Simple Unsupervised Object-Centric Learning for Complex and Naturalistic Videos

Gautam Singh*
Rutgers University
singh.gautam@rutgers.edu

Yi-Fu Wu
Rutgers University
yifu.wu@gmail.com

Sungjin Ahn
KAIST
sjn.ahn@gmail.com

Abstract

Unsupervised object-centric learning aims to represent the modular, compositional, and causal structure of a scene as a set of object representations and thereby promises to resolve many critical limitations of traditional single-vector representations such as poor systematic generalization. Although there have been many remarkable advances in recent years, one of the most critical problems in this direction has been that previous methods work only with simple and synthetic scenes but not with complex and naturalistic images or videos. In this paper, we propose STEVE, an unsupervised model for object-centric learning in videos. Our proposed model makes a significant advancement by demonstrating its effectiveness on various complex and naturalistic videos unprecedented in this line of research. Interestingly, this is achieved by neither adding complexity to the model architecture nor introducing a new objective or weak supervision. Rather, it is achieved by a surprisingly simple architecture that uses a transformer-based image decoder conditioned on slots and the learning objective is simply to reconstruct the observation. Our experiment results on various complex and naturalistic videos show significant improvements compared to the previous state-of-the-art. https://sites.google.com/view/slot-transformer-for-videos

1 Introduction

The goal of object-centric representation learning is to represent the inherent structure of the physical world such as compositionality and causality as a set of representation vectors (and their relations) each corresponding to a conceptual entity module such as an object (Greff et al., 2020; Schölkopf et al., 2021). Many previous works have demonstrated the potential of this approach as a way of resolving the key limitations of traditional single-vector representations (Greff et al., 2020; Schölkopf et al., 2021; Dittadi et al., 2021; Eslami et al., 2016; Bapst et al., 2019; Mambelli et al., 2022; Ghasemipour et al., 2022). For example, considering object-centric representations as independent mechanisms helps realize systematic and zero-shot generalization for image generation (Singh et al., 2022; Chen et al., 2021). Also, the ability to decompose a visual observation into a set of discrete knowledge modules has shown to be useful for visual reasoning (Zhou et al., 2021; Wu et al., 2021).

One of the most critical challenges remaining in object-centric representation learning is to successfully apply it to complex and natural images or videos. Achieving this has primary importance because it opens up a way to utilize the enormous amounts of images and videos available on the internet and thereby unleash the full potential of this method by applying it at scale — the machine learning community has recently observed the power of scale in large-scale language models (Brown et al., 2020; Bommasani et al., 2021).

*Correspondence to singh.gautam@rutgers.edu and sjn.ahn@gmail.com.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

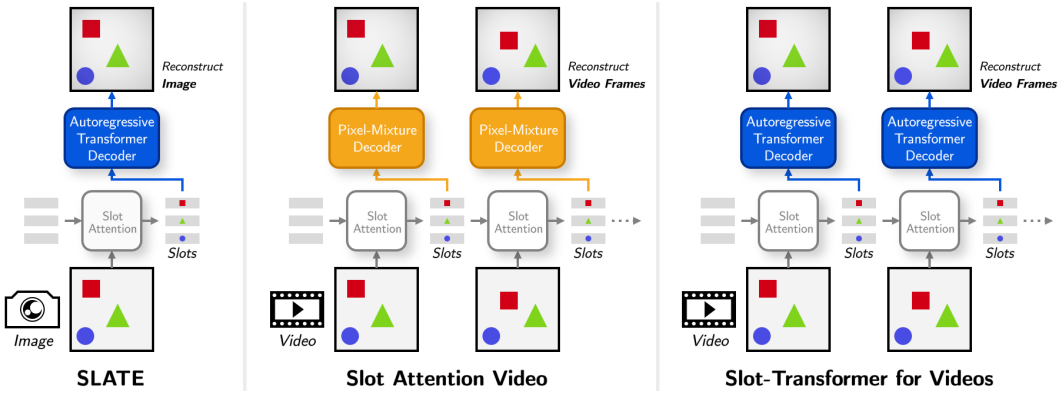


Figure 1: **Overview of the proposed model. Left:** SLATE auto-encoder combines the representational bottleneck of slots with an auto-regressive transformer decoder and shows object discovery in visually complex images. However, the question of whether this framework can deal with complex and naturalistic videos is unexplored. **Middle:** Conventional object-centric video models such as Slot Attention Video deal with videos by applying slot attention recurrently on the frames and applying a pixel-mixture decoder to reconstruct the frames. However, their ability to handle complex and naturalistic videos is lacking. **Right:** Our model *Slot Transformer for Videos* provides a simple and minimal architecture leveraging an auto-regressive transformer decoder that can effectively handle complex and naturalistic videos.

However, despite a significant amount of effort and many remarkable achievements in recent years, we do not yet have an unsupervised object-centric learning method that works well on complex and naturalistic videos. In fact, when considering the high complexity of natural images, the ultimate goal of making the concept of objects emerge without any supervision, i.e., only by observing, indeed remains quite elusive. As such, previous works have been shown to work only on toy or synthetic images/videos or resorted to utilizing some supervision such as optical flow or annotation on the initial frame (Kipf et al., 2021).

In this paper, we propose an unsupervised model for object-centric learning in videos, called STEVE (Slot-TransformEr for VidEos). Our proposed model makes a significant advancement by demonstrating its effectiveness for the first time on various complex and naturalistic videos unprecedented in this line of research. Interestingly, this is achieved by neither adding much complexity to the model architecture nor by introducing a new objective or weak supervision. Rather, it is achieved by a surprisingly simple architecture, comparable to or simpler than most of the previous methods that have not worked well on complex videos. Moreover, the learning objective is just to reconstruct. We believe this simplicity is one of the best strengths of our model.

Behind this success is the adoption of the transformer-based slot decoder recently proposed in SLATE (Singh et al., 2022). Because it is demonstrated in the paper that this decoder is the key to dealing with some visually complex synthetic images, it is of deep interest to the community (i) *whether and how* it can be extended to a temporal model utilizing dynamical observations, (ii) if it can eventually deal with complex naturalistic videos, and (iii) what lessons we can learn from this. To focus on investigating these central questions systematically, we intentionally choose not to pursue architectural novelty at the expense of adding complexity but to test a minimal architecture that combines the SLATE decoder with a standard slot-level recurrence model. Our experiment results on various complex and naturalistic videos show that the proposed architecture significantly outperforms previous state-of-the-art baseline models.

From these experiments, we discover important new knowledge to share with the community. We find that SLATE can deal with complex naturalistic images reasonably well despite not using temporal information. Although as a static model, SLATE has a fundamental limitation that it cannot take advantage of temporal information, e.g., maintaining consistent object identity across time, if we only look at its frame-level segmentation ability, it already performs better than the previous state-of-the-art temporal model that applies slot attention to videos (Kipf et al., 2021). Crucially, we also find that SLATE alone is not enough since our model, STEVE, shows much better frame segmentation than SLATE, while also providing consistent tracking of slots in videos, which SLATE cannot do.

In particular, we find that on some complex videos such as MOVi-Tex, SLATE fails significantly, suggesting that learning jointly from both temporal information and the powerful SLATE decoder is significant and essential to realize the full potential of object-centric representation learning in videos. Furthermore, our analysis shows that our model is robust to challenges such as static objects and camera motion and that it also generalizes well to out-of-distribution number of objects and unseen textures. Lastly, we propose several new datasets that are significantly more complex than those tackled by the previous unsupervised models.

2 Preliminaries

In unsupervised object-centric representation learning, the goal is to learn to obtain the representation of an observation as a set of object vectors, referred to as *slots* (Locatello et al., 2020). A common approach for this is to auto-encode: a slot-encoder $f_\phi^{\text{enc}}(\mathbf{x})$ extracts a set of slot representations $\mathbf{s} = \{\mathbf{s}_1, \dots \mathbf{s}_N\}$ from an image $\mathbf{x}$ and then a decoder $f_\phi^{\text{dec}}(\mathbf{s})$ makes a reconstruction $\hat{\mathbf{x}}$ from these slots, where $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ and $\mathbf{s}_n \in \mathbb{R}^D$. In the encoder, the slots compete with each other to learn about which area of the input will be explained by which slots. This provides a form of information bottleneck encouraging a slot to attend an area corresponding to a meaningful conceptual entity like an object. During training, a reconstruction objective provides the learning signal for training the complete model. One nice property of this method is that it only requires reconstructing the image and does not require any weak supervision or object-aware auxiliary loss terms. Ideally, we want to maintain this simplicity in video models, but current state-of-the-art requires weak supervision and optical flow to make it work on complex naturalistic videos (Kipf et al., 2021).

Mixture-based Decoder. The emergence of object-centric slots is strongly dependent on what decoder is used. The traditional way that most of the previous methods have taken is the mixture-based decoder or some variant of it. In this approach, the decoder decodes each slot $\mathbf{s}_n$ to obtain an object image $\hat{\mathbf{x}}_n$ and an alpha mask $\mathbf{m}_n$ via decoding functions g_θ^{RGB} and g_θ^{mask}, respectively. The decoded object images are then weighted-summed to obtain the full image $\hat{\mathbf{x}}$:

$$\hat{\mathbf{x}}_n = g_\theta^{\text{RGB}}(\mathbf{s}_n), \qquad \mathbf{m}_n = \frac{\exp g_\theta^{\text{mask}}(\mathbf{s}_n)}{\sum_{m=1}^{N} \exp g_\theta^{\text{mask}}(\mathbf{s}_m)}, \qquad \hat{\mathbf{x}} = \sum_{n=1}^{N} \mathbf{m}_n \cdot \hat{\mathbf{x}}_n.$$

These decoder networks are implemented using a CNN. The reconstruction objective is taken to be the squared error between the input and the reconstructed image i.e. $\mathcal{L}_{\text{mixture}} = \|\hat{\mathbf{x}} - \mathbf{x}\|^2$. A critical limitation of this approach is that it has never been successful in dealing with scenes with high visual complexity like natural images.

Autoregressive Slot-Transformer Decoder. Recently, Singh et al. (2022) have challenged this traditional perspective that we need a mixture decoder for the emergence of objectness in slots. Arguing that a mixture decoder severely limits the interaction among slots and the reconstruction quality—hence a problem in obtaining good learning signal for complex images—they proposed a new architecture, SLATE, using a powerful autoregressive decoder based on a transformer conditioned on the slots (Chen et al., 2020). However, many critical questions about this new approach still remain because the focus of the SLATE paper is on image generation ability. To get a deeper understanding of the object representations that a model like SLATE produces, we would need to thoroughly investigate the quantitative evidence about its scene decomposition ability, its ability to deal with visual complexity close to that of natural images, and finally, the effect of extending this architecture to a temporal model. We aim to answer these in this paper.

3 STEVE: Slot Transformer for Videos

To study the effect of extending the slot-transformer decoder to a temporal model that can deal with videos, we choose to propose a minimal architecture. Specifically, our proposed model, STEVE, combines three main components: (1) a CNN-based image encoder, (2) a recurrent slot encoder that updates slots temporally with recurrent neural networks (RNNs), and (3) the slot-transformer decoder of SLATE. Although we may obtain additional performance gains by exploring further architectural novelty at the cost of additional complexity, we choose to investigate this minimal architecture to focus on our goal of investigating slot-transformer decoder models for video.

Given a video consisting of frames $\mathbf{x}_1, \dots \mathbf{x}_T$, we want to maintain N slots $\mathbf{s}_t = \{\mathbf{s}_{t,1}, \dots, \mathbf{s}_{t,N}\}$ for each time-step $t \in \{1, \dots T\}$ by starting the temporal update from initial slots $\mathbf{s}_0$. We apply a recurrent slot encoder $f_\phi^{\text{slot-rnn}}$ at each step t to update the slot representations from $\mathbf{s}_{t-1}$ to $\mathbf{s}_t$ using the information from the input frame $\mathbf{x}_t$.

$$\mathbf{s}_t = f_\phi^{\text{slot-rnn}}(\mathbf{x}_t; \mathbf{s}_{t-1}), \qquad \mathbf{s}_0 = \texttt{initialize}().$$

It is expected that each slot n would represent one object and track it consistently over time in a given video. For each frame, the slot representation $\mathbf{s}_t$ is used to reconstruct the input frame $\mathbf{x}_t$ using the slot-transformer decoder. For doing such reconstruction, each frame $\mathbf{x}_t$ is treated as a sequence of discrete tokens provided by a discrete VAE encoder. Given the slots $\mathbf{s}_t$, the slot-transformer decoder learns to predict this sequence of tokens auto-regressively by minimizing a cross-entropy loss.

$$\mathcal{L}_{\text{CE}} = \sum_{t=1}^{T} \sum_{l=1}^{L} \text{CE}(\mathbf{z}_{t,l}, \mathbf{o}_{t,l}), \quad \mathbf{z}_{t,1}, \dots, \mathbf{z}_{t,L} = f_\phi^{\text{dVAE}}(\mathbf{x}_t), \quad \mathbf{o}_{t,l} = g_\theta^{\text{slot-transformer}}(\mathbf{s}_t; \mathbf{z}_{t,1} \dots \mathbf{z}_{t,l-1}).$$

where $\text{CE}(\cdot, \cdot)$ denotes cross-entropy loss, $\mathbf{z}_t = \{\mathbf{z}_{t,1}, \dots, \mathbf{z}_{t,L}\}$ are L discrete tokens that act as prediction targets for the transformer at time t and $\mathbf{o}_{t,l}$ contains the log-probabilities predicted by the transformer for the token at position l for frame t. To train the discrete VAE, an image reconstruction loss $\mathcal{L}_{\text{dVAE}}$ is used as in SLATE.

$$\mathcal{L}_{\text{dVAE}} = \sum_{t=1}^{T} \|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2, \qquad \mathbf{z}_{t,1}, \dots \mathbf{z}_{t,L} = f_\phi^{\text{dVAE}}(\mathbf{x}_t), \qquad \hat{\mathbf{x}}_t = g_\theta^{\text{dVAE}}(\mathbf{z}_{t,1}, \dots \mathbf{z}_{t,L}).$$

The complete model is trained using $\mathcal{L}_{\text{STEVE}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{dVAE}}$.

