AssetField: Assets Mining and Reconfiguration in Ground Feature Plane Representation

Yuanbo Xiangli[1*], Linning Xu[1*], Xingang Pan[2], Nanxuan Zhao[3], Bo Dai[4], Dahua Lin[1,4]
[1]The Chinese University of Hong Kong [2]Max Planck Institute for Informatics
[3]University of Bath [4]Shanghai AI Laboratory
{xy019,xl020,dhlin}@ie.cuhk.edu.hk xpan@mpi-inf.mpg.de
nanxuanzhao@gmail.com daibo@pjlab.org.cn

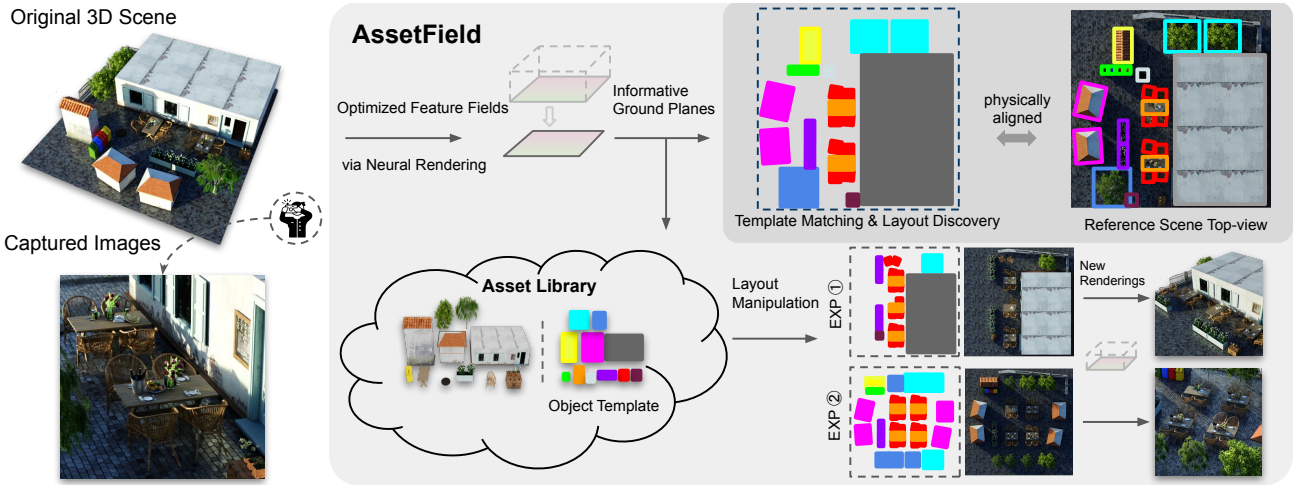

Figure 1. Man-made environments are populated with repetitive objects, such as tables, chairs, and trees, as showcased here. **AssetField** represents such environments with a set of learned informative ground feature planes well aligned with the physical scene ground plans. A variety of asset manipulations (*e.g.*, insertion, deletion, rotation, warping, etc.) and novel scene configuration can be performed directly on the ground feature plane and plugged into the original rendering network for novel scene rendering.

Abstract

Both indoor and outdoor environments (e.g. *restaurants and residential areas) are comprised of structured and repetitive components. Traditional modeling pipelines usually keep an asset library that stores template objects, where designers can constantly refer to and deploy their copies in the scene. Inspired by this observation, we present a novel neural scene representation system, namely AssetField, which is efficient and scalable by learning a set of object-centric ground feature planes, where an asset library can be automatically extracted. Unlike existing methods which require object masks to query the spatial points at inference time, our ground feature plane representation offers a natural visualization of the scene, on which a variety of operations* (e.g. *translation, deformation) can be performed to configure a new scene for rendering. Extensive experiments demonstrate that our system not only achieves competitive performance for novel-view synthesis, but also produces realistic rendering for new scene configurations with a variety of object editing.*

1. Introduction

Having an asset library is one of the key enabler to create large-scale environments in a virtual presence, where man-made environments like restaurants and neighborhoods are populated by lots of recurring items, as shown in Fig.1. A common practice in game development is to first create an *asset library* containing one *template* object of each category, either by manually modeling or capturing the physical world. Copies of the template are then deployed to build the scene. This mechanism is critical as it drastically saves memory footprint for large scene development and offers flexible editing choices for designers: altering the templates causes consistent changes for all deployed instances,

*Equal contribution.

whereas each instance can also be individually modified.

These days, the emergence of neural rendering methods offers an easier solution for scene modeling from captured 2D images. Previous methods [23, 32] tend to encode the entire scene into a single neural network, thus having difficulty to scale up due to the limited network capacity. The fact that contents are repetitive in man-made scenes inspires some work [14, 37] to tackle object-aware scene rendering in a bottom-up fashion by learning one model per object and then performing joint rendering. Despite being intuitive, one need to decide what object to model *in advance* and conduct full scans on each object to obtain their 3D models. Another branch of work learns to discover objects in the scene using instance masks [38], object motions [42], image features [20, 34], etc. Hybrid representations [8, 24, 40] on the other hand, explicitly model the scene by densely encoding local information on a feature grid and translating the interpolated features to density and radiance value via a small MLP network. [10, 12, 31] further factorize the 3D feature grids into 2D feature planes. Though demonstrating superior rendering efficiency and quality, their potential for dissecting scene structures and manipulating for reconfiguration is rarely explored.

