GENERATING SCENES WITH LATENT OBJECT MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a structured latent variable model that learns the underlying data-generating process for a dataset of scenes. Our goals are to obtain a compositional scene representation and to perform scene generation by modeling statistical relationships between scenes as well as between objects within a scene. To make inference tractable, we take inspiration from visual topic models and introduce an interpretable hierarchy of scene-level and object-level latent variables (i.e., slots). Since generating scenes requires modeling dependencies between objects, we cannot make a bag-of-words assumption to simplify inference. Moreover, assuming that slots are generated with an autoregressive prior requires decomposing scenes sequentially during inference which has known limitations. Our approach is to assume that the assignment of objects to slots during generation is a deterministic function of the scene latent variable. This removes the need for sequential scene decomposition and enables us to propose an inference algorithm that uses orderless scene decomposition to indirectly estimate an ordered slot posterior. Qualitative and quantitative analysis establishes that our approach successfully learns a smoothly traversable scene-level latent space. The hierarchy of scene and slot variables improves the ability of slot-based models to generate samples displaying complex object relations. We also demonstrate that the learned hierarchy of representations can be used for a scene-retrieval application with object-centric re-ranking.

1 INTRODUCTION

This paper considers the problem of learning the data-generating process for multi-object scenes. We want to train likelihood-based deep generative models that can encode scenes into a representation hierarchy of object-centric and scene-level latent variables and support novel scene generation. A scene representation captures statistical relationships between scenes and can model complex interactions between objects (Jiang & Ahn, 2020). Object-level representations increase compositionality, interpretability, and robustness of the model to out-of-distribution (OOD) data (Dittadi et al., 2021).

Much progress has been made on models that encode scenes into sets of object-centric distributed representations, or *slots* (Greff et al., 2017; Burgess et al., 2019; Greff et al., 2019; Engelcke et al., 2019; Locatello et al., 2020; Emami et al., 2021; Engelcke et al., 2021). These models have demonstrated strong generalization and object-level disentanglement and have been applied successfully to object-centric planning for visual model-based reinforcement learning (Veerapaneni et al., 2019). However, we understand little about how to jointly decompose scenes and sample novel scenes from noise with slot-based models. The limited prior work on this problem has approached it by factorizing the joint distribution over slots using the chain rule into a sequence of autoregressive distributions (Engelcke et al., 2019; 2021). Fixing the slot dependency graph vastly simplifies inference by removing the need to integrate out all ways to order the slots. However, these models do not possess a scene-level representation and are therefore limited in their ability to capture scene-level statistics and higher-order object relations (Jiang & Ahn, 2020). Moreover, inference is challenging because it requires sequentially decomposing the scene into objects without knowing the optimal order to follow. This was first approached with sequential attention that learned arbitrary and biased orderings (Burgess et al., 2019; Engelcke et al., 2019). A recent improvement to this decomposition approach proposes a randomly ordered decomposition algorithm (Engelcke et al., 2021). In principle, however, a *randomly ordered* decomposition provides a weak signal for training an autoregressive slot distribution to *sequentially* compose a scene.

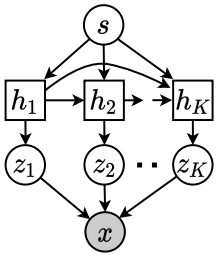


Figure 1: **The hierarchical graphical model for LOMs**. We assume that scenes x are generated from K object-centric slot latent variables $\mathbf{z}_{1:K}$. The slots are independent *conditional* on a causal sequence of deterministic variables (shown in boxes). This causal sequence is predicted from the top-level scene latent variable $\mathbf{s}$. By conditioning each slot on one of the deterministic variables, we provide the order-dependent relational information necessary for scene generation.

In this work, we introduce Latent Object Models (LOMs), a deep generative modeling framework for fitting the data-generating process for scenes with a latent variable hierarchy consisting of a scene-level variable and a set of slot variables. The analogy used to design the hierarchical graphical model is inspired by visual topic models (Hofmann, 1999; Blei et al., 2003; Sivic et al., 2005; Fritz & Schiele, 2008; Tuytelaars et al., 2010):

$$\begin{aligned} \text{documents} &- \text{scenes} \\ \text{topics} &- \text{slots} \\ \text{words} &- \text{percepts} \end{aligned}$$

where a percept is any discrete token that makes up a scene observation, such as a pixel or a 3D point. LOMs have a three-level hierarchy. Each percept is modeled as a K-component mixture, where each component is dependent on one of K intermediate-level slots. Then, the slot variables are dependent on a single top-level scene variable.

Topic models make a "bag-of-words" assumption that topics are orderless within a document. This is also known as *exchangeability*, which is a property of a set of random variables $\{\mathrm{z}_1, \ldots, \mathrm{z}_K\}$ whose joint distribution is invariant under permutation, e.g., $p(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3) = p(\mathbf{z}_2, \mathbf{z}_1, \mathbf{z}_3)$. However, multi-object scene generation requires modeling dependencies between slots. LOMs therefore cannot assume slots are orderless in a scene.

Instead of modeling slot generation as an autoregressive (i.e., stochastic) process, we handle the assignment of objects to slots during ordered generation deterministically. We train a function to take a single sample from the scene prior and predict a causal sequence of *deterministic* variables which we condition each slot prior with (Figure 1). Designing the slot prior to be independent when conditioned on a causal sequence, instead of autoregressive, prevents us from having to deal with deciding an order to follow for sequential scene decomposition during inference. We instead propose an inference algorithm for LOMs that assumes an *auxiliary* generative model which decomposes scenes into an orderless set of slots. This is used to infer a slot-order-invariant scene-level posterior distribution. To obtain an ordered slot posterior for maximum likelihood training, we compute a matching between the means of the orderless slot posterior and ordered slot prior. This matching is then used to permute the orderless posterior and convert it into an ordered distribution.

Contributions: We introduce a latent variable model and variational inference algorithm for modeling the data-generating process for scenes with a latent scene-slot hierarchy. We demonstrate an implementation of this framework that is capable of learning a low-dimensional scene-level manifold that can be smoothly traversed. We also show that generated scenes accurately capture higher-order object relations present in the dataset. We illustrate that the new ability to learn a hierarchical scene representation enables applications such as scene retrieval with object-centric re-ranking.

2 Related Work

We compare LOMs with related deep generative models in Table 1. Likelihood-based deep latent variable models based on the variational autoencoder (VAE) framework (Kingma & Welling, 2013) have recently demonstrated impressive scene generation capabilities for high-resolution images with spatial latent variables split across a hierarchy of stochastic layers (Sønderby et al., 2016; Kingma et al., 2016; Child, 2020; Vahdat & Kautz, 2020). Differently, LOMs model perceptual data using scene and slot latent variables. During inference and generation, each slot is bound to a single entity in the scene which improves interpretability and compositionality (Greff et al., 2020).

Table 1: Latent Object Models introduce the idea of using a scene-slot latent hierarchy to fit the data-generating process. Unlike spatial latents, slots are assigned to scene objects during inference and generation, improving interpretability and disentanglement of the learned representations.

Model	scene latent	spatial latents	object slots
VAE (Kingma & Welling, 2013)	✓		
NVAE (Vahdat & Kautz, 2020)		✓	
GNM (Jiang & Ahn, 2020)	✓	✓	
GENESIS (Engelcke et al., 2019)			✓
GENESIS-v2 (Engelcke et al., 2021)			✓
LOMs (Ours)	✓		✓

Generative Neurosymbolic Machines (GNMs) (Jiang & Ahn, 2020) present the idea of combining a scene-level prior and symbolic latent variables within a hierarchical VAE for multi-object scene generation. However, they make a strong assumption that the symbolic latent variables represent explicit object-level information within each cell of a fixed grid. While this works well for synthetic image datasets with similarly sized objects, LOMs tackle the more challenging problem of learning a *scene-slot* hierarchy. LOMs fall into the category of VAEs that represent scenes as mixture models whose components are generated by a set of slots (Burgess et al., 2019; Greff et al., 2019; Engelcke et al., 2019; 2021; Locatello et al., 2020; Emami et al., 2021). GENESIS (Engelcke et al., 2019) (GEN) and GENESIS-v2 (Engelcke et al., 2021) (GENv2) use an autoregressive slot prior for multi-object scene generation without a scene-level representation. Autoregressive priors rely on sequential scene decomposition to learn a good generation order. A randomly ordered decomposition provides no sequencing information to train the prior. An imposed order would be unnatural and limit performance (Engelcke et al., 2021). LOMs address this difficulty by *indirectly* estimating an ordered decomposition from an orderless decomposition.

Other VAEs with structured latent variables have been proposed for hierarchical scene generation (Anciukevicius et al., 2020; von Kügelgen et al., 2020) but make simplifying independence assumptions in their graphical models that prevents them from scaling to more complex scenes. The SetVAE (Kim et al., 2021) learns to generate set-structured data such as point clouds with a tree-structured hierarchy of sets of latent variables. However, unlike slot-based models, its latent variables are not explicitly trained to segregate the scene representation.

Generative adversarial methods for compositional scene synthesis (van Steenkiste et al., 2020; Nguyen-Phuoc et al., 2020; Liao et al., 2020; Ehrhardt et al., 2020; Niemeyer & Geiger, 2021; Hudson & Zitnick, 2021) avoid the computational challenges of Bayesian inference that LOMs address but lack the ability to *decompose* scenes into an interpretable hierarchy of representations.

