Object-Centric Multiple Object Tracking

Zixu Zhao\(^1\)  \quad Jiaze Wang\(^2\)  \quad Max Horn\(^1\)  \quad Yizhuo Ding\(^3\)  \quad Tong He\(^1\)  \quad Zechen Bai\(^1\)
Dominik Zietlow\(^1\)  \quad Carl-Johann Simon-Gabriel\(^1\)  \quad Bing Shuai\(^1\)  \quad Zhuowen Tu\(^1\)  \quad Thomas Brox\(^1\)
Bernt Schiele\(^1\)  \quad Yanwei Fu\(^3\)  \quad Francesco Locatello\(^1\)  \quad Zheng Zhang\(^1\)\(^4\)  \quad Tianjun Xiao\(^1\)
\(^1\) Amazon Web Services \quad \(^2\) The Chinese University of Hong Kong \quad \(^3\) Fudan University
{zhaozixu, jiazew, yizhuodi, htong, baizeche, bshuai, ztu, zhaz, tianjux}@amazon.com
{hornmax, zietld, cjsg, brox, bschiel, locatelf}@amazon.de, yanweifu@fudan.edu.cn

Abstract

Unsupervised object-centric learning methods allow the partitioning of scenes into entities without additional localization information and are excellent candidates for reducing the annotation burden of multiple-object tracking (MOT) pipelines. Unfortunately, they lack two key properties: objects are often split into parts and are not consistently tracked over time. In fact, state-of-the-art models achieve pixel-level accuracy and temporal consistency by relying on supervised object detection with additional ID labels for the association through time. This paper proposes a video object-centric model for MOT. It consists of an index-merge module that adapts the object-centric slots into detection outputs and an object memory module that builds complete object prototypes to handle occlusions. Benefited from object-centric learning, we only require sparse detection labels (0%-6.25%) for object localization and feature binding. Relying on our self-supervised Expectation-Maximization-inspired loss for object association, our approach requires no ID labels. Our experiments significantly narrow the gap between the existing object-centric model and the fully supervised state-of-the-art and outperform several unsupervised trackers. Code is available at https://github.com/amazon-science/object-centric-multiple-object-tracking.

1. Introduction

Visual indexing theory [45] proposes a psychological mechanism that includes a set of indexes that can be associated with an object in the environment. Each index retains its association with an object, even when that object moves, interacts with other objects, or becomes partially occluded. This theory was originally developed in the cognitive sciences, however, a very similar principle lies at the heart of object-centric representation learning. By learning object-level representations, we can develop models inferring object relations [39, 60, 62] and even their causal structure [36, 40]. Additionally, object-centric representations have shown to be more robust [13], allow for combinatorial generalization [36], and are beneficial for various downstream applications [60]. Since causal relations often unfold in time, it is only logical to combine object-centric learning (OCL) with temporal dynamics modeling, where consistent object representations are necessary.

Multiple object tracking (MOT) is a computer-vision problem that resembles visual indexing theory. MOT aims at localizing a set of objects while following their trajectories over time so that the same object keeps the same identity in the entire video stream. The dominant MOT methods follow the detect-to-track paradigm: First, employ an object detector to localize objects in each frame, then perform association on detected objects between adjacent frames to get tracklets. The development of state-of-the-art
MOT pipelines usually require large amounts of detection labels for the objects we are interested in, as well as video datasets with object ID labels to train the association module. Consequently, such approaches are label intense and do not generalize well in open-world scenarios.

Unsupervised object-centric representation learning tackles the object discovery and binding problem in visual data without additional supervision \cite{50}. Recent work, such as SAVI \cite{32} and STEVE \cite{52}, extended such models to the video domain, which hints at possible applications to MOT. However, existing approaches are primarily evaluated without heavy punishment if slots exchange “ownerships” of pixels and rather rely on clustering similarity metrics such as FG-ARI \cite{32}. An object may appear in different slots across time (a.k.a ID switch issue), which hinders downstream applications of OCL models, especially when directional relationship among objects and their dynamics must be reasoned upon (e.g., who acts upon whom). Additionally, the part-whole issues are not fully explored, allowing slots to only track parts of an object. Figure 1 visualizes the two problems of OCL models that are developed orthogonally with respect to MOT downstream tasks, leading to a significant gap with the state-of-the-art fully supervised MOT methods. Scalability challenges of unsupervised OCL methods only accentuate this gap.

