813
NeuRay [35]	25.35	0.198	0.818
GeoNeRF [23]	25.44	0.180	0.839
ContraNeRF [68]	25.44	0.178	0.842
GPNR [54]	25.72	0.175	0.880
GNT [56]	25.65	0.140	0.853
Ours	25.17	0.137	0.852

Table 8. Generalizable results on LLFF [39] compared with state-of-the-art methods using all reference views as input.

We present two examples from the LLFF [39] dataset in Figure 6 using zero 1-to-3 [33], shifting the input by 2 degrees vertically with its polar angle. When using zero 1-to-3 to render a scene-level image from a novel camera pose, the style of the prediction deviates significantly from the original input, exhibiting smoothing artifacts characteristic of the diffusion model. These diffusion-based methods offer good semantic completeness and less ambiguous boundaries, but at the expense of sacrificing the original style of the input images, which is not optimal for rendering.

Metric	Method	trex	fern	flower	leaves	room	fortress	horns	orchids
PSNR (↑)	LLFF [39]	27.48	28.72	20.72	21.13	24.54	21.79	23.22	18.52
	NeRF [41]	26.80	25.17	27.40	20.92	32.70	31.16	27.45	20.36
	NeX [63]	28.73	25.63	28.90	21.96	32.32	31.67	28.46	20.42
	GNT [56]	28.15	24.31	27.32	22.57	32.96	32.28	29.62	20.67
	Ours	28.30	25.63	28.29	23.11	32.21	32.47	29.56	21.52
LPIPS (↓)	LLFF [39]	0.222	0.247	0.174	0.216	0.155	0.173	0.193	0.313
	NeRF [41]	0.249	0.280	0.219	0.316	0.178	0.171	0.263	0.321
	NeX [63]	0.193	0.205	0.150	0.173	0.161	0.131	0.173	0.242
	GNT [56]	0.080	0.116	0.092	0.109	0.060	0.061	0.076	0.153
	Ours	0.076	0.111	0.068	0.095	0.057	0.055	0.071	0.139
SSIM (↑)	LLFF [39]	0.857	0.753	0.844	0.697	0.932	0.872	0.840	0.588
	NeRF [41]	0.880	0.792	0.827	0.690	0.948	0.881	0.828	0.641
	NeX [63]	0.953	0.887	0.933	0.832	0.975	0.952	0.937	0.765
	GNT [56]	0.936	0.846	0.893	0.852	0.963	0.934	0.935	0.752
	Ours	0.943	0.871	0.912	0.872	0.967	0.946	0.939	0.782

Table 9. Full table of evaluation for per-scene optimization for all eight categories on the LLFF [39] dataset comparing CaesarNeRF with state-of-the-art methods.