We present **AssetField**, a novel framework that enables asset mining and layout discovery from scenes, and easy object-level editing and scene-level reconfiguration. The core of our framework is a ground feature plane representation, which models the scene with an vertical z-axis feature vector as shown in Fig. 1. We find that such representation implicitly encourages the learning of a set of informative ground feature planes, on which one can use 2D object detection techniques to mine assets and infer scene layout. As we do not rely on any object-level labels, density and color alone are not sufficiently reliable to distinguish objects of different categories. We therefore introduce an off-the-shelf 2D image feature extractor [6] to provide external semantic supervision, such that the ground feature plane representation also encodes object semantics conveyed by the image features. Consequently, the ground feature planes can approximate objects' projections on the ground, from which object instances can be extracted and clustered according to their semantic pattern. We further show that extracted objects can be organized into an *asset library* with per-category object templates and all their instance-level variations. The asset library can be naturally expanded with assets from new scenes. AssetField also enables straightforward 3D-consistent scene editing by operating directly on 2D ground feature planes in a *visible* way, making radiance field manipulation easier for human perception.

Compared to previous methods, AssetField is advantageous in the following aspects: 1) *Higher efficiency*: it is built upon a compact ground feature plane representation. 2) *Shared template*: it automatically discovers repeated objects in the scene and represents them with a shared template, which further saves memory footprint and enables category-level editing. 3) *Easy perception*: it learns an informative ground plane which naturally visualizes the scene configuration. 4) *Robust to instance variations*: introducing a semantic field to the ground feature plane brings stronger correspondence between objects of the same category.

In summary, we propose the task of asset discovery from neural scene representations. Our key contributions are: 1) We demonstrate the versatility of extracting an *asset library* from the proposed ground feature plane representation for configuring and rendering novel scenes. 2) We suggest incorporating external self-supervised semantic guidance (*e.g.* DINO [1, 6]) into radiance fields to learn scene features that provide object- and category-level indications, while being robust to slight differences within-category. 3) Extensive experiments and demonstrations show that our system is comparable to SOTA at novel view synthesis, and also produces realistic renderings on novel scenes.

2. Related Works

Neural Implicit Representations and Semantic Fields. Since the introduction of neural radiance fields [23], many advanced neural scene representations have been proposed [9, 10, 22, 22, 24, 24, 40], demonstrating superior performance in terms of quality and speed for general scene renderings. However, most of these methods are semantic and content agnostic, and many assume sparsity to design a more compact structure for rendering acceleration [10, 22, 24]. We notice that the compositional nature of a scene and the occurrence of repetitive objects within can be further utilized, where we can extract a reusable asset library for more scalable usages, similar to those adopted in the classical modeling pipeline.

A line of recent neural rendering works has explored the jointly learning a semantic fields along with the original radiance field. Earlier works use available semantic labels [45] or existing 2D detectors for supervision [21]. The realized semantic field can enable category or object-level control. More recently, [20, 34] explore the potential of distilling self-supervised 2D image feature extractors [1, 7, 13] into NeRF, and showcasing their usages of support local editing. In this work, we target an orthogonal editing goal where the accurate control of high-level scene configuration and easy editing on object instances is desired.

Object Manipulation and Scene Composition. Traditional modeling [3, 5, 28–30] and rendering pipeline [16] are vastly adopted for scene editing and novel view synthesis in early approaches. For example, Karsch *et al*. [16] propose to realistically insert synthetic objects into legacy images by creating a physical model of the scene from user annotations of geometry and lighting conditions, then composing and rendering the edited scene. Cossairt *et al*. [11] consider

synthetic and real objects compositions from the perspective of light field, where objects are captured by a specific hardware system. [17, 18, 44] also consider the problem of manipulating existing 3D scenes by matching the objects to cuboid proxies or pre-captured 3D models.

These days, several works propose to tackle object-decomposite rendering under the context of newly emerged neural implicit representations [23]. Ost *et al*. [26] target dynamic scenes and learn a scene graph representation that encodes object transformation and radiance at each node, which further allows rendering novel views and re-arranged scenes. Kundu *et al*. [21] resort to existing 3D object detectors for foreground object extraction. Sharma *et al*. [31] disentangles static and movable scene contents, leveraging object motion as a cue. Guo *et al*. [14] propose to learn object-centric neural scattering functions to implicitly model per-object light transportation, enabling scene rendering with moving objects and lights. Neural Rendering in a Room [37] targets indoor scenes by learning a radiance field for each pre-captured object and putting objects into a panoramic image for optimization. While these methods need to infer object from motion, or require one model per object, ObjectNeRF [39] learns a decompositional neural radiance field, utilizing semantic masks to separate objects from the background to allow editable scene rendering. uORF [41] performs unsupervised discovery of object radiance fields without the need for semantic masks, but requires cross-scene training and is only tested on simple synthetic objects without textures.

AssetField differs from previous methods in several aspects. Unlike previous works that rely on pre-captured objects and instance masks, AssetField automatically mines assets and scene layouts from the learned neural scene representation. Moreover, AssetField supports scene manipulation explicitly on the ground feature planes, ranging from object-level editing to scene-level reconfiguration. Finally, AssetField possesses strong generalization ability across different scenes, making the asset library expandable and highly flexible. Therefore, it is scalable and straightforward, while being efficient with minimal extra cost.

3. Basic AssetField

Aiming to identify unique object categories and infer layouts from common man-made scenes, where objects are placed on some dominant horizontal surface, *e.g*. a tabletop or floor. Fig. 2 shows an overall pipeline of AssetField. We start by illustrating the general idea of using ground feature plane representation and asset extraction, where fields are *independently* modeled following the common practices [10]. Note that in addition to density and color field, one can optionally consider modeling a semantic field for better scene comprehension, as suggested in [45]. For clearer demonstration, we provide necessary concept clarifications below: (1) Objects with similar geometry, appearance, and semantics are considered to be in the same *category*. (2) Each category has a *template* object, which can be randomly chosen or assigned by the user; all other objects within the same category are *instances* of that template. (3) *Asset library* is a set of distinct template objects from one or across multiple scenes.

