Object Discovery from Motion-Guided Tokens

Zhipeng Bao*1 Pavel Tokmakov2 Yu-Xiong Wang3
Adrien Gaidon2 Martial Hebert1
1CMU 2Toyota Research Institute 3UIUC
{zbao, hebert}@cs.cmu.edu {pavel.tokmakov, adrien.gaidon}@tri.global yxw@illinois.edu

Abstract

Object discovery – separating objects from the background without manual labels – is a fundamental open challenge in computer vision. Previous methods struggle to go beyond clustering of low-level cues, whether handcrafted (e.g., color, texture) or learned (e.g., from auto-encoders). In this work, we augment the auto-encoder representation learning framework with two key components: motion-guidance and mid-level feature tokenization. Although both have been separately investigated, we introduce a new transformer decoder showing that their benefits can compound thanks to motion-guided vector quantization. We show that our architecture effectively leverages the synergy between motion and tokenization, improving upon the state of the art on both synthetic and real datasets. Our approach enables the emergence of interpretable object-specific mid-level features, demonstrating the benefits of motion-guidance (no labeling) and quantization (interpretability, memory efficiency).

1. Introduction

Objects are central in human and computer vision. In the former, they are a fundamental primitive used to decompose the complexity of the visual world into an actionable representation. This abstraction in turn enables higher-level cognitive abilities, such as casual reasoning and planning [32, 54]. In computer vision, object detection has achieved remarkable progress [7, 45] and is now an essential component in many applications (e.g., driving, robotics). However, these models require a large amount of manual labels from a fixed vocabulary of categories. Consequently, learning unsupervised, object-centric representations is an important step towards scaling up computer vision to the real world.

This topic has received renewed attention recently thanks to structured generative networks with iterative inference over a fixed set of variables [6, 16, 22, 38, 39]. These methods cluster pixels in the feature space of an auto-encoder, exhibiting behavior similar to grouping based on low-level cues, such as color or texture. Hence, they are restricted to toy images with colored geometric shapes on a plain background, and fail on more complex realistic scenes [4].

Two main types of works attempt to address this shortcoming. The first family of methods sets out to simplify the grouping problem by introducing more structure into the output space, e.g., reconstructing optical flow [36] or depth [15]. They, however, require supervision, either in the form of known poses [36] or ground truth bounding boxes [15], veering away from the unsupervised goal of object discovery. In contrast, Bao et al. [4] resolve the object-background ambiguity by explicitly integrating an unsupervised motion segmentation algorithm [11] into the pipeline, showing substantial progress on realistic scenes. The second main direction to improve object discovery focuses on improving the decoder part of auto-encoding architectures [52, 53], replacing convolutional decoders with transformers [43, 60] combined with discrete variational auto-encoders (DVAE) [47] to reduce memory footprint. These more sophisticated architectures improve performance without additional supervision, including on real-world sequences. However, these methods are evaluated with different protocols (metrics, datasets) and therefore no clear architectural principles have emerged yet for unsupervised object discovery.

In this work, we introduce a novel architecture, Motion-guided Tokens (MoTok), based on the combination of two fundamental structural principles: motion and discretization. We define objects as discrete entities that might have an independent motion. As prior works have shown encouraging results thanks to unsupervised motion guidance and better
In this work, we tackle the object discovery problem in realistic videos with object-centric representation by capitalizing on motion-guided tokens. We review the most relevant works in these areas below.

**Object Discovery** tackles the problem of separating objects from background without manual labels [4]. Classic computer vision methods use appearance-based perceptual grouping to parse a scene into object-like regions [2, 18, 37]. Among them, [2] first propose a multiscale fast normalized cuts algorithm and achieved robust object segmentation performance for static images.

Recently, the topic of object discovery has experienced renewed attention under the name of unsupervised object-centric representation learning. A wide variety of learning-based methods has been introduced [6, 16, 22, 23, 31, 38, 39, 52, 61, 64], usually with an encoder-decoder architecture [35, 46]. These methods allow to learn compositional feature representations, e.g., a set of variables that can bind to objects in an image [6, 16, 22, 23, 39, 52, 61, 64], or a video [4, 15, 31, 33, 36, 38, 48, 51, 53, 62]. Among them, [39] first formulate the SlotAttention framework, which is used to bind a set of variables, called slots, to image locations. The slots are then decoded individually and combined to reconstruct the image.

However, without additional constraints, such methods tend to converge to pixel grouping based on low-level cues, such as color, and do not generalize to realistic images or videos with complex backgrounds. To address this limitation, [36] and [15] extend the slot concept from static images to videos via reconstructing in the optical flow or depth space respectively. The intuition behind these methods is that this space provides stronger cues to separate the objects from the background. Separately, Bao et al. [4] use motion cues to guide the slots to find moving objects and then generalize to all the objects that can move.