Recurrent Slot Encoder. Several recent works have implemented recurrent object-centric models by incorporating an RNN-based dynamics model to update slots over timesteps (Veerapaneni et al., 2019; Jiang et al., 2020; Kossen et al., 2020; Lin et al., 2020a; Stanić & Schmidhuber, 2019). For our model, we leverage a slot-based recurrent encoder that is also used as the backbone of Slot Attention Video (Kipf et al., 2021). Our slot-based recurrent encoder works as follows. At the start of an episode, we provide initial slots $\mathbf{s}_0$ by randomly sampling from a Gaussian distribution with learned mean and variance. For each input frame $\mathbf{x}_t$, we compute an encoding in the form of a feature map using a backbone CNN. This feature map is flattened to obtain a set of input features $\mathbf{e}_t = \{\mathbf{e}_{t,1}, \mathbf{e}_{t,2}, \dots, \mathbf{e}_{t,HW}\}$. Next, the slots $\mathbf{s}_{t-1}$ perform attention on the features $\mathbf{e}_t$ and use the attention result to update the slots to $\mathbf{s}_t$. For this, the slots $\mathbf{s}_{t-1}$ compute attention weights $\mathcal{A}_t = \{\mathcal{A}_{t,1}, \dots, \mathcal{A}_{t,N}\}$ over the input features, where $\mathcal{A}_{t,n}$ are the attention weights for slot n. The attention weights are used to perform an attention-weighted sum of the input features. The attention result $\mathbf{r}_{t,n}$ is then used to update the respective slot using an RNN f_ϕ^{RNN}.

$$\tilde{\mathbf{s}}_{t,n} = f_\phi^{\text{RNN}}(\mathbf{r}_{t,n}, \mathbf{s}_{t-1,n}), \quad \mathbf{r}_{t,n} = \frac{\sum_{i=1}^{HW} \mathcal{A}_{t,n,i} \cdot v(\mathbf{e}_{t,i})}{\sum_{j=1}^{HW} \mathcal{A}_{t,n,j}}, \quad \mathcal{A}_t = \underset{N}{\texttt{softmax}}\left(\frac{q(\mathbf{s}_{t-1}) \cdot k(\mathbf{e}_t)^T}{\sqrt{D}}\right).$$

Here, q, k and v are linear projections and D is the output size of the projections. Finally, the slots are made to interact via an interaction network: $\mathbf{s}_t = f_\phi^{\text{interact}}(\tilde{\mathbf{s}}_t)$. Following Kipf et al. (2021), we use the slots $\tilde{\mathbf{s}}_t$ before the interaction step for reconstruction. For complete details of the architecture, see Appendix B.

4 Related Work

Unsupervised Object-Centric Representation Learning in Images. Unsupervised object-centric learning methods for static scenes have received significant interest in recent years (Greff et al., 2016; Burgess et al., 2019; Greff et al., 2019; Locatello et al., 2020; Greff et al., 2017; Engelcke et al., 2020, 2021; Eslami et al., 2016; Crawford & Pineau, 2019b; Lin et al., 2020b; Jiang & Ahn, 2020; Deng et al., 2020; Chen et al., 2021; Anciukevicius et al., 2020; von Kügelgen et al., 2020; Greff et al., 2020). These models learn through reconstruction, commonly adopting a mixture-based decoder. Another line of work has pursued object discovery by minimizing the mutual information between the predicted segments (Savarese et al., 2021; Yang et al., 2020). Approaches based on self-supervised representation learning have also been investigated (Caron et al., 2021; Löwe et al., 2020; Wang et al., 2022). Recently, scene decomposition ability has been shown to emerge in energy-based models (Du

et al., 2021a; Yu et al., 2021). Complex-valued neural networks have also shown such emergence Lowe et al. (2022). Closely related to our work, Lamb et al. (2021) and Singh et al. (2022) have shown that a slot-based encoder combined with a transformer decoder can enable object emergence in visually complex images. However, all of these approaches deal only with static images unlike ours which can be applied to videos.

Unsupervised Object-Centric Representation Learning in Videos. A large body of work has approached fully-unsupervised video segmentation and tracking by combining recurrent slot-based encoders with a reconstruction objective. In this class, (Kosiorek et al., 2018; Stanić & Schmidhuber, 2019; Jiang et al., 2020; Crawford & Pineau, 2019a; Lin et al., 2020a; Wu et al., 2021; Singh et al., 2021; He et al., 2019) use bounding boxes for tracking. A parallel line of research learns to localize objects via segmentation masks (Greff et al., 2017; Van Steenkiste et al., 2018; Veerapaneni et al., 2019; Watters et al., 2019; Weis et al., 2020; Du et al., 2020; Kipf et al., 2021; Kabra et al., 2021; Zoran et al., 2021; Besbinar & Frossard, 2021; Creswell et al., 2020, 2021). Our work lies along this line of research. Other approaches have considered using contrastive losses (Kipf et al., 2019; Carvalho et al., 2020) while some works have explored unsupervised object-centric learning in 3D scenes (Chen et al., 2021; Henderson & Lampert, 2020; Crawford & Pineau, 2020; Stelzner et al., 2021; Du et al., 2021b; Kabra et al., 2021). While dealing with 3D scenes would be an important future direction for our work, it is orthogonal to our current focus. However, unlike ours, all of the above works have only been successful on visually simple videos. In Appendix D, we discuss the related work that deals with unsupervised video object segmentation using motion cues and unsupervised segmentation propagation.

5 Experiments

Datasets. We evaluate our model on 8 datasets. These include 6 procedurally generated datasets: CATER (Girdhar & Ramanan, 2020), CATERTex, MOVi-Solid, MOVi-Tex, MOVi-D, and MOVi-E (Greff et al., 2022); and 2 natural datasets: Traffic and Aquarium. Out of these, 5 datasets are our contributions in this work which we now describe. *i) CATERTex.* We generated it by combining objects and textures of CLEVRTex (Karazija et al., 2021) with the object motion of CATER (Girdhar & Ramanan, 2020). *ii) MOVi-Solid.* We used Kubric (Greff et al., 2022) to generate this dataset and introduced textured backgrounds and more complex shapes to the *MOVi* dataset proposed by Kipf et al. (2021). *iii) MOVi-Tex.* We added the textures from the Describable Textures dataset (Cimpoi et al., 2014) as materials to the MOVi-Solid dataset to make this dataset. By applying the same material on all surfaces, the objects and their borders become significantly harder to distinguish, resulting in much higher visual complexity over MOVi-Solid. *iv) Traffic and Aquarium.* We created these two natural datasets by collecting a 6-hour long video stream from Youtube. For training, all models use only the raw videos as input with no other supervision or input whatsoever. For evaluation, we use the ground truth instance-level masks. For sample video frames and more details, see Appendix A. We will release all the proposed datasets which, we believe, will facilitate future research. For existing benchmarks i.e. CATER, MOVi-D and MOVi-E, we use the standard train and test splits as prescribed by their respective authors. We also note that in the MOVi benchmark i.e. MOVi-A to E, versions D and E are the two most challenging versions. Thus, we consider these sufficient for evaluating our hypothesis.

Baselines. We compare our performance with three baselines which are state-of-the-art in unsupervised object-centric scene decomposition. These include two baselines that use mixture-based reconstruction: Slot Attention Video (Kipf et al., 2021) and OP3 (Veerapaneni et al., 2019); and one baseline which uses transformer-based reconstruction: SLATE (Singh et al., 2022). For Slot Attention Video, we use the fully unsupervised version that is trained only using the image reconstruction objective without optical flow or label information in the first frame. For simplicity, we will refer to this unsupervised Slot Attention Video as SAVi. Comparison with SAVi is important because our model is most similar to it with one main difference: our model uses transformer-based decoding while SAVi uses mixture-based decoding. Therefore, a comparison of our model with SAVi will provide a clear insight into the effect of the decoding approach. For the baseline SLATE, we use the same CNN backbone as our model for a fair comparison. In line with the previous works, for SAVi and OP3, we take their decoding masks to be their predicted segmentation. Since our model STEVE and the baseline SLATE are based on transformer decoder, they do not provide decoding masks. Therefore, we use their input attention masks as our predicted segmentation. Note that this

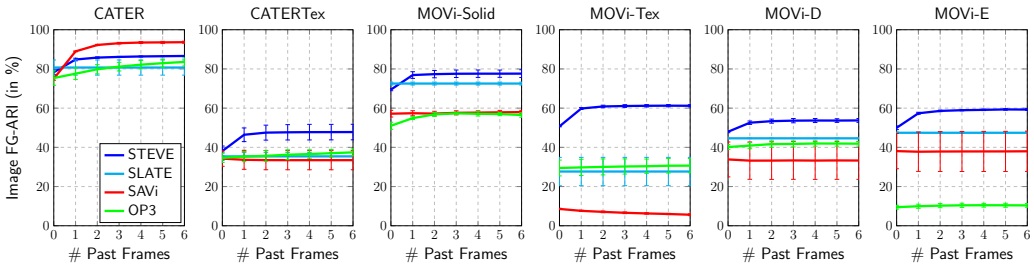


Figure 2: **Unsupervised Image Segmentation.** On the y-axis, we plot Image FG-ARI of per-frame segmentation. Along the x-axis, we show how the segmentation is affected as more frames are seen in a video. The FG-ARI of SLATE is computed on one-frame 'videos' and is broadcasted along the x-axis to facilitate comparison with other models.

difference gives an advantage to the baselines but not to our model as in section 5.3, our analysis will show that the decoding masks perform better than the input attention masks.

5.1 Unsupervised Image Segmentation

In this section, we evaluate how effective our model is for unsupervised image segmentation in videos. We report *Image FG-ARI* which measures how well the predicted segmentation matches the ground truth segmentation of a given single image. In line with the previous works (Locatello et al., 2020; Kabra et al., 2021; Kipf et al., 2021), we consider only the foreground segments of the ground truth. In Figure 2, we plot the Image FG-ARI of per-frame segmentation.

Benefit over Mixture-based Methods. Comparing STEVE with SAVi and OP3, we see that STEVE achieves significantly higher FG-ARI compared with the baselines in 5 out of 6 datasets. All of these 5 datasets are characterized by complex textures. Therefore, our superior performance shows that our model is more effective in dealing with textured scenes compared to the baselines. Because the main difference of STEVE with SAVi is the reconstruction approach, this result provides strong evidence that the transformer-based reconstruction is the key driver of this improvement. Analyzing the qualitative results in Figures 4 (left), 5 and 9 reveals that the baselines fail completely in CATERTex, MOVi-Tex, MOVi-D, and MOVi-E, most commonly by splitting the image into fixed patch regions instead of meaningful object regions. Unlike simple datasets used in prior works such as CLEVR or CATER that provide strong color contrast between objects, our datasets such as CATERTex and MOVi-Tex provide no such color cues. Despite this, our model can remarkably discover the objects. With the state-of-the-art baselines completely failing, it is for the first time in this line of research that such complex datasets have been effectively handled. In one dataset i.e. CATER, our performance is comparable to OP3 and slightly worse than SAVi. As CATER does not have complex textures, it is not surprising to see baselines performing well on this dataset. In Figure 2, we also find that as the models see more frames, the segmentation tends to improve.

Benefit of Temporal Learning. We also analyze whether training STEVE on videos has any benefit in image segmentation compared to SLATE. As SLATE is applicable only for static images, we train and test it on one frame 'videos' of our datasets. Comparing STEVE with SLATE, we note that STEVE performs consistently better in all datasets. Noteworthy is the gap in MOVi-Tex which is especially large. This suggests that training on temporal data i.e. videos can be helpful for learning to segment. We conjecture that due to the similar texture in the background and foreground, inferring correct object regions from a static image can be harder than inferring them when the objects are moving. This may explain the significantly larger gap with SLATE in MOVi-Tex compared to the other datasets. Note that this gap exists even when evaluating STEVE with zero past frames (i.e. a static image), indicating that training on temporal data helps STEVE infer segments on static images.

