Test-time Adaptation with Slot-Centric Models

Mihir Prabhudesai [1] **Anirudh Goyal** [2] **Sujoy Paul** [3] **Sjoerd van Steenkiste** [3] **Mehdi S. M. Sajjadi** [3]
Gaurav Aggarwal [3] **Thomas Kipf** [3] **Deepak Pathak** [1] **Katerina Fragkiadaki** [1]

Abstract

Current visual detectors, though impressive within their training distribution, often fail to parse out-of-distribution scenes into their constituent entities. Recent test-time adaptation methods use auxiliary self-supervised losses to adapt the network parameters to each test example independently and have shown promising results towards generalization outside the training distribution for the task of image classification. In our work, we find evidence that these losses are insufficient for the task of scene decomposition, without also considering architectural inductive biases. Recent slot-centric generative models attempt to decompose scenes into entities in a self-supervised manner by reconstructing pixels. Drawing upon these two lines of work, we propose Slot-TTA, a semi-supervised slot-centric scene decomposition model that at test time is adapted *per scene* through gradient descent on reconstruction or cross-view synthesis objectives. We evaluate Slot-TTA across multiple input modalities, images or 3D point clouds, and show substantial out-of-distribution performance improvements against state-of-the-art supervised feed-forward detectors, and alternative test-time adaptation methods. Project Webpage: http://slot-tta.github.io/

1. Introduction

While significant progress has been made in scene perception within the last decade, decomposing scenes into familiar entities often generalizes poorly outside the training distribution (Geirhos et al., 2020; Hendrycks et al., 2021). To tackle changes in the data distribution, Test-Time Adaptation (TTA) methods (Ghifary et al., 2016; Sun et al., 2020; Wang et al., 2020) adapt the model parameters at test-time to help generalization. In recent years, a variety of methods based on TTA have been proposed, focusing on few-shot adaptation (Ren et al., 2018) where the network is given access to a few *labeled* examples, or unsupervised domain adaptation (UDA) (Zhang, 2021) where the network is given access to many *unlabelled* examples from the new distribution. A popular approach in this setting is pseudo-labelling (Wang et al., 2020; Bateson et al., 2022), where the network uses its confident predictions in some examples as additional pseudo-labelled training data to improve its accuracy. However, this approach requires *multiple* confident examples for adaptation.

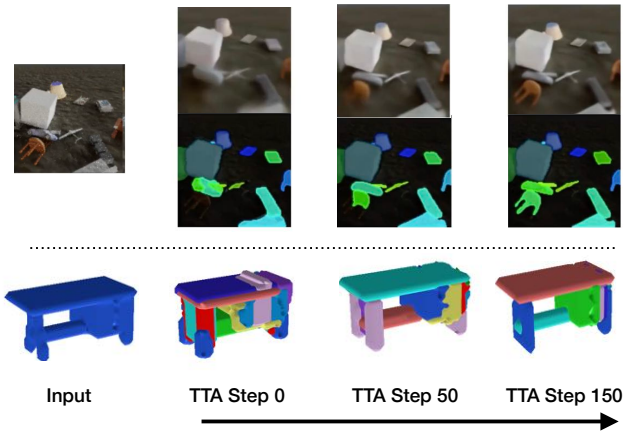

Figure 1. **Test-time adaptation in Slot-TTA**: Segmentation improves when optimizing reconstruction or view synthesis objectives via gradient descent at test-time on a single test sample.

We instead, study a specific unsupervised domain adaptation (UDA) setting where the network is adapted *independently* to each unlabelled example in the test set. This setting is analogous to a human taking more time to parse a difficult example while *not* having access to any additional information (Kahneman, 2011). Existing approaches in this setting typically devise a loss for a self-supervised pre-text task, such as rotation prediction in TTT (Sun et al., 2020) or instance discrimination in MT3 (Bartler et al., 2022), and then optimize this loss per image at test-time (Sun et al., 2020; Gandelsman et al., 2022; Bartler et al., 2022; Grill

[1]Carnegie Mellon University [2]Mila, DeepMind [3]Google Research. Correspondence to: Mihir Prabhudesai <mprabhud@cs.cmu.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

et al., 2020). While these methods have demonstrated success for the task of image classification, they are not equally effective when applied to other tasks, such as scene decomposition, that requires reasoning about objects, as we show in experiment Section 4.1. Specifically, when we apply the instance discrimination loss of MT3 (Bartler et al., 2022) to the state-of-the-art scene segmentation model Mask2Former of Cheng et al. (2021), the segmentation performance deteriorates during test-time adaptation. The question then becomes: what would be an effective TTA method for the task of scene decomposition?

Recent slot-centric generative models that attempt to segment scenes into object entities completely unsupervised, by optimizing a reconstruction objective (Eslami et al., 2016; Greff et al., 2016; Van Steenkiste et al., 2018; Goyal et al., 2021; Kosiorek et al., 2019; Locatello et al., 2020; Zoran et al., 2021) share the end-goal of scene decomposition and thus become a good candidate architecture for TTA. These methods differ in details but share the notion of incorporating a fixed set of entities, also known as *slots* or *object files*. Each slot extracts information about a single entity during encoding and is "synthesized" back to the input domain during decoding.

In light of the above, we propose Test-Time Adaptation with Slot-Centric models (Slot-TTA), a semi-supervised model equipped with a slot-centric bottleneck (Locatello et al., 2020) that jointly *segments* and *reconstructs* scenes. At training time, Slot-TTA is trained supervised to jointly *segment* and *reconstruct* 2D (multi-view or single-view) RGB images or 3D point clouds. At test time, the model adapts to a single test sample by updating its network parameters solely by optimizing the reconstruction objective through gradient descent, as shown in Figure 1. Slot-TTA builds on top of slot-centric models by incorporating segmentation supervision during the training phase. Until now, slot-centric models have been neither designed nor utilized with the foresight of Test-Time Adaptation (TTA). In particular, Engelcke et al. (2020) showed that TTA via reconstruction in slot-centric models fails due to a reconstruction-segmentation trade-off: as the entity bottleneck loosens, there's an improvement in reconstruction; however, segmentation subsequently deteriorates. We show that segmentation supervision aids in mitigating this trade-off and helps scale to scenes with complicated textures. We show TTA in semi-supervised slot-centric models significantly improves scene decomposition.

We test Slot-TTA in scene segmentation of multi-view posed images, single-view images and 3D point clouds in the datasets of PartNet (Mo et al., 2019), MultiShapeNet-Hard (Sajjadi et al., 2022b) and CLEVR (Johnson et al., 2017). The model segments objects and parts while reconstructing them in 2D or 3D. We compare its segmentation performance against state-of-the-art supervised feedforward RGB image and 3D point cloud segmentors of Mask2Former and Mask3D (Cheng et al., 2021; Schult et al., 2022), NeRF-based multi-view segmentation fusion methods of (Zhi et al., 2021) that adapt per scene through RGB and segmentation rendering, state-of-the-art test-time adaptation methods (Bartler et al., 2022), unsupervised entity-centric generative models (Locatello et al., 2020; Sajjadi et al., 2022a), and semi-supervised 3D part detectors (Wu et al., 2020; Tian et al., 2019). We show that Slot-TTA outperforms SOTA feedforward segmentators in out-of-distribution scenes, dramatically outperforms alternative TTA methods and alternative unsupervised or semi-supervised scene decomposition methods (Locatello et al., 2020; Sajjadi et al., 2022a; Wu et al., 2020; Tian et al., 2019), and better exploits multi-view information for improving segmentation over semantic NeRF-based multi-view fusion. Additionally, we show that test-time adaptation not only improves segmentation accuracy but also enhances the rendering quality of novel (unseen) views that were *not* used during test-time training.

Our contributions are as follows:

(i) We present an algorithm that significantly improves scene decomposition accuracy for out-of-distribution examples by performing test-time adaptation on each example in the test set independently.

(ii) We showcase the effectiveness of SSL-based TTA approaches for scene decomposition, while previous self-supervised test-time adaptation methods have primarily demonstrated results in classification tasks.

(iii) We introduce semi-supervised learning for slot-centric generative models, and show it can enable these methods to continue learning during test time. In contrast, previous works on slot-centric generative have neither been trained with supervision nor been used for test time adaptation.

(iv) Lastly, we devise numerous baselines and ablations, and evaluate them across multiple benchmarks and distribution shifts to offer valuable insights into test-time adaptation and object-centric learning.

Our code is publicly available to the community on our project webpage: `http://slot-tta.github.io`.