In another line of work, Singh et al. [53] show that a combination of a more powerful transformer decoder [13] and discrete variational auto-encoder [47] can enhance the object discovery performance. Different from these works, we propose a unified architecture for object discovery that is flexible to different choices of decoders and reconstruction space. We also introduce the vector quantized features as an additional reconstruction space with motion guidance.

Finally, several recent works also leverage 3D geometry as inductive biases to enforce the learning-based models to focus on object-like regions [10, 14, 26, 55]. Though these methods remain limited to the toy, synthetic environments, the underlying geometric priors are orthogonal to our approach and have a great potential to be combined with our proposed method as a future direction.

Vector quantization is originally developed for data compression in signal processing [20]. More recently, [58] propose the Vector-Quantized Variational Autoencoder (VQ-VAE), which learns a discrete representation of images, and models their distribution aggressively. This technique has motivated a series of investigations in solving different computer vision tasks including image synthesis [17, 24, 44], video generation [56, 63], and language modeling [24, 42, 43], to name a few. In this work, we adopt the vector quantization technique as a mid-level representation in our object discovery framework. The intuition is that reconstructing in a structured, low-dimensional feature space rather than the high-variance color space should simplify the task of resolving the object and background ambiguity. In comparison, [53] also introduce a mid-level feature space for object discovery in videos. However, they use DVAE, not VQ-VAE, and are motivated primarily by improving training efficiency. Moreover, they did not explore combining vector quantization with motion cues in an end-to-end trainable framework.

**Transformers** are originally proposed for sequence-to-sequence modeling in natural language processing [60]. They use a multi-head attention mechanism, instead of the recurrent memory units, to aggregate information from the input. Recently, vision transformer (ViT) [13] and its derivations have achieved state-of-the-art in several visual tasks [7, 29, 30, 41, 48, 49, 67]. Among them, Perceiver IO [29] design a computationally efficient architecture to handle arbitrary outputs in addition to arbitrary inputs, greatly enhancing the generalizability of the transformer architecture. We employ a variant of a Perceiver module as a decoder in our object discovery framework to combine high representation power with computational efficiency.
3. Method

We now explain the proposed Motion-guided Tokens (MoTok) framework in detail. Its architecture is shown in Figure 2. We first introduce a motion-guided slot learning framework [4] in Section 3.1. We then describe the slot decoder units in Section 3.2, which decode the slot features into a reconstruction space. In Section 3.3, we describe the choices of reconstruction space and explain the vector-quantized reconstruction space. In Section 3.3, we describe the choices for the slot decoder.

3.1. Preliminary: Motion-Guided Slot Learning

Our object-centric representation learning module is derived from [4] (Figure 2, left). Concretely, given a sequence of video frames $\{I^1, I^2, ..., I^T\}$, we first process each frame through an encoder CNN to obtain an individual frame representation $H^t = f_{enc}(I^t)$. These individual representations are aggregated by a spatio-temporal Convolutional Gated Recurrent Unit (ConvGRU) [3] to obtain video encoding via $H'^t = \text{ConvGRU}(R^{t-1}, H^t)$, where $R^{t-1} \in \mathbb{R}^{h' \times w' \times d_{	ext{map}}}$ is the recurrent memory state.

Next, we perform a single attention operation with $K$ slots to directly compute the slot state $S^t = W^t v(H'^t)$, where $W^t \in \mathbb{R}^{K \times N}$ is the attention matrix, $N = h' \times w'$ indicates the flapped shape dimension, and $v(\cdot)$ is the value embedding function. $W^t$ is computed using the slot state in the previous frame $S^{t-1}$ and the input feature $H'^t$. For the first frame, we use a learnable initial state $S^0$. At the same time, for each slot $s^t_i$, we can also obtain the attention mask $W^t_i$.

In [4], a motion cue is also added to guide the slots to find moving objects. A set of sparse, instance-level motion segmentation masks $\mathcal{M} = \{M^1, M^2, ..., M^T\}$ is assumed to be provided with every video, with $M^t = \{m^t_1, m^t_2, ..., m^t_{CN^t}\}$, where $CN^t$ is the number of moving objects that were successfully segmented in frame $t$, and $m^t_j \in \{0, 1\}^{h' \times w'}$ is a binary mask. The attention mask $W^t \in \mathbb{R}^{K \times N}$, is then supervised with the motion segments. $M^t$ is also considered as a set of length $K$ padded with $\emptyset$ (no object), and a bipartite matching is found between them with the lowest cost:

$$\hat{\sigma} = \arg \min_{\sigma} \sum_{i=1}^{K} \mathcal{L}_{\text{seg}}(m^t_i, W^t_i; \sigma(i)), \tag{1}$$

where $\mathcal{L}_{\text{seg}}(m^t_i, W^t_i; \sigma(i))$ is the segmentation loss between the motion mask $m^t_i$ and the attention map of the slot with index $\sigma(i)$. Once the assignment $\hat{\sigma}$ has been computed, the final motion supervision objective is defined as follows:

$$\mathcal{L}_{\text{motion}} = \sum_{i=1}^{K} \mathds{1}_{\{m^t_i \neq \emptyset\}} \mathcal{L}_{\text{seg}}(m^t_i, W^t_i; \hat{\sigma}(i)), \tag{2}$$

where $\mathds{1}_{\{m^t_i \neq \emptyset\}}$ denotes that the loss is only computed for the matched slots and $\mathcal{L}_{\text{seg}}$ is the binary cross entropy.

3.2. Slot Decoders

The goal of the Slot Decoder is to map the slot representation $(S^t, W^t)$ to a 2D feature map $F^t$ for the reconstruction space. As shown in Figure 2 (middle), we propose four choices for the slot decoder. Linear Decoder directly maps the slot features $S^t$ to their corresponding positions based on the attention mask $W^t$:

$$F^t_{\text{linear}}(x) = \frac{\sum_{i=1}^{K} s^t_i(x) W^t_i \cdot x}{\sum_{i=1}^{K} W^t_i \cdot x}, \tag{3}$$

where $x$ is an arbitrary 2D position. CNN Decoder further adds two convolutional layers to the 2D feature map formed by Equation 3:

$$F^t_{\text{CNN}} = \text{CNN} \left( \frac{\sum_{i=1}^{K} s^t_i W^t_i \cdot x}{\sum_{i=1}^{K} W^t_i \cdot x} \right). \tag{4}$$

Transformer Decoder decodes the feature by querying the slot representation with a 2D positional embedding through a transformer decoder:

$$F^t_{\text{transformer}} = \text{Transformer}(P, S^t, S^t), \tag{5}$$

Figure 2. Model architecture of the proposed Motion-guided Tokens (MoTok) framework. MoTok is a unified framework for video object discovery that is flexible with different choices of decoders and reconstruction spaces. Our framework effectively leverages the synergy between motion and tokenization, and enables the emergence of interpretable object-specific mid-level features.
We describe the VQ-space below.

When vector-quantized space is used. As we will show next, the query reconstruction directly, but rather supervise the feature map other three reconstruction spaces, here we do not predict the output of VQ-VAE, and then the discrete latent variables are calculated by a nearest neighborhood search among the discrete feature set E:

\[ z_i^t(x) = e_k, \quad \text{where } k = \arg\min_j ||z_i^t(x) - e_j||_2, \quad (6) \]

and x is an arbitrary 2D position. The reconstructed image \( \hat{I} \) is then decoded by \( \hat{I} = \text{Decoder}_{VQ}(z_i^d) \). The objective of VQ-VAE is:

\[ \mathcal{L}_{\text{VQVAE}} = \log P(I|z_i^d) + ||sg[z_i^d] - z_i^d||_2 + ||sg[z_i^e] - z_i^e||_2, \quad (7) \]

where \( sg[\cdot] \) is the stop-gradient operation.

Then we use the quantized feature map \( z_i^d \) as the target signal for the slot feature map \( F^t \). The final objective of VQ-VAE and the VQ reconstruction is:

\[ \mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{VQVAE}} + ||sg[F^t] - z_i^d||_2 + ||sg[z_i^e] - F^t||_2. \quad (8) \]

Motion-guided token representation. The last term in Equation 8, \( ||sg[F^t] - z_i^d||_2 \), enables the motion signal from slot learning to jointly optimize the token space through the output of the slot decoder. Furthermore, the token representation and the motion cues build a connection linked by the slot learning, thus enabling the emergence of interpretable object-specific mid-level features of tokens. In addition, reconstructing in a more compact token space also benefits the model by better utilizing the motion signal to achieve an improved slot representation and temporal consistency.

3.4. Optimization

Token contrastive constraint. The goal of reconstructing in the VQ-space is that it is more compact and of lower variation compared with the RGB space. To make the VQ-space more structured, we add an optional contrastive constraint below to the vector space, which increases the independence between latent vectors:

\[ \mathcal{L}_\text{contrastive} = ||I - \text{softmax}(E \cdot E^T)||_1, \quad (9) \]

where \( I \) is the identity matrix and \( E \in \mathbb{R}^{N \times d_{vq}} \) is the matrix of the feature embedding space \( S \).