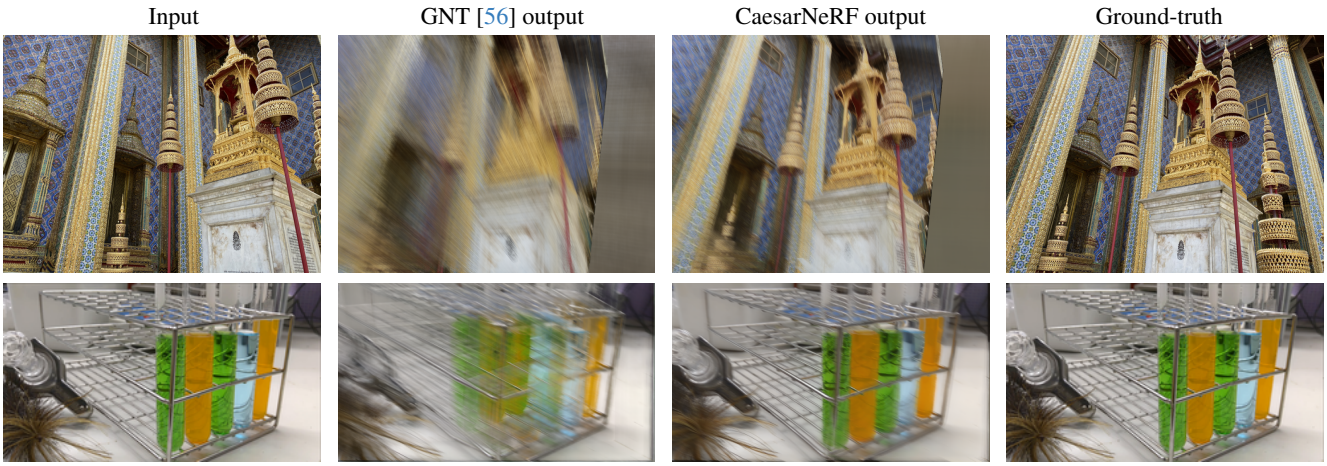


Figure 7. Visualization comparison of CaesarNeRF and GNT with the input frame using one reference view as input.

Generalizable Performance. Beyond few-view scenarios, we further analyze cases with more views and present these results in Table 8. When more views are available, our model demonstrates similar performance compared with other state-of-the-art methods, particularly GNT [56], our baseline method. This indicates that CaesarNeRF can match the performance of other state-of-the-art methods when all views are available and exhibits better performance when the number of input reference views is limited.

Per-scene Optimization on LLFF. We present detailed results for per-scene optimization in Table 9. We compare CaesarNeRF against the top-performing model from Table 5, GNT [56], as well as other methods, LLFF [39], NeX [63], and NeRF [41]. CaesarNeRF surpasses all other methods across all categories for LPIPS. It also shows the best performance in half of the LLFF dataset categories in terms of PSNR and SSIM. CaesarNeRF outperforms GNT [56], our baseline method, in all categories on three metrics with consistent improvements.

Comparison with input views. To investigate how CaesarNeRF handles a single reference view input, we present two visualizations in Figure 7, using the first two examples in Figure 1, "crest" and "lab" from the Shiny [63] dataset. We present the input view for these two examples along with the rendered results from GNT [56] and CaesarNeRF, comparing them with the ground-truth.

For a single-view input, the difference between the input and the target view can be decomposed into three parts: affine transformation, changes in occlusions, and information outside the image. CaesarNeRF can generate reason-

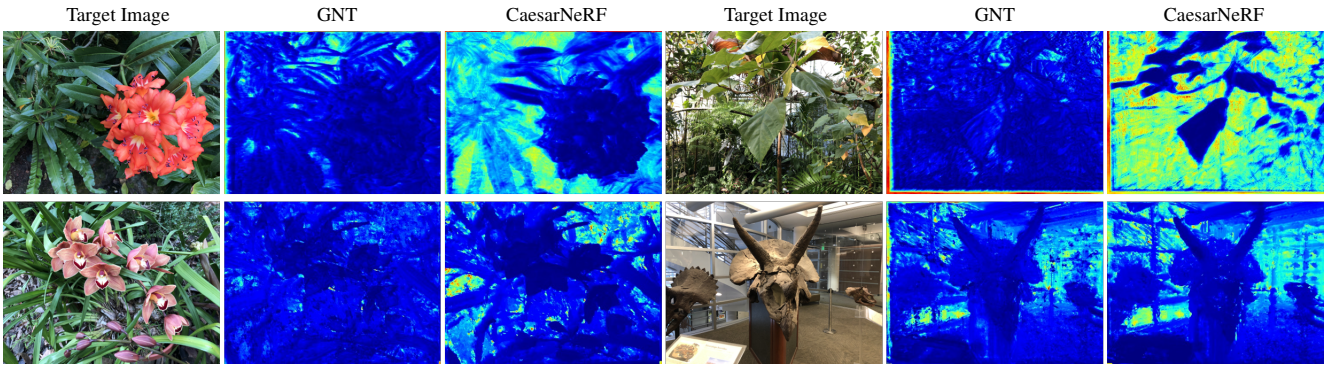


Figure 8. Depth estimation prediction using one reference view (first row) and two reference views (second row) as input from LLFF comparing CaesarNeRF with GNT.