2. Related Work

Test-time adaptation In test-time adaptation, model parameters are updated at test-time for the model to better generalize to data distribution shifts. In recent years, there has been significant development in this direction. Methods such as pseudo labelling and entropy minimization (Shin et al., 2022; Wang et al., 2020; Iwasawa & Matsuo, 2021; Bateson et al., 2022) have demonstrated that supervising the model using its confident predictions helps improve its

accuracy. Adaptive BatchNorm methods (Khurana et al., 2021; Chang et al., 2019) have shown that updating the BatchNorm parameters using a set of examples can help adaptation. Despite their impressive performance, these methods inherently require confident predictions or a batch of examples to adapt. Self-supervised learning (SSL) (Sun et al., 2020; Bartler et al., 2022; Gandelsman et al., 2022) based TTA methods on the other hand, train using a combination of the task and a SSL loss. During test time, they optimize using only the SSL loss. They can adapt to *individual* examples at test time. However, all methods in the SSL setting thus far focus on the image classification task and mainly differ in terms of the SSL loss employed. For example TTT (Sun et al., 2020) uses rotation angle prediction as their SSL loss, MT3 (Bartler et al., 2022) uses a BYOL (Grill et al., 2020) loss and TTT-MAE (Gandelsman et al., 2022) uses Masked autoencoding loss (Pathak et al., 2016; He et al., 2022). Our work targets TTA for the task of scene decomposition. In our work, we show that TTA with reconstruction loss in slot-centric models can help improve the segmentation peformance in out-of-distribution scenes.

Slot-centric generative models for scene decomposition *Entity-centric* (or *object-centric*) models represent a visual scene in terms of separate object variables, often referred to as *slots* or *object files* (Greff et al., 2020; Sabour et al., 2017; Kosiorek et al., 2018; Engelcke et al., 2019; Goyal et al., 2020; Ke et al., 2021; Burgess et al., 2019; Greff et al., 2019; Zablotskaia et al., 2020; Rahaman et al., 2020). Prominent examples of such models include MONet (Burgess et al., 2019), GENESIS (Engelcke et al., 2019), IODINE (Greff et al., 2019), and Slot Attention (SA) (Locatello et al., 2020), which are trained in a fully-unsupervised setting via a simple auto-encoding objective. Scene decomposition emerges via the inductive bias of the model architecture (and in some cases, additional regularizers). OSRT (Sajjadi et al., 2022a) builds on top of SA by replacing their autoencoding objective with a novel-view synthesis objective. Slot-TTA builds on top of slot-centric models by adding segmentation supervision at training time, and using reconstruction optimization per example for TTA.

3. Method

The goal of Slot-TTA is to decompose scenes into objects or parts. We consider three different settings: (i) 2D multi-view RGB images (ii) 2D single-view RGB images and (iii) 3D point clouds. In each setting, the model encodes the scene as a set of slot vectors that capture information about individual objects and decodes them back to either (novel-view) RGB images or 3D point clouds, depending on the setting. To compute slots, Slot-TTA uses Slot Attention (SA) (Locatello et al., 2020), where visual features are softly partitioned across slots through iterative attention.

3.1. Background

Current state-of-the-art detectors and segmentors instantiate slots, a.k.a. query vectors, from 2D visual feature maps or 3D point feature clouds (Cheng et al., 2021; Schult et al., 2022) via iterative cross-attention (features to slots) and self-attention (slots to slots) operations (Carion et al., 2020; Cheng et al., 2021). Recently, Slot Attention of Locatello et al. (2020) and Recurrent Independent Mechanisms (RIMs) of Goyal et al. (2021) popularized competition amongst slots and iterative routing to encourage a single location in the input to be assigned to a unique slot vector.

Given a set of feature vectors $M \in \mathbb{R}^{N \times C}$ obtained from an encoder, where N is the number of tokens in the encoded feature map and C is the dimensionality of each token. The Slot Attention module compresses these N tokens into a set of P slot vectors $S \in \mathbb{R}^{P \times D}$, where D is the dimensionality of each slot vector.

It does so by updating a set of learned latent embedding vectors $\hat{S} \in \mathbb{R}^{P \times D}$, conditioned on M. Specifically it computes an attention matrix $A \in \mathbb{R}^{N \times P}$ between feature map M and $\hat{S}$ using the equation 1

$$A_{i,p} = \frac{\exp\left(k(M_i) \cdot q(\hat{S}_p)^T\right)}{\sum_{p=0}^{P-1} \exp\left(k(M_i) \cdot q(\hat{S}_p)^T\right)} \tag{1}$$

$$\hat{M} = v(M)$$

here k, q, and v are learnable linear transformations. Specifically k and v are applied element-wise to map $M \in \mathbb{R}^{N \times C}$ to $\mathbb{R}^{N \times D}$. Similarly, q applies an element-wise transformation from $\hat{S} \in \mathbb{R}^{P \times D}$ to $\mathbb{R}^{P \times D}$. The softmax normalization over the slot axis in equation 1 ensures competition amongst them to attend to a specific feature vector in M. It then extracts slot vectors S from M by updating $\hat{S}$ using a GRU.

$S = \mathrm{GRU}(\hat{S}, U)$ where the update vector U is calculated by taking a weighted average of elements in $\hat{M}$ using the re-normalized attention matrix $\hat{A}$:

$$U = \hat{A}^T \hat{M} \in \mathbb{R}^{P \times C}, \text{where } \hat{A}_{i,p} = \frac{A_{i,p}}{\sum_{i=0}^{N-1} A_{i,p}}$$

We iterate 3 times over equations 1 to 4, while setting $\hat{S} = S$ each time.

3.2. Test-time Adaptation with Slot-Centric Models (Slot-TTA)

We first describe the encoders and decoders of Slot-TTA for each modality. Next, we detail how we train Slot-TTA and how we adapt it at test time.

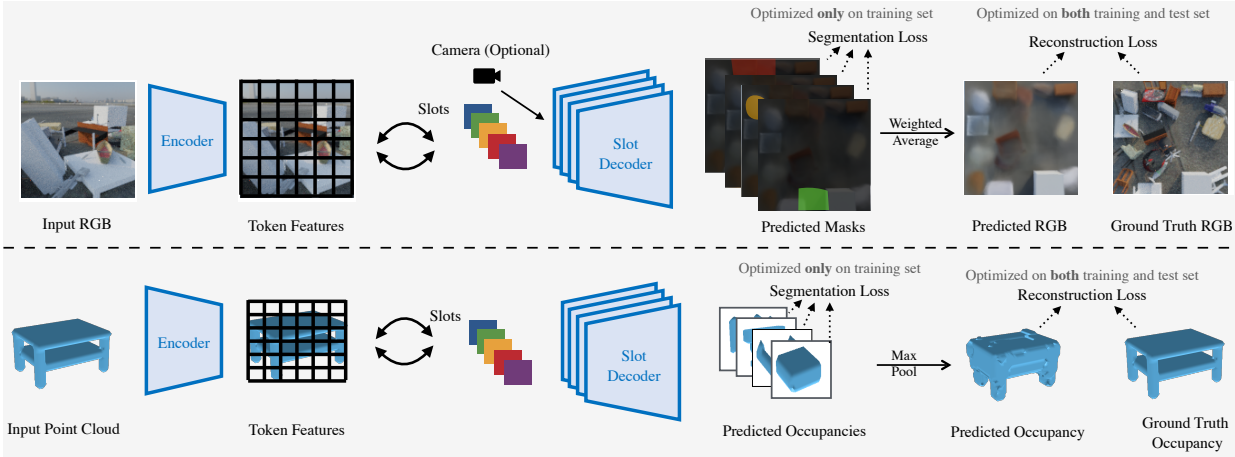

Figure 2. **Model architecture for Slot-TTA for posed multi-view or single view RGB images (*top*) and 3D point clouds (*bottom*).** Slot-TTA maps the input (multi-view posed) RGB images or 3D point cloud to a set of token features with appropriate encoder backbones. It then maps these token features to a set of slot vectors using Slot Attention. Finally, it decodes each slot into its respective segmentation mask and RGB image or 3D point cloud. It uses weighted averaging or maxpooling to fuse renders across all slots. For RGB images, we show results for multi-view and single-view settings, where in the multi-view setting the decoder is conditioned on a target camera-viewpoint. We train Slot-TTA using reconstruction and segmentation losses. At test time, we optimize only the reconstruction loss.

3.2.1. Encoding and Decoding Backbones

Posed multi-view 2D RGB images The architecture of Slot-TTA for the the multi-view RGB setting is illustrated in Figure 2 *top*. Our model's architecture is built upon OSRT (Sajjadi et al., 2022a), which is an object-centric, geometry-free novel view synthesis method. Given a set of posed RGB images as input, a CNN encodes each input image I_i into a feature grid, which is then flattened into a set of tokens with camera pose and ray direction information added in each of the tokens, similar to SRT (Sajjadi et al., 2022b). These are then encoded into a set of latent features using a transformer (Vaswani et al., 2017) Enc with multiple self-attention blocks $M = \mathrm{Enc}(\mathrm{CNN}(\mathrm{I}_i))$. The latent features M are then mapped into a set of slots S using Slot Attention (Section 3.1).