3 Latent Object Models

3.1 Generative Model

We restrict the focus of this paper to unlabeled datasets $\mathcal{D}$ of i.i.d. samples of images $\mathbf{x} \in \mathbb{R}^{N \times C}$, $N = HW$, for clarity. Our goal is to fit the underlying data-generating process, which we propose to approach by using a three-level latent hierarchy. To that end, we define a scene-level latent variable $\mathbf{s} \in \mathbb{R}^{D_s}$, K object-level latent variables $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_K \in \mathbf{z}_k \in \mathbb{R}^{D_z}$ (or *slots*, which we abbreviate with $\mathbf{z}_{1:K}$, and pixel observations $\mathbf{x}_i \in \mathbb{R}^C$, $i = 1, \ldots, N$. The joint distribution over observed and latent variables can be factorized as follows:

$$p_\theta(\mathbf{x}, \mathbf{z}_{1:K}, \mathbf{s}) = p(\mathbf{s}) p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s}) p_\theta(\mathbf{x} \mid \mathbf{z}_{1:K}). \tag{1}$$

Since Bayesian inference for the posterior is intractable and we would like to parameterize these distributions with deep neural networks, we adopt the VAE framework for approximate Bayesian inference (Kingma & Welling, 2013; Rezende et al., 2014).

The scene-level prior $p(\mathbf{s})$ is left as a standard Gaussian in this work for simplicity. However, the slot prior is more complicated, since generating coherent scenes requires modeling relational information

between slots. Unlike topic models, we cannot assume slots are i.i.d. when conditioned on $\mathbf{s}$. We instead assume a *fixed and deterministic order* on the slots informed by $\mathbf{s}$:

$$p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s}) = \prod_{k=1}^{K} p_\theta(\mathbf{z}_k \mid \boldsymbol{h}_k), \text{ where } \boldsymbol{h}_k = f_\theta(\boldsymbol{h}_{1:k-1}, \boldsymbol{s}), \tag{2}$$

where f_θ is a deterministic function that transforms a sample $\boldsymbol{s}$ from the scene prior and $\boldsymbol{h}_1 = f_\theta(\boldsymbol{s})$. Each $\boldsymbol{h}_k$ contains the order-aware information about which object the kth slot should generate. By designing the slot prior such that the kth slot indirectly depends on slots $\mathbf{z}_{1:k-1}$ through $\boldsymbol{h}_k$, the joint distribution over slots factorizes as a product of Gaussian conditionally independent (but not identically distributed) distributions. Notice that if we replaced each $\boldsymbol{h}_k$ with $\boldsymbol{s}$ the LOM generative model becomes a bag-of-words model.

The choice of image-likelihood strongly influences the complexity of the learned scene representation. We use a slot order-invariant Gaussian likelihood with the mean given by a sum of the product of the image-shaped decoded masks m_k and components x_k:

$$p_\theta(\mathbf{x} \mid \mathbf{z}_{1:K}) = \prod_{i=1}^{N} \mathcal{N}\left(\sum_{k=1}^{K} m_{i,k} x_{i,k}, \sigma^2\right). \tag{3}$$

The Gaussian likelihood is well-suited for generating complex scenes since it does not penalize solutions that use multiple slots to fit parts of the background (Locatello et al., 2020; Emami et al., 2021). By contrast, the Mixture-of-Gaussians likelihood used by some slot-based models (Burgess et al., 2019; Greff et al., 2019; Engelcke et al., 2019; 2021) heavily penalizes the model for splitting parts of the scene across multiple slots. A scene can be sampled from a LOM as follows:

1. Sample $\mathbf{s}$ from the scene prior $p(\mathbf{s})$.
2. Compute $\boldsymbol{h}_1 = f_\theta(\mathbf{s}), \boldsymbol{h}_2 = f_\theta(\boldsymbol{h}_1, \mathbf{s}), \ldots, \boldsymbol{h}_K = f_\theta(\boldsymbol{h}_{1:K-1}, \mathbf{s})$.
3. Sample one slot from each of the $p_\theta(\mathbf{z}_1 \mid \boldsymbol{h}_1), p_\theta(\mathbf{z}_2 \mid \boldsymbol{h}_2), \ldots, p_\theta(\mathbf{z}_K \mid \boldsymbol{h}_k)$.
4. Render the image by decoding the K slots in parallel into masks m_k and pixel components x_k and then aggregating them with Equation 3.

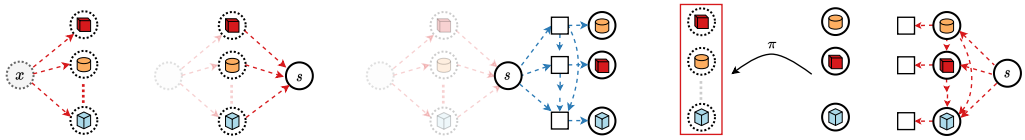
3.2 Inference

We have introduced how LOMs model the scene generation process. Now, we turn our attention to the inference algorithm for the slot and scene latent variables. We want to avoid directly estimating an ordered slot posterior using sequential scene decomposition since an optimal ordering to follow is unknown and difficult to learn without supervision. Our inference algorithm, visualized in Figure 2, uses an *auxiliary* generative model that assumes slots are orderless in scenes to indirectly obtain an ordered slot posterior. Orderless scene decomposition uses a randomized iterative assignment algorithm to bind percepts to the slot variables (Greff et al., 2019; Locatello et al., 2020; Engelcke et al., 2021; Emami et al., 2021). An important consequence of using an orderless slot posterior is that we can define a slot-order-invariant scene-level posterior:

$$q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K})), \tag{4}$$

where g_ϕ is a permutation-invariant function. Orderless slots are denoted by $\mathbf{o}_{1:K}$. Each step of LOM inference is summarized here and visualized in Figure 2:

1. **Estimate orderless slot posterior:** Compute and sample from the orderless posterior q_ω estimated by the auxiliary model (with inference network parameters ω): $\mathbf{o}_{1:K} \sim q_\omega(\mathbf{o}_{1:K} \mid \mathbf{x}) = \prod_{k=1}^{K} q_\omega(\mathbf{o}_k \mid \mathbf{x})$.
2. **Estimate scene posterior:** Compute and sample from the scene-level posterior: $\mathbf{s} \sim q_\phi(s \mid g_\phi(\mathbf{o}_{1:K}))$.
3. **Estimate scene-conditioned ordered slot prior:** Compute the ordered prior $p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s})$ using a single sample $\mathbf{s}$ and the causal function f_θ.
4. **Align the orderless slot posterior:** Compute a matching π between the K means from $p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s})$ and the K means from $q_\omega(\mathbf{o}_{1:K} \mid \mathbf{x})$, e.g., by minimizing the pairwise $L2$ distance, then permute the order of the slot posterior means by π to get $\mathbf{o}_{\pi(1):\pi(K)}$.



(a) Step 1. (b) Step 2. (c) Step 3. (d) Step 4. (e) Step 5.

Figure 2: Inference for Latent Object Models. Circles are slot latents. Dotted edges signify variables from the auxiliary orderless model and solid edges are variables from the LOM model. Boxes are deterministic variables. In steps 1-2 we infer an orderless slot posterior and the scene posterior. In Steps 3 we use the scene posterior to generate slots via the deterministic variables. In Step 4 we align the orderless slot posterior to the prior. Finally in step 5 we estimate new order-aware scale parameters for the aligned slot posterior through another set of deterministic variables.

5. **Estimate an ordered slot posterior:** Compute new order-aware scale parameters $\sigma^{\mathbf{z}}_{\pi(k)}$ for the permuted slot posteriors to obtain *ordered* slot posteriors $q_\phi(\mathbf{z}_{\pi(k)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x})$.

Here $\sigma^{\mathbf{z}}_{\pi(k)} = w_\phi(\mathbf{h}'_{\pi(k)})$, with $\boldsymbol{h}'_{\pi(k)} = f'_\phi(\mathbf{o}_{\pi(1):\pi(k-1)}, \mathbf{s})$ and w_ϕ is a linear transformation. We introduce another causal function f'_ϕ to transform a sample from the scene posterior and the permuted orderless slots $\mathbf{o}_{\pi(1):\pi(K)}$ into the sequence $\mathbf{h}'_{\pi(1):\pi(K)}$. A simple and robust choice for computing π is an $O(K)$ greedy alignment (Algorithm 1 in the appendix) that sequentially finds the best match for each slot. Each $\mathbf{h}'_{\pi(k)}$ provides the ordered slot posterior with its order-aware scale parameter:

$$\prod_{k=1}^{K} q_\phi(\mathbf{z}_{\pi(k)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x}) = \prod_{k=1}^{K} \mathcal{N}(\mu^{\mathbf{o}}_{\pi(k)}, (\sigma^{\mathbf{z}}_{\pi(k)})^2 I). \tag{5}$$

Here, $\mu^{\mathbf{o}}_{\pi(k)}$ is just the kth mean of the permuted orderless slot posterior.