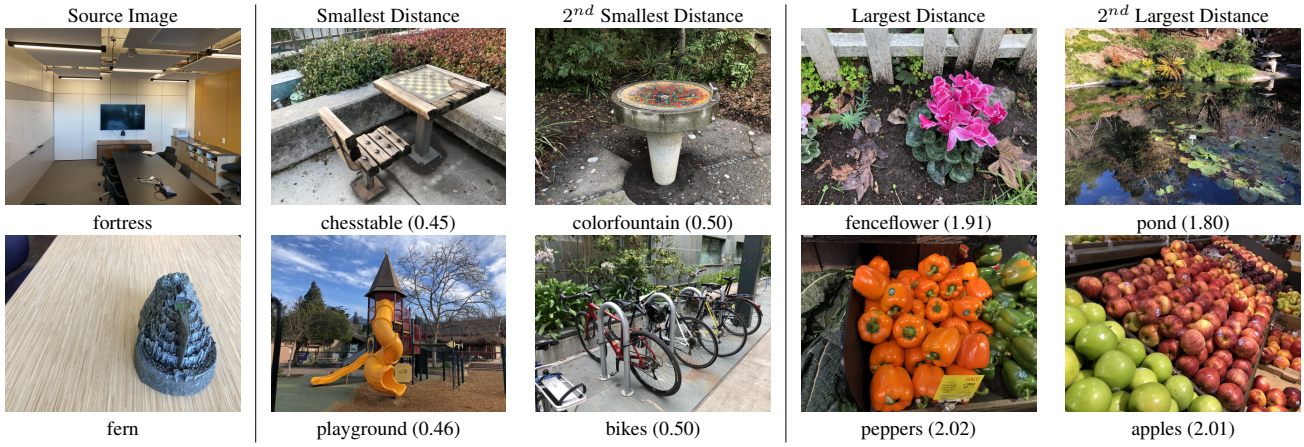


Figure 9. Largest and smallest distances for two examples from LLFF test split when matching with training scenes. Numbers ($\times 10^{-2}$) in the brackets are the L-2 distance to the source image.

able results for the affine transformation that is visible in the input view. For different occlusions between the input and target views, such as the center bottom of the first example, where the base occludes more of the pillar in the background in the rendered image compared with the input view, CaesarNeRF can predict the occlusion correctly, indicating it can handle the object relationships in the image instead of treating the scene as a flat image. For areas not captured in the input images, CaesarNeRF cannot provide rendering results if there is a large patch missing. Additionally, compared with large view shifts in the first example, small shifts between the poses of the input and target views, as in the second example, yield more accurate rendering.

Depth estimation. In addition to RGB image predictions, we also explore depth prediction on the LLFF [39] dataset, specifically in scenarios with few reference views, such as one or two. We compare the performance of CaesarNeRF and GNT [56] in Table 8. We find that GNT struggles to accurately represent the relative positions of objects within the scene when provided with few reference views. For instance, in the flower case with just one view, CaesarNeRF accurately shows that the flower is closer to the camera than the leaves in the background, whereas GNT doesn't show a distinguishable difference in depth prediction.

Furthermore, the depth estimations provided by CaesarNeRF are more consistent. In the horn example involving two views, CaesarNeRF offers better boundary delineation, particularly when confronted with reflective surfaces like the glass in the background. While CaesarNeRF performs well in capturing the relative depths within the scene, it tends to predict the background as being farther from the camera. The absolute depth values are not accurate enough to resolve all ambiguities in the reconstructed depth, constituting a potential limitation that could be addressed in future work, as discussed in Section 5.

Semantic analysis for $\tilde{S}$. As the averaged calibrated semantic representation $\tilde{S}$ includes information from frames across the same scene, we conduct further analysis to deter-

mine if such a feature from the encoder indeed encompasses semantic details about the scene. We use examples from the LLFF test set [39] to match with LLFF training examples and examine the highest and lowest response based on their L-2 distance between features from two scenes. Since we lack specific labels, the L-2 distance serves as an implicit reflection of the similarity between two vectors. A smaller distance indicates more similarity between them, and vice versa. To extract the scene-level representation for each scene, we consider the first image of each category in the LLFF training and test sets as a reference image and apply CaesarNeRF to extract the scene-level representation for the nearest ten views surrounding the reference image.