In this work, we take steps to bridge the gap between object-centric learning and fully-supervised multiple object tracking pipelines. Our design focuses on improving OCL framework on two key issues: 1) track objects as a whole, and 2) track objects consistently over time. For these, we insert a memory model to consolidate slots into memory and roll out the representations of the memory forward (to improve temporal consistency). Overall, our model provides a label-efficient alternative to the otherwise costly MOT pipelines that rely on detection and ID labels. Our contributions can be summarized as follows:

1. We develop a video object-centric model that can be applied to MOT task with very few detection labels (0-6.25%) and no ID labels.

2. OC-MOT leverages an unsupervised memory to predict completed future object states even if occlusion happens. Besides, the index-merge module can tackle the part-whole and duplication issues specific to OC models. The two cross-attention design is simple but nontrivial, serving as the “index” and “merge” functions with their key and query being bi-directional.

3. We are the first to introduce the object-centric representations to MOT that are versatile enough in a way of supporting all the association, rolling-out, and merging functions, and can be trained with low labeling cost.

2. Related Works

**Unsupervised Object-centric Learning.** Unsupervised object-centric learning describes approaches which aim at tackling the binding problem of visual input signals to objects without additional supervision \cite{26}. This is often accomplished using architectural inductive biases which force the model to encode input data into a set-structured bottleneck where object representations exert competition \cite{16, 37, 57} or exclusive binding to features \cite{25, 24, 6, 15}. Since their initial development on synthetic image data, these approaches have been extended to more complicated images by adapting the reconstruction objective \cite{51, 50}, to the decomposition of 3D scenes \cite{9, 42, 53}, to synthetic videos \cite{23, 28, 11, 30, 32, 52} and to real-world videos by exploiting additional modalities and priors \cite{32, 1, 14}. Our work is most closely related to the last group of methods which apply object-centric learning methods to real-world videos, yet in contrast does not focus on the derivation of object-centric representations themselves. Instead we focus on how object-centric representations can be used to perform multiple object tracking via long-term memory. Our work presents the first dedicated memory module, which, independent of the origin of the object-centric representation can match occurrences of objects to previously discovered objects and thus track these over time.

**Self-supervised MOT.** Most works study MOT in supervised settings, where the models are trained with object-level bounding box labels and ID labels \cite{10, 64, 66, 63, 7}. Tracktor++ \cite{3} uses a ready-made detector \cite{20} to generate object bounding boxes and propagates them to the next frame as region proposals. MOTR \cite{63} simultaneously performs object detection and association by autoregressively feeding a set of track queries into a Transformer decoder at the next timestep. To reduce the hand-label annotations, several recent approaches leverage the self-supervised signals to learn object associations from widely available unlabeled videos. For example, CRW \cite{58} and JSTG \cite{65} learns video correspondences by applying a cycle-consistent loss. Without fine-tuning, these models track objects at inference time by propagating the annotations from the first frame.

Our work is mostly related to the unsupervised detecto-track approaches that assume a robust detector is available. SORT \cite{4} and IOU \cite{5} associate detections using heuristic cues such as Kalman filters and intersection-of-union of bounding boxes. Such models do not need training but fail to handle scenarios with frequent occlusion and camera motion. A recent related method uses cross-input consistency \cite{2} to train the tracker: given two distinct inputs from the same video sequence, the model is encouraged to produce consistent tracks. Unfortunately, it suffers performance degradation once the detection boxes are not accurate, e.g., the grouping results from the object-centric model. For both supervised and unsupervised track-
3. Method

Our OC-MOT improves over traditional OCL frameworks in terms of tracking objects as a whole, and consistently over time. This is achieved by extending the traditional OC framework with a self-supervised memory to: i) Store historical information in the memory to fight against noise and occlusion. This helps improve temporal consistency. ii) Use the complete representation read-out from the memory to consolidate parts captured in different slots, which resolves the part-whole problem. The overall framework of OC-MOT is shown in Figure 2. Given slots \( \{ S_t \}_{t=1}^T \) extracted from \( T \) video frames by an object-centric grouping module, OC-MOT first uses the memory rollout \( \tilde{M}_t \) to perform slot-to-memory indexing. Then, it merges the slots as \( M_t \) to update the memory.