Architectures for f_θ,f'_ϕ and g_ϕ: In this work we use a Transformer (Vaswani et al., 2017) with multiple blocks of single-head *causal* self-attention (SA) to implement f_θ. The first token in the input is a sample $\mathbf{s} \sim p(\mathbf{s})$ and the next K tokens are a shared trainable embedding initialized with zeros. We implement f'_ϕ using nearly the same causal Transformer architecture as f_θ. The differences are that the first token is now a sample $\mathbf{s}$ from the scene posterior and instead of using a trainable embedding as input for the other tokens we pass the K permuted orderless slots $\mathbf{o}_{\pi(1):\pi(K)}$. For g_ϕ we use a permutation-invariant Transformer with multiple blocks of residual SA and no positional encoding or causal masking. We aggregate the set output with a global sum pooling layer, which we find essential for capturing complex object relations in scenes. Details are provided in the appendix.

3.3 Training

LOMs can be seen as a hybrid model that jointly trains an orderless VAE and an ordered VAE. Gradients are prevented from flowing back to the orderless VAE from the ordered VAE during training (e.g., via `stop_grad`$(\mathbf{o}_{1:K})$) to ensure that orderless scene decomposition remains unbiased. To estimate $q_\omega(\mathbf{o}_{1:K} \mid \mathbf{x})$, we can use any slot-based VAE with orderless decomposition. In this work, we choose EfficientMORL (Emami et al., 2021) (EMORL) since it can be used without modification for our purposes. EMORL is a hierarchical VAE with a data-dependent i.i.d. slot prior that combines efficient bottom-up stochastic slot attention with top-down iterative refinement.

Objective: The ELBO for training LOMs is maximized jointly with the ELBO for the auxiliary orderless generative model. The LOM ELBO resembles the same ELBO used by previous deep generative models with a similar topic-model-inspired hierarchical factorization (Edwards & Storkey, 2016). We show it here presented as a sum of three terms and defer its derivation to the appendix.

First is a log-likelihood (i.e., reconstruction) term:

$$\mathcal{L}_{\text{LL}} = \mathbb{E}_{q_\phi(\mathbf{s}|g_\phi(\mathbf{o}_{1:K}))} \left[\mathbb{E}_{q_\phi(\mathbf{z}_{\pi(1):\pi(K)}|\boldsymbol{h}'_{\pi(1):\pi(K)},\mathbf{x})} \left[\log p_\theta(\mathbf{x} \mid \mathbf{z}_{1:K}) \right] \right]. \tag{6}$$

Then, we have a reverse Kullback-Leibler (KL) term for the slots, which can be factorized as a sum over K since the slot posterior and prior are conditionally independent and aligned via π:

$$\mathcal{L}_{\text{slotKL}} = -\mathbb{E}_{q_\phi(\mathbf{s}|g_\phi(\mathbf{o}_{1:K}))}\left[\sum_{k=1}^{K} D_{KL}\big(q_\phi(\mathbf{z}_{\pi(k)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x}) \parallel p_\theta(\mathbf{z}_k \mid \boldsymbol{h}_k)\big)\right]. \tag{7}$$

Finally, we have a scene-level reverse KL term:

$$\mathcal{L}_{\text{sceneKL}} = -D_{KL}\big(q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K})) \parallel p(\mathbf{s})\big). \tag{8}$$

Altogether, we maximize $\mathcal{L}$, the sum of these three terms and the auxiliary orderless model's ELBO:

$$\mathcal{L} = \mathcal{L}_{\text{orderless}} + \mathcal{L}_{\text{LL}} + \mathcal{L}_{\text{slotKL}} + \mathcal{L}_{\text{sceneKL}}. \tag{9}$$

Impact of maximizing the ELBO on f_θ: Intuitively, maximizing the ELBO will lead f_θ to seek out *consistent* generation orders that are easy to match to the orderless posterior, hence minimizing $\mathcal{L}_{\text{slotKL}}$. In practice, we observe that f_θ often learns to generate parts (or all) of the background with the same slot(s) and will generate the foreground objects using the same subset of slots (Figure 3).

Optimization: To dynamically balance the reconstruction and KL terms for both VAEs, we use GECO (Rezende & Viola, 2018). GECO reformulates maximization of the ELBO into a constrained minimization of the KL terms, where the constraint dictates a desired reconstruction quality C. Please refer to the appendix for a discussion on choosing C as it is crucial for obtaining good performance. All other implementation details including architecture description, hyperparameters, and compute requirements are also deferred to the appendix.

4 Experiments

Datasets: We use three synthetic multi-object datasets: *Objects Room* and *CLEVR6* from the standard Multi-Object Dataset (Kabra et al., 2019) and *ShapeStacks* (Groth et al., 2018). All three datasets consist of rendered images of 3D scenes containing 3+ objects and have variable illumination, shadow, and camera perspectives. Successful object-centric scene generation requires reasoning about local object relations (e.g., occlusion in CLEVR, placement of objects relative to the floor in Objects Room) as well as global relations (e.g., stacking multiple blocks in ShapeStacks). We focus on synthetic scenes in this work since we wish to first establish correctness of the proposed LOM framework before tackling real-world scenes.

Baselines: For our main evaluation, we focus our comparison on the state-of-the-art method for object-centric scene generation, GENESIS-v2 (GENv2). Using code released by the authors, we train GENv2 on CLEVR6 as these results were not reported in their paper. GENv2 has a well-tuned schedule for trading off the reconstruction and KL terms and we are able to achieve sharp reconstructions and high-quality segmentation on all datasets. We also train a variant of GENv2 with the same Gaussian likelihood as the LOM on all three datasets. On CLEVR6 we report results for a down-sized NVAE that we trained using the official code release. The NVAE is a powerful VAE baseline that only uses spatial latents. Results for GENESIS (GEN) on Objects Room and ShapeStacks are from Engelcke et al. (2021). See Section 2 for a discussion of these models and the appendix for baseline training details and qualitative results.

Metrics: The Frechet Inception Distance (**FID**) (Heusel et al., 2017) is a standard metric for quantifying the ability of a model to generate images similar to the training set. We follow Engelcke et al. (2021) and use 10K real and generated samples to calculate this score. However, we found that the FID score heavily penalizes blurriness in the generated images while ignoring structural accuracy. Therefore, inspired by the evaluation of GNM (Jiang & Ahn, 2020), on ShapeStacks we generate 100 images with each model and count the fraction of images containing a stack where all blocks in the stack are touching each other (structural accuracy, or **S-Acc**). We compute disentanglement **DCI** scores (Eastwood & Williams, 2018) on CLEVR6, which provides ground truth latent factors, to compare the ability to jointly achieve high-quality generation and object-centric disentanglement between NVAE and slot-based models.

4.1 Exploring the Scene-level Latent Space

We first verify that LOMs are able to fit the data-generating process for each dataset. Random unconditional samples from the prior are visualized in Figure 3. We can observe by inspecting the

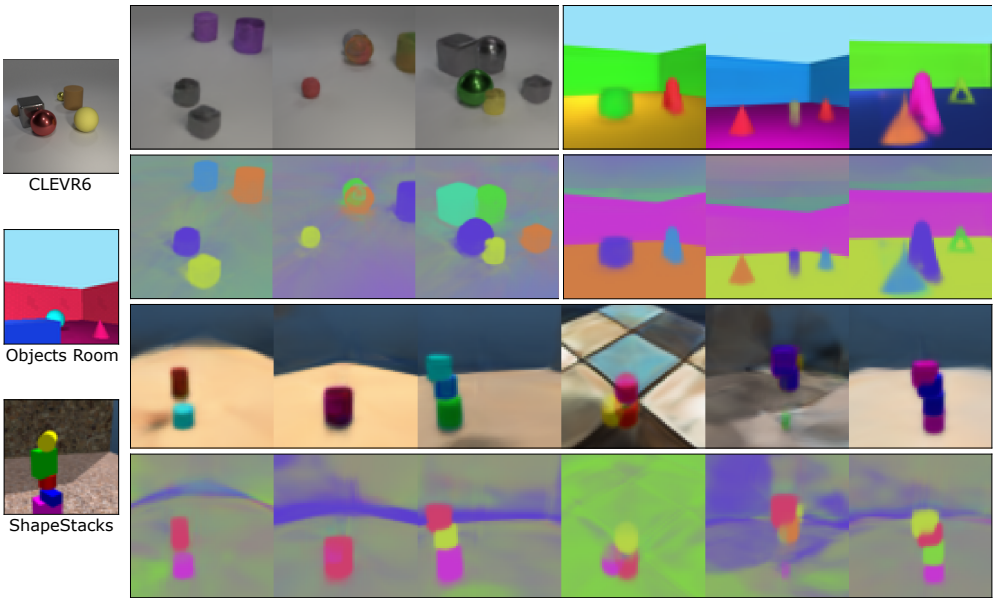


Figure 3: Unconditional generation from the LOM prior. The generated masks are visualized beneath each sample. Mask color corresponds with slot number. Dataset examples are shown on the left. *Top left)* CLEVR6. *Top right)* Objects Room. *Bottom)* ShapeStacks.

generated masks in Figure 3 that the model has learned a fixed strategy for how it generates objects with the slots for each dataset. For example, in Objects Room, the back wall is always generated using the same (e.g., first) slot. In ShapeStacks, a group of 3-4 slots are used to generate blocks. The ability to learn to consistently generate certain objects with the same slots (by virtue of the deterministic causal variables) is a key advantage our design. Concretely, we show that this prevents generating nonsensical images that have floating objects or missing/extra parts of the background in a controlled ablation in Section 4.2. Note that we slightly scale down the standard deviations in the prior using a temperature of $\tau = 0.8$ when sampling, which is common practice to help keep samples in high probability regions for hierarchical VAEs (Vahdat & Kautz, 2020).