3.1. Ground Feature Plane Representation

A ground plan is an informative representation commonly used for both indoor and outdoor scene modelings [12, 27, 31], where object placement within a scene is mainly reflected on a horizontal plane. We therefore parameterize such scenes with 2D *ground feature planes* expanding along the xy-dimension, where a globally encoded z-axis feature vector is paired to each feature plane to recover the 3D information. During training, a query point is projected onto the ground feature plane with its value retrieved via bilinear interpolation. Take density and color field as an example, let $\mathcal{M}$=($\mathcal{M}_\sigma, \mathcal{M}_c$) denote the set of ground feature planes for each field collectively. The two planes have the shape of $L \times W \times N$, with N=N_σ and N=N_c channel components respectively [1]. The paired z-axis features are represented with $\mathcal{H}$=($\mathcal{H}_\sigma, \mathcal{H}_c$) of shape $H \times N_\sigma$ and $H \times N_c$ correspondingly. The queried plane feature m=(m_σ, m_c) and the line feature h=(h_c, h_c) for each point at coordinate x, y, z are:

$$\begin{aligned} m_\sigma, m_c &= \text{Interp}(\mathcal{M}_\sigma, (x, y)), \text{Interp}(\mathcal{M}_c, (x, y)), \\ h_\sigma, h_c &= \text{Interp}(\mathcal{H}_\sigma, z), \text{Interp}(\mathcal{H}_c, z). \end{aligned} \tag{1}$$

The retrieved plane feature m is then combined with line features h, and decoded into density σ and color c values by two small MLP networks f_σ, f_c. Ray points are volumetrically integrated [23] to reconstruct the pixel value with:

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp\left(-\sigma_i \delta_i\right)\right) \mathbf{c}_i, \tag{2}$$

where $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$, and supervised by the 2D image reconstruction loss with $\sum_{\mathbf{r}} (\|\hat{C}(\mathbf{r}) - C(\mathbf{r})\|_2^2)$, where $C(\mathbf{r})$ is the ground truth pixel color of the ray $\mathbf{r}$.

Intuitively, the learned neural ground plane is a 2D grid of features aligned with the scene's physical ground plane. The direct advantage of adopting ground plane representation is that they match our human way of high-level editing and graphic design. Artists and designers mainly sketch on 2D canvas to reflect the 3D scene. The globally encoded z-axis feature encourages learning an informative ground plane, where 3D scene contents are abstracted into a highly

[1] We eliminate the optional semantic field here for notation brevity. The derivation of semantic field is similar to the color field. For example, the output dimension is changed to 384 when applying DINO [1] supervision.

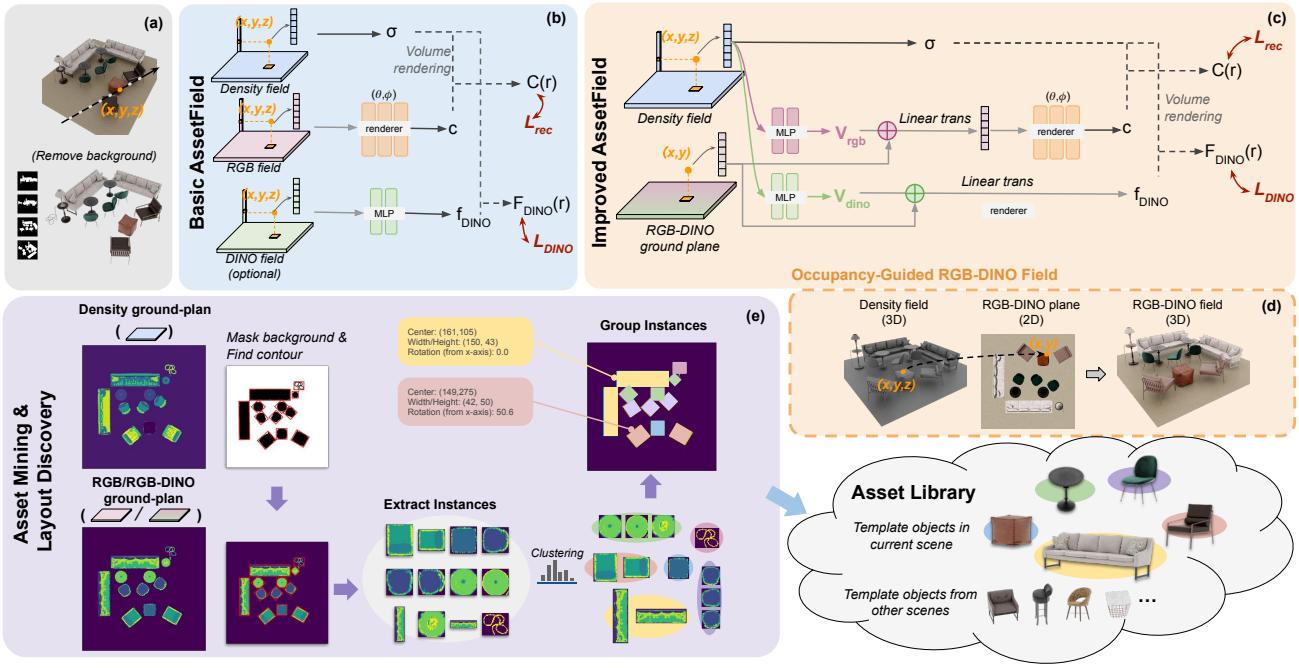

Figure 2. Overview of *AssetField*. The blue box(b) and orange box(c-d) show the pipeline of Basic and Improved AssetField respectively. Basic AssetField models a set of separate density and RGB (optionally DINO) fields. Improved AssetField unifies color and semantic field into RGB-DINO ground feature plane that is decoded into 3D with geometry guidance from density occupancy. This further encourages learning an object-centric RGB-DINO plane that is suitable for (e) asset mining and layout discovery. The extracted category *template* from the asset and layout discovery procedure is then added to the maintained asset library for future usage.

compact feature plane. This compact representation is also beneficial for training from sparse view images, where the 3-dimensional feature grids are easy to overfit with noisy values under insufficient supervision.