3.1. Object-centric Grouping

The object-centric grouping module uses Slot Attention\[37\] to turn the set of encoder features from video frames into a set of slot vectors \( \{ S_t \}_{t=1}^T \). The model is trained with a self-supervised reconstruction loss \( L_{oc\_rec} = || y - Dec(S) ||^2 \), where \( y \) can be the raw frame pixels, or feature representations extracted from the frames. The decoder has a compete-to-explain inductive bias to encourage binding of objects into individual slots.

3.2. Memory Module

We store the historical representations of all tracked objects into memory buffers \( M \in \mathbb{R}^{M \times T \times d} \) where \( M \) is the buffer number and \( d \) denotes the representation dimension. The memory is implemented with a first-in-first-out data structure and reserves a maximum of \( T_{max} \) time steps for each object. At time step \( t \), the detection results are

...
\( M_t = \{ m_1^t, ..., m_M^t \} \) if we denote \( m_t \) as the object representation. Intuitively, each buffer is a tracklet.

**Memory rollout.** At time step \( t \), the memory rolls the past states forward, and predicts the current object representations for all slots to index. The rollout process integrates the multi-view object representations together and handles the part-whole matching in the occlusion scenarios. Without losing generality, we denote all the past representations as \( M_{<t} \). The rollout \( M_t \in \mathbb{R}^{M \times d} \) is obtained by:

\[
\hat{M}_t = \text{Rollout}(M_{<t}).
\]

We adopt a mini GPT-2 model \([46]\) containing only 1.6M parameters as the rollout module. It performs temporal reasoning via an auto-regressive transformer.

### 3.3. Index-Merge Module

The index-merge module is used as a discrete interface between memory buffers and slots. To achieve this, we split the object association process into the index step and merge step, as shown in Figure 3, which can be achieved by standard multi-head attention (MHA) \([56]\) blocks.

#### Slot-to-memory index.

The index matrix \( I_t \in \mathbb{R}^{N \times M} \) indicates soft slots-to-buffer assignment. To compute it, we train a MHA block that takes the slots \( S_t \in \mathbb{R}^{N \times d} \) as query, and rollout \( \hat{M}_t \) as key and values, where \( N \) is slot number:

\[
I_t = \text{MHA}(k, v = \hat{M}_t, q = S_t).\text{attn\_weight} \quad (2)
\]

#### Memory-to-slot merge.

Our goal is to make sure a buffer represents one object by pooling from the slots that belong to that object, while simultaneously dealing with slots that represent parts of an object or duplicates. Thus, we stack another MHA block to merge the slots, using \( I_t \) as masked attention weights. Specifically, the merging function is defined as below:

\[
m_t = \text{MHA}(k, v = S_t, q = \hat{M}_t, \text{attn\_mask} = I_t). \quad (3)
\]

Here, the query is the rollout \( \hat{M}_t \); the key and value are slots \( S_t \). We apply \( I_t \) as the attention mask in MHA such that the re-normalized attention weights can be used for merging. This helps us to deal with wrongly-assigned slots. For example, if there are three slots and two of them are matched to one buffer, the attention weight could be \([0.8, 0.2, 0] \) indicating that the second slot does not belong to this buffer.

### 3.4. Model Training under EM Paradigm

**Losses.** The key of training detect-to-track models is to minimize the assignment costs for object associations. Usually, the weights of the pre-trained detector are frozen during training \([7, 3]\). Therefore, in our scenario, we freeze the object-centric model and only train the memory module. Assume we use \( L_{\text{assign}} \) to measure the assignment costs between slots \( S_t \in \mathbb{R}^{N \times d} \) and memory buffers \( M_t \in \mathbb{R}^{M \times d} \). The training loss can be formulated as:

\[
L_{\text{MOT}} = \sum_{t=1}^{T} \sum_{i=1}^{N} 1[Z_t[i] = j]L_{\text{assign}}(S_t[i], M_t[j]), \quad (4)
\]

where \( Z_t \in \mathbb{R}^N \) denotes the assignments and \( Z_t[i] = j \) means the \( i^{th} \) slots matches to the \( j^{th} \) buffer. Specifically, we have three options to calculate the assignment cost: 1) use a binary cross-entropy loss on the decoded masks to promote the consistency of object attributes such as shape; 2) use a pixel-wise squared reconstruction loss on the object reconstructions (pixel reconstructions multiplied by object masks) to learn the color information; 3) use the same loss as 2) but directly apply on the feature space. The assignment cost could be a combination of the three losses:

\[
L_{\text{assign}}(S_t[i], M_t[j]) = \lambda_1 \text{BCELoss}(\text{Dec}(S_t[i]), \text{Dec}(M_t[j])) + \lambda_2 ||\text{Dec}(S_t[i]) - \text{Dec}(M_t[j])||^2 + \lambda_3 ||S_t[i] - M_t[j]||^2,
\]

where \( \lambda_1, \lambda_2, \) and \( \lambda_3 \) are the balancing weights. We use the frozen decoder from the object-centric model to decode object representations into pixel reconstructions and masks.