The final loss function is a combination of the reconstruction objective, the motion objective, and the optional contrastive constraint:

\[ \mathcal{L} = \lambda \mathcal{L}_\text{motion} + \mathcal{L}_\text{recon} + \lambda_c \mathcal{I}_{VQ} \mathcal{L}_\text{contrastive}, \quad (10) \]

where \( \lambda \) and \( \lambda_c \) are weighting factors and \( \mathcal{I}_{VQ} \) is an indicator function. For the reconstruction loss, we set \( \mathcal{L}_\text{recon} = \mathcal{L}_VQ \) when performing reconstruction in the VQ-space. Otherwise, we use an \( L_2 \) loss for reconstruction in the other three spaces.

---

**Algorithm 1: Perceiver decoder**

\[
\begin{align*}
\text{Perceiver}(S^t, P): \\
S^t &\to \text{Norm}(S^t) \\
\hat{S}^t &= \text{SelfAttention}(S^t) + S^t \\
\tilde{S}^t &\to \text{Norm}(\tilde{S}^t) \\
\hat{S}^t &= \text{MLP}(\tilde{S}^t) + \hat{S}^t \\
S^t &\to \text{Norm}(S^t) \\
F_{\text{Perceiver}}^t &= \text{CrossAttention}(P, \hat{S}^t, \tilde{S}^t)
\end{align*}
\]

where the query \( P \in \mathbb{R}^{N \times d_p} \) is a learnable positional embedding.

Compared with the previous two linear decoders, the transformer decoder further considers the global connections between the slot features and the input query, so that it will form a more powerful feature map. However, an obvious limitation is that the transformer decoder applies self-attention to the input positional query, which is 1) redundant since the positional embedding itself is learnable, and 2) not computationally efficient therefore limits the scalability of the whole model. To solve this limitation, we further propose the perceiver decoder.

**Perceiver Decoder** is inspired by [29], which designs a computationally efficient architecture to handle arbitrary outputs in addition to arbitrary inputs. Different from the original architecture, we only add a self-attention layer to the slot representations, followed by a cross-attention layer for the output and positional embedding queries. The whole procedure is illustrated in Algorithm 1.

After forming the 2D feature map \( F^t \), we decode it to a reconstruction space with a CNN-based decoder, except for when vector-quantized space is used. As we will show next, an additional decoder is redundant in this case.

3.3. Reconstruction Space

There are four choices of reconstruction space in our framework. The plain RGB space contains the most information but also is the most complex to resolve the object/background ambiguity. The flow and depth spaces, as shown in Figure 2 (right), are more structured. Reconstructing in these spaces makes the grouping problem easier. However, there are still shortcomings, e.g. non-moving objects are not captured in the flow space and depth cannot easily distinguish between-objects that are near each other. Moreover, these two spaces are not as informative as the RGB space. In addition, we introduce the VQ-space, which is end-to-end trainable and is both structured and informative. We describe the VQ-space below.

**Vector-quantized reconstruction space.** Different from the other three reconstruction spaces, here we do not predict the reconstruction directly, but rather supervise the feature map \( F^t \) to match the latent embedding space of VQ-VAE.

In particular, following [58], we define a latent embedding space \( E \) as the set of \( M \) vectors \( e_i \) of dimension \( d_{vq} \):

\[ E = \{ e_i \in \mathbb{R}^{d_{vq}} | i = 1, 2, \ldots, M \} \]

Given an input image \( I^t \), VQ-VAE first processes it with an encoder to get the output \( z_i^t = \text{Encoder}_{VQ}(I^t) \), and then the discrete latent variables \( z \) are calculated by a nearest neighborhood search among the discrete feature set \( E \):

\[ z_i^t(x) = e_k, \quad \text{where } k = \arg\min_j ||z_i^t(x) - e_j||_2, \quad (6) \]

and \( x \) is an arbitrary 2D position. The reconstructed image \( \hat{I} \) is then decoded by \( \hat{I} = \text{Decoder}_{VQ}(z_i^d) \). The objective of VQ-VAE is:

\[ \mathcal{L}_{\text{VQVAE}} = \log P(I|z_i^d) + ||sg[z_i^d] - z_i^d||_2 + ||sg[z_i^e] - z_i^e||_2, \quad (7) \]

where \( sg[\cdot] \) is the stop-gradient operation.