5.2 Unsupervised Video Segmentation

In this section, we evaluate how well our predicted segmentation tracks the ground truth segmentation in both space and time without switching identities. For this, we compute *Video FG-ARI* which considers the full trajectory of a segment over the video length as one cluster. In line with the previous

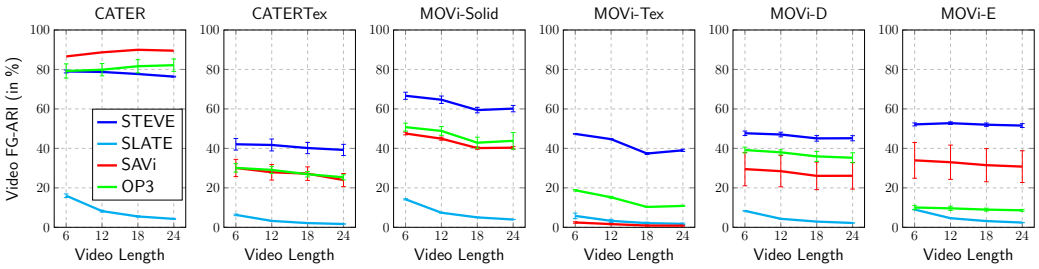


Figure 3: **Unsupervised Video Segmentation.** On the y-axis, we plot Video FG-ARI and compare our performance with the baselines. Along x-axis, we show how the performance is affected by the length of the video at test time.

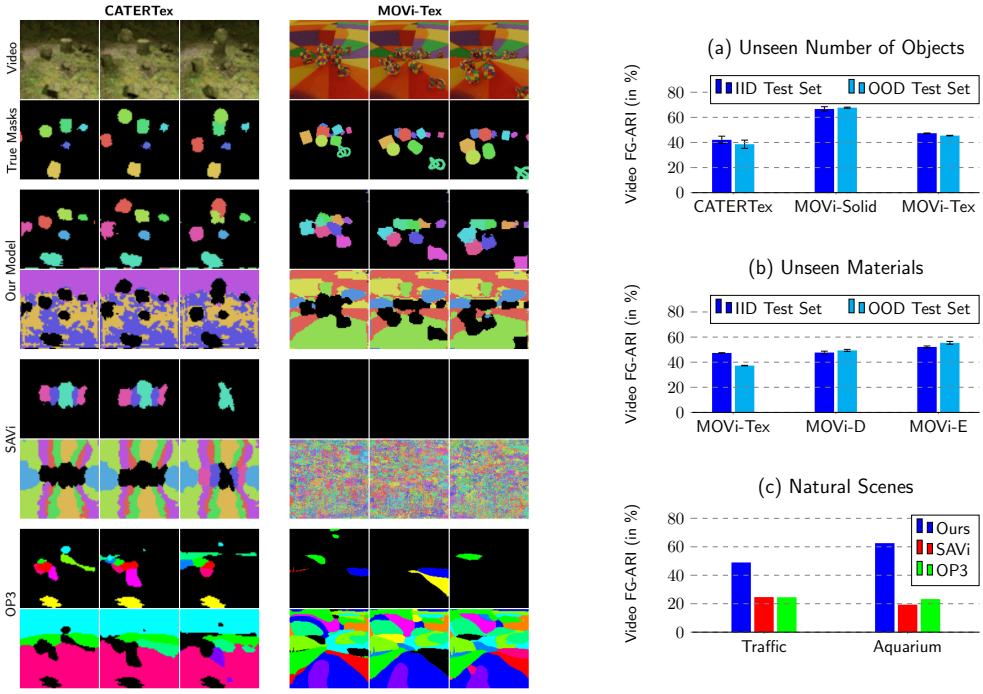


Figure 4: **Left: Visualization of Unsupervised Video Segmentation in CATERTex and MOVi-Tex.** We show the video frames and their true masks followed by foreground and background segmentation of each model. We separated the visualization of the foreground segments to make the result easier to interpret by applying a threshold on the segment area. See the supplementary material for more visualizations. **Right: Unsupervised Video Segmentation in OOD and Natural Datasets.** We plot the Video FG-ARI. In (a) and (b), we compare the performance of our model on IID and OOD test sets. In (c), we evaluate our performance in unsupervised video segmentation on natural scenes.

works Locatello et al. (2020); Kabra et al. (2021); Kipf et al. (2021), we consider only the foreground segments of the ground truth.

Video Segmentation Performance. In Figure 3, we see that in 5 out of 6 datasets, all of which are textured, STEVE significantly outperforms the baselines. In CATER, the baselines are able to perform comparably or slightly better than ours. As with the segmentation result of Section 5.1, this is not surprising given that CATER is visually simple enough for the baselines to handle well. We also observe the one major drawback of SLATE in these results i.e. SLATE cannot provide aligned slot representations for a video. This is because SLATE can only be applied to individual frames and the slots of each frame would be randomly permuted. We also show how the performance is affected by the length of the video. The models were trained on 3-length videos (6-length for CATER) while we evaluate on up to 24-length videos. As the likelihood of switching identities should increase

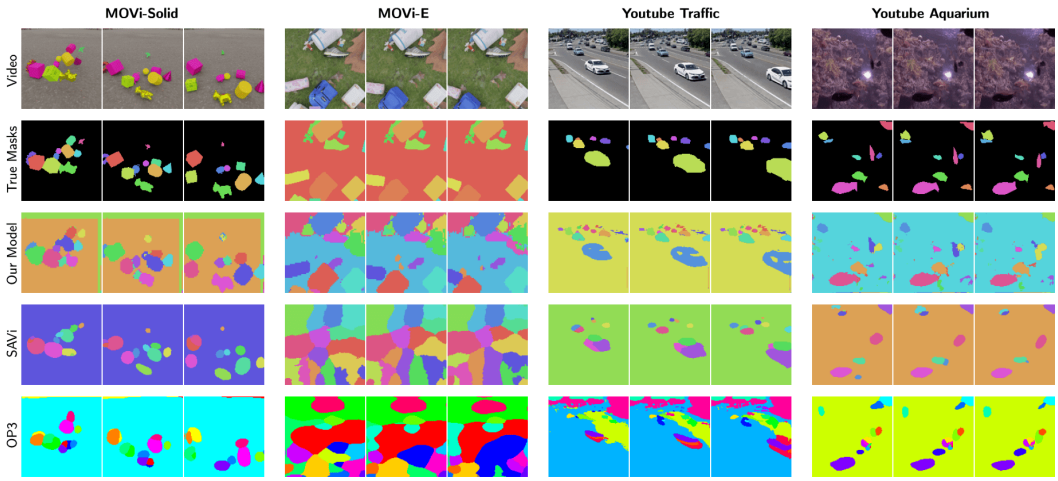


Figure 5: **Visualization of Unsupervised Video Segmentation in MOVi-Solid, MOVi-E, and Natural Scenes.** The rows (top to bottom) show the input video, the true segmentation followed by the predicted segments of the models. See the supplementary material for more visualizations.

with the video length, all models show some downward slope. Going from video length 6 to 24 in CATERTex, MOVi-D, and MOVi-E, our model deteriorates noticeably less than the baselines.

Out-of-Distribution Generalization. In Figure 4 (a) and (b), we show how well our trained model can generalize at test time to out-of-distribution videos which have more number of objects and unseen objects and materials. We see that our model generalizes well and retains a similar performance as that on the IID test set. One exception is MOVi-Tex in which our model generalizes well to more objects but suffers on novel textures. Due to the modular structure of the recurrent encoder, each slot binds to input objects quite independently. We conjecture that this modularity is responsible for good generalization to more objects. However, for new materials, generalization depends on how well the backbone CNN can generalize. As the training set of MOVi-Tex was generated using a library of 240 textures, it would not be surprising if this CNN provides poor features of unseen textures. This, we believe, is affecting the performance of our model.

Natural Scenes. In Figure 4 (c), we evaluate video segmentation on two natural datasets taken from real-world videos: Traffic and Aquarium. We see that our model performs significantly better than the baselines, approximately doubling the FG-ARI in the Traffic dataset and tripling the FG-ARI in the Aquarium dataset. Noteworthy is the visual complexity of the Aquarium dataset in which the color and texture of fish are strongly camouflaged against the background and seeing them requires careful inspection even for a human (see Figure 5). Despite this, we note that our model is able to deal with this challenge effectively.

5.3 Analysis

Diagnosing Slot Representation in STEVE. In previous sections, we focused on segmentation quality and used input attention masks of STEVE. However, input attention masks do not provide a direct indication of the information content of slots. We would like to diagnose how well the slots represent the object-centric structure of the scene. For this, we take the slots from the pre-trained encoder of STEVE and, without propagating the gradient to the encoder, we train a mixture-based decoder to reconstruct the observation from the slots. We call this model *STEVE-Diagnostic*. In Figure 6 (a), we compare the performance of unsupervised video segmentation with the baselines using the decoding masks of STEVE-Diagnostic. The decoder of STEVE-diagnostic is made to be exactly the same as that of SAVi for a fair comparison. We find that STEVE-Diagnostic performs better than the baselines in all textured datasets, in line with our earlier results in Figures 2 and 3. This shows that the slots in STEVE represent the object-centric structure better than the baselines. In Table 9 in Appendix C, we further analyze the slot contents by training a linear regressor to predict the object positions and find that STEVE performs favorably relative to the baselines.

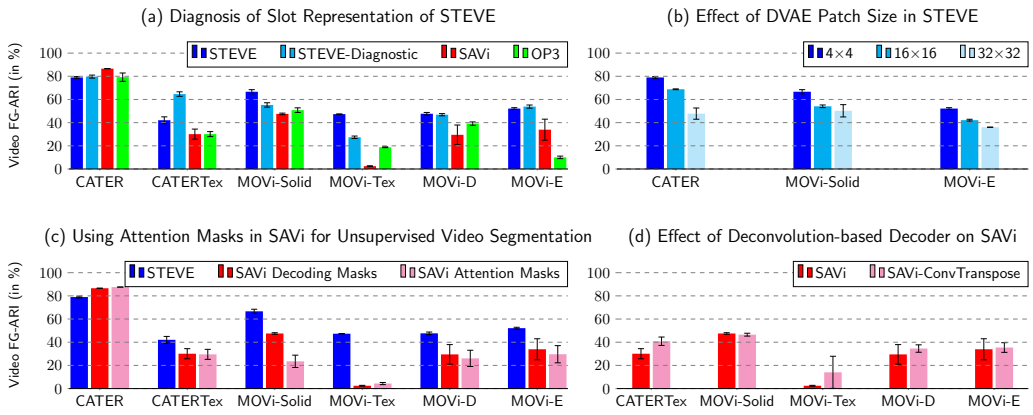


Figure 6: **Analysis.** Based on the analyses described in Section 5.3 we evaluate various aspects of the models. In all plots, we report Video FG-ARI computed on videos of length 6.

Effect of Granularity of dVAE on STEVE. In this section, we analyze the effect of granularity of discrete VAE on the segmentation performance of STEVE. In the default configuration of our model, the discrete VAE divides the image uniformly into patches of size 4×4 and represents each patch with one discrete token. That is, an image of size 128×128 would be represented by a sequence of length 1024. In this analysis, we test the effect on segmentation performance if dVAE represents the image using larger patches i.e. 16×16 and 32×32. In our results shown in Figure 6, we note that using larger patches results in a worsening of the performance. This suggests a possible scaling law that smaller patches lead to better performance. However, due to the prohibitive memory needs of the transformer on longer sequences, we were not able to experiment with smaller patches.

Effect of Transformer Capacity on STEVE. In Fig. 7 in the appendix, we analyze whether our segmentation performance is sensitive to the capacity of the decoder. For this, we test the effect of increasing the number of layers in the transformer decoder. We find that our segmentation performance is unaffected by this change, showing that our model is robust.

Comparison between Decoding Masks and Input Attention Masks in SAVi. In previous sections on unsupervised image and video decomposition, we used input attention masks of STEVE while for the baselines, we used the decoding masks. We would like to test how different are the performances of input attention masks and decoding masks. While OP3 only provides decoding masks, fortunately, SAVi provides both input masks and decoding masks which we compare in Figure 6 (c). We find that input attention masks tend to be worse than decoding masks of SAVi. Therefore, the attention masks of STEVE outperform both the input and decoding masks of the baseline SAVi.