For decoding, we adopt a spatial broadcast decoder (Watters et al., 2019), where a render MLP takes as input the slot vector S_k and the pixel location p parameterized by the camera position and the ray direction pointing to the pixel to be decoded, and outputs an RGB color c_k and an unnormalized alpha score a_k for each pixel location $c_k, a_k = \mathrm{Dec}(p, S_k)$. The a_k's are normalized using a Softmax and used as weights to aggregate the predicted RGB values c_k for each slot. We ablate other decoder choices, such as the Slot Mixer decoder (Sajjadi et al., 2022a) in Appendix Section 9.1.

Single-view 2D RGB images The pipeline of Slot-TTA for the the single-view RGB setting is the same as the multi-view setting (Figure 2 top), except we do not condition the decoder with the camera information. In this setting, Slot-TTA uses a convolutional encoder the same as Locatello et al. (2020), to encode the input RGB image into a feature grid. We then add positional vectors to the feature grid and map them to a set of slot vectors using Slot Attention. Similar to the multi-view setting, each slot vector is decoded to the RGB image and an alpha mask using an MLP renderer. We parameterize pixel location p as (x, y) points on the grid instead of camera information.

3D point clouds The architecture of Slot-TTA for the 3D point cloud setting is illustrated in Figure 2 *bottom*. Our model's architecture uses a 3D point transformer (Zhao et al., 2021) which maps the 3D input points to a set of M feature vectors of C dimensions each. We set M to 128 and C to 64 in our experiments. Point feature vectors are mapped to slots with Slot Attention. Slot-TTA decodes 3D point clouds from each slot using implicit functions (Mescheder et al., 2019). Specifically, each decoder takes in as input the slot vector S_k and an (X, Y, Z) location and returns the corresponding occupancy score $o_{k,x,y,z} = \mathrm{Dec}(S_k, (x, y, z))$, where Dec is a multi-block ResNet MLP similar to that of Lal et al. (2021). We then max-pool over the slot dimension k to get an occupancy value $o_{x,y,z}$ for each 3D point in the scene. More details on our encoder and decoder architectures are included in the Appendix Section 8.

Information bottleneck in the decoder A very important ingredient for scene decomposition via optimizing reconstruction in slot-centric models is the information bottleneck in the decoder (Engelcke et al., 2020; Locatello et al., 2020; Sajjadi et al., 2022a). In slot-centric models, the decoder $\mathrm{Dec}(S_k, (x, y, z))$ decodes the segmentation mask *condi-*

tioned only on S_k, where S_k is a C dimensional slot vector, whereas in supervised image segmentors (Cheng et al., 2021; Carion et al., 2020) the decoder $\mathrm{Dec}(S_k, M)$ decodes the segmentation mask *conditioned on both the slot vector* S_k *and the feature map* M. Specifically, in the latter case, they use a dot product between the vectors in the encoded feature map M and slot vectors S_k, thus not having an information bottleneck. These different decoder choices create interesting trade-offs between test-time adaptation and fitting well to the training distribution. We discuss this further in Section 5.

3.2.2. Training and Test-time Adaptation

Slot-TTA is supervised from entity segmentation masks and self-supervised from image and 3D point cloud reconstruction objectives.

Training for joint segmentation and reconstruction Our model is trained to jointly optimize a (novel view) image synthesis or point cloud reconstruction objective alongside a task-specific segmentation objective over all the n examples in the training set. The optimization reads:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \lambda_s l_{seg}(x_i, y_i; \theta) + \lambda_r l_{recon}(x_i; \theta), \qquad (2)$$

where x represents the input scene and y the segmentation labels. For RGB image reconstruction, we minimize the mean squared error between the predicted and ground truth RGB images. For segmentation, we supervise the alpha masks a_i of each slot as provided by the decoders. We use Hungarian matching (Kuhn, 1955), a combinatorial optimization algorithm that solves assignment problems, to associate the ground truth masks with the predicted masks, and upon association we apply a categorical cross-entropy loss l_{seg}. For 3D point cloud reconstruction, we supervise the predicted occupancy probability o. We use a binary cross-entropy loss for l_{recon}. For l_{seg} we use Hungarian matching with a categorical cross-entropy loss over o_k. We weight the respective losses by λ_s and λ_r.

Test-time adaptation We refer to a single forward pass through our trained model without any test-time adaptation as *direct inference* (same as regular inference). During test-time adaptation, we adapt the parameters θ of our model by backpropagating through *only* the reconstruction objective of Eq.2 for 150 steps per scene example. For the multi-view posed image case, we test-time adapt the model considering RGB reconstruction on target RGB views, and measure segmentation performance similarly on the same target views (that are different from the input views). We visualize the test-time adaptation results across variable number of iterations in Figure 1. Further in our appendix Section 9.1 we ablate different choices of parameters to update during TTA.

4. Experiments

We test Slot-TTA in its ability to segment multi-view posed RGB images, single-view RGB images and 3D point clouds. Further, we test Slot-TTA's ability to render and decompose image views from novel (unseen) viewpoints. Our experiments aim to answer the following questions:

- How does Slot-TTA compare against state-of-the-art 2D and 3D segmentors, Mask2Former (Cheng et al., 2021) and Mask3D (Schult et al., 2022), within and outside of the training distribution?
- How does Slot-TTA compare against previous state-of-the-art test-time adaptation methods (Bartler et al., 2022)?
- How does Slot-TTA compare against NeRF-based methods that do multi-view semantic fusion (Zhi et al., 2021)?
- How much do different design choices of our model contribute to performance? We investigate decoder architecture, mask segmentation supervision, and the use of Slot Attention.

We use Adjusted Random Index (ARI) as our segmentation evaluation metric (Rand, 1971). ARI measures cluster similarity while being invariant to the ordering of the cluster centers. ARI of 0 indicates random clustering, while 1 indicates a perfect match. Note that we *do include the background component* in our ARI metric. We use the publicly available implementation of Kabra et al. (2019).

4.1. Decomposing RGB images in multi-view scenes

Dataset We evaluate Slot-TTA on the MultiShapeNet-Hard (MSN) dataset from SRT of (Sajjadi et al., 2022b). The dataset is constructed by rendering 51K ShapeNet objects using Kubric (Greff et al., 2022) simulator against 382 real world HDR backgrounds. Each scene has 9 posed RGB rendered images that are randomly assigned into input and target views for our model. We consider a train-test split where we ensure that *there is no overlap between object instances and between number of objects present in the scene* between training and test sets. Specifically, scenes with 5-7 object instances are in the training set, and scenes with 16-30 objects are in the test set. Increasing the amount of clutter or occlusions in the scene has shown to be a common test for a model's strong generalization (Cai et al., 2020). In the Appendix, we test our model on a different distribution shift where we introduce instances from *unseen object categories* from Google Scanned objects dataset (Downs et al., 2022) in the test set (Table 6). Further, in appedix Table 7, we evaluate Slot-TTA on multi-view CLEVR dataset (Johnson et al., 2017).

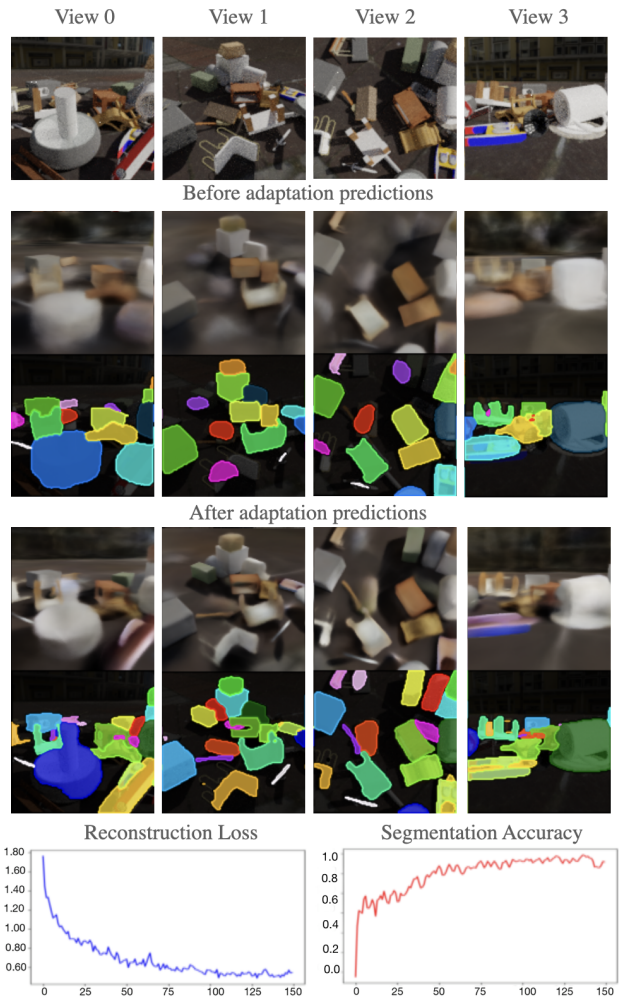

Figure 3. **Test-time adaptation in Slot-TTA for multi-view RGB images**. We visualize image reconstruction loss (blue curve) and segmentation ARI accuracy (red curve) during TTA iterations. Segmentation accuracy increases as reconstruction loss decreases.