3.2. Asset and Layout Discovery

Object Discovery. The first step to constructing an asset library is to filter all the objects of interest from the background. Enforced by the neural rendering equation (Eq. 2), the model tends to learn a clean and accurate density field to guide the rendering of color and semantic fields. We observe that the learned density planes appear sharper object boundaries, making them suitable for object detection. We first roughly cluster the density feature plane into a predefined number of groups, from which a binary mask that separates objects from the background can be obtained. On the binary image, we perform contour detection [4, 33] to detect the borders of the objects and localize them. This results in a set of minimum rotated rectangles surrounding the objects, denoted by their centers, sizes, and orientations from the x-axis, as shown in the purple box in Fig. 2. In practice, as Basic AssetField can have a set of separate fields, object discovery can be conducted on any field that provides the most distinctive features in a scene dependent manner. For example, DINO field is found useful [20, 34] for object discovery in real-world scenes, where occlusions and overlays can cause vague boundaries for density fields.

Template Matching. With the assumption that objects with close appearance and functionality are considered in the same category, template matching is hence performed on color (and semantic) planes. Bounding boxes from previous step can be used to obtain object patches in this step. However, since the object pose in each bounding box is unknown, pixel-wise feature comparison among patches is not ideal. We therefore propose to compare the color (and semantic) distribution of object patches. To do so, we first discretize the feature plane, *e.g.* with clustering, then correspondingly crop out label patches using object bounding boxes. The similarity between label patches p_i, p_j are measure by the Jensen-Shannon Divergence (JSD) over the label distribution, denoted by $\mathrm{JSD}(p_i||p_j)$. We then perform Agglomerative clustering [25] on label patches using JSD as the distance metric. As illustrated in the grey box in Fig. 2, this results in objects grouped in categories, from which a *template* object can be selected either randomly or in a user-defined manner to be added to the *asset library*. Note that label patches are only guidance to template matching, the asset field stores the actual feature patch of density and color (and semantic) field used for future renderings.

Layout Discovery. By far, we have inferred object bounding boxes and their category from Basic AssetField. Scene layout can further be extracted by computing the canonical pose between label patches of the category template and its instances. We rotate the instance patch 3 times by 90° and

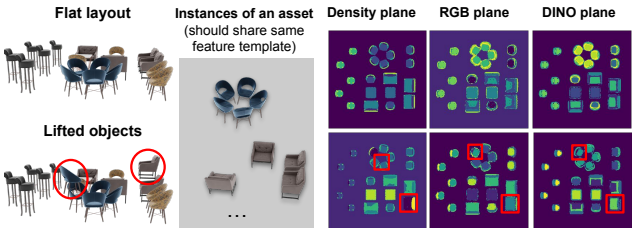

Figure 3. The ground feature planes via direct xy-z factorization are sensitive to height variation, where objects from a same category could be categorized as two different groups.

compare the amount of matching pixels with the template patch. The one with the smallest difference is considered to be the actual relative rotation. With the extracted object patches and scene layout, manipulations of object, category, and scene level can all be applied. We may also replace the original ground planes with a set of template feature patches cropped from the ground feature planes and store them with lower memory cost, where an optional template refinement step is also allowed. Details are provided the Sec. 5.3.

4. Improved AssetField

Recall that in Basic AssetField, the set of different fields is independently modeled. While such parameterization usually yields high reconstruction quality, little information is shared among fields, which may cause discrepancies at the template matching step. For example, in real-world, objects can be placed at different elevations, *e.g.* on a slight slope or a stair. This setting introduces inconsistency in ground plane features within the object category as illustrated in Fig. 3, because such variations need to be encoded into neural ground plans to be correctly decoded with the shared z-axis feature. To mitigate this discrepancy, we propose an improved version of AssetField, which learns a set of occupancy-guided ground feature planes that enforce color and semantic to constrain each other and be agnostic to object geometry and location.

Integrating RGB and DINO fields. The transformer-based method DINO [6] has recently demonstrated impressive capabilities in segmenting salient objects. A line of recent NeRF-based methods also incorporates DINO to guide learning a semantic-aware radiance field for more controllability over radiance field [20,34]. It can be noticed that color and semantic fields describe objects from two different angles, providing complementary information. For instance, objects from different categories but with similar colors can have very close RGB features that make them to be clustered together. On the other hand, semantic-rich features tend to group semantically related objects and parts, such as legs of chairs and tables, regardless of their appearance differences, as has also been pointed out in [34]. Inspired by this observation, we also resort to DINO image feature, and propose to enforce color and semantic fields to constrain each other by sharing the same set of ground feature planes, dubbed RGB-DINO field. The ground feature plane of RGB-DINO field serves as a reliable 2D plane indicating the object category for each xy location.

Occupancy-Guided RGB-DINO Field. We also observed that all the ground feature planes learned from Basic AssetField are inevitably entangled with height information, making them unreliable when objects within the same category have shape and position variation. We thus propose to treat the joint color and semantic field as a 2D field, which is decoded into 3D-aware features when queried by scene density features. The idea is illustrated in the orange boxes (c-d) in Fig. 2. Concretely, the density field is modeled the same as in Basic AssetField with $\mathcal{M}_\sigma$ and $\mathcal{H}_\sigma$ in full 3D. We then convert the extracted density feature f_σ to features v_{rgb} and v_{dino} in color and semantic latent space via two single-layer MLPs respectively. v_{rgb} and v_{dino} are decoded into scene color c and semantic f_{DINO} along with the plane feature $m_{rgb-dino} = \text{Interp}(\mathcal{M}_{rgb-dino}, (x, y))$ interpolated from the RGB-DINO plane with two small MLPs.