**Optimization.** In contrast to prior supervised trackers \([7, 63]\) that use ID labels to find the assignments, our model learns the index matrix \( I_t \) without any supervision. One naive solution is to convert \( I_t \in \mathbb{R}^{N \times M} \) to \( Z_t \in \mathbb{R}^N \) by
performing argmax along the buffer dimension. However, the argmax function is non-differentiable. Even though we apply the straight-through trick [27] to make it trainable, the optimization easily gets stuck in a local minimum because the model has no chance to evaluate other possible assignments. To tackle this problem, we take inspiration from the Expectation-maximization (EM) paradigm which optimizes the assignments from seeing all possible assignments in \( \mathcal{I}_t \).

We formulate the expectation of \( S_i \) matches to \( \mathcal{M}_t \) as:

\[
Q(\theta^*, \theta) = \mathbb{E}[\ln p(S_i, \mathcal{M}_t | \theta^*)] = \sum_i \sum_j p(\mathcal{M}_t^i | S_i^t) \ln p(S_i^t, \mathcal{M}_t^i | \theta^*) = -\sum_i \sum_j \mathcal{I}_t[i,j] \mathcal{L}_{\text{assign}}(S_i^t, \mathcal{M}_t^i).
\]

Here, \( \theta \) is the learnable parameters in the memory module. \( p(\mathcal{M}_t^i | S_i^t) \) denotes the probability of the \( i^{th} \) slot being assigned to the \( j^{th} \) buffer, which, in our model, it exactly equals \( \mathcal{I}_t[i,j] \). Further, we can use \( \mathcal{L}_{\text{assign}} \) to represent \( \ln p(S_i^t, \mathcal{M}_t^i | \theta^*) \). We optimize the parameters of the model in order to maximize the expectation via SGD [49], for which we rewrite equation (4) as:

\[
\mathcal{L} = \sum_{t=1}^{T} \sum_{i=1}^{N} \sum_{j=1}^{M} \mathcal{I}_t[i,j] \mathcal{L}_{\text{assign}}(S_i^t, \mathcal{M}_t^i).
\]

The above loss (7) is applied to both the merged results \( \mathcal{M}_t \) and rollout \( \mathcal{M}_t \) with each combination weight set as 1.

### 3.5. Model Inference

During inference, we binarize the indexing matrix \( \mathcal{I}_{t,\text{hard}} \in \{0, 1\}^{N \times M} \) to strictly assign one slot to one buffer. Specifically, \( \mathcal{I}_{t,\text{hard}}[i,j] = 1 \) only if \( j = \text{argmax}(\mathcal{I}_t[i]) \) for \( i \in [1, N] \); otherwise, \( \mathcal{I}_{t,\text{hard}}[i,j] = 0 \). The discrete index supports the object in-n-out logic by indicating the presence of an object.

**Object-in logic.** For the first frame, we filter out duplicate slots before using them to initialize memory buffers. Slots with high mask IoU (bigger than \( \tau_{\text{in}} \)) to other slots will be discarded. For the next frames, we activate new buffers for new objects if slots have no substantial IoU with any masks of the memory rollout from the last timestep \( \{\mathcal{M}_1^{i-1}, \ldots, \mathcal{M}_k^{i-1}\} \), where \( k \) is the active buffer number. Note that, for training, we replace the rollout with slots from the last timestep \( \{S_1^{i-1}, \ldots, S_N^{i-1}\} \) because the rollout is not reliable at the early training stage.

**Object-out logic.** To re-track an object, we keep the buffer alive for \( \tau_{\text{out}} \) consecutive frames when the object is occluded or disappears. In other words, if an object disappears for more than \( \tau_{\text{out}} \) frames, the buffer will be terminated.