Then we use the quantized feature map \( z_i^d \) as the target signal for the slot feature map \( F^t \). The final objective of VQ-VAE and the VQ reconstruction is:

\[ \mathcal{L}_{\text{VQ}} = \mathcal{L}_{\text{VQVAE}} + ||sg[F^t] - z_i^d||_2 + ||sg[z_i^e] - F^t||_2. \quad (8) \]

Motion-guided token representation. The last term in Equation 8, \( ||sg[F^t] - z_i^d||_2 \), enables the motion signal from slot learning to jointly optimize the token space through the output of the slot decoder. Furthermore, the token representation and the motion cues build a connection linked by the slot learning, thus enabling the emergence of interpretable object-specific mid-level features of tokens. In addition, reconstructing in a more compact token space also benefits the model by better utilizing the motion signal to achieve an improved slot representation and temporal consistency.
Figure 3. Frame samples from the video datasets used in our experiments. MOVi [21] (left) is a multi-object video dataset created by simulating rigid body dynamics. TRI-PD [4] (middle) is a collection of photo-realistic, synthetic driving videos. KITTI [19] (right) is a real-world benchmark with city driving scenes.

4. Experimental Evaluation

4.1. Experimental Settings

**Benchmarks.** We evaluate our approach on three popular benchmarks with different complexity (Figure 3). MOVi [21] is a synthetic multi-object video dataset, created by simulating rigid body dynamics. We use this benchmark to ablate and analyze our model architecture. Following previous works [15, 53], we use the most complex subset, MOVi-E, for evaluation. MOVi-E contains both moving and static objects (maximum 20 objects) with linear random camera motion. The resolution of this dataset is 128 × 128 and each video contains 24 frames with a sample rate of 12 FPS. We use standard train and test split for MOVi-E.

TRI-PD [4, 25] is a synthetic dataset based on street driving scenarios collected by using a state-of-the-art synthetic data generation service [1]. The dataset comes with a collection of accurate annotations (e.g., for flow or depth), which we use to ablate the impact of additional information and its quality on various models. Each video in TRI-PD is 10 seconds captured at 20 FPS. There are 924 videos in the training set and 51 videos in the test set. We crop and resize each frame to the resolution of 480 × 968.

KITTI [19] is a real-world benchmark with city driving scenes. We use the whole 151 KITTI videos for training and the instance segmentation subsets with 200 single-frame images for evaluation. The frames are resized to 368 × 1248.

**Baselines.** We compare our methods against the most recent learning-based object discovery models. In particular, SAVi [36] and SAVi++ [15] are two direct extensions of the original slot attention [39] using optical flow and depth to facilitate object-centric learning. Alternatively, STEVE [53] uses a more powerful transformer decoder to enhance the object discovery performance. Bao et al. [4], Karazija et al. [33] utilize motion cues to guide object discovery. In addition, in the supplementary, we compare our approach to [51], which is a concurrent work that relies on ImageNet [12] pre-training [9]. For [4, 15, 36, 53], we use either the official code or public implementation (see supplementary for details). Due to the lack of implementation, we reuse reported results with the two very recent approaches [33, 51].

**Evaluation Metrics.** Following prior works, we use the Foreground Adjusted Rand Index (FG. ARI) to evaluate the performance of the models, which captures how well the predicted segmentation masks match ground-truth masks in a permutation-invariant fashion. Notice that FG. ARI is measured in terms of the whole video, so that the temporal consistency of the slots is considered with this metric.

**Implementation Details.** We use the same ResNet-18 ConvGRU encoder backbone for all the compared methods following [4]. We set the number of slots as 24 for MOVi-E and 45 for PD and KITTI based on the maximal number of objects in these datasets. On TRI-PD and KITTI, we further downsample the images by 4 for SAVi and SAVi++ to fit the GPU memory. Notice that, the other methods also produce downsampled slot masks, leading to the evaluation under the same resolution for all the methods. All the models are trained for 500 epochs using Adam [34]. During training, we train all the models with a randomly sampled video sequence of a fixed length (6, 5, 5 for MOVi-E, TRI-PD, and KITTI respectively). During the inference time, all the models are evaluated frame by frame until the end. For the FG. ARI measurement on TRI-PD, we discard any instance labels covering an area of 0.5% or less of the first sampled video frame following [15]. We also evaluate FG. ARI at 5 FPS due to memory limitation. We use batch size 64 for MOVi-E, and 8 for TRIPD and KITTI.

Our method and Bao et al. [4] require motion segmentation signals. We apply [11], a powerful method pre-trained on the toy FlyingThings3D dataset [40], taking the ground-truth flow (MOVi-E) or RAFT [57] flow as the input. For SAVi and SAVi++, we do not use the first-frame bounding box supervision for a fair comparison. We generate the optical flow and depth annotation for KITTI using two state-of-the-art methods, RAFT [57] and VIDAR [25]. We also use them for ablations on TRI-PD. More details about the hyper-parameters, training scheme, and annotation generation are provided in the supplementary.
4.2. Object Discovery on MOVi

We use MOVi-E dataset to ablate our model with different choices of slot decoder and reconstruction spaces, aiming to elucidate their roles in object discovery. We also compare the performance with state-of-the-art models in this section.