Baselines We consider the following baselines:

(i) Mask2Former (Cheng et al., 2021), a state-of-the-art 2D image segmentor that extends detection transformers (Carion et al., 2020) to the task of image segmentation via using multiscale segmentation decoders with masked attention.

(ii) Mask2Former-BYOL which combines the segmentation model of Cheng et al. (2021) with test time adaptation using BYOL self-supervised loss of Bartler et al. (2022).

(iii) Mask2Former-Recon which combines the segmentation model of Cheng et al. (2021) with a RGB rendering module and an image reconstruction objective for test-time adaptation.

(iv) Semantic-NeRF (Zhi et al., 2021), a NeRF model which adds a segmentation rendering head to the multi-view RGB rendering head of traditional NeRFs. It is fit per scene *on all available 9* RGB posed images and corresponding segmentation maps from Mask2Former as input.

Method	in-dist : 5-7 instances		out-dist : 16-30 instances	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Slot-TTA-w/o supervision	0.32	0.30	0.33	0.29
Mask2Former	0.93	N/A	0.74	N/A
Mask2Former-BYOL	0.93	**0.95**	0.75	0.74
Mask2Former-Recon	0.93	0.92	0.74	0.67
Semantic-NeRF	N/A	0.94	N/A	0.77
Slot-TTA (Ours)	0.92	**0.95**	0.70	**0.84**

Table 1. **Instance Segmentation ARI accuracy** (higher is better) in the multi-view RGB setup for in-distribution test set of 5-7 object instances and out-of-distribution 16-30 object instances.

(v) Slot-TTA-w/o supervision, a variant of our model that does not use any segmentation supervision; rather is trained only for cross-view image synthesis similar to OSRT (Sajjadi et al., 2022a).

Results We show quantitative segmentation results for our model and baselines on target camera viewpoints in Table 1 and qualitative TTA results in Figure 3. Our conclusions are as follows:

(i) Slot-TTA with TTA outperforms Mask2Former in out-of-distribution scenes and has comparable performance within the training distribution.

(ii) Mask2Former-BYOL does not improve over Mask2Former, which suggests that adding self-supervised losses of SOTA image classification TTA methods (Bartler et al., 2022) to scene segmentation methods does not help.

(iii) Slot-TTA-w/o supervision (model identical to Sajjadi et al. (2022a)) greatly underperforms a supervised segmentor Mask2Former. This means that unsupervised slot-centric models are still far from reaching their supervised counterparts.

(iv) Slot-TTA-w/o supervision does not improve during test-time adaptation. This suggests *segmentation supervision at training time is essential for effective TTA*.

(v) Semantic-NeRF which fuses segmentation masks across views in a geometrically consistent manner outperforms single view segmentation performance of Mask2Former by 3%.

(vi) Slot-TTA which adapts model parameters of the segmentor at test time greatly outperforms Semantic-NeRF in OOD scenes.

(vii) Mask2Former-Recon performs worse with TTA, which suggests that the decoder's design is very important for aligning the reconstruction and segmentation tasks.

For qualitative comparisions with Mask2former and additional qualitative results, please refer to Figure 7 and Figure 8 in the Appendix. Further in Section 9.1 of the Appendix, we show results on a different distribution shift, where we include object instances from novel object categories that are not present in the train set. Additionally we also include results on multi-view CLEVR dataset.

4.1.1. Synthesizing and Decomposing Unseen Viewpoints

We evaluate Slot-TTA's ability to render and decompose novel (unseen) RGB image views. We consider the same dataset and train-test split as above, where in the MSN-Hard dataset we have 5-7 object instances in the training set and 16-30 object instances in the test set. Our model takes two views as input, and uses two views for TTA, and the remaining five (unseen) views are used to evaluate rendering quality. We evaluate the pixel-accurate reconstruction quality PSNR and segmentation ARI accuracy for the remaining five novel unseen viewpoints, which are not seen during TTA training. Views are randomly sampled in the different sets. We fit SemanticNeRF in the same 4 (input and target) views, and evaluate it in the same remaining five views as our model.

We see in Table 2 that Slot-TTA's rendering quality on novel (unseen) viewpoints improves with test-time adaptation. This means that test-time adaptation does not only improve segmentation on the test-time adapted viewpoints, but also improves the view synthesis quality and segmentation accuracy on novel (unseen) viewpoints. Further, we find that Semantic-NeRF does not generalize as well to novel (unseen) viewpoints. NeRFs are well known to perform poorly with a small number of views (Yu et al., 2021; Johari et al., 2022). This is because the model does not have any inductive biases of scene structure. Rather, an MLP renderer is optimized for each scene separately.

Method	PSNR		ARI	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Semantic-NeRF	N/A	18.9	N/A	0.51
Slot-TTA (Ours)	19.7	**22.6**	0.57	**0.68**

Table 2. **RGB rendering and segmentation accuracy** (higher is better) in out-of-distribution test set of 16-30 object instances.

Method	in-dist: CLEVR		out-of-dist: ClevrTex	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Slot-TTA-w/o supervision	0.21	0.22	0.15	0.37
Mask2Former	**0.97**	N/A	0.64	N/A
Slot-TTA (Ours)	0.95	**0.97**	0.35	**0.68**

Table 3. **Instance Segmentation ARI accuracy** (higher is better) for single-view RGB images. Out-of-distribution scenes are sampled from ClevrTex having different object shapes and materials compared to the in-distribution train set of CLEVR.

4.2. Decomposing RGB images in single-view scenes

We test Slot-TTA in it's ability to segment single-view RGB images on CLEVR (Johnson et al., 2017) and ClevrTex (Karazija et al., 2021) which are standard object-centric learning datasets. We compare Slot-TTA against state-of-the-art supervised and unsupervised scene decomposition methods such as Mask2Former (Cheng et al., 2021) and Slot Attention (Locatello et al., 2020).

Dataset We consider CLEVR (Johnson et al., 2017) and it's out-of-distribution variant ClevrTex (Karazija et al., 2021). For supervised training, we use the CLEVR dataset open-sourced by Kabra et al. (2019). The train set consists of standard CLEVR scenes sampled from 3 object shapes, 8 object colors and 2 object materials. For the test set we use the ClevrTex dataset of Karazija et al. (2021) which is sampled from 8 object shapes and 85 object materials. Specifically we use their publicly available ClevrTex-PlainBG dataset, thus resulting in a significant distribution shift in terms of object properties. Both the datasets contain 3-10 object instances per scene.

Baselines We compare against the following baselines:

(i) Mask2Former (Cheng et al., 2021) a state-of-the-art 2D image segmentor.

(ii) Slot-TTA-w/o supervision, has the same model architecture as our method except is not trained using supervised segmentation loss. This method is similar to Slot Attention (Locatello et al., 2020).

Results We show our results in Table 3. Our findings are similar to Section 4.1, for instance Slot-TTA significantly outperforms Mask2former on out-of-distribution scenes after doing test-time adaptation. Similarly Slot-TTA gets similar results to Mask2former on the in-distribution set after adaptation. Further training Slot Attention with supervision is important to get high ARI accuracy.

4.3. Decomposing 3D point clouds

We test Slot-TTA in its ability to segment 3D object point clouds into parts. We consider two types of distribution shifts: (i) Part-to-object distribution shift, where our model and baselines are supervised from a dataset of generic 3D part primitives and are tested on segmenting 3D object point clouds. (ii) Cross-object-category distribution shift, where our model and baselines are supervised from 3D object part segmentations and tested on segmenting instances of *novel (unseen) categories* into parts.

4.3.1. PART-TO-OBJECT DISTRIBUTION SHIFT

Dataset We consider the dataset split of Shape2Prog (Tian et al., 2019). Our supervised training set consists of scenes that contain 2-3 primitive parts, resized and translated in different 3D locations of a blank 3D canvas. The part primitives (akin to generalized cylinders of Marr (1982)) consist of differently sized cubes, cuboids, and discs. Our test set consists of unseen object categories specifically chairs and tables from PartNet, each composed of 6 to 16 parts.

Baselines We consider the following semi-supervised 3D baselines which show results for the aforementioned train-test split in their respective papers:

(i) PQ-Nets of Wu et al. (2020), which assumes access to a set of primitive 3D parts for pre-training. Specifically they first learn a primitive part decoder, then they learn a sequential encoder that encodes the 3D point cloud into a 1D latent vector and sequentially decodes parts using the pretrained part decoder. We use the publicly available code to train the model.