It is also helpful to inspect the learned scene-level latent space by spherically interpolating between two random samples from the prior (Figure 4). The intermediate generated images resemble the training data. The images smoothly change camera perspective, colors, shapes, and relative positions of objects. This suggests that LOMs have successfully learned a low-dimensional scene manifold (we use $|\mathbf{s}| = 128$ and $|\mathbf{z}| = 64$) that captures scene similarity in part based on object-centric attributes. More qualitative examples are provided in the appendix.

4.2 Ablations and Variations

We ablate various aspects of LOMs on a reduced resolution variant of Objects Room with five training runs per model, including expressivity of the Transformers by varying the number of residual blocks, the impact of computing new order-aware scale parameters, and sharing parameters between f_θ and f'_θ. Details and KL/FID scores for all are in the appendix. We focus our discussion here on two autoregressive prior variants of the LOM. The first (LOM-u) uses an unconditional RNN prior (like GEN and GENv2) and the other (LOM-s) conditions the first slot of an RNN prior on the scene latent $\mathbf{s}$. Both are trained by adding the proper auxiliary KL terms to $\mathcal{L}_{\text{orderless}}$. We found that adding the scene latent $\mathbf{s}$ greatly improved the FID score considerably but both models produce relatively poor results. As can be seen in Figure 5a (top right), the LOM-s can generate nonsensical scenes, for example with floating objects and multiple walls. This is because the autoregressive prior is trained to generate slots in a random order which leads to such errors. We observed the same phenomena for the GENv2 model we trained with Gaussian likelihood on Objects Room (Figure 5a). This also explains the artifacts visible in ShapeStacks samples from GENv2 (bottom images in Figure 5a)—there are multiple backgrounds being generated and blended together.

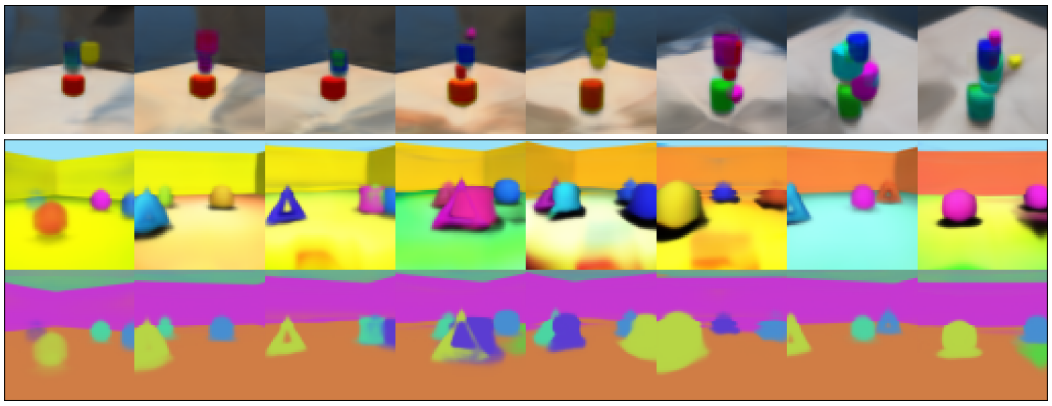


Figure 4: **LOMs can generate interpolated scenes between two random samples from the scene prior by smoothly changing individual object and scene-level attributes**. We use spherical interpolation to generate the intermediate latents between two random samples from $p(\mathbf{s})$. This suggests the LOM has learned a good fit to the underlying data-generating process.

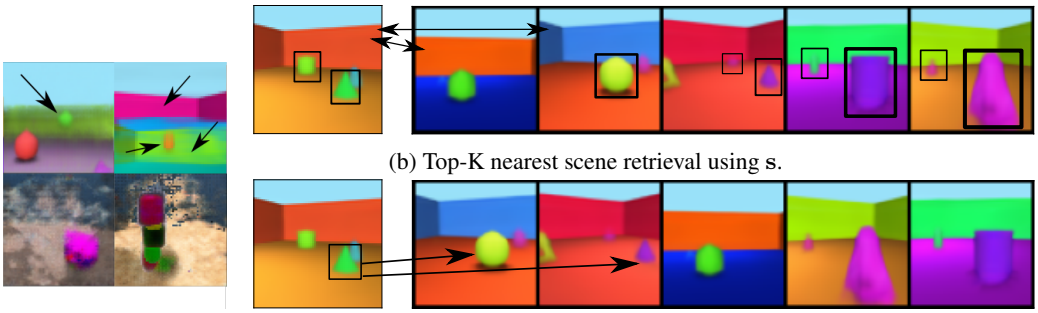
Table 2: Metrics are averaged over three training runs. ShapeStacks S-Acc is computed for the best performing model by FID and with pre-trained models released by Engelcke et al. (2021).

	Objects Room	**ShapeStacks**		**CLEVR6**	
Model	FID↓	FID↓	S-Acc (%)↑	FID↓	Disent.↑
Spatial latents					
NVAE	-	-	-	$28.3_{\pm 4}$	$0.19_{\pm 0.0}$
Slot latents					
GENESIS (Engelcke et al., 2021)	$62.8_{\pm 3}$	$186.8_{\pm 18}$	0	-	-
GENv2 (Engelcke et al., 2021)	$52.6_{\pm 3}$	$112.7_{\pm 3}$	0.59	-	-
GENv2 (MoG)	-	-	-	$61.4_{\pm 2}$	$0.56_{\pm 0.03}$
GENv2 (G)	$87.6_{\pm 4}$	$115.3_{\pm 6}$	0.42	$61.0_{\pm 3}$	$0.54_{\pm 0.01}$
Scene and slot latents					
LOM (–scene sum pool)	-	$139.1_{\pm 10}$	0.40	-	-
LOM	$67.4_{\pm 3}$	$132.2_{\pm 15}$	**0.76**	$46.7_{\pm 5}$	$0.44_{\pm 0.02}$

4.3 Quantitative Evaluation

First, we examine whether adding a scene-level representation enables generating scenes that accurately reflect higher-order multi-object relations in the data compared to models that only use slots. Results are presented in Table 2. LOM achieves the highest ShapeStacks S-Acc by a margin of 17%. Random samples from GEN's prior are mostly unintelligible on this dataset, hence the score of 0. Removing sum pooling in the scene encoder g_ϕ decreases LOM accuracy by 36%, highlighting the essential role it plays in encoding relations involving many objects. Replacing sum pooling with a more expressive function should further increase S-Acc as our implementation still somewhat struggles to generate towers with 4+ blocks.

Next, we compare the generation quality of the slot-based models and LOM. On CLEVR6, LOM achieves the lowest FID. This dataset has a plain background so GENv2 receives no particular advantage in FID score from its superior reconstruction quality and low KL. This suggests LOM achieves a better fit to the data-generating process on this dataset. GENv2 achieves lower FID scores on ShapeStacks since it can generate the background texture with higher fidelity, unlike our LOM implementation. This is despite the fact that LOM demonstrates higher S-Acc and its samples do not have artifacts from generating multiple backgrounds. Adapting GenV2's randomly ordered decomposition algorithm into the LOM framework should be possible by pairing it with an i.i.d. prior, which we believe could lead to sharper generation quality. GENv2 and GEN achieve slightly lower FID scores than LOM on Objects Room (as reported in Engelcke et al. (2021)). This dataset is less challenging than ShapeStacks for GEN, which it totally fails on due to its sequential decomposition



(a)

(b) Top-K nearest scene retrieval using $\mathbf{s}$.

(c) Re-ranking results based on similarity to the green cone.

Figure 5: a) **Orderless decomposition with autoregressive generation leads to poorly structured scenes and artifacts.** Top right image is our LOM with the scene-conditioned autoregressive prior and the other three are from the GENv2 baseline. **Hierarchical retrieval example**. b) The top 5 retrieved scenes based on Euclidean distance to the query $\mathbf{s}$ (image on left) show similar: camera angle, wall and floor colors, and relative positions of multiple objects. We highlight some of these with boxes and arrows. c) We use the slot for the green cone (boxed) to re-rank the results based on matching against the slots for each retrieved scene. Notice that the top two scenes now have objects that share a similar size, location, or shape with the green cone (arrows).

inference. As the authors of GENv2 describe, occasionally its prior can learn a fixed generation order (e.g., always generating the background with the last slot), which enables consistent generation of coherent scenes and therefore lower FID scores. However, this is not always the case, and the GENv2 model we train with Gaussian likelihood achieves a higher FID than LOM because it cannot reliably compose the separately segmented wall, floor and objects together.

Finally, we observe that despite the NVAE's ability to easily fit the CLEVR6 dataset and generate high-fidelity samples, its relatively low disentanglement scores compared to slot-based models shows that its spatial latents are encoding local low-level image statistics rather than interpretable object-centric information (full DCI scores provided in the appendix). We expect that LOM might close the FID gap with NVAE (28.3 vs. 46.7) with a more carefully tuned implementation.

4.4 Hierarchical Scene Retrieval with Object-based Re-ranking

We provide an illustrative example of how the representations inferred by LOMs can be used for a hierarchical scene retrieval application. The basic idea is to use the learned scene-level latent space to retrieve scenes similar to a query based on only scene statistics. Then, we *re-rank* the retrieval results by selecting a single slot from the query and matching it against the slots of the retrieved scenes. A database of 50K scenes stored as $(\mathbf{s}, \mathbf{z})$ vectors is created from the Objects Room training set. We visualize one query result (Figure 5) with more examples provided in the appendix.