Set up. We first fix the reconstruction space as the VQ-space and ablate the decoder architectures (first 4 lines in Table 1), and then we fix the decoder architecture with the best component and reconstruct in different reconstruction space (Line 5 to 9 in Table 1). Since SAVi++ [15] argues that reconstructing in the combined space of flow and depth is the key to their success, we add a variant Flow + Depth, which also reconstructs in this combined space. Furthermore, as VQ-VAE can be trained to reconstruct the image in the flow space as well, we also report a VQ (flow) variant for completeness. We do not include motion cues for the architecture analysis. A more comprehensive ablation is reported in the supplementary.

Decoder analysis. By comparing the results on the first four lines in Table 1, similar to [53], we find that the capacity of the decoder indeed plays a key role in the object discovery performance. A simple linear decoder fails, but adding an additional CNN decoder can bring limited performance gains. Introducing powerful transformer-based decoders, on the other hand, greatly improves the object discovery capabilities of the model. Between the two transformer variants, the more advanced perceiver decoder achieves better results, while being more computationally efficient.

Reconstruction space analysis. For the four reconstruction spaces, the VQ-space yields the best object discovery performance. We analyze the learned slots and token representations in Figure 4 and make the following discoveries. Firstly, the model trained using the VQ-space better separates the objects from the background and shows stronger temporal consistency. This is due to the more structured, compact, and lower-variance reconstruction space provided by the quantized features, which simplifies the task of grouping, compared to the raw RGB space.

Secondly, by comparing Flow and Flow+Depth variants, we find that the latter indeed carries more information, allowing the model to better group the objects, which is consistent with [15]. However, interestingly, we also find that given a sufficiently strong decoder, reconstructing in depth or flow space does not bring further improvements compared to RGB, and can even decrease the performance. Finally, comparing Flow and VQ(flow), we find that although adding the quantization improves performance significantly, this variant still lags behind reconstructing in the raw RGB space, reinforcing our previous conclusions.

Comparison to the state of the art. In Table 2, we compare our approach to the state-of-the-art methods. Our model achieves the best performance among them. Additionally, compared to SAVi and SAVi++, even our variants that reconstruct in the same space outperform them, indicating that the perceiver decoder and single-stage decoding strategy are the optimal choices. Our model also outperforms STEVE, due to the fact that, in contrast to DVAE, the codebook in VQ-VAE can be jointly optimized with slot learning (see Equation 8). Compared with Bao et al., our model shows better performance even without motion cues used by that method, but further incorporating them into our approach allows it to achieve top results.

Notice that, we compare with the variants of SAVi and SAVi++ that do not rely on ground-truth bounding boxes at test time. Their published numbers are included at the bottom of the table for reference, but they are not comparable to the other unsupervised methods.

4.3. Object Discovery with Realistic Driving Videos

Set up. Firstly, following [4], we capitalize on the ground-truth annotations available in TRI-PD, and evaluate the importance of the quality of external signals for various approaches. In particular, we separately evaluate using ground-truth (GT) flow, depth, and motion segmentation, as well as estimating those using state-of-the-art algorithms [11,25,57]. In addition, we report a variant STEVE-m for which we add

<table>
<thead>
<tr>
<th>Motion</th>
<th>Space</th>
<th>Decoder</th>
<th>FG. ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗</td>
<td>VQ</td>
<td>Linear</td>
<td>14.6</td>
</tr>
<tr>
<td>✗</td>
<td>VQ</td>
<td>Linear-CNN</td>
<td>16.3</td>
</tr>
<tr>
<td>✗</td>
<td>VQ</td>
<td>Transformer</td>
<td>40.1</td>
</tr>
<tr>
<td>✗</td>
<td>VQ</td>
<td>Perceiver</td>
<td>52.4</td>
</tr>
<tr>
<td>✗</td>
<td>RGB</td>
<td>Perceiver</td>
<td>49.2</td>
</tr>
<tr>
<td>✗</td>
<td>Flow</td>
<td>Perceiver</td>
<td>36.3</td>
</tr>
<tr>
<td>✗</td>
<td>Depth</td>
<td>Perceiver</td>
<td>18.4</td>
</tr>
<tr>
<td>✗</td>
<td>Flow + Depth</td>
<td>Perceiver</td>
<td>38.0</td>
</tr>
<tr>
<td>✗</td>
<td>VQ (flow)</td>
<td>Perceiver</td>
<td>44.6</td>
</tr>
</tbody>
</table>

Table 1. Model architecture analysis with different choices of the slot decoder and reconstruction space on MOVi-E. Perceiver decoder + VQ reconstruction space yields the best performance, indicating that (1) the capacity of the decoder plays a key role in object discovery; (2) learnable, vector-quantized reconstruction space outperforms fixed alternatives like depth or flow.