(ii) Shape2Prog of Tian et al. (2019), which is a shape program synthesis method that is trained supervised to predict shape programs from object 3D point clouds. The program represents the part category, location, and the symmetry relations among the parts (if any).

We further consider the following ablative versions of Slot-TTA:

(i) Slot-TTA w/o supervision, a variant of our model that does not use any segmentation supervision.

(ii) Slot-TTA w/o SlotAttention, which instead of Slot Attention it maps 3D point features to slots via iterative layers of cross (query to point) and self (query-to-query) attention layers on learnable query vectors similar to DETR (Carion et al., 2020). Please note that slots and queries represent the same thing, but we use the terminology of DETR (Carion et al., 2020) in this case.

(iii) Slot-TTA w/o SlotDecoder, *does not use a information bottleneck during mask decoding*. Rather it decodes the mask by computing a dot product between slot vectors and the feature grid, similar to Mask3D. For the reconstruction head it uses the same decoder as Slot-TTA.

Results We show quantitative results of our model and baselines in Table 4. Our conclusions are as follows:

(i) Slot-TTA significantly outperform PQ-Nets (Wu et al., 2020) and Shape2Prog (Tian et al., 2019).

(ii) Slot-TTA outperforms Slot-TTA w/o SlotAttention, which indicates that competition among slots is important for TTA.

Method	**in-dist:**		**out-dist:**	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Shape2Prog	0.65	0.71	0.26	0.38
PQ-Nets	0.63	0.67	0.19	0.28
Slot-TTA w/o SlotAttention	0.71	0.74	0.41	0.52
Slot-TTA w/o Supervision	0.42	0.38	0.38	0.27
Slot-TTA w/o SlotDecoder	**0.79**	**0.77**	0.42	0.33
Slot-TTA (Ours)	0.69	0.75	**0.44**	**0.58**

Table 4. **Instance Segmentation ARI accuracy** (higher is better) in instances from generic primitives dataset (in-distribution) and Chair and Table categories (out-of-distribution) when trained using the supervision from generic primitive compositions (part-to-object distribution shift).

(iii) Test-time adaptation through reconstruction feedback increases 3D part segmentation accuracy on both our model *and* our baselines. We think this is due to the common slot bottleneck present in the baselines. However, the difference is that the baselines infer slot vectors one at a time using sequential RNN operations whereas our model infers them jointly using SlotAttention.

(iv) Slot-TTA w/o supervision does not improve during TTA, much like in the multi view RGB image case of Section 4.1.

(v) The direct inference version of Slot-TTA w/o SlotDecoder significantly outperforms Slot-TTA in distribution, however it fails to improve with TTA in out-of-distribution setting. This suggests that information bottleneck in the decoder is a plus for test-time adaptation, however it is a minus for fitting to the training distribution.

Please, refer to Section 9.2 of the Appendix for further ablations and qualitative comparison against baselines. Further please refer to our project webpage <link> for videos of 3D object segmentation during TTA iterations. Finally, please refer to Appendix Figure 6 for visualization of the 3D primitive dataset.

4.3.2. CROSS-CATEGORY DISTRIBUTION SHIFT

Dataset We divide object categories in the PartNet benchmark (Mo et al., 2019) into train and test sets such that there is no overlap of categories between the two sets. Specifically, the model has access to the ground-truth point cloud segmentation of eight categories and is tested on the remaining 9 PartNet categories. We use annotation for the finest segmentation level available (level 3).

Baselines We compare our model against the following baselines:

(i) Mask-3D (Schult et al., 2022), a state-of-the-art 3D instance segmentation model, which is a 3D adaptation of the 2D state-of-the-art segmentor Mask2Former.

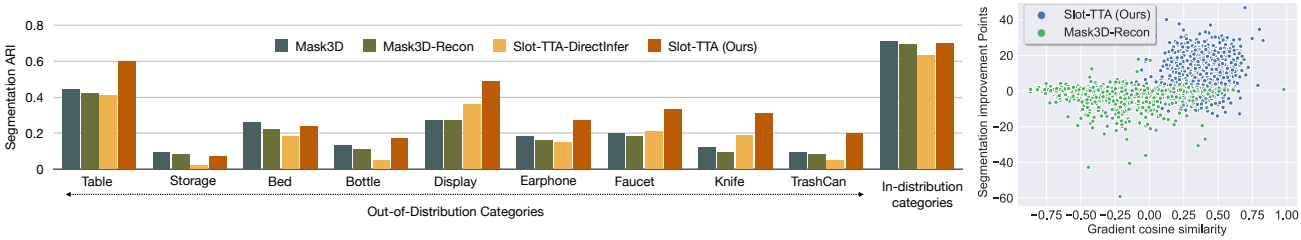

Figure 4. **Left:** Instance segmentation ARI accuracy on out-of-distribution categories for Slot-TTA and baselines. **Right:** Segmentation-reconstruction gradient similarity versus segmentation improvement. We plot the cosine similarity of gradients from reconstruction and segmentation losses for various examples in the test-set paired with their corresponding difference in segmentation accuracy before and after TTA (Improvement Points).

(ii) Mask-3D-Recon, where we add a reconstruction decoding head to the Mask-3D baseline, for doing test-time adaptation.

Please refer to Section 8.3 of the Appendix for additional details on the baselines.

Results We show quantitative results in Figure 4 on the *left* side and qualitative results in Figure 2 and in the Appendix. Our conclusions are as follows:

(i) Slot-TTA outperforms Mask3D in OOD categories and has comparable performance within the training distribution.

(ii) Mask3D-Recon does not improve during test-time adaptation. This suggests that the decoder's architecture in Slot-TTA are important for aligning the segmentation and reconstruction objectives. To illustrate this point further, we visualize in Figure 4 on the *right* side, the cosine similarity between gradient vectors of segmentation and reconstruction loss during TTA of multiple individual scene examples for our model and for the Mask3D-Recon baseline. As can be seen, higher gradient similarity correlates with larger improvements in segmentation performance from reconstruction gradient descent during TTA. We find that Slot-TTA has higher gradient similarity than Mask3D-Recon, which explains their performance difference during TTA.

(iii) Direct inference in Slot-TTA has a lower performance than Slot-TTA, Mask3D and Mask3D-Recon baselines.

5. Discussion - Limitations

As can be inferred from the results in Sections 4.1, 4.3.1 and 4.3.2, the direct inference versions of baseline models such as Mask2former or Mask3D, significantly outperforms the direct inference version of Slot-TTA. On the other hand, Slot-TTA after TTA significantly outperforms the above baselines on out-of-distribution examples. This stark difference in performance between direct inference and after TTA setting can be attributed to the information bottleneck of the segmentation decoder, which we ablate in Section 4.3.1. We find that the presence of an information bottleneck within the decoder adversely impacts the direct inference performance. This effect is found to escalate exponentially when dealing with complex datasets like MS-COCO (Lin et al., 2014), even when trained supervised using human-annotated segmentations. This significantly diminished direct inference capability compromises the model's potential for doing test-time adaptation. Exploration of architectures that can both fit on large scale training data, such as COCO, and be test time adapted is a direct avenue of our future work. Our present work sheds lights to limitations and opportunities of slot-centric models when combined with entity segmentation supervision.

6. Conclusion

We presented Slot-TTA, a novel semi-supervised scene decomposition model equipped with a slot-centric image or point-cloud rendering component for test time adaptation. We showed Slot-TTA greatly improves instance segmentation on out-of-distribution scenes using test-time adaptation on reconstruction or novel view synthesis objectives. We compared with numerous baseline methods, ranging from state-of-the-art feedforward segmentors, to NERF-based TTA for multiview semantic fusion, to state-of-the-art TTA methods, to unsupervised or weakly supervised 2D and 3D generative models. We showed Slot-TTA compares favorably against all of them for scene decomposition of OOD scenes, while still being competitive within distribution.

Acknowledgements We thank Mike Mozer, Klaus Greff and Ansh Khurana for the helpful discussions and Aravindh Mahendran for the paper feedback. Part of the Work was done during Mihir Prabhudesai's internship at Google. This material is based upon work supported by DARPA Young Investigator Award, a NSF CAREER award, DARPA Machine Common Sense, ONR N00014-22-1-2096 and ONR MURI N00014-22-1-2773.

References

Bartler, A., Bühler, A., Wiewel, F., Döbler, M., and Yang, B. Mt3: Meta test-time training for self-supervised test-time adaption. In *International Conference on Artificial Intelligence and Statistics*, pp. 3080–3090. PMLR, 2022.