We present two examples, "room" and "fortress", accompanied by the first image of the category with the top-2 highest and lowest responses from the LLFF training set in Figure 9. In extracting the scene-level representation, the representation predominantly emphasizes structural information as well as certain similar object categories. For instance, with the source image "room", the highest responses align with table-like structures, and the scene is mainly object-centric. Conversely, the lowest responses display images of flowers or a pond in open spaces, categories not present in the source images. A similar observation applies to the second example, "fortress", where the top two responses reflect spiral structures like playground slides and bicycle pegs. In contrast, the lowest two responses from the LLFF training scenes originate from scenes with displayed objects. It's worth noting that color is not the paramount factor for scene-level semantic representation. This is evident from the large yellow and dark patches in the last example (pond) of the first row, which still has the second-lowest response when matched with the source image.

8. Visualizations.

In this section, we provide more visualizations and analysis on the fours datasets we used in our experiment. We present two different variations, framewise results as attached to this document, and the video results in the form of GIF files, which are included in the supplementary material.

Framewise results. We provide more examples comparing CaesarNeRF to GNT [56], our baseline method, as it has outperformed other baseline methods in Table 1. We show results on the LLFF [39] and Shiny [63] datasets in Figure 10 and on the mip-NeRF 360 [4] and MVImgNet [72] datasets in Figure 11. For each dataset, we present two examples using 1, 2, and 3 views as input reference views. We observe that with a limited number of input views, the overall reconstruction quality is constrained, especially for mip-NeRF 360 [4], where the differences between reference and target camera views are substantial. Nevertheless, when compared to GNT [56] across varying numbers of views, CaesarNeRF consistently generates images with better rendered quality, featuring sharper boundaries and fewer miscolored areas.

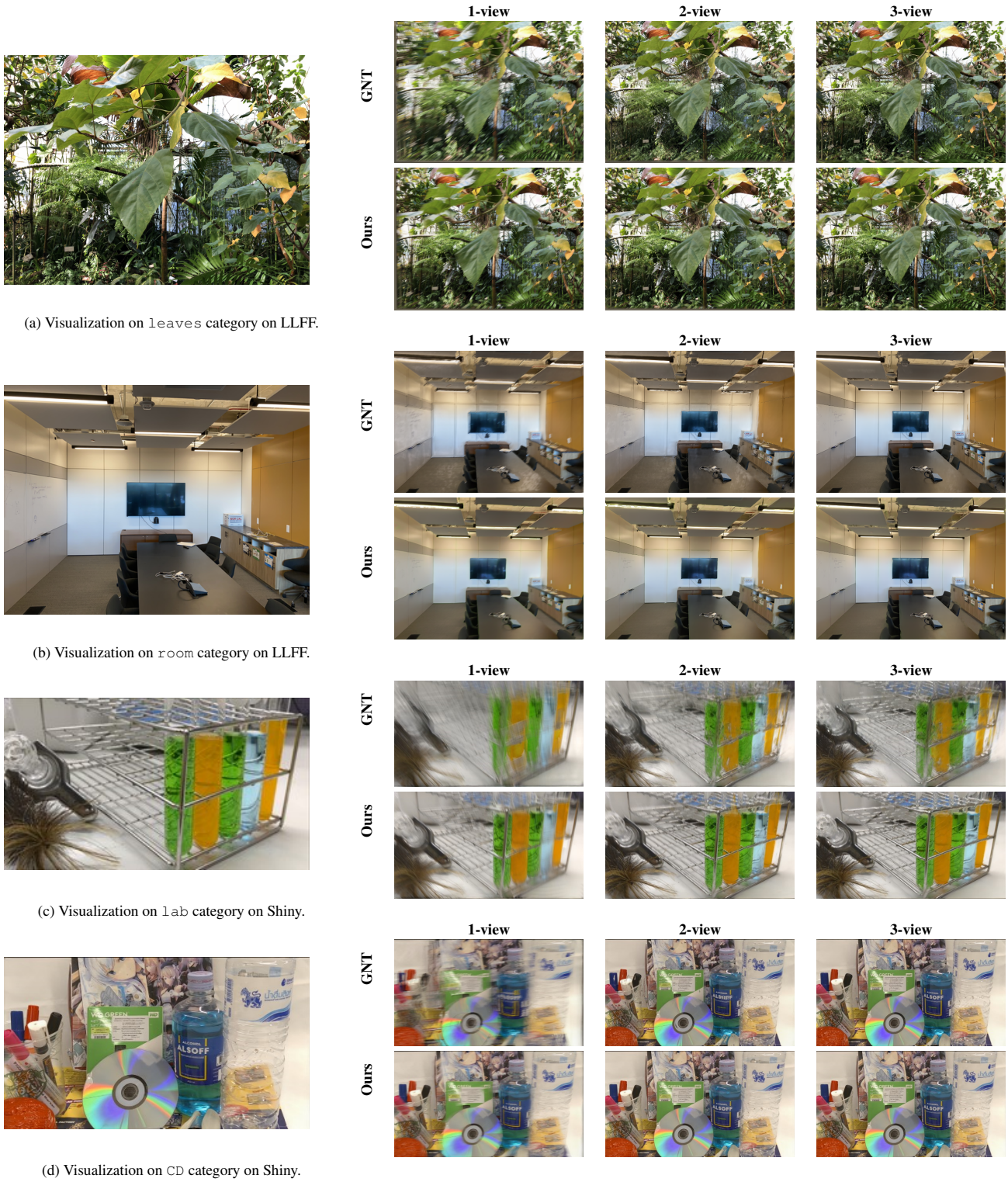
In addition to scenes with multiple objects, we also present additional visualizations from the recently introduced MVImgNet [72] dataset in Figure 12. These scenes in the MVImgNet [72] dataset mostly focus on object-centric scenarios, and we use just one reference view as input as it is a simpler case. Different from scenes featuring multiple object combinations and intricate geometrical relationships, the object-centric scenes in MVImgNet [72] provide enhanced quality with even a single input view for both GNT [56] and our CaesarNeRF. CaesarNeRF markedly surpasses our baseline method, GNT [56].