<table>
<thead>
<tr>
<th>Model</th>
<th>FG. ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bao et al.</td>
<td>51.6</td>
</tr>
<tr>
<td>STEVE</td>
<td>51.2</td>
</tr>
<tr>
<td>SAVi</td>
<td>28.1</td>
</tr>
<tr>
<td>SAVi++</td>
<td>31.7</td>
</tr>
<tr>
<td>MoTok (no motion)</td>
<td>52.4</td>
</tr>
<tr>
<td>MoTok</td>
<td>63.8</td>
</tr>
<tr>
<td>SAVi (G.T. box)</td>
<td>53.4</td>
</tr>
<tr>
<td>SAVi++ (G.T. box)</td>
<td>84.1</td>
</tr>
</tbody>
</table>

Table 2. Comparison to the state-of-the-art object discovery approaches on the validation sets of MOVi-E using FG. ARI. Our approach outperforms all the recent methods with the help of motion cues and vector quantization.
Figure 5. Reconstruction, slot representation, and token masks for [4], STEVE-m [53], and our method. We use ground-truth motion cues for supervision, and visualize top-10 masks excluding background slots. MoTok effectively leverages the synergy between motion and tokenization, enabling the emergence of interpretable object-specific mid-level features, which simplify the problem of object discovery.

Table 3. Object discovery evaluation of different models on TRIPD dataset. GT denotes the ground-truth guidance given to each model and EST denotes the estimated cues. State-of-the-art models fail to work without motion cues; introducing token representation helps our model better utilize the motion signal.

From Table 3, we can observe that our method strongly outperforms all the baselines. Additionally, we find that even with the ground-truth flow or depth, SAVi and SAVi++ do not work well for realistic videos with complex background and crowded scenes. Without motion cues, even equipped with a powerful transformer decoder, STEVE fails to work in this challenging setting as well. Finally, our model achieves better performance compared to Bao et al. with both GT and estimated motion segments, indicating its better robustness.

Impact of motion-guided tokens: In Figure 5, we visualize the RGB reconstruction, slot representation, and token masks (if possible) for Bao et al., STEVE-m, and our MoTok. There are several key observations. Firstly, compared with STEVE-m, MoTok enables the emergence of more interpretable mid-level features, even though both models are trained with motion cues and tokenization. This is due to the interplay between object and slot discovery in our end-to-end framework, whereas STEVE-m trains DVAE separately from the rest of the model.

Secondly, our model achieves the best RGB reconstruction result, even for the latter frames. The high quality of the reconstruction indicates that the model can better take advantage of the appearance signal to optimize the slot representation. Finally, our model demonstrates stronger temporal consistency compared with the baselines, thanks to the structured reconstruction space. To sum up, our architecture effectively leverages the synergy between motion cues and tokenization, enabling the emergence of interpretable mid-level features, which greatly simplifies the task of object discovery.

Interpretable tokens. To better illustrate the interpretability of motion-guided tokens, we measured their alignment with the ground-truth semantic labels using cluster purity [50]. Vanilla VQ-VAE (no motion guidance) achieves a purity of 40.1, whereas ours reaches 65.7 (chance performance: 38.5). Qualitative comparison in Figure 6 clearly demonstrates our better semantic alignment.

Scalability to realistic videos: We additionally report the GPU memory usage per batch in Table 3. The per-slot decoding strategy and the use of transformer decoder limit the scalability of the other baselines. In comparison, our model achieves the best performance with the lowest memory consumption, showing a great generalization capability to real-world videos, thanks to the efficient perceiver decoder and the single-shot decoding strategy.

Impact of objectives. Finally, we ablate the objective design of our model with the GT annotations in Table 4. We build three variants of our model, one without the additional token contrastive constraint defined by Equation 10 (no contrastive), the second without the motion cues (no motion), and the last trained with ground-truth instance masks for all objects to indicate the performance upper bound (*). By comparing the performance of these variants, we make the following observations: (1) in realistic synthetic videos even the strongest architectures fail to resolve the object/background ambiguity in the absence of motion cues, demonstrating the difficulty of object discovery tasks in the real world; (2)

Figure 6. Visualization of the learned tokens for vanilla VQ-VAE and MoTok. The proposed MoTok model shows a better alignment with semantic categories.
Table 4. Ablation study on different learning signals. MoTok++ is our model trained with ground-truth instance segmentation. The motion cue and the contrastive constraint help improve the capability of the proposed model, almost reaching the performance upper bound (MoTok++).