5 Conclusion

We presented a probabilistic framework for learning an interpretable hierarchy of scene and slot latent variables. The learned representation shares advantages of both a single scene-level representation as well as object-centric slots. This work improves our understanding on how to develop object-centric VAEs, particularly with respect to accurately modeling scene-level statistics.

We believe the interpretation of the deterministic variables in the LOM graphical model could benefit from further study. They are essential to the success of LOMs but trade off the ability to model stochasticity in the order in which objects are generated. We observed that LOMs occasionally generate objects that intersect each other due to a lack of 3D understanding in the latent representations. Although unsupervised 3D scene inference is particularly difficult due to ambiguities introduced by perspective geometry, obtaining 3D object representations should improve generation quality. Our LOM implementation could also be improved by replacing the standard Gaussian scene prior with a more flexible distribution such as a mixture-of-Gaussians.

REFERENCES

Titas Anciukevicius, Christoph H Lampert, and Paul Henderson. Object-centric image generation with factored depths, locations, and appearances. *arXiv preprint arXiv:2004.00642*, 2020.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *arXiv preprint arXiv:1901.11390*, 2019.

Rewon Child. Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

Andrea Dittadi, Samuele Papa, Michele De Vita, Bernhard Schölkopf, Ole Winther, and Francesco Locatello. Generalization and robustness implications in object-centric learning. *arXiv preprint arXiv:2107.00637*, 2021.

Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *International Conference on Learning Representations*, 2018.

Harrison Edwards and Amos Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.

Sebastien Ehrhardt, Oliver Groth, Aron Monszpart, Martin Engelcke, Ingmar Posner, Niloy Mitra, and Andrea Vedaldi. Relate: Physically plausible multi-object scene synthesis using structured latent spaces. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 11202–11213. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/806beafe154032a5b818e97b4420ad98-Paper.pdf.

Patrick Emami, Pan He, Sanjay Ranka, and Anand Rangarajan. Efficient iterative amortized inference for learning symmetric and disentangled multi-object representations. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2970–2981. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/emami21a.html.

Martin Engelcke, Adam R Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. *arXiv preprint arXiv:1907.13052*, 2019.

Martin Engelcke, Oiwi Parker Jones, and Ingmar Posner. Genesis-v2: Inferring unordered object representations without iterative refinement. *arXiv preprint arXiv:2104.09958*, 2021.

Mario Fritz and Bernt Schiele. Decomposition, discovery and detection of visual categories using topic models. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, 2008.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, pp. 6691–6701, 2017.

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2424–2433, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/greff19a.html.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

Oliver Groth, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 702–717, 2018.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 50–57, 1999.

Drew A Hudson and Larry Zitnick. Generative adversarial transformers. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4487–4499. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/hudson21a.html.

Jindong Jiang and Sungjin Ahn. Generative neurosymbolic machines. *arXiv preprint arXiv:2010.12152*, 2020.

Rishabh Kabra, Chris Burgess, Loic Matthey, Raphael Lopez Kaufman, Klaus Greff, Malcolm Reynolds, and Alexander Lerchner. Multi-object datasets. https://github.com/deepmind/multi-object-datasets/, 2019.

Jinwoo Kim, Jaehoon Yoo, Juho Lee, and Seunghoon Hong. Setvae: Learning hierarchical composition for generative modeling of set-structured data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15059–15068, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751, 2016.

Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3d controllable image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5871–5880, 2020.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. Biva: A very deep hierarchy of latent variables for generative modeling. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/9bdb8b1faffa4b3d41779bb495d79fb9-Paper.pdf.

James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

Thu Nguyen-Phuoc, Christian Richardt, Long Mai, Yong-Liang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *arXiv preprint arXiv:2002.08988*, 2020.

Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464, 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Danilo Jimenez Rezende and Fabio Viola. Taming vaes. *arXiv preprint arXiv:1810.00597*, 2018.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *31st International Conference on Machine Learning, ICML 2014*, 4:3057–3070, 2014.

Josef Sivic, Bryan C Russell, Alexei A Efros, Andrew Zisserman, and William T Freeman. Discovering objects and their location in images. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pp. 370–377. IEEE, 2005.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pp. 3738–3746, 2016.

Tinne Tuytelaars, Christoph H Lampert, Matthew B Blaschko, and Wray Buntine. Unsupervised object discovery: A comparison. *International journal of computer vision*, 88(2):284–302, 2010.

Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19667–19679. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/e3b21256183cf7c2c7a66be163579d37-Paper.pdf.

Sjoerd van Steenkiste, Karol Kurach, Jürgen Schmidhuber, and Sylvain Gelly. Investigating object compositionality in generative adversarial networks. *Neural Networks*, 130:309–325, 2020. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2020.07.007. URL https://www.sciencedirect.com/science/article/pii/S0893608020302483.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua B Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. *arXiv preprint arXiv:1910.12827*, 2019.

Julius von Kügelgen, Ivan Ustyuzhaninov, Peter Gehler, Matthias Bethge, and Bernhard Schölkopf. Towards causal generative scene models via competition of experts. *arXiv preprint arXiv:2004.12906*, 2020.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of AAAI*, 2021.

A APPENDIX

A.1 ARCHITECTURES

We use GELU non-linear activations and use *xavier* initialization for all neural nets.

Residual self-attention block: We construct each Transformer architecture used to implement the permutation-invariant scene encoder and the causal Transformers by stacking residual self-attention blocks. Each such block applies single-head scaled dot-product attention to input $\boldsymbol{x} \in \mathbb{R}^{N \times D}$:

$$\alpha = \texttt{softmax}(\frac{k(\boldsymbol{x})q(\boldsymbol{x})^\intercal}{\sqrt{D}})v(\boldsymbol{x}) \tag{10}$$

$$\boldsymbol{x} = \boldsymbol{x} + \texttt{MLP}(\texttt{LayerNorm}(\boldsymbol{x} + \alpha)). \tag{11}$$

Each k, q, v linear map and the MLP project to the same dimension D'. This is fixed to $|\mathbf{z}| = 64$ in all architectures in our experiments to enable stacking the residual blocks.

Scene encoder g_ϕ: The scene encoder g_ϕ uses 3 residual SA blocks in all main experiments. It takes as input an orderless set of slots $\mathbf{o}_{1:K}$ where each slot is a 64-dim vector. We concatenate a 64-dim trainable scene embedding feature to the input so that the input is a set of size $K + 1$. After 3 residual SA blocks, we applying a sum pooling layer that aggregates the set by summing the $K + 1$ vectors and projecting the result to the scene latent dimension. Then, two linear layers project this vector to the mean and softplus-activated standard deviation for the Gaussian scene posterior.

Causal Transformers f_θ: A straightforward architectural choice for implementing f_θ is a recurrent neural network (RNN) such as a GRU (Cho et al., 2014). However, Transformers have only a constant time dependency on the sequence length, unlike RNNs, and have superior ability to capture long-range interactions between parts of the sequence due to the use of self-attention (Radford et al., 2019; Brown et al., 2020; Zhou et al., 2021).

We use a standard Transformer with single-head *causal* self-attention (SA) to implement f_θ. In detail, the first token in the input sequence is the sample $\mathbf{s} \sim p(\mathbf{s})$ and the subsequent K tokens in the sequence are set to a shared trainable embedding of dimension $|\mathbf{z}|$ initialized with zeros. The scene sample is projected to the same dimension as the slots with a single layer MLP. To provide extra positional information we add a positional encoding to the input sequence. This is implemented by mapping the values $[0/(K+1), 1/(K+1), \ldots, (K+1)/(K+1)]$ to $|\mathbf{z}|$-dimensional vectors with a linear layer and adding them to the input. We use the standard approach of masking out attention values in each of the two stacked residual SA blocks to implement causal self-attention. The length K output sequence, starting from the second output token, are the $\boldsymbol{h}_{1:K}$. These are each mapped to Gaussian parameters for each $p_\theta(\mathbf{z}_k \mid \boldsymbol{h}_k)$ with two linear layers. We train f_θ end-to-end alongside the inference and generation networks to maximize the ELBO, which we discussed in Section 3.3.

Causal Transformer f'_θ: For the causal Transformer that predicts the order-dependent scale parameters for the ordered slot posterior, we use the same architecture except for a few minor differences. Primarily, the inputs are now a sample from the scene posterior and the permuted set of orderless slots. The output of the two stacked residual SA blocks is mapped with a single linear layer to a softplus-activated standard deviation for the ordered slot posterior Gaussian of dim $|\mathbf{z}|$. The mean of this Gaussian is fixed as the same mean from the permuted orderless posterior.

A.2 GREEDY MATCHING ALGORITHM

Pseudocode for the greedy matching algorithm we use is provided in Algorithm 1. Alternative matching algorithms such as the Hungarian algorithm (Munkres, 1957) can be used instead at a higher cost. We did not explore other matching algorithms since the greedy solution worked well initially.

Algorithm 1 Greedy matching algorithm for $\boldsymbol{\pi}$. Uses $O(K)$ time and $O(K^2)$ space.