The above procedure can be interpreted as, the scene density feature informatively encodes the spatial occupancy information to guide the decoding of the RGB-DINO feature plane into 3D. The density field is learned on-the-fly and is agnostic to scene semantics to guarantee high reconstruction quality. Compared to providing explicit z-axis encoding to the plane features, inferring scene geometry from learned occupancy encourages RGB-DINO ground plane to learn object-centric features rather than fitting the scene.

5. Experiments

In this section, we first quantitatively compare the reconstruction quality of AssetField to general baselines, then qualitatively show the asset mining and scene manipulation results step-by-step following our proposed framework. Results of ablating various hyper-parameters (*e.g.* number of components used in ground plan representation, the combination of feature vectors of three fields, number of clusters used in template matching) are provided in supplementary.

5.1. Experimental Setup

Dataset. To better fit our scenario, we create a synthetic dataset for experimental evaluation. We compose 10 scenes resembling common man-made environments such as conference room, living room, dining hall, office, and terrace. Each scene contains objects from $3 \sim 12$ categories with a fixed light source. For each scene, we render 50 views with viewpoints sampled on a half-sphere, among which 40 are used for training and the rest for testing. Experiments are also conducted on complex real-world environments from Mip-NeRF 360 [2] for manipulations.

Implementation. We use NeRF [23] and TensoRF [10] as

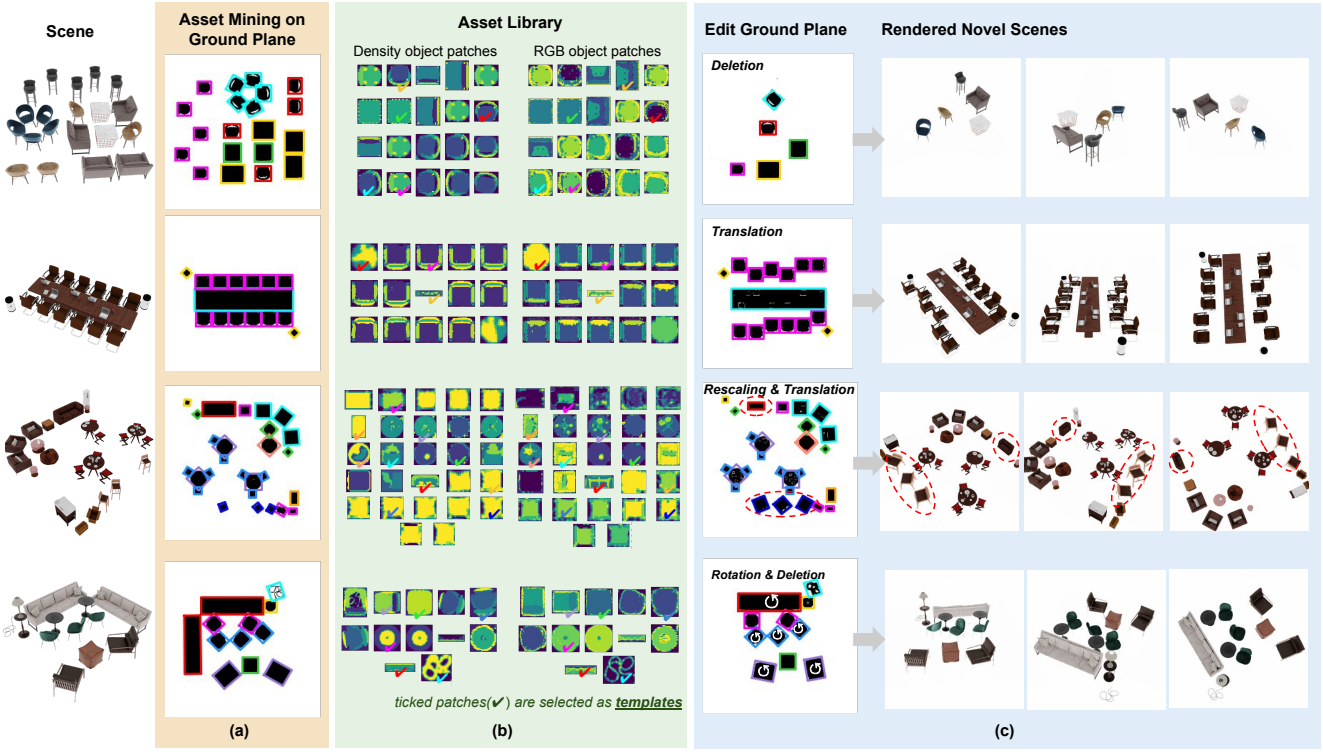

Figure 4. Results of asset mining and scene editing with *Improved AssetField*. (a) Our approach learns informative density and RGB-DINO ground feature planes that support object detection and categorization. (b) With joint training, an asset library can be constructed by storing ground *feature plane patches* of the radiance field (we show label patches here for easy visualization). (c) The proposed ground plane representation provides an explicit visualization of the scene configuration, which be directly manipulated by users. The altered ground feature plane is then fed to the global MLP renderer along with the shared z-axis feature to render views of the novel scenes. Basic operations such as object removal, translation, rotation and rescaling are demonstrated on the right.

baselines to evaluate the rendering quality of the original scenes. For a fair comparison, all methods are implemented to model an additional DINO field. Specifically, (1) NeRF is extended with an extra head to predict view-independent DINO feature [1] in parallel with density. (2) For TensoRF, we additionally construct a DINO field which is factorized along 3 directions the same as its radiance field. (3) Basic AssetField separately models a density, RGB, and DINO field where ground plane features and z-axis features are combined via outer-product, following [10]. (4) Improved AssetField models a density field the same as Basic AssetField, and a 2D RGB-DINO ground feature plane. The resolution of feature planes in TensoRF baseline and AssetField are set to 300×300. Detailed model adaptation can be found in the supplementary. We train NeRF for $200k$ iterations, and $50k$ iterations for TensoRF and AssetField using Adam [19] optimization with a learning rate set to $5e^{-4}$ for NeRF and 0.02 for TensoRF and AssetField.