We further present the results for different numbers of views using CaesarNeRF in Figure 13. When the number of views exceeds 2, the overall quality of the reconstructed images remains consistent for CaesarNeRF, resulting in high-quality outcomes as evident in Table 4. In scenarios where the scene is object-centric with a straightforward background, CaesarNeRF excels with relatively fewer input views, maintaining its high quality across different numbers of images used as reference views.

Video results. Along with the framewise rendering, we also include rendered videos in the form of GIF files along with this document in the supplementary material. As we have focused on generalizable rendering with one or two reference views in framewise reconstructions, for video rendering, we provide examples for two other cases, including the rendering results with three reference views for generalizable rendering and per-scene optimization.

For the generalizable setting with three reference views, we have selected four scenes from LLFF with high-frequency pattern changes, including "flower", "horns", "leaves", and "orchids". We compare CaesarNeRF with its baseline, GNT [56]. When the input views are limited but sufficient, GNT exhibits more inconsistent fragments, such as the boundaries of leaves and flowers. In contrast, our proposed CaesarNeRF demonstrates more consistent boundaries with the introduction of calibrated semantic representation across views, enabling a holistic understanding.

For per-scene optimization, we present an example involving "orchids", comparing CaesarNeRF with GNT [56]. GNT primarily focuses on pixel-level rendering and lacks a holistic scene-level understanding. Consequently, the overall illumination across different viewpoints changes consistently in GNT, while CaesarNeRF produces a more consistent rendering. Additionally, we observe that in two patches we cropped out, GNT generates inconsistent structures across different views, whereas our rendering results remain more stable.