Input: Ordered prior means $\mu_1^{\mathbf{z}}, \ldots, \mu_K^{\mathbf{z}}$, orderless posterior means $\mu_1^{\mathbf{o}}, \ldots, \mu_K^{\mathbf{o}}$, each $\mu_k \in \mathbb{R}^{D_z}$
$\boldsymbol{C}[i,j] = \|\mu_i^{\mathbf{z}} - \mu_j^{\mathbf{o}}\|_2, \ \forall (i,j)$
$\pi = [\,]$
for index $i = 1 \ldots K$ **do**
  $j^* = \arg\min \boldsymbol{C}[i,:]$
  $\boldsymbol{C}[:,j^*] = +\inf$
  $\pi[i] = j^*$
end for
return Permutation π

A.3 Derivation of the LOM ELBO

We assume that we are given a sample $\mathbf{o}_{1:K}$ from an orderless slot posterior estimated by an auxiliary orderless generative model. We can derive a lower-bound on the log marginal likelihood as follows:

$$\log p(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{s}, \mathbf{z}_{\pi(1):\pi(K)}) d\mathbf{s}, d\mathbf{z}_{1:K} \tag{12}$$

$$= \log \int \frac{q_\phi(\mathbf{s}, \mathbf{z}_{\pi(1):\pi(K)} \mid \mathbf{x})}{q_\phi(\mathbf{s}, \mathbf{z}_{\pi(1):\pi(K)} \mid \mathbf{x})} p_\theta(\mathbf{x}, \mathbf{s}, \mathbf{z}_{1:K}) d\mathbf{s}, d\mathbf{z}_{1:K} \tag{13}$$

$$= \log \mathbb{E}_q \left[\frac{p_\theta(\mathbf{x}, \mathbf{s}, \mathbf{z}_{1:K})}{q_\phi(\mathbf{s}, \mathbf{z}_{\pi(1):\pi(K)} \mid \mathbf{x})} \right]. \tag{14}$$

Applying Jensen's inequality, we get:

$$\log p(\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{s}, \mathbf{z}_{\pi(1):\pi(K)} | \mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x}, \mathbf{s}, \mathbf{z}_{1:K})}{q_\phi(\mathbf{s}, \mathbf{z}_{\pi(1):\pi(K)} \mid \mathbf{x})} \right] \tag{15}$$

$$= \mathbb{E}_{q_\phi(\mathbf{s} | g_\phi(\mathbf{o}_{1:K}))} \left[\mathbb{E}_{q_\phi(\mathbf{z}_{\pi(1):\pi(K)} | \boldsymbol{h}'_{\pi(k)}, \mathbf{x})} \left[\log \frac{p_\theta(\mathbf{x} \mid \mathbf{z}_{1:K}) p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s}) p(\mathbf{s})}{q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K})) q_\phi(\mathbf{z}_{\pi(1):\pi(K)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x})} \right] \right]. \tag{16}$$

We can treat the inner log product of terms as a sum of logs to get the following three loss terms. First is a log-likelihood loss term:

$$\mathcal{L}_{\text{LL}} = \mathbb{E}_{q_\phi(\mathbf{s} | g_\phi(\mathbf{o}_{1:K}))} \left[\mathbb{E}_{q_\phi(\mathbf{z}_{\pi(1):\pi(K)} | \boldsymbol{h}'_{\pi(k)}, \mathbf{x})} \left[\log p_\theta(\mathbf{x} \mid \mathbf{z}_{\pi(1):\pi(K)}) \right] \right]. \tag{17}$$

Second is a slot reverse KL divergence loss:

$$\mathcal{L}_{\text{KL_z}} = \mathbb{E}_{q_\phi(\mathbf{s} | g_\phi(\mathbf{o}_{1:K}))} \left[\mathbb{E}_{q_\phi(\mathbf{z}_{\pi(1):\pi(K)} | \boldsymbol{h}'_{\pi(k)} \mathbf{x})} \left[\log \frac{p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s})}{q_\phi(\mathbf{z}_{\pi(1):\pi(K)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x})} \right] \right] \tag{18}$$

$$= -\mathbb{E}_{q_\phi(\mathbf{s} | g_\phi(\mathbf{o}_{1:K}))} \left[D_{KL} \big(q_\phi(\mathbf{z}_{\pi(1):\pi(K)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x}) \parallel p_\theta(\mathbf{z}_{1:K} \mid \mathbf{s}) \big) \right]. \tag{19}$$

Since both the ordered slot posterior and the ordered slot prior are products of K conditionally independent distributions, this reverse KL further factorizes as a sum over K:

$$\mathcal{L}_{\text{KL_z}} = -\mathbb{E}_{q_\phi(\mathbf{s} | g_\phi(\mathbf{o}_{1:K}))} \left[\sum_{k=1}^{K} D_{KL} \big(q_\phi(\mathbf{z}_{\pi(k)} \mid \boldsymbol{h}'_{\pi(k)}, \mathbf{x}) \parallel p_\theta(\mathbf{z}_k \mid \boldsymbol{h}_k) \big) \right]. \tag{20}$$

Finally, there is a scene-level reverse KL divergence loss:

$$\mathcal{L}_{\text{KL_s}} = \mathbb{E}_{q_\phi(\mathbf{s}|g_\phi(\mathbf{o}_{1:K}))}\left[\mathbb{E}_{q_\phi(\mathbf{z}_{\pi(1):\pi(K)}|\boldsymbol{h}'_{\pi(k)},\mathbf{x})}\left[\log \frac{p(\mathbf{s})}{q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K}))}\right]\right] \tag{21}$$

$$= \mathbb{E}_{q_\phi(\mathbf{s}|g_\phi(\mathbf{o}_{1:K}))}\left[\log \frac{p(\mathbf{s})}{q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K}))}\right] \tag{22}$$

$$= -\mathbb{E}_{q_\phi(\mathbf{s}|g_\phi(\mathbf{o}_{1:K}))}\left[\log \frac{q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K}))}{p(\mathbf{s})}\right] \tag{23}$$

$$= -D_{KL}\big(q_\phi(\mathbf{s} \mid g_\phi(\mathbf{o}_{1:K})) \parallel p(\mathbf{s})\big). \tag{24}$$

Therefore, we have the following ELBO we wish to maximize with respect to parameters θ, ϕ:

$$\log p(\mathbf{x}) \geq \mathcal{L}_{\text{LL}} + \mathcal{L}_{\text{KL_z}} + \mathcal{L}_{\text{KL_s}}. \tag{25}$$

A.4 Experiment details

An open source implementation of the code for replicating all experiments will be released with link placed here. Videos demonstrating latent space random walks will also be released.

Hyperparameters: We base our implementation on EfficientMORL (Emami et al., 2021), which the authors have released an open source implementation for.[1]. We keep their hyperparameters except that the number of iterative refinement iterations is fixed at 2 for all experiments. Also, all LOM models use the Gaussian image likelihood with a global standard deviation of 0.7 for all datasets. We set the slot dimension to 64 and the scene dimension at 128. Like EfficientMORL, we use the Adam optimizer (Kingma & Ba, 2014) with default PyTorch (Paszke et al., 2019) hyperparameters and a learning rate of 1e-4. We keep their learning rate schedule, which is a linear warmup for 10K steps then multiplicatively decays with by a rate of 0.5 every 100K steps.

Datasets: Following Engelcke et al. (2021), we use $K = 7$ and $K = 9$ slots respectively for Objects Room and ShapeStacks. For CLEVR6, we use $K = 7$ slots following standard practice on this popular multi-object benchmark. We pre-process all images by converting RGB values to the range $[-1, 1]$ before passing them as input to our model. We use the Objects Room training set which can be accessed freely online.[2] This dataset contains 64x64 RGB training images. The ShapeStacks dataset is also freely available for download online and also has 64x64 RGB images.[3]. We use the same preprocessing protocol for CLEVR6 as (Emami et al., 2021), which is to center crop the images to 192x192 and the resize them to size 96x96. This dataset is also available as part of the Multi-Object Dataset.

Balancing reconstruction and KL: LOM uses the same GECO schedule as EfficientMORL except we tune the target for the Gaussian likelihood with a standard deviation of 0.7. For Objects Room, we used a per-pixel and per-channel target of -0.352 (for an unnormalized Gaussian log-likelihood). Due to the increased visual complexity of ShapeStacks, we lowered this target to -0.351 to achieve a better trade-off of KL and reconstruction. On CLEVR we increased this target to -0.356 as the background is of a low complexity so that the reconstruction error starts out much lower than the other two datasets. The rule of thumb used to tune the GECO target is that the target should be reached after about 15-20% of the training steps, so that the GECO Lagrange parameter can decrease and return to 1 so that the proper ELBO is fit by the end of training. We use a GECO learning rate for updating its Lagrange parameter of 1e-6, and multiply this by 5 following GENESISv2 once the target is reached. On ShapeStacks, we got best results by instead increasing this acceleration factor to 50.

We found it beneficial to also use a *freebits* (Maaløe et al., 2019) parameter λ on the KL terms to help avoid posterior collapse for EfficientMORL's KL terms as well as the LOM scene KL (but not for the LOM slot KL since the posterior means are already fixed). To adapt this for slot-based distributions, we set λ to represent the minimum value for the sum over K slot KL terms. This is set to $\lambda = 0.75 \times |\mathbf{z}|$ for each stochastic layer in EfficientMORL and $\lambda = 0.25 \times |\mathbf{s}|$ for the scene KL.

[1] https://github.com/pemami4911/EfficientMORL
[2] https://github.com/deepmind/multi_object_datasets
[3] https://ogroth.github.io/shapestacks/

We experimented with annealing λ down slightly for the scene KL term during the last 25K steps of training and found that this had a slight increase in performance.