5.2. Results

Novel View Rendering. We compare the general rendering quality of Basic AssetField and Improved AssetField with the adapted NeRF [23] and TensoRF [10] as described above. Quantitative results are provided in Tab. 1. It is noticeable that grid-based methods (*i.e.* AssetField and TensoRF) always outperform NeRF while also being highly efficient at both training and rendering. On the other hand, AssetField achieves comparable performance as TensoRF, indicating the suitability of adopting ground plane representations for such scenes.

	Scene1			*Scene2*			*Scene3*			*Scene4*		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
NeRF	26.207	0.931	0.695	29.373	0.967	0.260	27.559	0.969	0.286	29.927	0.968	0.366
TensoRF	35.751	0.990	0.057	**38.184**	**0.995**	**0.027**	36.933	0.994	0.034	**37.795**	**0.993**	**0.059**
Basic	36.471	0.992	0.049	36.856	0.993	0.037	36.753	0.994	0.038	37.445	0.990	0.065
Improved	**36.526**	**0.991**	**0.047**	37.271	0.994	0.035	**37.249**	**0.995**	**0.032**	37.716	0.991	0.060

Table 1. Quantitative comparison on test views for the 4 scenes in Fig. 4. We report PSNR(↑), SSIM(↑) [36] and LPIPS(↓) [43] score for evalution. The **best** and second best results are highlighted.

Object Detection and Categorization. In Fig. 5 we show the ground-aligned feature plane learned from NeRF, TensoRF and AssetField. Specifically, we regard the xy-plane from TensoRF as its ground feature plane. As NeRF does not learn such an explicit feature plane aligned with ground, we instead visualize its bird-eye view feature map. It can be observed, the xy-plane feature (corresponds to our ground feature plane) learned by TensoRF is noisy and less infor-

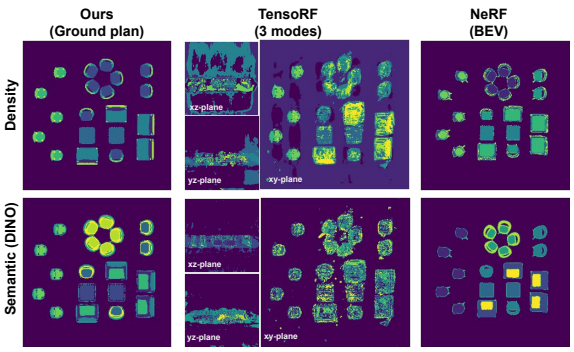

Figure 5. Our ground plane representation yields informative feature plane that clearly illustrated scene contents and layout after discretization. TensoRF with full 3 modes however produces noisy feature planes which cannot be used for object detection. NeRF does not learn such explicit feature plane but produces a relatively clean BEV feature map that can be used to infer layout.

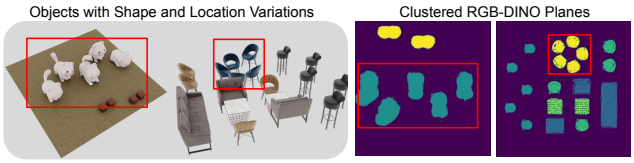

Figure 6. RGB-DINO planes are applicable to object instances within a category with both shape and location variations. To further constrain the instances with a shareable template feature grids, a separate feature plane modeling object variations could be used.

mative compared to our ground plan representation. NeRF on the other hand yields a decent hidden feature plane and a DINO feature plane in bird eye view, from which we can acquire a plausible layout to guide scene decomposition. However, unlike AssetField where the user can explicitly configure the scene beforehand, manipulating radiance fields is less intuitive and is performed at the inference time following a typical procedure as described in [20,35], which requires more computation and extra caution on the physical violation. We show that AssetField is able to identify most of the scene contents, whereas Improved AssetField is more robust to height displacement and slight appearance changing of recurring object instances, as shown in Fig. 6.

Scene Editing. Techniques on 2D image manipulation can be smoothly applied to ground feature planes. Fig. 4 shows that AssetField supports scene editing via manipulation on the ground feature plane, such as object removal, insertion, translation, rescaling, and warping. Scene-level reconfiguration is also feasible by composing a set of density and color ground plans from scratch. Note that Improved AssetField associated RGB-DINO field with space occupancy, which provides more flexible editing at the object-level. Fig. 8 demonstrates a case of topology deformation, where the blue bottle's density field is warped to the region of the brown bottle, while keeping their RGB(-DINO) feature unchanged. Results show that Improved AssetField successfully preserves object structure and part semantics, while Basic AssetField fails to render the cork correctly.

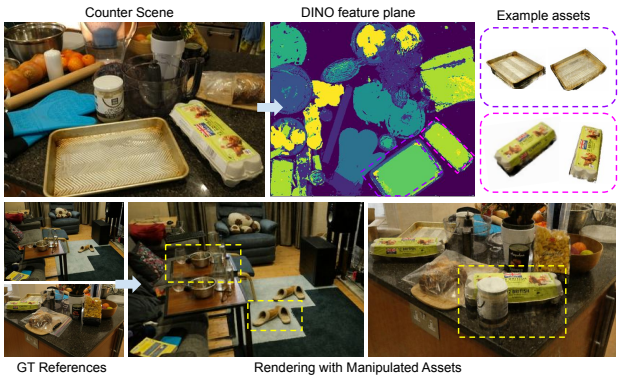

Figure 7. Editing on real-world scenarios [2]. We use DINO plane here for asset discovery. The side table and the slippers are duplicated by placing their templates on nearby ground planes.