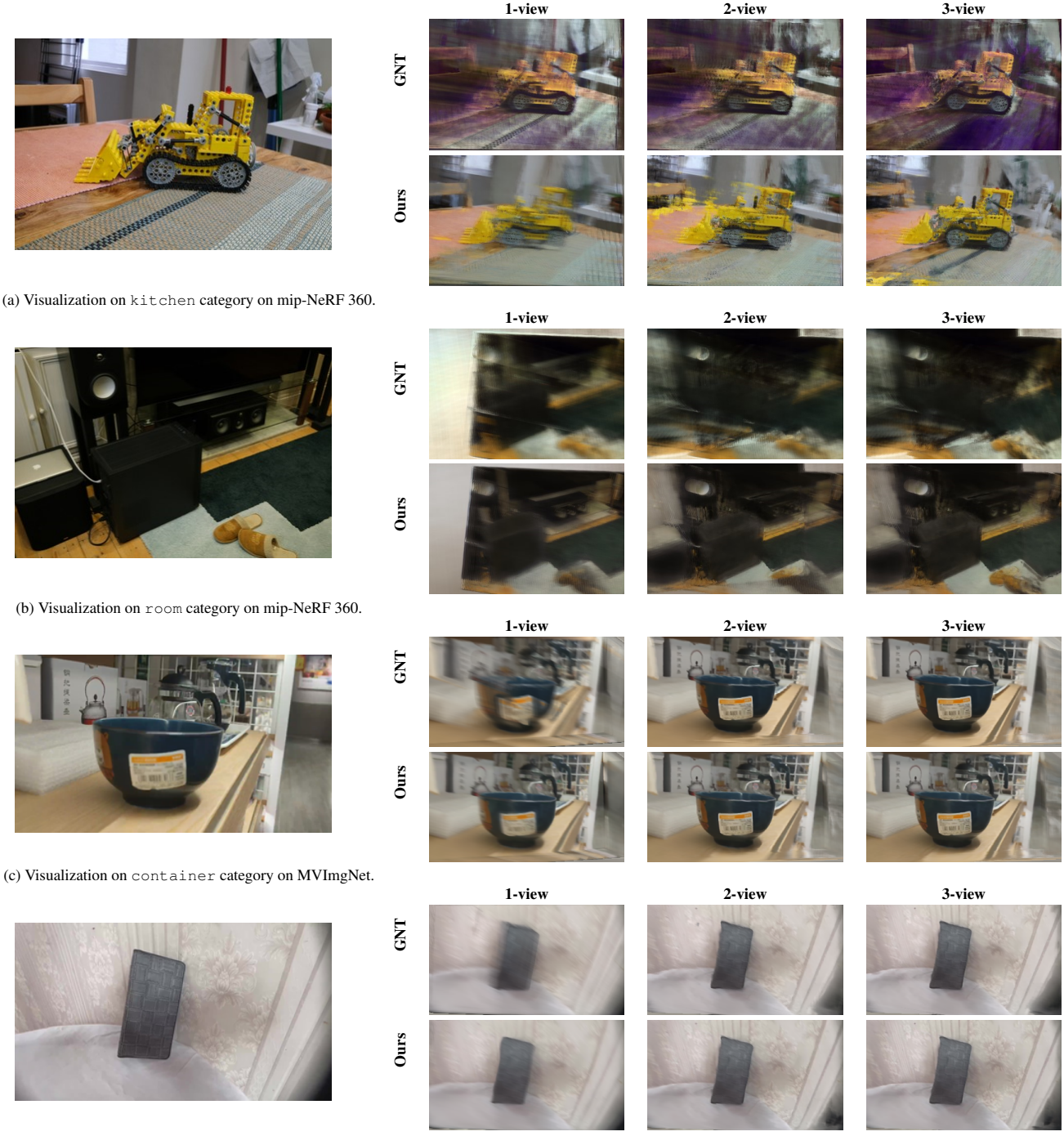
(a) Visualization on `leaves` category on LLFF.

(b) Visualization on `room` category on LLFF.

(c) Visualization on `lab` category on Shiny.

(d) Visualization on `CD` category on Shiny.

Figure 10. Additional Visualizations on LLFF [39] and Shiny [63] datasets when using 1, 2 and 3 reference views as input comparing ours with GNT [56].