Baselines: Code for training GENESISv2 is provided by the authors.[4] They also release pre-trained weights for GENESIS and GENESISv2 on Objects Room and ShapeStacks which we use to compute the structure accuracy metric. To train GENESISv2 on CLEVR6, we had to lower the default standard deviation for the mixture of Gaussians from 0.7 to 0.1 to get it to properly discover and segment the objects. We adjusted their GECO target accordingly, tuning it to -2.265 (this is the per-pixel and per-channel value. We based this off of the target value of -61000 for an unnormalized mixture-of-Gaussians summed over pixels and channels with standard deviation of 0.1 used by Emami et al. (2021)). For the variant of GENESISv2 trained with the Gaussian likelihood, we used the same standard deviation of 0.7 as LOM. For Objects Room, ShapeStacks, and CLEVR6 we used a per-pixel and per-channel target of -0.353 (for unnormalized Gaussian log-likelihood). We found that we needed to decrease the GECO learning rate from 1e-5 to 1e-6.

The NVAE is trained with the open source implementation provided by the authors.[5]. We changed the image likelihood from the discrete mixture of logistic distribution to the Gaussian likelihood with standard deviation of 0.7 for a fair comparison with other considered models. We also modify the default architecture for 64x64 images for use with CLEVR6. To decrease memory consumption, we remove the normalizing flows and use 16 initial channels in the encoder and decoder towers. The NVAE has three scales of spatial latents, and we extract the top-level at the lowest resolution of 12x12. Since each spatial latent group has 20 channels, this becomes a 2880-dim vector when flattened. We follow the same protocol as (Greff et al., 2019; Dittadi et al., 2021) to compute disentanglement scores for NVAE of imposing a fixed order on objects in the seen based on their attributes.

Compute: We use a batch size of 32 for LOM split across 8 GPUs for our main experiments. We train for 400K steps on Objects Room and ShapeStacks, which takes approximately 27 wall-clock hours each run. We set a compute budget upper time limit of 24 hours for CLEVR, as both GENESISv2 and LOM could continue training with gradual performance improvement for longer. LOM reaches 200K training steps in this time.

On Objects Room and ShapeStacks, training GENESISv2 with 4 GPUs and a batch size of 32 takes about 20 hours to reach 500K steps, which follows the authors's training protocol. On CLEVR, we reach 300K steps in 24 hours. The heavier compute requirement of the LOM compared to GENESISv2 comes from the use of EfficientMORL; the LOM transformers and extra reconstruction term add only a small increase in compute.

The NVAE converges fairly quickly on CLEVR because we are able to use a large batch size of 256 split across 8 GPUs. We found it takes about 10-14 hours to converge in this setup.

Metrics: We provide the generated samples and our labels used to compute the structure accuracy metric in the supplementary material.

A.5 Additional Results

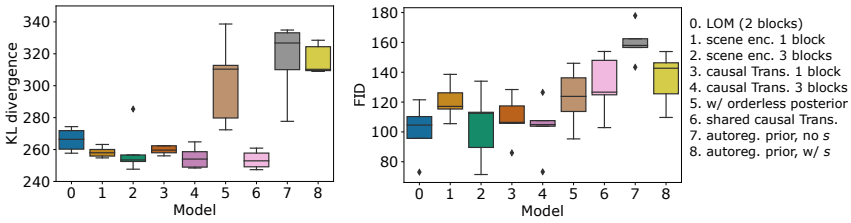
A.5.1 Additional Ablation Studies

See Figure 6. First, we found that increasing the expressivity of the scene encoder g_ϕ by using three SA blocks instead of two (the baseline) slightly increased FID, whereas decreasing the number of SA blocks to one in g_ϕ or in the causal Transformers f_θ, f'_θ lead to a pronounced decline in performance. Next, we validated the importance of computing new order-aware scale parameters for the permuted slot posterior by directly re-using the scale parameters estimated by the auxiliary orderless model. This resulted in a high test KL and FID scores because the order-dependent prior struggles to keep up with the orderless posterior. Then, we attempted to share parameters between f_θ and f'_θ, but found this severely degraded the FID.

A.5.2 Disentanglement scores

Full DCI scores are reported in Table 3.

[4] https://github.com/applied-ai-lab/genesis
[5] https://github.com/NVlabs/NVAE



(a) (b)

Figure 6: Ablations and variations results. b) Test KL for ablated models. Lower is better. c) FID scores for ablated models. Lower is better.

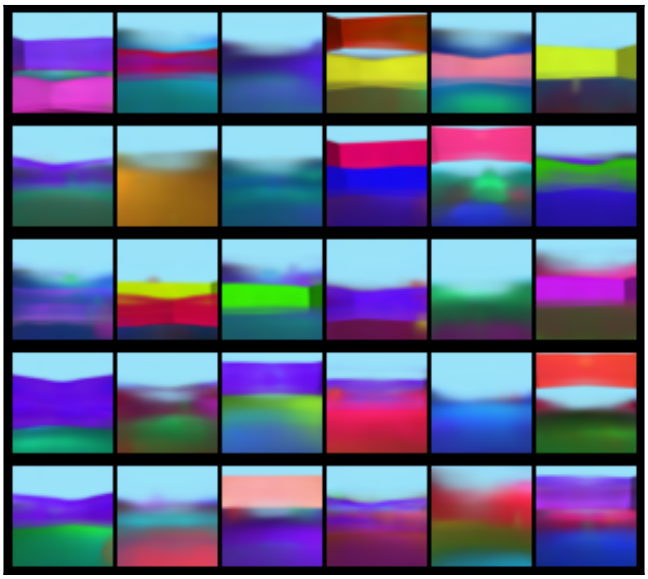


Figure 7: Objects Room samples from the EMORL's data-dependent i.i.d. prior.

A.5.3 Additional Qualitative Results

We visualize samples from EfficientMORL i.i.d. prior in Figure 7 for Objects Room. It is interesting to see that it can generate walls, floors, and sky. Without any dependencies between slots modeled, it cannot generate coherent scenes.

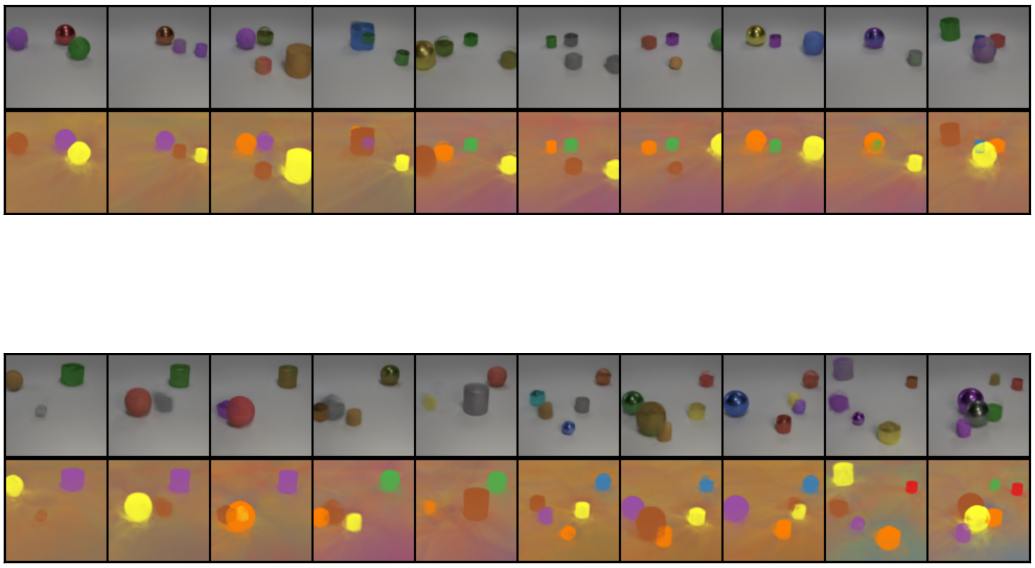


Figure 8: LOM spherical interpolations between two random samples from the prior.

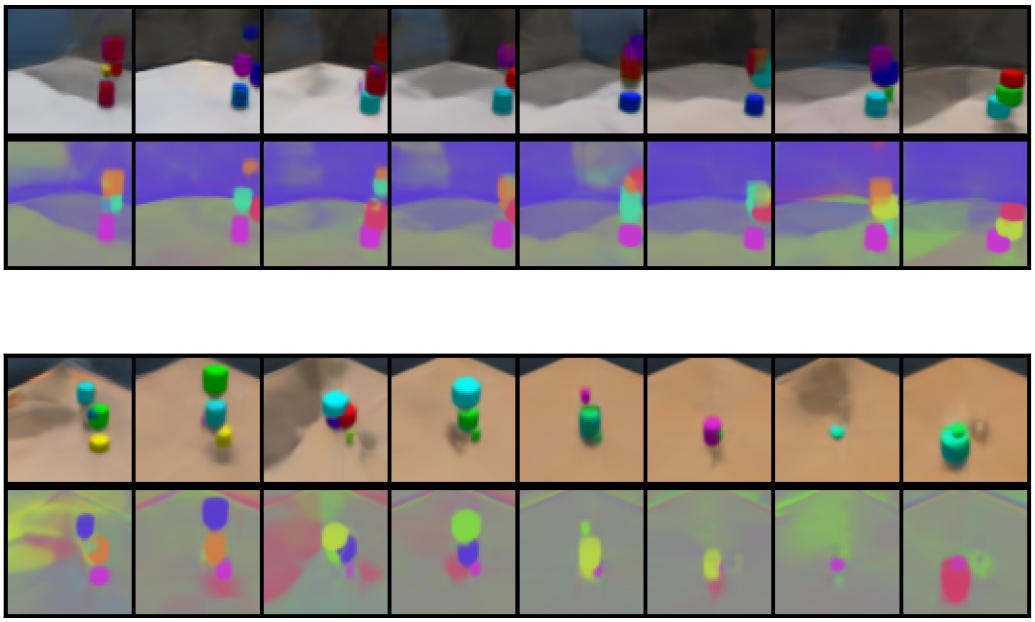


Figure 9: LOM spherical interpolations between two random samples from the prior.