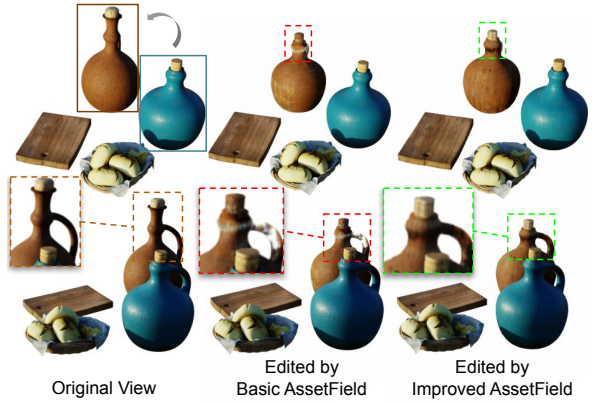

Figure 8. Density warping from the blue bottle to the region of the brown one. Basic AssetField loses the structure of the brown bottle in terms of part semantics, while Improved AssetField gives plausible editing result with appropriate structure transfer.

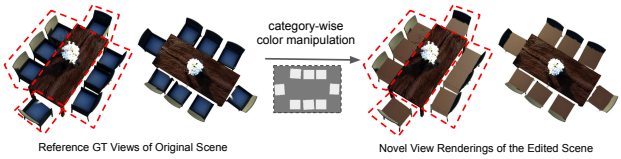

Figure 9. We apply batch-wise color changing for all instances of the chair, by replacing the template RGB feature map solely.

5.3. Applications

Manipulation with Ground Feature Planes. As demonstrated in Fig. 4 and Fig. 7-10, AssetField allows different levels of manipulation, ranging from instance-level to scene-level, 1) At *instance*-level, user can move, rotate, insert and delete an object instance in the scene. Basic deformation like warping and rescaling are also feasible. 2) *Category*-level editing can be achieved by modify the *template* object, which will affect all instances in the scene. 3) *Scene*-level reconfiguration is also intuitive, where the user

Figure 10. Expanding the 2D ground plane back to 3D feature grids, explicit control on full 3D space is allowed. We remove the ceiling light by setting the density grids as zero at the target region.

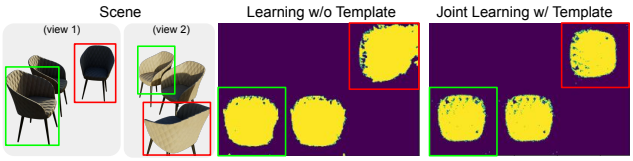

Figure 11. With appearance variations and insufficient training views for corner objects, the instance feature maps are not consistent and can not faithfully encode the category geometry. The object template, when trained among all instances within the scene, produces more accurate feature map compared to the isolated ones.

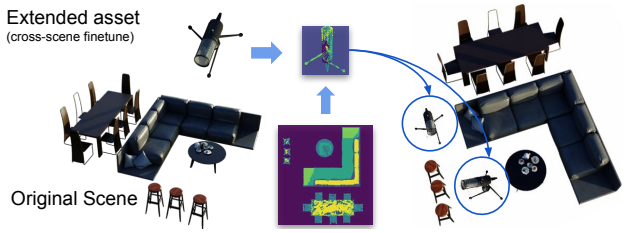

Figure 12. We expand the asset library from the living room with the newly included asset *misc* from [23]. The template of misc is in the shared latent space with the living room and can thus naturally composed together for rendering.

can compose a set of new ground plans with objects from the *asset library*. 4) Explicit 3D control is also available by expanding ground feature planes back to 3D feature grids with the pairing z-values.

Template Refinement. The learned feature planes can be vulnerable to training views, where grids with insufficient point supervision can get noisy and inaccurate values. As a result, instances from the same category often obtain feature maps with undesired variations. Moreover, the appearance differences caused by *e.g.*, lighting condition, occlusion, etc, are unwanted for a clean template map. We therefore propose *template refinement* (optional) to exclude the appearance variations and obtain a cleaner template. Fig. 11 gives a comparative example showing the refined density map with template joint training. The original scene casts shadows on instances, and the instance on the corner receive less supervision from the training view where observing angles from back side is missing. In general, once the layout configuration is obtained, we replace all instances with their representative category template and optimize this set of feature patches instead of the full ground planes. Consequently, the template can integrate supervisions from all instances from the scene and eliminate the effects of appearance variations or sparse views. Note that, storing a scene as a collection of template also saves a lot memory storage, especially for real-world scenes with many repetitive objects such as a conference room or theatre with arrays of chairs.

Expanding Asset Library. As a neural radiance field is typically scene-specific, the extracted assets are exclusive too. However, a desired feature for an asset library is to hold template objects across scenes and freely combine their instances in new scenes. To incorporate a new scene S_i, we fix $\mathcal{H}_0$ from the existing scene and learn a set of exclusive ground plans $\mathcal{M}_i$ to reconstruct S_i. The intuition is that the ground feature plane representation encourages local scene information to be encoded in the 2D feature plane as much as possible, where z-axis becomes generalizable not both within a scene and across scenes. Consequently, we can decode ground plane patches from different scenes with the shared $\mathcal{H}_0$ into cross-scene objects. An example is given in Fig. 12. Technical details can be found in supplementary.

6. Discussion and Conclusion

AssetField presents a novel system that extracts an asset library from highly structured and repetitive man-made scenes with simultaneous layout discovery. The core is to represent scenes with a set of informative ground feature planes to model density, RGB and semantic fields. Object- and scene-level editing can be easily conducted through a variety of manipulations on the learned 2D ground planes. Extensive experiments are conducted to show the easy control over multiple scenes and the realistic rendering results given novel scene configurations. We also show that the derived asset library has the potential to further reduces memory cost for large scene modeling.