<table>
<thead>
<tr>
<th>Model</th>
<th>FG. ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoTok</td>
<td>60.6</td>
</tr>
<tr>
<td>MoTok (no contrastive)</td>
<td>57.4</td>
</tr>
<tr>
<td>MoTk (no motion)</td>
<td>15.6</td>
</tr>
<tr>
<td>MoTok</td>
<td>64.7</td>
</tr>
</tbody>
</table>

Table 5. Evaluation of object discovery in the real-world KITTI dataset. Karazija et al. [33] (WL) is a variant trained with their proposed wrapping loss. Our model achieves state-of-the-art in real-world object discovery.

<table>
<thead>
<tr>
<th>Model</th>
<th>FG. ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bao et al. [1]</td>
<td>47.1</td>
</tr>
<tr>
<td>SAVi</td>
<td>20.0</td>
</tr>
<tr>
<td>SAVi++</td>
<td>23.9</td>
</tr>
<tr>
<td>STEVE</td>
<td>11.9</td>
</tr>
<tr>
<td>Karazija et al. [33]</td>
<td>50.8</td>
</tr>
<tr>
<td>Karazija et al. [33] (WL)</td>
<td>51.9</td>
</tr>
<tr>
<td>MoTok</td>
<td>64.4</td>
</tr>
</tbody>
</table>

consistent constraint facilitates diversity during vector quantization, increasing the information content of each token, which helps reduce the object/background ambiguity; (3) the performance of our model is close to the upper bound, indicating the effectiveness of the motion-guided tokens.

4.4. Object Discovery in the Real World

Set up. To train the object discovery model on the real-world KITTI benchmark, we first pre-train all the models on PD using ground-truth flow and depth and then fine-tune them using estimated annotations on KITTI.

The comparisons on KITTI are shown in Table 5 and Figure 7. We find that, consistent with our observations on TRI-PD, without motion guidance, both more powerful decoders (STEVE) and the simpler reconstruction space (SAVi and SAVi++) fail to work. However, our model still captures good dynamic object segmentation results. Compared to Bao et al., we still achieve a better FG. ARI score and a more solid segmentation mask, indicating the benefit of the model design and the motion-guided tokens. Finally, we outperform the very recent approach of Karazija et al. [33] even when they include an additional warping loss objective.

4.5. Limitations and Future Work

Object/background ambiguity in the real world. Even after successfully parsing the world into object and background slots, separating those from each other in an unsupervised way is an open challenge. Promising directions include temporal contrastive learning [5, 28], unsupervised clustering [8, 59], and utilizing geometric cues [65, 66].

Slot drift. From Figure 5, we notice that the final RGB reconstruction quality decreases with longer videos, which indicates that the distribution of the slot representation has shifted in the latter frames. This issue leads to degraded temporal consistency and ambiguous object masks. Better training schemes and model improvements, such as adding data augmentation during training [15], replacing GRU units with long-short memory units [27], and enforcing temporal consistency in the token space can help.

Better measuring object discovery in the real world. Measuring the performance of object discovery in the real world is challenging. As noted in [15], one slot may re-bind to another object after the tracked object moves out of the frame and a new object enters the scene. The FG. ARI score is not suitable for such scenarios but currently, there are no better metrics for object discovery which is able to tackle the re-binding issue. Developing a novel metric for this problem could have a significant impact on the community.

5. Conclusion

This work proposes MoTok, a unified architecture that leverages motion-guided tokenization for object discovery. By jointly training the slot representation with motion cues and vector quantization, our model enables the emergence of interpretable mid-level features which simplifies the problem of object discovery. Comprehensive evaluation on both synthetic and real-world benchmarks shows that with sufficient capacity of the slot decoder, motion guidance alleviates the need for labels, optical flow, or depth decoding, thanks to tokenization, achieving state-of-the-art results.

Acknowledgements. We thank Dian Chen, Alexei Efros, and Andrew Owens for their valuable comments. This research was supported by Toyota Research Institute. YXW was supported in part by NSF Grant 2106825, NIFA Award 2020-67021-32799, and the NCSA Fellows program.
References


[26] Paul Henderson and Christoph H Lampert. Unsupervised object-centric video generation and decomposition in 3D. In NeurIPS, 2020. 2


[56] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In CVPR, 2019. 2
[57] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In ECCV, 2020. 5, 6