4. Experiments

We show that 1) OC-MOT consolidates “objects” in memory and greatly improves the temporal consistency of object-centric representations; 2) the gap between object-centric learning and MOT can be narrowed by involving partial labels to improve the grouping performance; 3) the ablation studies demonstrate the effectiveness and feasibility of each module in the framework. Finally, we turn to KITTI [18] to discuss our limitations.

**Datasets.** CATER [19] is a widely used synthetic video dataset for object-centric learning. It is rendered using a library of 3D objects with various movements. Tracking multiple objects requires temporal reasoning about the long-term occlusions, a common issue in this dataset. FISHBOWL [54] consists of 20,000 training and 1,000 validation and test videos recorded from a publicly available WebGL demo of an aquarium, each with a resolution of 480x320px and 128 frames. Compared to CATER, FISHBOWL records more complicated scenes and has even more severe object occlusions. Besides, we also work on the real-world driving dataset KITTI [13] to analyze the limitation of the proposed object-centric framework.

**Metrics.** Following the standard MOT evaluation protocols [48, 41], we use Identity F1 score (IDF1), Multiple-Object Tracking Accuracy (MOTA), Mostly Tracked (MT), Mostly Lost (ML), and Identity Switches (IDS) as the metrics. Specifically, IDF1 highlights the tracking consistency, and MOTA measures the object coverage. To weight down the effect of detection accuracy and focus on the association performance, we set the IoU distance threshold as 0.7. We also introduce Track mAP [12], which is more sensitive to identity switches by matching the object bounding boxes to ground-truth through the entire video using 3D IoU.

**Implementation details.** We train OC-MOT using the Adam optimizer [31] with a learning rate of \( 2 \cdot 10^{-4} \) and an exponentially decaying learning rate schedule. The mod-
els were trained on 8 NVIDIA GPUs with batchsize of 8. We set $\tau_{out}$ as 5 for buffer termination. The IoU threshold $\tau_{iou}$ is set as 0.9. For the experiments on CATER, we pretrain a SA Vi model for object grouping without any annotation. We set $N = 11$ and $M = 15$. The hyperparameters in the training loss $\lambda_1, \lambda_2, \lambda_3$ are selected as 1, 0.1, 0. For the experiments on FISHBOWL, we used a pretrained image-level DINOSAUR [50] as the grouping module and selected $\lambda_1, \lambda_2, \lambda_3$ as 1, 0, 1. We set $\lambda_2$ to 0 due to GPU memory limitations when combining the EM loss computation with the high dimensional DINOSAUR features. We set $N = 24$ and $M = 40$. In complex scenes of FISHBOWL, we noticed a performance drop due to more severe part-whole issues and over-segmentation on the background as illustrated in Figure 1. To avoid tracking background objects and reduce over-segmentation on big objects, we suggest further improving object-centric grouping by utilizing temporal sparse labels. To be more specific, we apply supervised DETR [8]-style loss on the decoded masks of slots. Since the object grouping loss already takes the heavy-lifting of discovering objects and parts, we only require very few mask labels to inject semantics about which objects are interesting and how to merge parts into a whole object. In practice, we utilized 6.25% (randomly label 8 frames in 128-frame videos) mask labels for DINOSAUR pre-training, with both DETR loss and self-supervised reconstruction loss.

### 4.1. Comparison with the State-of-the-art Methods

#### Baselines.
We compare OC-MOT with one object-centric method (SAVi [32]), three unsupervised MOT methods (IOU [5], SORT [4], and Visual-Spatial [2]), and one fully supervised MOT method (MOTR [63]). For the SAVi evaluation, we remove the background slots and treat each slot prediction as a tracklet. When training SAVi on FISHBOWL, we also provide 6.25% temporal sparse mask labels to be comparable to our own setting. For fair comparisons, we use the same pre-trained object-centric model (DINOSAUR with 6.25% detection labels) as the detector for IOU, SORT, and Visual-Spatial. MOTR utilizes a query-based transformer for both object detection and association but requires object-level annotations (both boxes and object ID) for model training. This model and its follow-ups have achieved SOTA results on several MOT benchmarks.

#### Results on CATER.
As shown in Table 1, OC-MOT substantially outperforms the video object-centric model and other unsupervised baselines. Our approach is also competitive with supervised MOTR [63] trained on expensive object-level annotations, yielding only slightly lower IDF1 and MOTA. OC-MOT can keep tracking more objects but produces lower ID switches. For example, it achieves 82.3% Mostly Tracked (MT) and 13.9% Mostly Lost (ML), and shows only 5658 IDS. Moreover, in Table 2 SAVi achieves 90.2% of FG-ARI but performs bad in terms of other MOT metrics such as 42.8% Track mAP, indicating that the FG-ARI is not a good metric for measuring object-level temporal consistency.