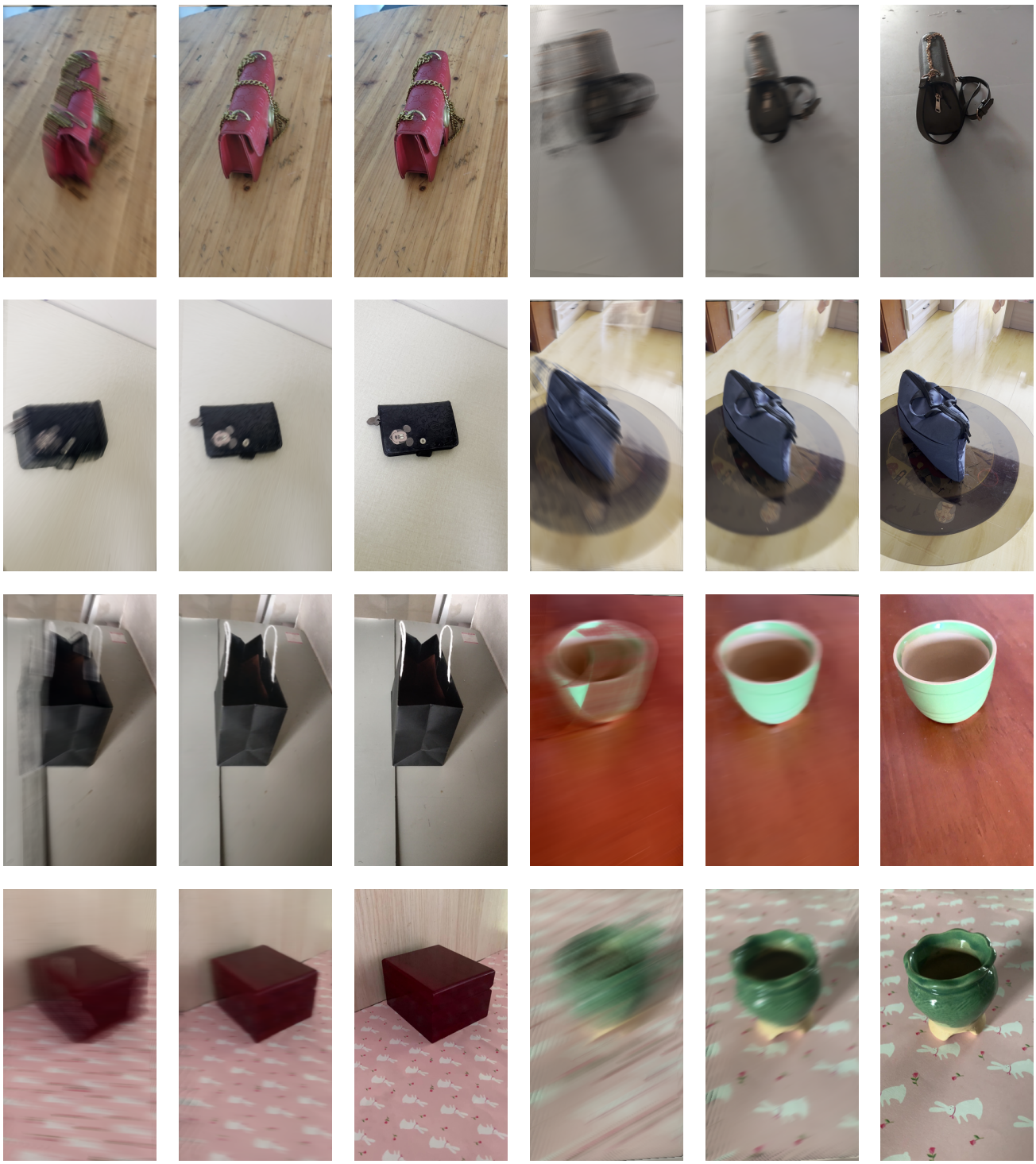
(a) Visualization on `kitchen` category on mip-NeRF 360.

(b) Visualization on `room` category on mip-NeRF 360.

(c) Visualization on `container` category on MVImgNet.

(d) Visualization on `container` category on MVImgNet.

Figure 11. Additional Visualizations on mip-NeRF 360 [4] and MVImgNet [72] datasets when using 1, 2 and 3 reference views as input comparing our proposed method, CaesarNeRF, with our baseline, GNT [56].



(a) GNT (b) Ours (c) Groundtruth (a) GNT (b) Ours (c) Groundtruth

Figure 12. Visualization for one reference view input comparing GNT [56] with CaesarNeRF on MVImgNet [72].

(a) 1-view (b) 2-view (c) 3-view (d) 4-view (e) 5-view (f) Groundtruth

Figure 13. Visualization for different numbers of reference views with CaesarNeRF on MVImgNet [72].