Mixture-based decoding without Spatial Broadcast. Mixture-based decoding requires a CNN to decode the object image and the mask from each slot. To implement this CNN, the baselines prescribe using a Spatial Broadcast decoder, a type of CNN decoder that limits the expressiveness of the decoder to facilitate the discovery of objects. We would like to test if a more flexible CNN implemented completely using transposed convolutions would be enough to improve the performance or not. In Figure 6 (d), our results on the 5 textured datasets show that this change leads to a similar or slightly higher performance than the original decoder. However, the model still fails by splitting the image into fixed patches similarly to before and still performs significantly worse than our model. This shows that simply using a more flexible CNN with mixture-based decoding is not enough and the expressiveness of an auto-regressive transformer is important for achieving significant gains.

Computational Requirements of STEVE. In Table 6 in the Appendix, we empirically compare the computational requirements of STEVE and SAVi. We assess by how much the computational requirement change in going from the mixture-based to the transformer-based approach. We find that in videos with image size 64×64, STEVE requires similar resources as SAVi and about twice for image size 128×128. Given the quadratic memory complexity of transformers, the larger memory demand of STEVE is not surprising. However, one should also consider that the mixture-based

baselines fail almost completely on datasets like CATERTex, MOVi-Tex, MOVi-D, and MOVi-E, unlike ours. We also provide a big-O analysis of the memory costs in Appendix B.1.

Effect of Camera Motion. Our segmentation results on MOVi-D and MOVi-E in Figures 2 and 3 provide insight into the effect of camera motion as the MOVi-E dataset introduces a moving camera to MOVi-D. We observe that camera motion in MOVi-E leads to a slight improvement in performance over MOVi-D. We conjecture that this is because a moving camera causes relative motion between objects and can help better in bringing out patterns that tend to remain stable under motion.

Static Objects. We also note that our model is effective even when static objects are present in the scene such as in CATER, CATERTex, MOVi-D, and MOVi-E. In methods that rely purely on optical flow for discovering objects can suffer significantly with static objects. For MOVi-D and MOVi-E, Greff et al. (2022) report FG-ARI of 19.4 and 2.7 using SAVi trained with optical flow supervision. Compared to this, our model achieves a significantly better 47.67 and 52.15 on these datasets, respectively, despite the static objects and without using any supervision whatsoever.

6 Conclusion

In this paper, we study the effect of a powerful transformer-based slot decoder for unsupervised object-centric video learning. For this, we propose a simple model, STEVE, and demonstrate that it can deal with various complex and naturalistic videos without any supervision by significantly outperforming previous state-of-the-art baseline models. In experiments, we also find that although the slot-transformer decoder alone is an important component, for videos it is essential to learn also from temporal information for dealing with complex videos. The results of this paper and the SLATE paper jointly suggest using a powerful reconstruction decoder for unsupervised object-centric learning which is contrary to the traditional view advocating the use of weak mixture decoders. This raises a question to the community: *is accurate reconstruction from slots all we need to capture the objectness?* In the future, it seems worth investigating further in this direction. Also, because we choose to investigate a minimal architecture in this paper, it would be interesting to see what gains we could obtain additionally by exploring more advanced architectures. Lastly, it would be interesting to investigate applying this method to large-scale datasets. We further elaborate on these in Appendix F.

Acknowledgments and Disclosure of Funding

This work is supported by Brain Pool Plus (BP+) Program (No. 2021H1D3A2A03103645) and Young Researcher Program (No. 2022R1C1C1009443) through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT.

References

Anciukevicius, T., Lampert, C. H., and Henderson, P. Object-centric image generation with factored depths, locations, and appearances. *arXiv preprint arXiv:2004.00642*, 2020.

Bao, Z., Tokmakov, P., Jabri, A., Wang, Y.-X., Gaidon, A., and Hebert, M. Discovering objects that can move. *ArXiv*, abs/2203.10159, 2022.

Bapst, V., Sanchez-Gonzalez, A., Doersch, C., Stachenfeld, K. L., Kohli, P., Battaglia, P. W., and Hamrick, J. B. Structured agents for physical construction. *arXiv preprint arXiv:1904.03177*, 2019.

Besbinar, B. and Frossard, P. Self-supervision by prediction for object discovery in videos. *arXiv preprint arXiv:2103.05669*, 2021.

Bideau, P. and Learned-Miller, E. G. It's moving! a probabilistic model for causal motion segmentation in moving camera videos. *ArXiv*, abs/1604.00136, 2016.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Brox, T. and Malik, J. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010.

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.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *ICCV 2021 - International Conference on Computer Vision*, 2021.

Carvalho, W., Liang, A., Lee, K., Sohn, S., Lee, H., Lewis, R. L., and Singh, S. Reinforcement learning for sparse-reward object-interaction tasks in first-person simulated 3d environments. *arXiv preprint arXiv:2010.15195*, 2020.

Chen, C., Deng, F., and Ahn, S. Roots: Object-centric representation and rendering of 3d scenes. *Journal of Machine Learning Research*, 22(259):1–36, 2021.

Chen, M., Radford, A., Child, R., Wu, J. K., Jun, H., Luan, D., and Sutskever, I. Generative pretraining from pixels. In *ICML 2020: 37th International Conference on Machine Learning*, volume 1, pp. 1691–1703, 2020.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , and Vedaldi, A. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Crawford, E. and Pineau, J. Exploiting spatial invariance for scalable unsupervised object tracking. *arXiv preprint arXiv:1911.09033*, 2019a.

Crawford, E. and Pineau, J. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of AAAI*, 2019b.

Crawford, E. and Pineau, J. Learning 3d object-oriented world models from unlabeled videos. In *Workshop on Object-Oriented Learning at ICML*, 2020.

Creswell, A., Nikiforou, K., Vinyals, O., Saraiva, A., Kabra, R., Matthey, L., Burgess, C., Reynolds, M., Tanburn, R., Garnelo, M., et al. Alignnet: Unsupervised entity alignment. *arXiv preprint arXiv:2007.08973*, 2020.

Creswell, A., Kabra, R., Burgess, C., and Shanahan, M. Unsupervised object-based transition models for 3d partially observable environments. *arXiv preprint arXiv:2103.04693*, 2021.

Deng, F., Zhi, Z., Lee, D., and Ahn, S. Generative scene graph networks. In *International Conference on Learning Representations*, 2020.

Dittadi, A., Papa, S., De Vita, M., Schölkopf, B., Winther, O., and Locatello, F. Generalization and robustness implications in object-centric learning. *arXiv preprint arXiv:2107.00637*, 2021.

Du, Y., Smith, K., Ulman, T., Tenenbaum, J., and Wu, J. Unsupervised discovery of 3d physical objects from video. *arXiv preprint arXiv:2007.12348*, 2020.

Du, Y., Li, S., Sharma, Y., Tenenbaum, J., and Mordatch, I. Unsupervised learning of compositional energy concepts. *Advances in Neural Information Processing Systems*, 34, 2021a.

Du, Y., Smith, K. A., Ulman, T., Tenenbaum, J. B., and Wu, J. Unsupervised discovery of 3d physical objects from video. *ArXiv*, abs/2007.12348, 2021b.

Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I. Genesis: Generative scene inference and sampling with object-centric latent representations. In *ICLR 2020 : Eighth International Conference on Learning Representations*, 2020.

Engelcke, M., Jones, O. P., and Posner, I. Genesis-v2: Inferring unordered object representations without iterative refinement. *arXiv preprint arXiv:2104.09958*, 2021.

Eslami, S. A., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., and Hinton, G. E. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems*, pp. 3225–3233, 2016.

Ghasemipour, S. K. S., Freeman, D., David, B., Kataoka, S., Mordatch, I., et al. Blocks assemble! learning to assemble with large-scale structured reinforcement learning. *arXiv preprint arXiv:2203.13733*, 2022.

Girdhar, R. and Ramanan, D. CATER: A diagnostic dataset for Compositional Actions and TEmporal Reasoning. In *ICLR*, 2020.

Greff, K., Rasmus, A., Berglund, M., Hao, T., Valpola, H., and Schmidhuber, J. Tagger: Deep unsupervised perceptual grouping. In *Advances in Neural Information Processing Systems*, pp. 4484–4492, 2016.

Greff, K., van Steenkiste, S., and Schmidhuber, J. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, pp. 6691–6701, 2017.

Greff, K., Kaufmann, 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. *arXiv preprint arXiv:1903.00450*, 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., Gnanapragasam, D., Golemo, F., Herrmann, C., Kipf, T., Kundu, A., Lagun, D., Laradji, I. H., Liu, H.-T., Meyer, H., Miao, Y., Nowrouzezahrai, D., Oztireli, C., Pot, E., Radwan, N., Rebain, D., Sabour, S., Sajjadi, M. S. M., Sela, M., Sitzmann, V., Stone, A., Sun, D., Vora, S., Wang, Z., Wu, T., Yi, K. M., Zhong, F., and Tagliasacchi, A. Kubric: A scalable dataset generator. *ArXiv*, abs/2203.03570, 2022.

He, Z., Li, J., Liu, D., He, H., and Barber, D. Tracking by animation: Unsupervised learning of multi-object attentive trackers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1318–1327, 2019.

Henderson, P. and Lampert, C. H. Unsupervised object-centric video generation and decomposition in 3d. *ArXiv*, abs/2007.06705, 2020.

Janai, J., Güney, F., Ranjan, A., Black, M. J., and Geiger, A. Unsupervised learning of multi-frame optical flow with occlusions. In *ECCV*, 2018.

Jiang, J. and Ahn, S. Generative neurosymbolic machines. *arXiv preprint arXiv:2010.12152*, 2020.

Jiang, J., Janghorbani, S., de Melo, G., and Ahn, S. SCALOR: generative world models with scalable object representations. In *International Conference on Learning Representations*, 2020.

Jonschkowski, R., Stone, A., Barron, J. T., Gordon, A., Konolige, K., and Angelova, A. What matters in unsupervised optical flow. In *ECCV*, 2020.

Kabra, R., Zoran, D., Erdogan, G., Matthey, L., Creswell, A., Botvinick, M., Lerchner, A., and Burgess, C. P. Simone: View-invariant, temporally-abstracted object representations via unsupervised video decomposition. *arXiv preprint arXiv:2106.03849*, 2021.

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

Kipf, T., van der Pol, E., and Welling, M. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

Kipf, T., Elsayed, G. F., Mahendran, A., Stone, A., Sabour, S., Heigold, G., Jonschkowski, R., Dosovitskiy, A., and Greff, K. Conditional Object-Centric Learning from Video. *arXiv preprint arXiv:2111.12594*, 2021.

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

Kossen, J., Stelzner, K., Hussing, M., Voelcker, C., and Kersting, K. Structured object-aware physics prediction for video modeling and planning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=B1e-kxSKDH`.

Lai, Z. and Xie, W. Self-supervised learning for video correspondence flow. *ArXiv*, abs/1905.00875, 2019.

Lai, Z., Lu, E., and Xie, W. Mast: A memory-augmented self-supervised tracker. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6478–6487, 2020.

Lamb, A., He, D., Goyal, A., Ke, G., Liao, C.-F., Ravanelli, M., and Bengio, Y. Transformers with competitive ensembles of independent mechanisms. In *arXiv e-prints*, 2021.

Lin, Z., Wu, Y.-F., Peri, S. V., Jiang, J., and Ahn, S. Improving generative imagination in object-centric world models. In *International Conference on Machine Learning*, pp. 4114–4124, 2020a.

Lin, Z., Wu, Y.-F., Peri, S. V., Sun, W., Singh, G., Deng, F., Jiang, J., and Ahn, S. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. In *International Conference on Learning Representations*, 2020b.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention, 2020.

Löwe, S., Greff, K., Jonschkowski, R., Dosovitskiy, A., and Kipf, T. Learning object-centric video models by contrasting sets. *arXiv preprint arXiv:2011.10287*, 2020.

Lowe, S., Lippe, P., Rudolph, M. R., and Welling, M. Complex-valued autoencoders for object discovery. *ArXiv*, abs/2204.02075, 2022.

Mambelli, D., Träuble, F., Bauer, S., Schölkopf, B., and Locatello, F. Compositional multi-object reinforcement learning with linear relation networks. *arXiv preprint arXiv:2201.13388*, 2022.

Oh, S. W., Lee, J.-Y., Xu, N., and Kim, S. J. Video object segmentation using space-time memory networks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9225–9234, 2019.

Ren, Z., Yan, J., Ni, B., Liu, B., Yang, X., and Zha, H. Unsupervised deep learning for optical flow estimation. In *AAAI*, 2017.

Savarese, P., Kim, S. S. Y., Maire, M., Shakhnarovich, G., and McAllester, D. Information-theoretic segmentation by inpainting error maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4029–4039, 2021.

Schölkopf, B., Locatello, F., Bauer, S., Ke, N. R., Kalchbrenner, N., Goyal, A., and Bengio, Y. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Singh, G., Peri, S., Kim, J., Kim, H., and Ahn, S. Structured world belief for reinforcement learning in pomdp. In *International Conference on Machine Learning*, pp. 9744–9755. PMLR, 2021.