Bateson, M., Lombaert, H., and Ben Ayed, I. Test-time adaptation with shape moments for image segmentation. In *MICCAI*, pp. 736–745. Springer, 2022.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Cai, Z., Zhang, J., Ren, D., Yu, C., Zhao, H., Yi, S., Yeo, C. K., and Change Loy, C. Messytable: Instance association in multiple camera views. In *European Conference on Computer Vision*, pp. 1–16. Springer, 2020.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229. Springer, 2020.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

Chang, W.-G., You, T., Seo, S., Kwak, S., and Han, B. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 7354–7362, 2019.

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention mask transformer for universal image segmentation. *CoRR*, abs/2112.01527, 2021. URL https://arxiv.org/abs/2112.01527.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Downs, L., Francis, A., Koenig, N., Kinman, B., Hickman, R., Reymann, K., McHugh, T. B., and Vanhoucke, V. Google scanned objects: A high-quality dataset of 3d scanned household items. *arXiv preprint arXiv:2204.11918*, 2022.

Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I. Genesis: Generative scene inference and sampling with object-centric latent representations. *arXiv preprint arXiv:1907.13052*, 2019.

Engelcke, M., Jones, O. P., and Posner, I. Reconstruction bottlenecks in object-centric generative models. *arXiv preprint arXiv:2007.06245*, 2020.

Eslami, S., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Kavukcuoglu, K., and Hinton, G. E. Attend, infer, repeat: Fast scene understanding with generative models. *arXiv preprint arXiv:1603.08575*, 2016.

Gandelsman, Y., Sun, Y., Chen, X., and Efros, A. A. Test-time training with masked autoencoders. *arXiv preprint arXiv:2209.07522*, 2022.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Ghifary, M., Kleijn, W. B., Zhang, M., Balduzzi, D., and Li, W. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European conference on computer vision*, pp. 597–613. Springer, 2016.

Goyal, A., Lamb, A., Gampa, P., Beaudoin, P., Levine, S., Blundell, C., Bengio, Y., and Mozer, M. Object files and schemata: Factorizing declarative and procedural knowledge in dynamical systems. *arXiv preprint arXiv:2006.16225*, 2020.

Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., and Schölkopf, B. Recurrent independent mechanisms. In *ICLR*, 2021.

Greff, K., Rasmus, A., Berglund, M., Hao, T. H., Schmidhuber, J., and Valpola, H. Tagger: Deep unsupervised perceptual grouping. *arXiv preprint arXiv:1606.06724*, 2016.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

Greff, K., Van Steenkiste, S., and Schmidhuber, J. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

Greff, K., Belletti, F., Beyer, L., Doersch, C., Du, Y., Duckworth, D., Fleet, D. J., Gnanapragasam, D., Golemo, F., Herrmann, C., et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3749–3761, 2022.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z., Gheshlaghi Azar, M., et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021.

Iwasawa, Y. and Matsuo, Y. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34: 2427–2440, 2021.

Johari, M. M., Lepoittevin, Y., and Fleuret, F. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18365–18375, June 2022.

Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2901–2910, 2017.

Kabra, R., Burgess, C., Matthey, L., Kaufman, R. L., Greff, K., Reynolds, M., and Lerchner, A. Multi-object datasets. https://github.com/deepmind/multi-object-datasets/, 2019.

Kahneman, D. *Thinking, fast and slow*. Macmillan, 2011.

Karazija, L., Laina, I., and Rupprecht, C. Clevrtex: A texture-rich benchmark for unsupervised multi-object segmentation. *arXiv preprint arXiv:2111.10265*, 2021.

Ke, N. R., Didolkar, A., Mittal, S., Goyal, A., Lajoie, G., Bauer, S., Rezende, D., Bengio, Y., Mozer, M., and Pal, C. Systematic evaluation of causal discovery in visual model based reinforcement learning. *arXiv preprint arXiv:2107.00848*, 2021.

Khurana, A., Paul, S., Rai, P., Biswas, S., and Aggarwal, G. Sita: Single image test-time adaptation. *arXiv preprint arXiv:2112.02355*, 2021.

Kosiorek, A., Kim, H., Teh, Y. W., and Posner, I. Sequential attend, infer, repeat: Generative modelling of moving objects. *Advances in Neural Information Processing Systems*, 31:8606–8616, 2018.

Kosiorek, A., Sabour, S., Teh, Y. W., and Hinton, G. E. Stacked capsule autoencoders. *Advances in neural information processing systems*, 32, 2019.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Lal, S., Prabhudesai, M., Mediratta, I., Harley, A. W., and Fragkiadaki, K. Coconets: Continuous contrastive 3d scene representations, 2021.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Marr, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman and Co., New York, NY, 1982.

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., and Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4460–4470, 2019.

Mo, K., Zhu, S., Chang, A. X., Yi, L., Tripathi, S., Guibas, L. J., and Su, H. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 909–918, 2019.

Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., and Efros, A. A. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2536–2544, 2016.

Rahaman, N., Goyal, A., Gondal, M. W., Wuthrich, M., Bauer, S., Sharma, Y., Bengio, Y., and Schölkopf, B. S2rms: Spatially structured recurrent modules. *arXiv preprint arXiv:2007.06533*, 2020.

Rand, W. M. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., and Zemel, R. S. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

Sabour, S., Frosst, N., and Hinton, G. E. Dynamic routing between capsules. *arXiv preprint arXiv:1710.09829*, 2017.

Sajjadi, M. S., Duckworth, D., Mahendran, A., van Steenkiste, S., Pavetić, F., Lučić, M., Guibas, L. J., Greff, K., and Kipf, T. Object scene representation transformer. *arXiv preprint arXiv:2206.06922*, 2022a.

Sajjadi, M. S., Meyer, H., Pot, E., Bergmann, U., Greff, K., Radwan, N., Vora, S., Lučić, M., Duckworth, D., Dosovitskiy, A., et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6229–6238, 2022b.

Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., and Leibe, B. Mask3d for 3d semantic instance segmentation. *arXiv preprint arXiv:2210.03105*, 2022.

Shin, I., Tsai, Y.-H., Zhuang, B., Schulter, S., Liu, B., Garg, S., Kweon, I. S., and Yoon, K.-J. Mm-tta: Multi-modal test-time adaptation for 3d semantic segmentation. In *CVPR*, 2022.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Tian, Y., Luo, A., Sun, X., Ellis, K., Freeman, W. T., Tenenbaum, J. B., and Wu, J. Learning to infer and execute 3d shape programs. *arXiv preprint arXiv:1901.02875*, 2019.

Van Steenkiste, S., Chang, M., Greff, K., and Schmidhuber, J. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *arXiv preprint arXiv:1802.10353*, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., and Darrell, T. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Watters, N., Matthey, L., Burgess, C. P., and Lerchner, A. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. *arXiv preprint arXiv:1901.07017*, 2019.

Wu, R., Zhuang, Y., Xu, K., Zhang, H., and Chen, B. Pq-net: A generative part seq2seq network for 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Yu, A., Ye, V., Tancik, M., and Kanazawa, A. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4578–4587, 2021.

Zablotskaia, P., Dominici, E. A., Sigal, L., and Lehrmann, A. M. Unsupervised video decomposition using spatio-temporal iterative inference. *arXiv preprint arXiv:2006.14727*, 2020.

Zhang, Y. A survey of unsupervised domain adaptation for visual recognition. *arXiv preprint arXiv:2112.06745*, 2021.

Zhao, H., Jiang, L., Jia, J., Torr, P. H., and Koltun, V. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16259–16268, 2021.

Zhi, S., Laidlow, T., Leutenegger, S., and Davison, A. J. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15838–15847, 2021.

Zoran, D., Kabra, R., Lerchner, A., and Rezende, D. J. Parts: Unsupervised segmentation with slots, attention and independence maximization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10439–10447, 2021.

Appendix

The structure of this appendix is as follows: In Section 7 we cover the details on the datasets. In Section 8 we specify further implementation details. In Section 9 we provide additional qualitative and quantitative results for the experiments in Section 4 of our main paper.

7. Datasets

7.1. Multi-view RGB

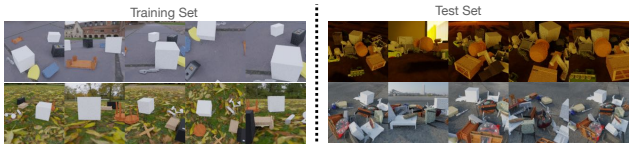

Figure 5. We visualize samples from the train-test split used by us in experiment Section 4.1. Different rows correspond to different scenes and different columns correspond to different viewpoints.