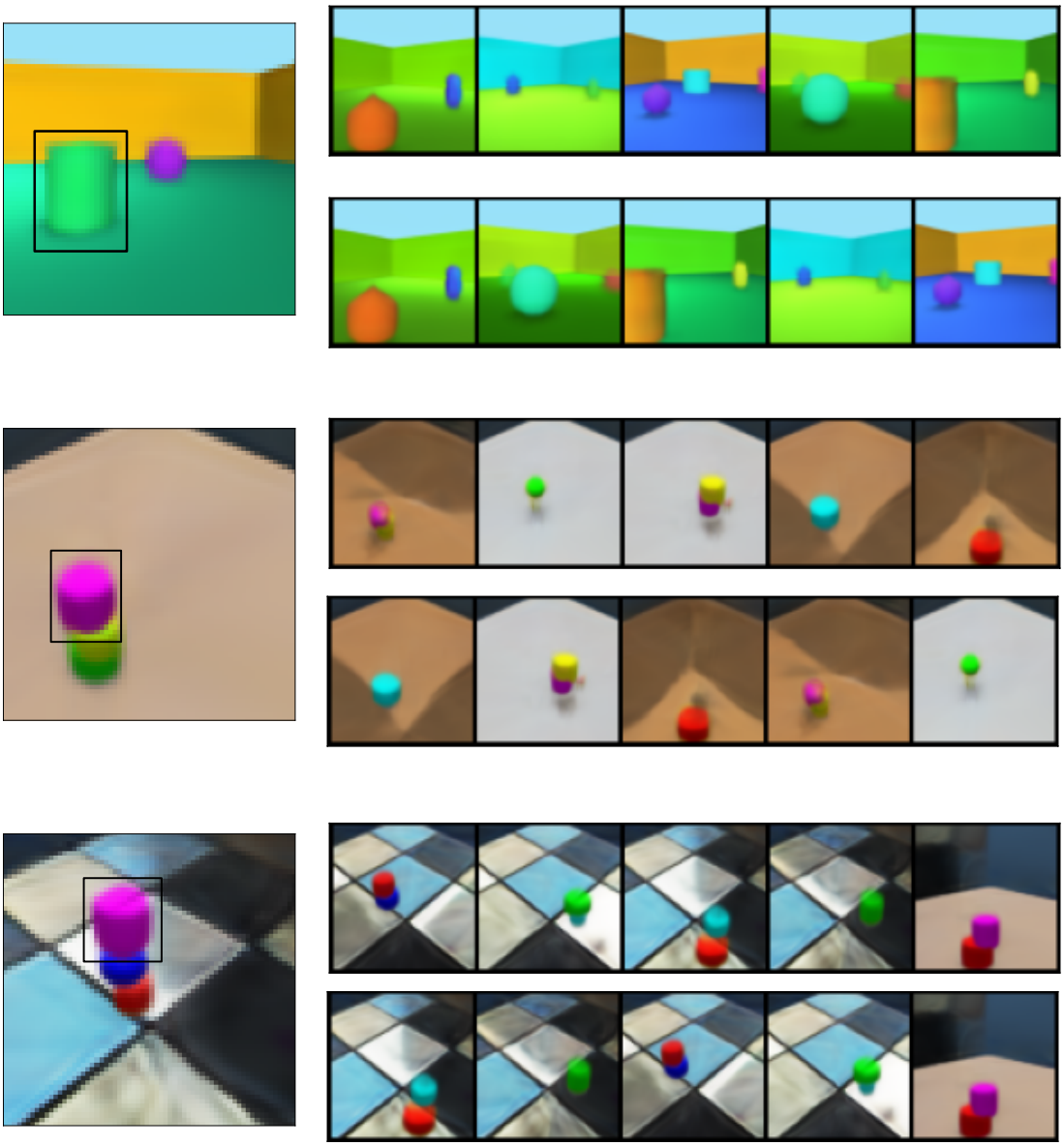


Figure 10: Additional hierarchical retrieval results. The top row show the initially retrieved scenes and the bottom row shows the re-ranking based on the selected slot (shown with black boxes).

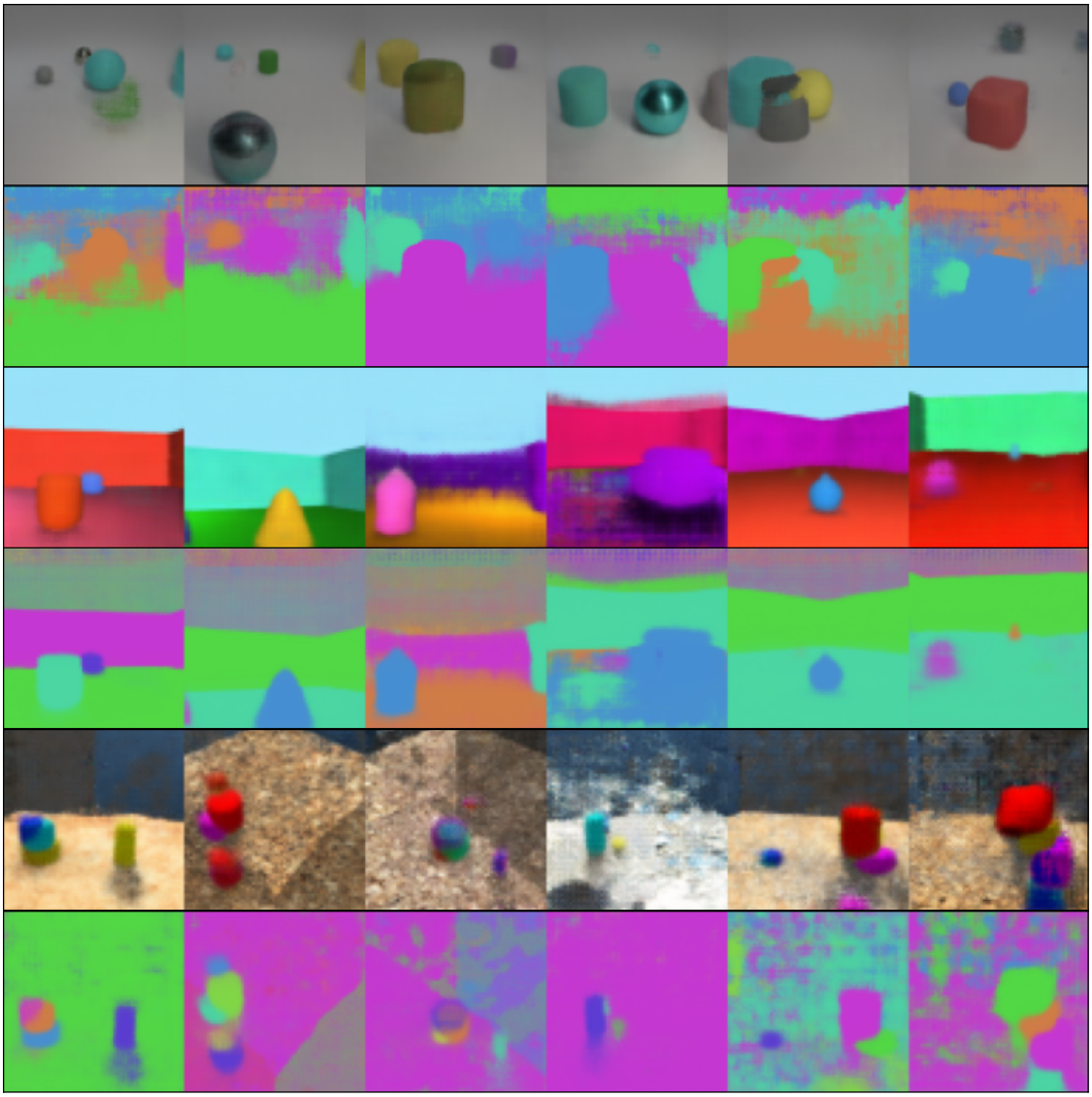


Figure 11: GENESISv2 random samples from the models we trained on each dataset. The ShapeStacks quality matches Engelcke et al. (2021) and achieves similar FID scores.

Table 3: DCI scores averaged across three training runs.

DCI	**LOM**	**GENESISv2 (G)**	**GENESISv2 (MoG)**	**NVAE**
Disentanglement	$0.44_{\pm 0.02}$	$0.54_{\pm 0.01}$	$0.56_{\pm 0.04}$	$0.19_{\pm 0.0}$
Completeness	$0.51_{\pm 0.04}$	$0.60_{\pm 0.04}$	$0.57_{\pm 0.05}$	$0.16_{\pm 0.0}$
Informativeness	$0.23_{\pm 0.03}$	$0.41_{\pm 0.03}$	$0.42_{\pm 0.05}$	$0.0_{\pm 0.0}$