Singh, G., Deng, F., and Ahn, S. Illiterate dall-e learns to compose. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=h0OYV0We3oh`.

Stanić, A. and Schmidhuber, J. R-sqair: relational sequential attend, infer, repeat. *arXiv preprint arXiv:1910.05231*, 2019.

Stelzner, K., Kersting, K., and Kosiorek, A. R. Decomposing 3d scenes into objects via unsupervised volume segmentation. *arXiv preprint arXiv:2104.01148*, 2021.

Stone, A., Maurer, D., Ayvaci, A., Angelova, A., and Jonschkowski, R. Smurf: Self-teaching multi-frame unsupervised raft with full-image warping. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3886–3895, 2021.

Tangemann, M., Schneider, S., von Kügelgen, J., Locatello, F., Gehler, P., Brox, T., Kümmerer, M., Bethge, M., and Schölkopf, B. Unsupervised object learning via common fate. *arXiv preprint arXiv:2110.06562*, 2021.

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. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Veerapaneni, R., Co-Reyes, J. D., Chang, M., Janner, M., Finn, C., Wu, J., Tenenbaum, J. B., and Levine, S. Entity abstraction in visual model-based reinforcement learning. *arXiv preprint arXiv:1910.12827*, 2019.

von Kügelgen, J., Ustyuzhaninov, I., Gehler, P., Bethge, M., and Schölkopf, B. Towards causal generative scene models via competition of experts. *arXiv preprint arXiv:2004.12906*, 2020.

Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., and Murphy, K. P. Tracking emerges by colorizing videos. In *ECCV*, 2018.

Wang, N., Song, Y., Ma, C., gang Zhou, W., Liu, W., and Li, H. Unsupervised deep tracking. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1308–1317, 2019a.

Wang, X., Jabri, A., and Efros, A. A. Learning correspondence from the cycle-consistency of time. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2561–2571, 2019b.

Wang, Y., Yang, Y., Yang, Z., Zhao, L., and Xu, W. Occlusion aware unsupervised learning of optical flow. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4884–4893, 2018.

Wang, Y., Shen, X., Hu, S. X., Yuan, Y., Crowley, J. L., and Vaufreydaz, D. Self-supervised transformers for unsupervised object discovery using normalized cut. *ArXiv*, abs/2202.11539, 2022.

Watters, N., Matthey, L., Bosnjak, M., Burgess, C. P., and Lerchner, A. Cobra: Data-efficient model-based rl through unsupervised object discovery and curiosity-driven exploration. *arXiv preprint arXiv:1905.09275*, 2019.

Weis, M. A., Chitta, K., Sharma, Y., Brendel, W., Bethge, M., Geiger, A., and Ecker, A. S. Unmasking the inductive biases of unsupervised object representations for video sequences. *arXiv preprint arXiv:2006.07034*, 2, 2020.

Wu, Y.-F., Yoon, J., and Ahn, S. Generative video transformer: Can objects be the words? In *International Conference on Machine Learning*, pp. 11307–11318. PMLR, 2021.

Yang, C., Lamdouar, H., Lu, E., Zisserman, A., and Xie, W. Self-supervised video object segmentation by motion grouping. *arXiv: Computer Vision and Pattern Recognition*, 2021.

Yang, Y., Loquercio, A., Scaramuzza, D., and Soatto, S. Unsupervised moving object detection via contextual information separation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 879–888, 2019.

Yang, Y., Chen, Y., and Soatto, S. Learning to manipulate individual objects in an image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6558–6567, 2020.

Yu, P., Xie, S., Ma, X., Zhu, Y., Wu, Y. N., and Zhu, S.-C. Unsupervised foreground extraction via deep region competition. In *NeurIPS*, 2021.

Zhou, H., Kadav, A., Lai, F., Niculescu-Mizil, A., Min, M. R., Kapadia, M., and Graf, H. P. Hopper: Multi-hop transformer for spatiotemporal reasoning. *arXiv preprint arXiv:2103.10574*, 2021.

Zhu, F., Zhang, L., Fu, Y., Guo, G., and Xie, W. Self-supervised video object segmentation. *ArXiv*, abs/2006.12480, 2020.

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.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] In supplementary material, we provide the architectural details and the hyperparameters used for our experiments. We will share the code and our proposed datasets at `https://sites.google.com/view/slot-transformer-for-videos`.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We provide the amount of resources used in Section B.1 in Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We will share them upon acceptance.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Datasets

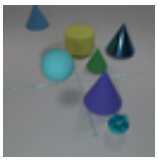
	Sample Frame	Textured FG	Textured BG	Static Objects	Camera Motion	Natural
CATER (Girdhar & Ramanan, 2020)		No	No	**Yes**	**Yes**	No
CATERTex		**Yes**	**Yes**	**Yes**	No	No
MOVi-Solid		No	**Yes**	No	No	No
MOVi-Tex		**Yes**	**Yes**	No	No	No
MOVi-D (Greff et al., 2022)		**Yes**	**Yes**	**Yes**	No	No
MOVi-E (Greff et al., 2022)		**Yes**	**Yes**	**Yes**	**Yes**	No
YT-Traffic		**Yes**	**Yes**	**Yes**	No	**Yes**
YT-Aquarium		**Yes**	**Yes**	No	No	**Yes**

Table 1: Characteristics of the video datasets used in our evaluation.

Dataset Descriptions. We generated MOVi-Solid dataset using Kubric (Greff et al., 2022) which introduces textured background and more complex shapes to the MOVi dataset Kipf et al. (2021). For this, we used 100 HDR backgrounds and Kubric's KuBasic shapes. We also generate two test sets: one which has the same number of objects as the training set (3-10 objects) and other which has higher out-of-distribution number of objects (11-12 objects). Similarly to MOVi-Solid, for MOVi-Tex, we use Kubric to generate the dataset. In MOVi-Tex, we modify our MOVi-Solid dataset to have complex textures applied to the objects taken from the Describable Textures dataset Cimpoi et al. (2014). Describable Textures dataset provides several categories of textures. We used 240 textures each for generating the training set. For this, we used 120 textures that are from the banded category and 120 textures that are from the chequered category. We also generated two out-of-distribution test sets: one which has the same number of objects as the training set (3-10 objects) but never-before-seen textures and the other which as the same textures but higher number of objects (11-12 objects). For the out-of-distribution test set with never-before-seen textures, 120 textures were taken from the grid category and 120 were taken from the grooved category of the Describable Textures dataset. For CATER, MOVi-D and MOVi-E, we used the standard training and test splits provided in the respective releases of CATER (with masks) and MOVi datasets. We generated CATERTex by integrating the codes for CATER and CLEVRTex (Girdhar & Ramanan, 2020; Karazija et al., 2021). We generated two sets: the training and the IID test sets which contain 3-8 objects and an out-of-distribution test set which contains 9-10 objects. We collected the Youtube datasets by downloading a 6-hour long video streams, center-cropping and resizing them. The Youtube-Traffic dataset was generated from a video feed collected by a traffic camera that watches a fixed intersection. The camera does not move during the video. The lighting conditions change slightly over time and this has a slight effect on the background and the overall scene from one clip to another. The vehicles appear to show diversity in size, color, and type. The Youtube-Aquarium dataset was generated from a video collected by a static camera mounted inside a fish exhibit. The background, lighting conditions, and other characteristics remain stationary throughout the video. The size, color, and types of fish show some diversity. As these were collected from Youtube and did not have ground truth masks, we manually annotated 50 videos of the test set. We will release the training and the test splits of all the datasets.

B STEVE: Architecture Details

Recurrent Slot Encoder. The design of our slot-based recurrent encoder $f_\phi^{\text{slot-rnn}}$ is inspired by Kipf et al. (2021). At the start of an episode, we provide initial slots $\mathbf{s}_0$ by randomly sampling from a Gaussian distribution with learned mean and variance.

$$\mathbf{s}_{0,n} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}), \qquad \forall n \in \{1, \ldots, N\}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are learned parameters. For each input frame $\mathbf{x}_t$, we compute an encoding in the form of a feature map using a backbone CNN. The architecture of this CNN is described in Table 2. The resulting feature map has size $H_{\text{enc}} \times W_{\text{enc}} \times D_{\text{enc}}$. It is then flattened on spatial dimensions to obtain

Layer	Kernel Size	Stride	Padding	Channels	Activation
Conv	5×5	1 (2)	2	32 (64)	ReLU
Conv	5×5	1	2	32 (64)	ReLU
Conv	5×5	1	2	32 (64)	ReLU
Conv	5×5	1	2	32 (64)	None

Table 2: Configuration of the CNN encoder used in our model. The values in parentheses are applicable for image size 128.

a set of $H_{\text{enc}} \cdot W_{\text{enc}}$ features. A positional embedding $\mathbf{p}$ is added to these followed by a 2-layer MLP. With these, we obtain a set of input features $\mathbf{e}_t = \{\mathbf{e}_{t,1}, \mathbf{e}_{t,2}, \ldots, \mathbf{e}_{t,HW}\}$. The positional embedding is computed by applying a linear map to the 4-dimensional spatial position as introduced by Locatello et al. (2020) and Kipf et al. (2021). Next, the slots of the previous time-step $\mathbf{s}_{t-1}$ perform attention on the features $\mathbf{e}_t$ and use the attention result to update the slots to $\mathbf{s}_t$. For this, the slots in $\mathbf{s}_{t-1}$ first compute the attention weights $\mathcal{A}_t = \{\mathcal{A}_{t,1}, \ldots, \mathcal{A}_{t,N}\}$ over the input features. Using the attention weights, the attention result $\mathbf{r}_{t,n}$ is computed for each slot via an attention-weighted sum of input

features.

$$\mathcal{A}_t = \texttt{softmax}\left(\frac{q(\mathbf{s}_{t-1}) \cdot k(\mathbf{e}_t)^T}{\sqrt{D}}, \text{dim} = \texttt{slots}\right) \qquad \mathbf{r}_{t,n} = \frac{\sum_{i=1}^{HW} \mathcal{A}_{t,n,i} \cdot v(\mathbf{e}_{t,i})}{\sum_{j=1}^{HW} \mathcal{A}_{t,n,j}}.$$

where q, k and v are linear projections and D is the output size of the projections. The attention result $\mathbf{r}_{t,n}$ is then used to update the respective slot: $\tilde{\mathbf{s}}_{t,n} \leftarrow f_{\text{RNN}}(\mathbf{r}_{t,n}, \mathbf{s}_{t-1,n})$ where the update function f_{RNN} is implemented as a GRU (Cho et al., 2014). For additional expressiveness, a 2-layer MLP (with residual connection) and LayerNorm is applied on the output of the GRU. In practice, these attention and update steps can be executed multiple times per frame. We use 2 iterations per frame in our experiments. Lastly, the slots are made to interact using an interaction function $\mathbf{s}_t \leftarrow f_\phi^{\text{interact}}(\tilde{\mathbf{s}}_t)$ implemented using a 1-layer transformer with 4 heads. Following Kipf et al. (2021), we use the slots $\tilde{\mathbf{s}}_t$ before the interaction step for reconstruction. For fairness of comparison between our model STEVE and SAVi, we use the same implementation of this backbone recurrent slot encoder in both models.

Discrete VAE. The implementation of our discrete VAE is based on that proposed by (Singh et al., 2022). It consists of an encoder network f_ϕ^{dVAE} and a decoder network g_θ^{dVAE}.

The encoder network essentially divides the whole image $\mathbf{x}_t$ into patches of size 4×4 and encodes each patch as one discrete token belonging to a vocabulary $\mathcal{V}$. We take the size of the vocabulary to be 4096 in our experiments. This is achieved in parallel for all patches applying a convolutional neural network on the image of size $H \times W \times 3$ as follows.

Layer	Kernel Size	Stride	Padding	Channels	Activation
Conv	4×4	4	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	$\lvert\mathcal{V}\rvert$	None

Table 3: Configuration of discrete VAE encoder used in our model.

This results in a feature map of size $(H/4) \times (W/4) \times |\mathcal{V}|$. Each cell $i \in 1, \ldots, L$ (where $L = HW/16$) of this feature map can be seen as a vector $\mathbf{o}_{t,i}^{\text{dVAE}}$ containing log-probabilities over $|\mathcal{V}|$ vocabulary tokens. For each cell i, a discrete token is sampled by constructing a categorical distribution using these log-probabilities.

$$\mathbf{z}_{t,i} \sim \text{Categorical}(\mathbf{o}_{t,i}^{\text{dVAE}}).$$

In the discrete VAE decoder, the input is a discrete feature map of size $(H/4) \times (W/4) \times |\mathcal{V}|$ whose each cell is a one-hot vector $\mathbf{z}_i$ and the decoder tries to reconstruct the original image from this discrete feature map input. This mapping is implemented using a convolutional neural network as described below, finally outputting an image of size $H \times W \times 3$. Finally, the whole network is trained by minimizing a image reconstruction loss implemented using squared-error.

$$\mathcal{L}_{\text{dVAE}} = \sum_{t=1}^{T} \|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_2^2, \qquad \mathbf{z}_{t,1}, \ldots \mathbf{z}_{t,L} = f_\phi^{\text{dVAE}}(\mathbf{x}_t), \qquad \hat{\mathbf{x}}_t = g_\theta^{\text{dVAE}}(\mathbf{z}_{t,1}, \ldots \mathbf{z}_{t,L}).$$

As there is a discrete sampling step required, we use the Gumbel-Softmax relaxation during training while we use hard sampling for generating discrete targets for the transformer, following Singh et al. (2022). The temperature parameter for the Gumbel-Softmax is decayed from 1.0 to 0.1 in the first 30K iterations of training as prescribed by Singh et al. (2022).