While the ground feature plane is intuitive for editing, 1) it needs to be converted to 3D feature grids when dealing with *overlaying objects* in complex scenes. An ideal way is to separate objects into layers as in the industrial practices, which is potentially achievable via cross-scene learning. 2) *Partially observed objects* require a hierarchy of part-semantics to find their best template matches. 3) Self-supervised features tend to group object semantics by parts, and lack 3D consistency where 2D supervisions from different views conflict, resulting in noisy ground planes. The extracted features may also be inaccurate when facing noisy backgrounds or severe occlusions. This module can be improved by using more recent [15] or future advances.

AssetField offers a new way to interact with neural radiance fields. We believe the potential of such implicit scene representation can be further explored to support more advanced manipulations and large-scale scenes, *e.g.*, developing neural *procedural modeling* in a programmable manner.

References

[1] Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021. 2, 3, 6

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 5, 7

[3] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo - stereo matching with slanted support windows. In *BMVC*, 2011. 2

[4] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 4

[5] Adrian Broadhurst, Tom Drummond, and Roberto Cipolla. A probabilistic framework for space carving. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, 1:388–393 vol.1, 2001. 2

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9630–9640, 2021. 2, 5

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021. 2

[8] Eric Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, Jonathan Tremblay, S. Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16102–16112, 2022. 2

[9] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021. 2

[10] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. 2, 3, 5, 6

[11] Oliver S Cossairt, Shree K. Nayar, and Ravi Ramamoorthi. Light field transfer: global illumination between real and synthetic objects. *ACM SIGGRAPH 2008 papers*, 2008. 2

[12] Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W Taylor, and Joshua M Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14304–14313, 2021. 2, 3

[13] Zhiwen Fan, Peihao Wang, Yifan Jiang, Xinyu Gong, Dejia Xu, and Zhangyang Wang. Nerf-sos: Any-view self-supervised object segmentation on complex scenes. *arXiv preprint arXiv:2209.08776*, 2022. 2

[14] Michelle Guo, Alireza Fathi, Jiajun Wu, and Thomas A. Funkhouser. Object-centric neural scene rendering. *ArXiv*, abs/2012.08503, 2020. 2, 3

[15] Olivier J Hénaff, Skanda Koppula, Evan Shelhamer, Daniel Zoran, Andrew Jaegle, Andrew Zisserman, João Carreira, and Relja Arandjelović. Object discovery and representation networks. *arXiv preprint arXiv:2203.08777*, 2022. 8

[16] Kevin Karsch, Varsha Hedau, David A. Forsyth, and Derek Hoiem. Rendering synthetic objects into legacy photographs. *Proceedings of the 2011 SIGGRAPH Asia Conference*, 2011. 2

[17] Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014. 3

[18] Young Min Kim, Niloy J Mitra, Dong-Ming Yan, and Leonidas Guibas. Acquiring 3d indoor environments with variability and repetition. *ACM Transactions on Graphics (TOG)*, 31(6):1–11, 2012. 3

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015. 6

[20] Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. Decomposing nerf for editing via feature field distillation. In *Advances in Neural Information Processing Systems*, volume 35, 2022. 2, 4, 5, 7

[21] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022. 2, 3

[22] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 2

[23] 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. In *ECCV*, 2020. 2, 3, 5, 6, 8

[24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. 2

[25] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *ArXiv*, abs/1109.2378, 2011. 4

[26] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2855–2864, 2021. 3

[27] Avishkar Saha, Oscar Alejandro Mendez Maldonado, Chris Russell, and R. Bowden. Translating images into maps. *2022 International Conference on Robotics and Automation (ICRA)*, pages 9200–9206, 2022. 3

[28] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016. 2

[29] Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 2

[30] Steven M. Seitz and Charles R. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35:151–173, 1997. 2

[31] Prafull Sharma, Ayush Tewari, Yilun Du, Sergey Zakharov, Rares Ambrus, Adrien Gaidon, William T Freeman, Fredo Durand, Joshua B Tenenbaum, and Vincent Sitzmann. Seeing 3d objects in a single image via self-supervised static-dynamic disentanglement. *arXiv preprint arXiv:2207.11232*, 2022. 2, 3

[32] Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *ArXiv*, abs/1906.01618, 2019. 2

[33] Satoshi Suzuki and Keiichi Abe. Topological structural analysis of digitized binary images by border following. *Comput. Vis. Graph. Image Process.*, 30:32–46, 1985. 4

[34] Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2022. 2, 4, 5

[35] Bing Wang, Lujia Chen, and Bo-Hsiang Yang. Dm-nerf: 3d scene geometry decomposition and manipulation from 2d images. *ArXiv*, abs/2208.07227, 2022. 7

[36] Zhou Wang, Alan Conrad Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004. 6

[37] Bangbang Yang, Yinda Zhang, Yijin Li, Zhaopeng Cui, S. Fanello, Hujun Bao, and Guofeng Zhang. Neural rendering in a room. *ACM Transactions on Graphics (TOG)*, 41:1 – 10, 2022. 2, 3

[38] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *International Conference on Computer Vision (ICCV)*, October 2021. 2

[39] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13759–13768, 2021. 3

[40] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5491–5500, 2022. 2

[41] Hong-Xing Yu, Leonidas J. Guibas, and Jiajun Wu. Unsupervised discovery of object radiance fields. In *International Conference on Learning Representations*, 2022. 3

[42] Wentao Yuan, Zhaoyang Lv, Tanner Schmidt, and S. Lovegrove. Star: Self-supervised tracking and reconstruction of rigid objects in motion with neural rendering. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13139–13147, 2021. 2

[43] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 6

[44] Youyi Zheng, Xiang Chen, Ming-Ming Cheng, Kun Zhou, Shi-Min Hu, and Niloy J Mitra. Interactive images: Cuboid proxies for smart image manipulation. *ACM Trans. Graph.*, 31(4):99–1, 2012. 3

[45] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15838–15847, 2021. 2, 3