We use the MultiShapeNet-Hard dataset of Scene Representation Transformer, a complex photo-realistic dataset for Novel View Synthesis (Sajjadi et al., 2022b). Our train split consists of 5-7 ShapeNet objects placed at random locations and orientations in the scene. The backgrounds are sampled from 382 realistic HDR environment maps. Our test set consists of 16-30 novel object instances placed at novel arrangements. We sample objects from a pool of 51K ShapeNet objects across all categories, we divide the pool into train and test such that the test set consists of objects not seen during training. The train split has 200K scenes, and the test set consists of 4000 scenes, each with 9 views. We had to regenerate the dataset for this specific train-test split.

7.2. Single-view RGB

We use the CLEVR dataset of Johnson et al. (2017), which includes RGB images and segmentation masks rendered using Blender. For the training set we use the official dataset opensourced by Kabra et al. (2019). For the test-set we use the official dataset of ClevrTex by Karazija et al. (2021). Specifically we use their publicly available ClevrTex-PlainBG dataset. Due to computational cost of TTA, we only use the first 1000 scenes in the dataset for testing.

7.3. Point Cloud

7.3.1. Generic primitive part dataset.

We use the primitive dataset of (Tian et al., 2019) as supervision in Experiment Section 4.3.1. The dataset consists of 200K primitive instances sampled from the primitive templates that are visualized in Figure 6. Examples are sampled from the templates by changing their sizes and placing 2-3 primitives uniformly in random locations. The primitives are represented using a 32x32x32 binary voxel grid.

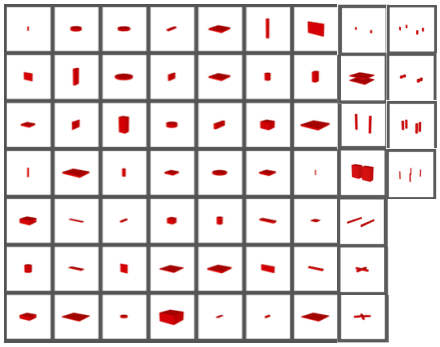

Figure 6. We visualize all the generic primitive templates of (Tian et al., 2019), as you can see, they mainly consist of Cubes, Cuboids, and Discs.

7.3.2. PartNet dataset.

We use the official level-3 train-test split of PartNet (Mo et al., 2019). We randomly split the categories in PartNet into train and test. Our train categories consist of: Chair, Lamp, Clock, Refrigerator, Microwave, Dishwasher, Door and Vase. Our test categories consist of: Table, Storage, Bed, Bottle, Display, Earphone, Faucet, Knife and TrashCan. We use this as the train-test split in Experiment Section 4.3.2. We set the value of number of slots K as 16 for this dataset. We provide all 10K points as input to Slot-TTA and baselines. For evaluation, we calculate ARI segmentation accuracy on occupied points after voxelizing 10K points into a 32x32x32 binary voxel grid.

8. Implementation details

8.1. Posed multi-view 2D RGB images

Training details and computational complexity. We use a batch size of 256 in this setting. We set our learning rate as 10^{-4}. We use an Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. For training, our model takes about 4 days to converge using 64 TPUv2 chips. Test-time adaptation for each example takes about 10 seconds on a single TPUv2 chip. Similarly, a forward pass through our model takes about 0.1 seconds. During training, instead of decoding all the pixels, we decode only a sample of them. Specifically, we randomly pick 1024-pixel locations for each example in the batch during each iteration of training. During test-time adaptation, instead of uniformly sampling pixel locations, we use an error-weighted sampling strategy which we describe below.

Inputs. During training and test-time adaptation, our model takes in as input multi-view RGB images along with their ground-truth egomotion. For each scene, we randomly select four input and five target views. We use a resolution of 128x128 for our input and target images.

Encoder. Here we follow the original implementation of OSRT (Sajjadi et al., 2022b). The model encodes each input image I_i, its camera extrinsic and intrinsics into a set representation via a shared CNN and transformer backbone. Specifically, the CNN outputs a feature grid for each image conditioned on the camera extrinsic and intrinsics, which are then flattened into a set of flat patch embeddings. The patch embeddings are then processed by a transformer that outputs a set of latent embeddings. The latent embeddings have a dimensionality of 1535. The CNN consists of 3 blocks of convolutions, with a ReLU activation after each convolution. The transformer contains 5 blocks of Multi-Head Self-attention.

Slot Attention. The latent embeddings from the encoder are then mapped to a Slot Attention module. We use the original implementation by (Locatello et al., 2020), however instead of initializing the slots from a multi-variate gaussian we have them as learnable embedding vectors. We keep our slot vectors dimensionality as 1536. We set the number of slots as 32 in this setting.

Decoder. We use the broadcast decoder of (Sajjadi et al., 2022a) for decoding the slots to their RGB image conditioned on the target viewpoints. Our slot decoder consists of a 4-layer MLP with a hidden dimensionality of 1536 and ReLU activation. Our target viewpoints are parameterized using 6D light-field parametrization of camera position and normalized ray direction.

Error-conditioned pixel sampling To accelerate test-time adaptation, we sparsely sample a subset of pixels from the target images, where we prioritize the pixels with a high reconstruction error. To this end, we calculate the reconstruction error over all pixels and apply a Softmax with a temperature $\tau = 0.01$ along the pixel dimension.

8.2. 3D point clouds

Training details and computational complexity. We use a batch size of 16 for point cloud input. We set our learning rate as 40^{-4}. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. Our model takes 24 hours (approximately 200k iterations) to converge. Our test-time adaptation per example takes about 1 min (500 iterations). A forward pass through the proposed model takes about 0.15 secs. We use a single V100 GPU for training and inference.

Inputs. We subsample the input point clouds of 10K points to a standard size of 2048 points, before passing it to the encoder.

Encoder. We adopt the point transformer (Zhao et al., 2021) architecture as our encoder. Point transformer encoder is essentially layers of self-attention blocks. Specifically, a self-attention block includes sampling of query points and updating them using their N most neighboring points as key/value vectors. In the architecture, we apply 5 layers of self-attention which look as follows: 2048-16-64, 2048-16-64, 512-16-64, 512-16-64, 128-16-64, 128-16-64. We use the notation of S-N-C, where S is the number of subsampled query points from the point cloud, N is the number of neighboring points and C is the feature dimension. We thus get an output feature map of size 128×64.

Decoder. We obtain point occupancies by querying the slot feature vector $slot_k$ at discrete locations (x, y, z) specifically $o_{x,y,z} = \mathrm{Dec}(slot_k, (x, y, z))$. The architecture of Dec is similar to that of (Lal et al., 2021). Given $slot_k$, which is one of the slot feature vector. We encode the coordinate (x, y, z) into a 64-D feature vector using a linear layer. We denote this vector as z. The inputs $slot_k$ and z are then processed as follows: $out_k = RB_i(z + FC_1(slot_k)) \cdots)$ We set $i = 3$. FC_i is a linear layer that outputs a 64 dimensional vector. RN_i is a 2 layer ResNet MLP block (He et al., 2016). The architecture of ResNet block is: ReLU, 64-64, ReLU, 64-64. Here, $i - o$ represents a linear layer, where i and o are the input and output dimension. Finally out_k is then passed through a ReLU activation function followed by a linear layer to generate a single value for occupancy.

8.3. Baselines

Mask2former (Cheng et al., 2021) Mask2former is a recent state-of-the-art 2D RGB segmentation network, that scales transformer-based DETR (Carion et al., 2020) for the task of segmentation. They improve DETR's transformer decoder by adding masked and multi-scale attention, which helps them achieve SOTA results on panoptic, instance and semantic segmentation on the COCO dataset. We use their publicly available code to train on the MultiShapeNet-Hard dataset. We use a batch size of 256 and train their network on 8 V100s GPUS for four days until convergence. We set the number of slots in their network as 32, similar to our model.

Mask2former-BYOL Following the implementation of MT3 (Bartler et al., 2022), we add a byol head on top of the slot vectors Mask2former. Specifically, we compress the slot vectors into a single vector also commonly known as <CLS TOKEN> in ViT(Dosovitskiy et al., 2020). We then follow the implementation of MT3 where add a BYOL head

on top of this vector. We use all the augmentations originally used by Mask2former for computing the non-contrastive loss.

Mask2former-Recon Similar to Slot-TTA, we add an RGB decoder on top of the slot vectors of Mask2former. Specifically, we use the same implicit broadcast decoder and alpha composting of Slot-TTA to predict the scene RGB. Note that we only predict the RGB and not the segmentation mask from this decoder.

Mask3D. (Schult et al., 2022) Mask3D is a 3D re-implementation of Mask2former (Cheng et al., 2021)(a state-of-the-art object 2D segmentation method) for the task of instance segmentation. Mask3D doesn't officially show any results on PartNet dataset, so we adapt their code to fit the resolution of PartNet dataset while keeping their core architecture the same.

Mask3D-Recon We follow the same design choice as Mask2former-Recon, however, we add the reconstruction decoder to Mask3D instead of Mask2former.