Transformer Decoder. We use the standard design of transformer as introduced by Vaswani et al. (2017) and adopted by Singh et al. (2022). Like SLATE, we use learned positional embeddings for each position $i \in \{1, \ldots, L\}$ in the sequence.

Layer	Kernel Size	Stride	Padding	Channels	Activation
Conv	1×1	1	0	64	ReLU
Conv	3×3	1	1	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	256	ReLU
PixelShuffle (`upscale_factor=2`)	-	-	-	-	-
Conv	3×3	1	1	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	64	ReLU
Conv	1×1	1	0	256	ReLU
PixelShuffle (`upscale_factor=2`)	-	-	-	-	-
Conv	1×1	1	0	3	None

Table 4: Configuration of discrete VAE decoder used in our model.

During training, the transformer decoder takes the slots and the dVAE tokens as input and learns to predict the next token. We learn to do this next-token prediction for all tokens in parallel by applying causal masking in the style of GPT. In further detail, the dVAE tokens given as input to the transformer are obtained via argmax on the logits output by the dVAE encoder. That is, we take the dVAE logits $\mathbf{o}_{t,i}^{\text{dVAE}}$ and first perform argmax. For this argmax index, we retrieve a learned embedding from a codebook. To these retrieved embeddings, we add learned positional encodings and the resulting embeddings are then given as input to the transformer decoder. Similarly to SLATE, we do not let gradient be propagated across the argmax operation. The prediction targets are the argmax indices of the next dVAE token which we learn to predict using a cross-entropy loss.

Difference between SLATE and STEVE. The key difference between STEVE and SLATE is that STEVE temporally extends SLATE by adopting a recurrent slot encoder. In doing this, STEVE retains the same decoder architecture as that of SLATE.

Training. We trained all models upto a maximum of 200K training iterations (except for CATERTex for which we trained all models to 400K iterations). These amounted to training time of approximately 2 days. Similarly to SLATE, we applied a learning rate warmup from 0 to the peak learning rate in the first 30K training iterations. After that we decay the learning rate exponentially with a half-life of 250K training iterations.

The loss function is $\mathcal{L}_{\text{STEVE}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{dVAE}}$ as described in Section 3. The gradients from the $\mathcal{L}_{\text{dVAE}}$ loss are only used to train the dVAE encoder and the dVAE decoder. The gradients from the $\mathcal{L}_{\text{CE}}$ cross-entropy loss are only used for training the slot encoder and the transformer decoder. This is due to the fact that we do not let the gradients be propagated accross the argmax operation used to obtain the discrete representation of the input frame. All 3 components i.e. the dVAE, the slot encoder, and the transformer are trained simultaneously in a single training run.

B.1 Computational Requirements

In Table 6, we empirically compare the computational requirements of STEVE and SAVi. We assess by how much the computational requirement change in going from the mixture-based to the transformer-based approach. We find that in videos with image size 64×64, STEVE requires similar resources as SAVi and about twice for image size 128×128. Given the quadratic memory complexity of transformers, the larger memory demand of STEVE is not surprising. However, one should also consider that the mixture-based baselines fail almost completely on datasets like CATERTex, MOVi-Tex, MOVi-D, and MOVi-E, unlike ours. Another feature of our method is that the memory demand of transformer decoder is approximately constant in the number of slots but linear for mixture-based decoders.

GPU Resources. We performed our experiments on Quadro RTX 8000 GPUs. We used 2 GPUs per each training seed, each of which took approximately 2-3 days to complete training.

Module	Hyperparameter	Image Size	
		64	128
General	Batch Size	24	24
	Episode Length	3	3
	Training Steps	200000	200000
Encoder	Corrector Iterations	2	2
	Slot Size	192	192
	MLP hidden size	192	192
	# Predictor Blocks	1	1
	# Predictor Heads	4	4
	Learning Rate	0.0001	0.0001
Transformer Decoder	# Decoder Blocks	4	8
	# Decoder Heads	4	4
	Hidden Size	192	192
	Dropout	0.1	0.1
	Learning Rate	0.0003	0.0003
DVAE	Learning Rate	0.0003	0.0003
	Patch Size	4×4 pixels	4×4 pixels
	Vocabulary Size	4096	4096
	Temperature Start	1.0	1.0
	Temperature End	0.1	0.1
	Temperature Decay Steps	30000	30000

Table 5: Hyperparameters of STEVE used in our experiments.

Image Size	Memory (GB)		Seconds per iteration	
	SAVi	STEVE	SAVi	STEVE
64	9.8	10.7	0.24	0.20
128	29.8	57.8	0.53	0.74

Table 6: Computational requirements of STEVE and SAVi. The configurations of STEVE are as described in Tables 5 and 2. The configurations of SAVi are matched with STEVE except for the its decoder which is a 4-layer spatial broadcast network (as described in the original SAVi paper). Here, all values are computed using same the batch size and episode length (taken to be 24 and 3, respectively) for an informative comparison.

Big-O Analysis. The memory complexity of the transformer decoder is dependent on the cost of self-attention between all the dVAE tokens and the cross-attention between the dVAE tokens and the slots. Let the number of dVAE tokens be L and the number of slots be N. Then the number of dVAE tokens is $L = HW/K^2$ where H and W are image height and width, respectively; and K is the dVAE patch size. To perform self-attention among all L tokens requires a memory complexity of $O(L^2)$. Furthermore, the L tokens also perform cross-attention on the N slots which leads to an attention matrix with memory cost $O(LN)$. Adding these attention costs leads to a total cost of $\approx O(L^2) + O(LN)$. But since it is common to have just a few slots N (e.g. $N = 11$ slots for CATER) and a much larger number of tokens (e.g. $L = 256$ tokens for CATER), the first term $O(L^2)$ is typically the dominant one in comparison to the other term $O(LN)$. Replacing $L = HW/K^2$ gives an approximate cost of $O((HW)^2/K^4)$. This can be contrasted with the memory cost of mixture decoder. Each object component can be decoded using a CNN with a cost of $O(HW)$ and there are N such components. Thus, the total cost is approximately $O(HWN)$. These suggest that as we continue of scale up the image size H and W (keeping everything else same), the memory needs of the transformer decoder would significantly surpass that of the mixture decoder.

B.2 Mixture-based Decoding without Spatial Broadcast

In Section 5.3, we performed an ablation of SAVi where we replaced its original Spatial Broadcast decoder with a more flexible CNN implemented entirely using transposed convolutions. In Tables 7 and 8, we describe these two CNN architectures in detail. We note that the decoder used in

Layer	Kernel Size	Stride	Channels	Activation
Spatial Broadcast	-	-	`slot_size`	-
Positional Encodings	-	-	`slot_size`	-
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	1 (2)	64	ReLU

Table 7: Configuration of the CNN of the Spatial Broadcast decoder used in SAVi (Kipf et al., 2021). The values in parentheses are applicable for image size 128.

Layer	Kernel Size	Stride	Channels	Activation
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	2	64	ReLU
ConvTranspose	5×5	1 (2)	64	ReLU

Table 8: Configuration of the CNN decoder implemented only using transposed convolutions which is used in our ablation of SAVi in Section 5.3. The values in parentheses are applicable for image size 128.

the ablation of SAVi has more layers and is therefore more expressive. It applies 7 transposed convolution layers while the original decoder had 4. Furthermore, unlike the original decoder that uses a Spatial Broadcast in the first layer which is intended to lower its flexibility, our ablation of SAVi uses transposed convolutions in all the layers of the CNN decoder.

C Additional Results.

In this section, we provide additional evaluation results for our model.

	STEVE	SAVi	OP3
CATER	**0.911** $\pm$ 0.015	0.818 $\pm$ 0.025	0.682 $\pm$ 0.045
CATERTex	**0.755** $\pm$ 0.024	0.224 $\pm$ 0.051	0.299 $\pm$ 0.055
MOVi-Solid	0.848 $\pm$ 0.034	**0.875** $\pm$ 0.005	0.568 $\pm$ 0.042
MOVi-Tex	**0.741** $\pm$ 0.011	-3.085 $\pm$ 0.236	0.040 $\pm$ 0.036
MOVi-D	**0.589** $\pm$ 0.008	0.391 $\pm$ 0.12	0.316 $\pm$ 0.028
MOVi-E	**0.567** $\pm$ 0.032	0.436 $\pm$ 0.113	0.216 $\pm$ 0.017

Table 9: **Comparison of property prediction performance with the baselines.** Given a frozen slot, we predict the 2D position coordinates of the object using linear regression. For training the regressors, we match the slots with the ground truth position labels via Hungarian matching using IoU score between the true and the predicted masks. We report the R^2 score, similarly to Kabra et al. (2021). We note that our model outperforms the baselines significantly, except in MOVi-Solid where our performance is comparable to that of SAVi.

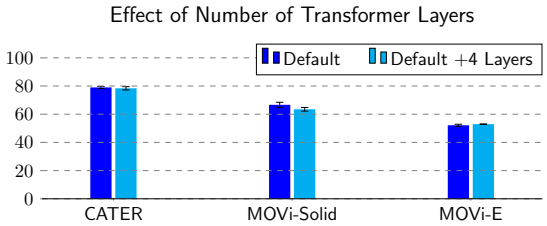


Figure 7: Effect of number of layers in the transformer decoder. The aim of this experiment is to test how susceptible is our performance to the flexibility of the transformer decoder. We compare our default transformer decoder configuration with a transformer decoder which has 4 additional layers in it. As this introduces more parameters to the decoder, it makes the decoder more flexible. We report the FG-ARI (in %) on videos of length 6. We find that the performances are comparable in both configurations. This shows that a careful tuning of decoder strength is not needed and this can be seen as a benefit of our model.

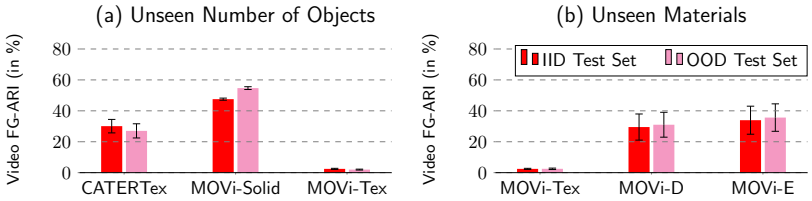


Figure 8: Out-of-distribution generalization to unseen number of objects and unseen materials in SAVi. We report the FG-ARI (in %) on videos of length 6.

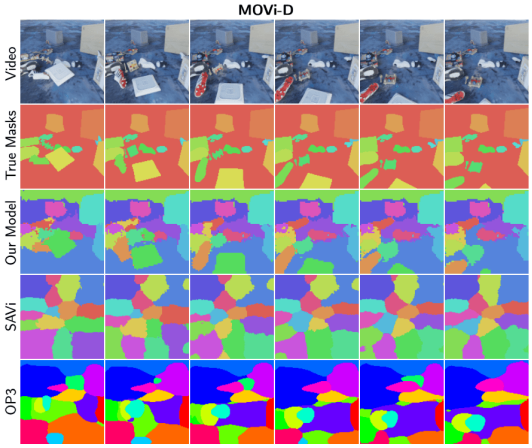


Figure 9: **Unsupervised Video Segmentation in MOVi-D.** We qualitatively compare video segmentation of STEVE with the baselines SAVi and OP3. The rows show the input video, the true segmentation, followed by the predicted segments of STEVE and the baselines.

Figure 10: Additional Samples of Unsupervised Video Segmentation in MOVi-E.