Shape2Prog (Tian et al., 2019) Shape2Prog is a shape program synthesis method that is trained supervised to predict shape programs from object 3D point clouds. Shape2Prog introduced two synthetically generated datasets that helped the model parse 3D pointclouds from ShapeNet (Chang et al., 2015) into shape programs without any supervision: i) Generic primitive set (Figure 6) we discussed earlier in which they use to pre-train their part decoders. Shape2Prog assumes access to a Synthetic whole shape dataset of chairs and tables generated programmatically alongside its respective ground-truth programs. Their model requires supervised pre-training on the dataset of synthetic whole shapes paired with programs. Inorder to maintain the OOD shift, we don't assume access to synthetic whole shapes dataset, however instead we train their encoder to predict programs/segment multiple instances of primitive parts. We use their open-sourced code for comparision with our model. We change the value of number of blocks similar to the number of slots in our model.

PQ-Nets. (Wu et al., 2020) PQ-Nets is a sequential encoder-decoder architecture, that takes 3D point cloud as input and sequentially encodes it into multiple 1D latents which are then decoded to part point clouds. It achieves this decomposition by pre-training their decoder to predict part point clouds. We use their open-sourced architecture and code for comparision with our model. We train their model using our datasets from scratch. We change the value of number of slots in their model based on the maximum number of parts in the dataset.

9. Additional Experiments

9.1. Segmenting RGB images in multi-view scenes

Method	in-dist (5-7 instances)		out-of-dist (16-30 instances)	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Slot-TTA-SlotMixer_Decoder	**0.94**	0.89	0.65	0.72
Slot-TTA-SRT_Decoder	0.92	0.88	0.60	0.63
Slot-TTA-tta_All_param	N/A	0.92	N/A	0.82
Slot-TTA-tta_Norm_param	N/A	0.94	N/A	0.79
Slot-TTA-tta_Slot_param	N/A	0.94	N/A	0.76
Slot-TTA w/o Weighted_Sample	N/A	0.93	N/A	0.81
Slot-TTA (Ours)	0.92	**0.95**	**0.70**	**0.84**

Table 5. **Instance Segmentation ARI accuracy** (higher is better) in the in-distribution test set of 5-7 object instances and out-of-distribution 16-30 object instances.

Method	in-dist (ShapeNet categories)		out-of-dist (GSO categories)	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Mask2Former	0.93	N/A	**0.93**	N/A
Mask2Former-BYOL	0.93	0.95	0.92	0.93
Mask2Former-Recon	0.93	0.92	0.92	0.91
Slot-TTA (Ours)	0.92	**0.95**	0.92	**0.95**

Table 6. **Instance Segmentation ARI accuracy** (higher is better) in the in-distribution test set of ShapeNet object categories(Chang et al., 2015) and out-of-distribution test set of GSO object categories (Downs et al., 2022).

In Table 6, we tested our model on a different distribution shift. In the test set instead of increasing the number of instances in the scene in Table 1, we introduced instances from new object categories. Specifically the MSN (Sajjadi et al., 2022b) train-set consists of ShapeNet object categories(Chang et al., 2015) (Tables, Chairs etc), whereas the new test-set consists of Google Scanned Object (Downs et al., 2022) (GSO) categories (Shoes, Stuffed toys etc).

In Table 7, we tested our model on CLEVR dataset of (Johnson et al., 2017) using the same train-test setup as Section 4.1, where in the train set we use 4-7 objects and in the test set we use 7-10 objects. As can be seen Slot-TTA achieves close to perfect results out-of-dist shift after TTA.

Method	in-dist (4-7 instances)		out-of-dist (7-10 instances)	
	Direct Infer.	with TTA.	Direct Infer.	with TTA.
Slot-TTA (Ours)	0.96	**0.97**	0.92	**0.97**

Table 7. **Instance Segmentation ARI accuracy** (higher is better) in the in-distribution test set of 4-7 object instances and out-of-distribution 7-10 object instances of CLEVR dataset.

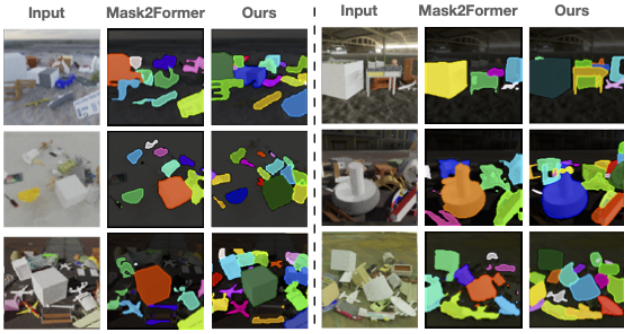

Figure 7. We compare Mask2former with Slot-TTA-with TTA on out-of-dist test set of 16-30 object instances, following the setup of Section 4.1.

We conduct various ablations of Slot-TTA considering the same setting as 4.1 in Table 1. In Figure 8, we show additional qualitative results comparing Slot-TTA-DirectInfer and Slot-TTA-with TTA. In Figure 7, we qualitatively compare Mask2former with Slot-TTA.

(i) We ablate different decoder choices in the topmost section where instead of using the broadcast decoder we use the Scene representation transformer (SRT) decoder (Sajjadi et al., 2022b) which we refer to as **Slot-TTA-SRT_Decoder** or the SlotMixer decoder (Sajjadi et al., 2022a), referred to as **Slot-TTA-SlotMixer_Decoder**.

(ii) We ablate what parameters to adapt at test time. As it's unclear since TENT (Wang et al., 2020) optimizes BatchNorm or LayerNorm parameters, but TTT (Sun et al., 2020) optimizes the shared parameters between the SSL and the task-specific branch, which in our case will be all the parameters in the network. In Table 5, **Slot-TTA-tta_All_param** is when we adapt all the network parameters, **Slot-TTA-tta_Norm_param** adapts only the Layer or BatchNorm parameters and **Slot-TTA-tta_Slot_param** adapts only the learnable slot embeddings. We find that optimizing only the encoder parameters works the best for our setting.

(iii) Further, we ablate error-conditioned pixel sampling where **Slot-TTA w/o Weighted_Sample** refers to our model that uses uniform sampling instead of the error weighted sampling.

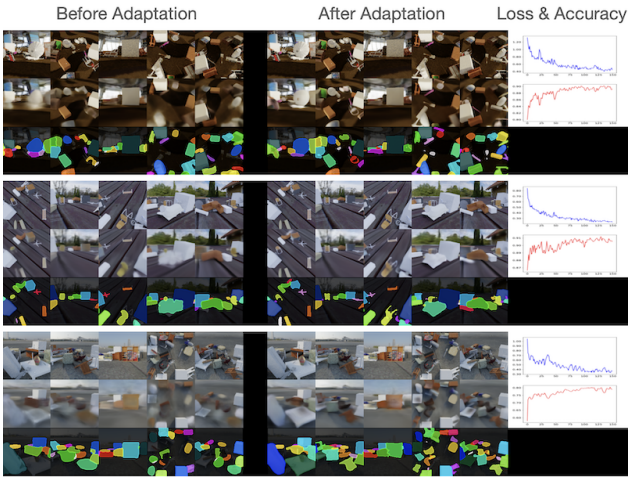

Figure 8. On the left, we visualize Slot-TTA-DirectInfer. In the middle, we visualize Slot-TTA-with TTA. In the first row we visualize the ground truth target RGB views. In the second and third row we visualize Slot-TTA predicted target RGB views and their segmentation masks. On the right-most column we visualize the RGB loss and segmentation accuracy during adaptation.

9.2. Decomposing 3D point clouds

We show additional qualitative results for Section 4.3.1 in Figure 9.

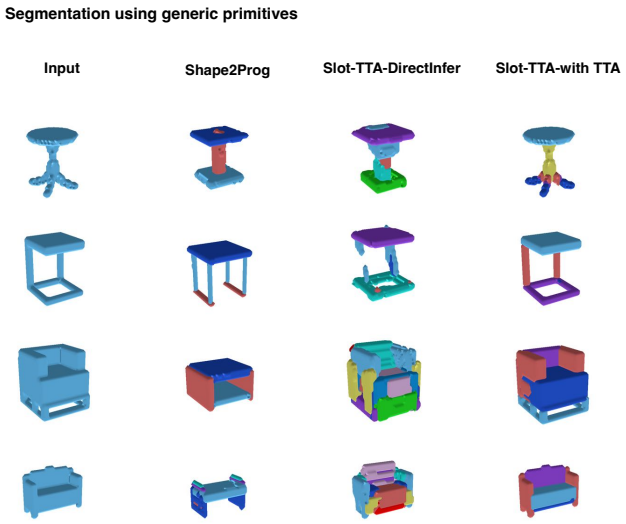

Figure 9. Additional segmentation results on out-of-distribution categories when supervised from generic primitives. Same setting as Section 4.3.1