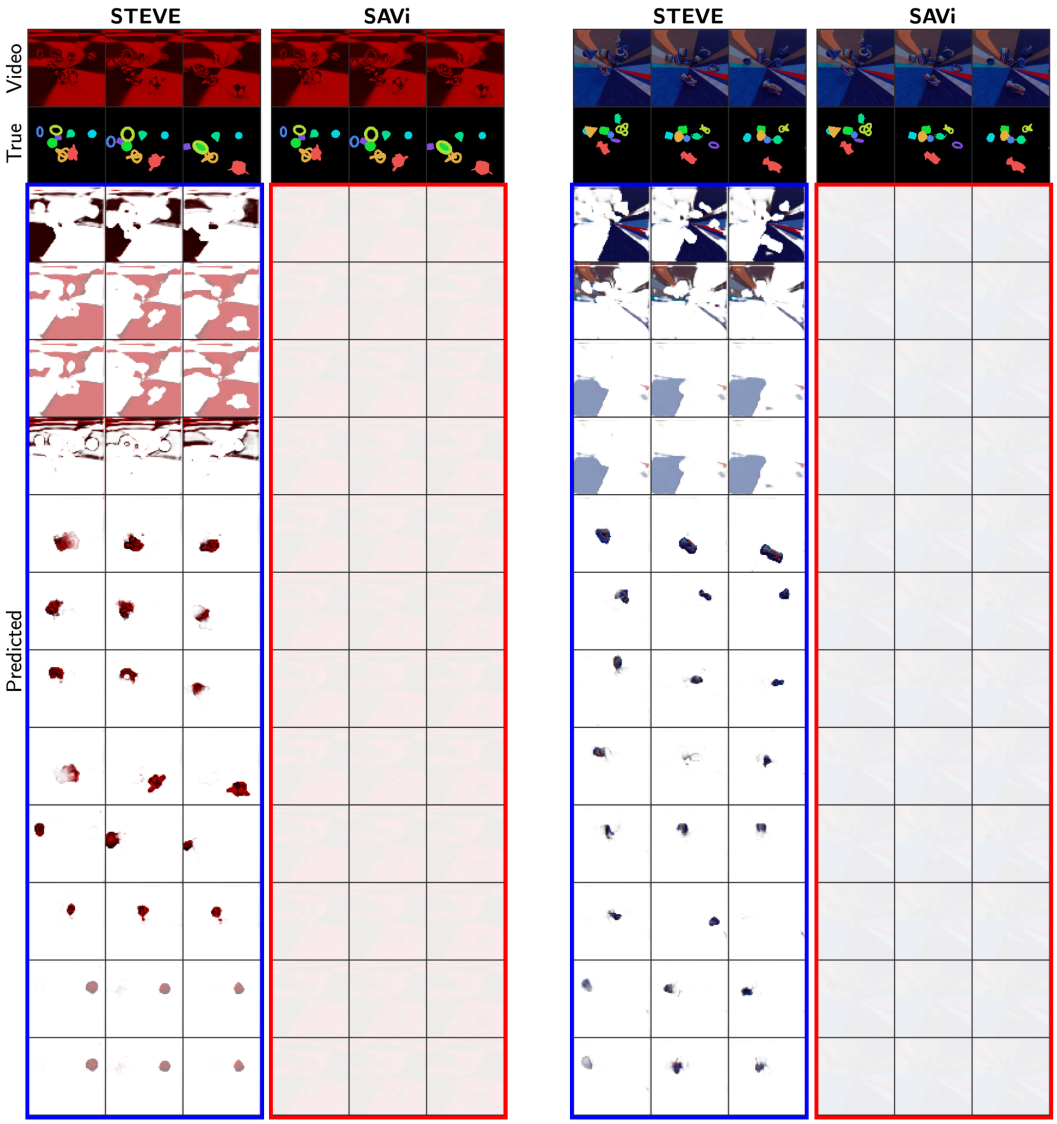


Figure 11: Additional Samples of Unsupervised Video Segmentation in MOVi-Tex.

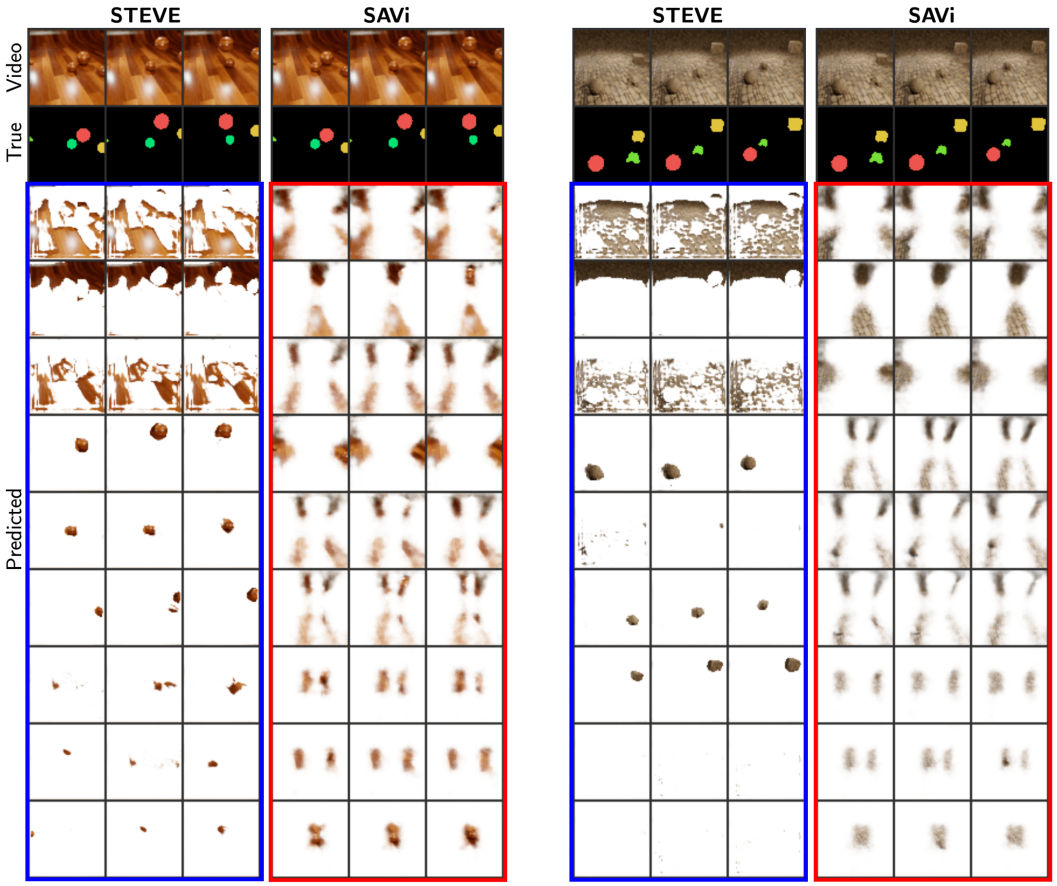


Figure 12: Additional Samples of Unsupervised Video Segmentation in CATERTex.

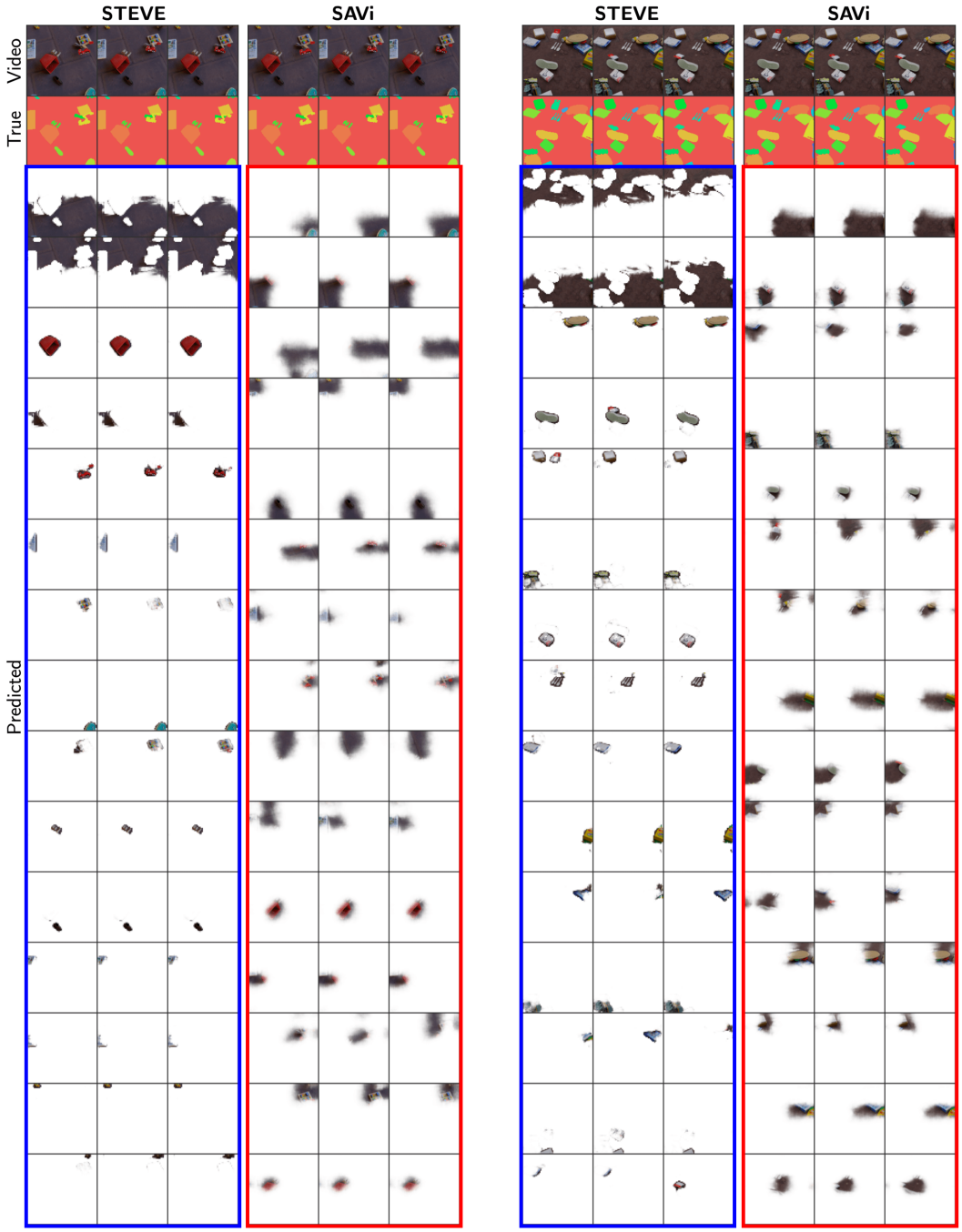


Figure 13: Additional Samples of Unsupervised Video Segmentation in MOVi-D.

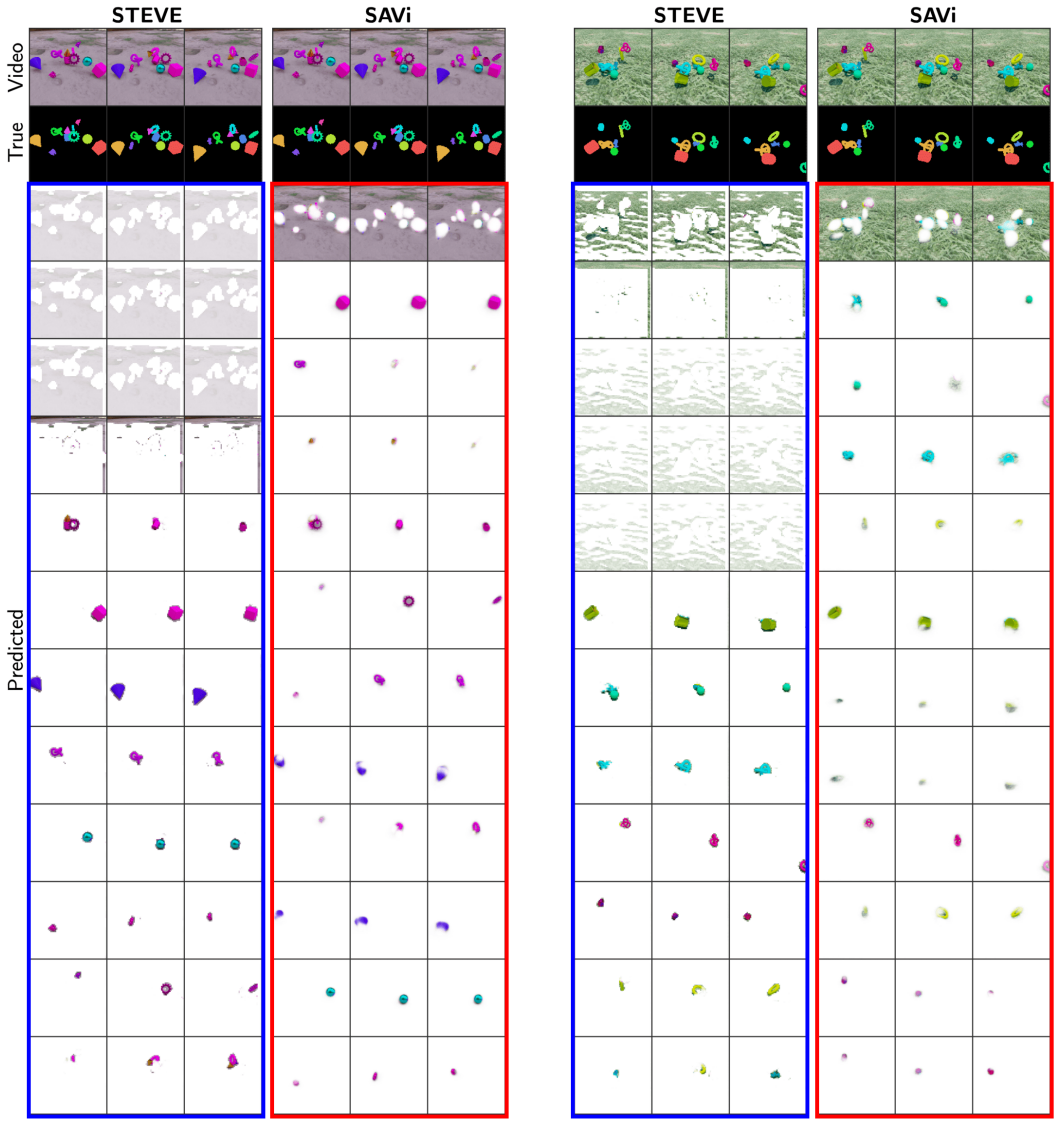


Figure 14: Additional Samples of Unsupervised Video Segmentation in MOVi-Solid.

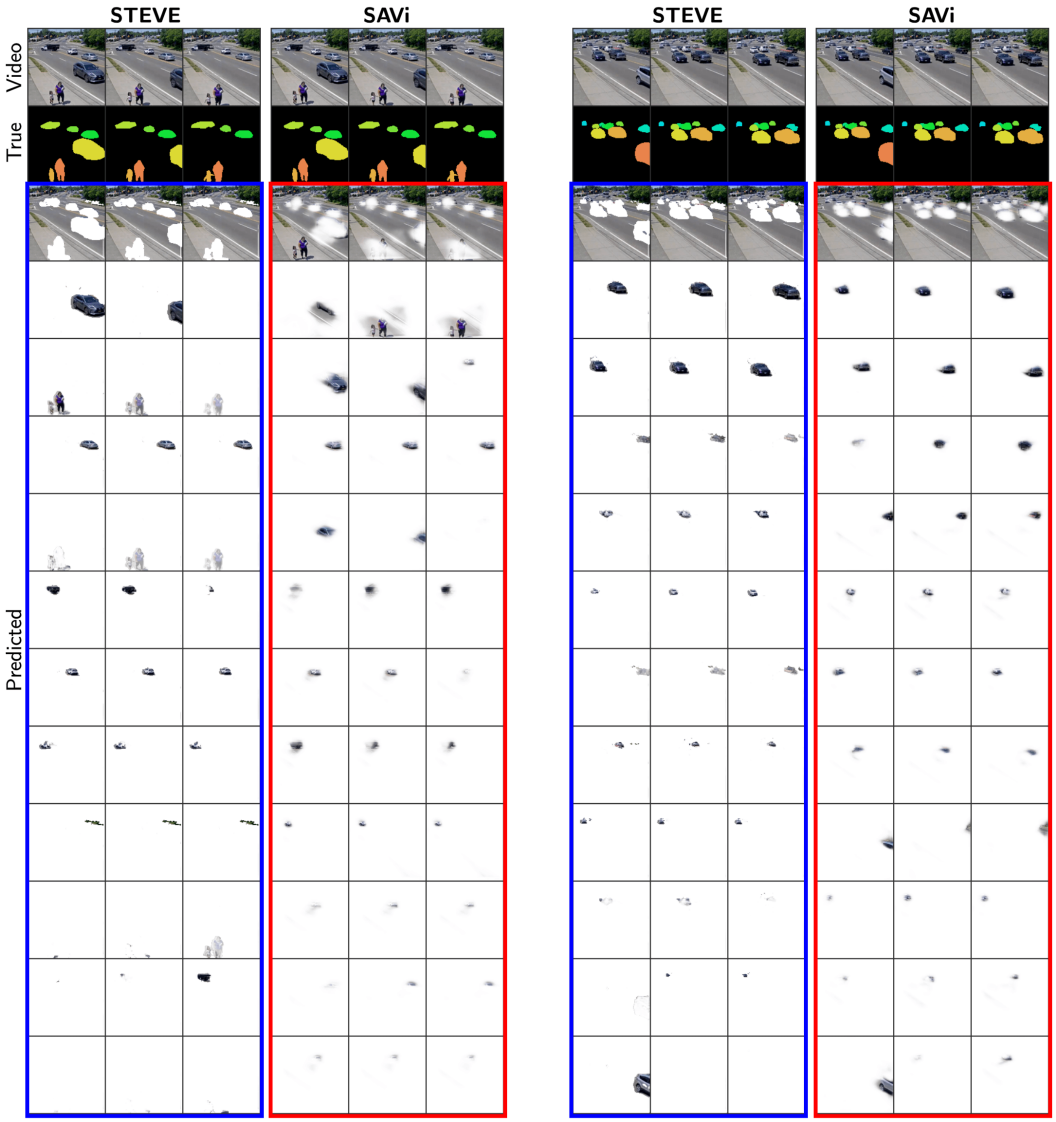


Figure 15: Additional Samples of Unsupervised Video Segmentation in Youtube Traffic.

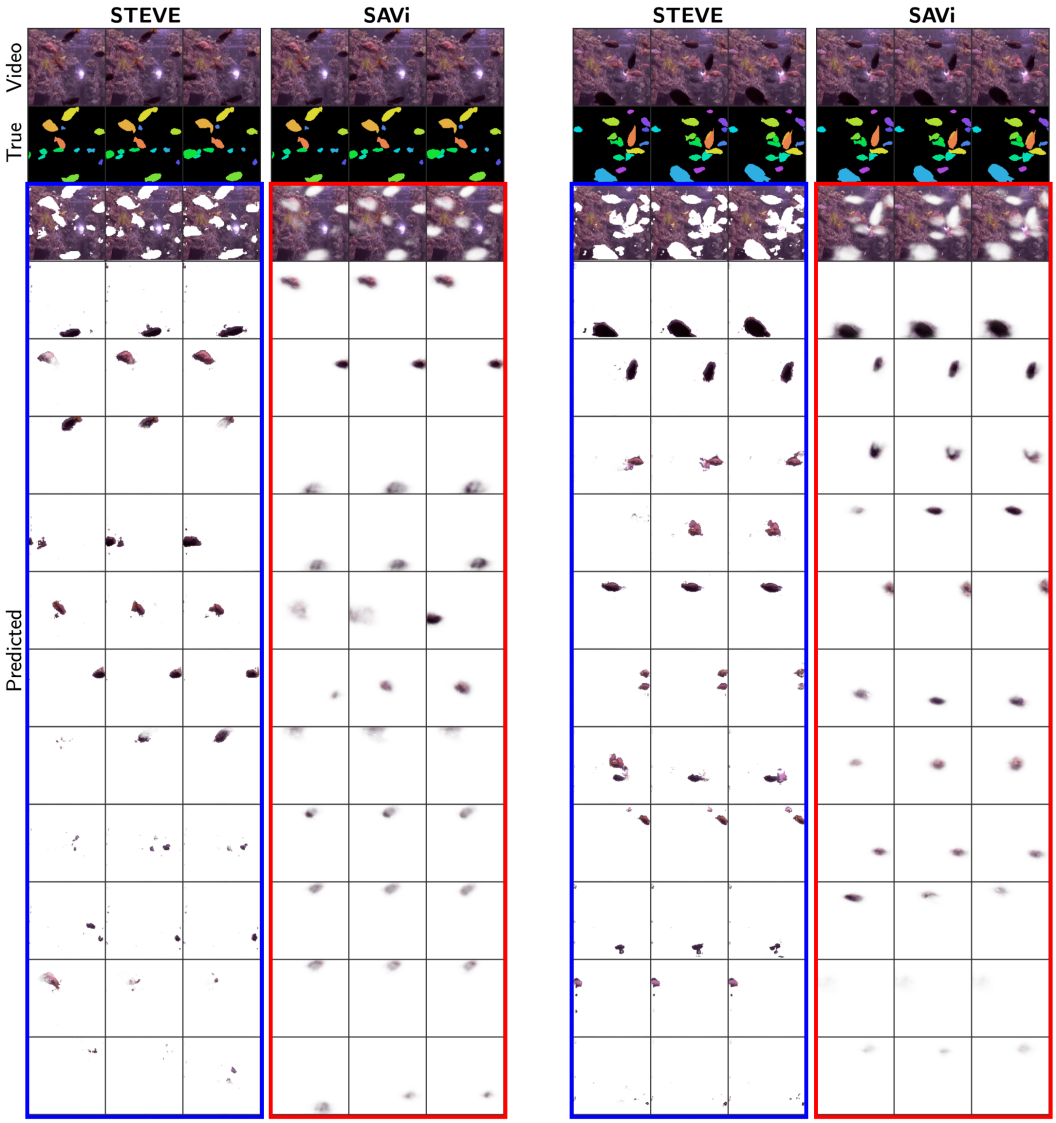


Figure 16: Additional Samples of Unsupervised Video Segmentation in Youtube Aquarium.

D Additional Related Work

Unsupervised Video Object Segmentation using Motion Cues. When objects in the video are moving independently, optical flow provides a strong signal of objectness as pixels with similar flow can be treated as an object. Therefore, several works pursue estimating optical flow in an fully unsupervised way (Ren et al., 2017; Janai et al., 2018; Wang et al., 2018; Jonschkowski et al., 2020; Stone et al., 2021). Leveraging flow, several methods have shown success in visually complex videos (Brox & Malik, 2010; Bideau & Learned-Miller, 2016; Yang et al., 2019; Tangemann et al., 2021; Yang et al., 2021). However, these can require access to optical flow even during deployment which can be challenging to obtain. In response, recent methods aim to infer objects via RGB-only input, but by using optical flow to supervise the training (Kipf et al., 2021; Bao et al., 2022). Even though these can handle naturalistic videos, their learning depends primarily on the flow information. If flow is absent e.g. in scenes with static objects, these can suffer (Greff et al., 2022). In comparison, our model is trained given only RGB video frames and we consider the use of motion information as orthogonal to our approach.

Unsupervised Segmentation Propagation. Several methods learn to propagate mask from one frame to another via unsupervised training and are shown to handle natural videos (Lai et al., 2020; Vondrick et al., 2018; Lai & Xie, 2019; Zhu et al., 2020; Oh et al., 2019; Wang et al., 2019a,b). However these methods only learn how to propagate the masks, and still require ground truth segmentation or bounding boxes in the first frame to actually perform tracking. Therefore, these methods are not fully unsupervised like ours.

E Additional Experiment Details

Evaluating the Effect of Number of Past Frames on Image Segmentation. We randomly take a video clip from the test set having T frames: $\mathbf{x}_1, \ldots, \mathbf{x}_T$. We would like to measure how the segmentation of the T-th frame $\mathbf{x}_T$ would change as a function of the number of past frames k shown to the model. For this, we predict the object masks $\mathcal{M}_T^k$ for the frame $\mathbf{x}_T$ given that the model has seen k frames $\mathbf{x}_{T-k}, \ldots, \mathbf{x}_{T-1}$ in the past. We compute the Image FG-ARI for the masks $\mathcal{M}_T^k$ by varying the value of $k \in \{0, \ldots, T-1\}$ and make the plot in Figure 2. We take $T = 7$.

Evaluation of Out-of-Distribution Generalization to More Objects. In this experiment, a model is tested on more objects than it was trained on. For this experiment, during testing, we increase the number of slots by an amount equal to the difference in maximum number of objects between the test and the training set.

F Limitations

We now describe the main limitations of STEVE and the future directions. First, there is some opportunity to improve the performance further within the proposed framework by adopting more advanced architectures than the minimal architecture we choose in this work. We would say that this is more of a future work than a limitation of this framework. Second, more computational efficiency is required in the future to handle images of higher resolution because the transformer decoder has a quadratic memory complexity and also because token-by-token generation can be slow. Third, the potential to scale the model to internet-scale datasets is discussed in this work but not investigated. Our current datasets typically contain about 200K video frames with an image resolution of 128. For scaling this up, we may also need to develop a new model in this framework. Fourth, although our focus in this paper is not on synthetic image generation, we found that the quality of image generation in naturalistic datasets has some room for further improvement. We believe that this could not only provide the ability to synthesize images and videos but also improve object decomposition as well. Fifth, even if our model outperforms previous works in general, we also observe some failure modes in MOVi-E such as splitting of large objects or merging of very small objects. To resolve this, we believe that it's worth investigating a much larger data on a bigger model or with a more sophisticated architecture. Lastly, the CNN backbone of our slot encoder is rather lightweight, having just 4-layers and a feature-size 64. For scaling this model to more complex natural images, it is likely that this lightweight CNN will not be enough.

G Impact Statement

There are no imminent negative societal consequences of our current work, however, future applications should be mindful to avoid malicious uses such as in surveillance and to minimize environmental impact when training larger transformers. At the same time, benign applications in scene understanding, robotics and autonomous navigation would benefit positively from our work.