#### Table 1. Evaluation results on CATER and FISHBOWL.
For CATER, the object-centric grouping module is pre-trained without any label. For FISHBOWL, the grouping module is pre-trained with 6.25% mask labels to improve the detection accuracy. The supervised MOTR [63] is trained with 100% box labels and ID labels. The best results of unsupervised trackers are marked in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Detection Label</th>
<th>ID Label</th>
<th>IDF1 ↑</th>
<th>MOTA ↑</th>
<th>MT ↑</th>
<th>ML ↓</th>
<th>FP ↓</th>
<th>FN ↓</th>
<th>IDS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAVi [32]</td>
<td>73.2%</td>
<td>52.5%</td>
<td>75.2%</td>
<td>21.2%</td>
<td>305027</td>
<td>130810</td>
<td>20352</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IOU [5]</td>
<td>83.0%</td>
<td>77.4%</td>
<td>73.3%</td>
<td>17.4%</td>
<td>35480</td>
<td>173595</td>
<td>8259</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SORT [4]</td>
<td>84.5%</td>
<td>79.2%</td>
<td>71.8%</td>
<td>24.1%</td>
<td>43097</td>
<td>148068</td>
<td>8219</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visual-Spatial</td>
<td>85.8%</td>
<td>80.3%</td>
<td>76.6%</td>
<td>20.8%</td>
<td>51348</td>
<td>129680</td>
<td>7562</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>OC-MOT</strong></td>
<td><strong>88.6%</strong></td>
<td><strong>82.4%</strong></td>
<td><strong>82.3%</strong></td>
<td><strong>13.9%</strong></td>
<td><strong>57792</strong></td>
<td><strong>105054</strong></td>
<td><strong>5658</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MOTR [63]</strong></td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
<td><strong>93.8</strong></td>
<td><strong>88.6</strong></td>
<td><strong>82.4</strong></td>
<td><strong>42.8</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>OC Metric</th>
<th>MOT Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FG-ARI ↑</td>
<td>IDF1 ↑</td>
</tr>
<tr>
<td>SAVi [32]</td>
<td>90.2</td>
<td>72.3</td>
</tr>
<tr>
<td>OC-MOT</td>
<td><strong>93.8</strong></td>
<td><strong>88.6</strong></td>
</tr>
</tbody>
</table>

Table 2. Comparisons with video object-centric models on CATER. Note that FG-ARI [32] is a commonly used OC metric.
Results on FISHBOWL. FISHBOWL is a more challenging benchmark with serious occlusions and complicated backgrounds. In this scenario, SAVi [32] tends to split the complicated background into multiple slots, causing the high number of false positive (FP). Table 1 shows that OC-MOT achieves state-of-the-art performance among the unsupervised tracking methods. By getting a much lower IDS number, our approach shows its advantage in solving the occlusion problem. The non-linear transformation in frequent occlusions cannot be handled by IoU-based association (IOU [5]) or Kalman filter (SORT [4]). Compared to supervised MOTR, what would like to highlight is the impressive association capability of OC-MOT. We point out that the lower IDF1 and MOTA are mainly caused by the detection limitation of existing OC models (e.g., DINOSAUR decoder predicts masks at a low feature resolution, making it hard to get pixel-level accuracy).

4.2. Visualization

The MOT results on the occlusion cases are visualized in Figure 5. OC-MOT associates the slots from the object-centric model and generates consistent predictions even when objects frequently interact with each other. Due to the severe occlusions, SAVi [32] fails to track objects even using the track query as input, thereby causing more ID switches. Moreover, SAVi produces more false positives due to over-segmentation.

In Figure 6, we visualize the memory rollout results by decoding the representations to object reconstructions. The memory starts to roll out after the first frame, and, at t = 1, we visualize the existing memory features. We can observe that the rollouts achieve good temporal consistency and, even more interestingly, that the memory can predict a complete object even when it has been partially occluded.

4.3. Ablation Studies

Component analysis. Table 3 compares different design choices for the key components in OC-MOT. For the index-
merge module, a naive solution is to use a parameter-
free dot-product to calculate the feature similarity, inspired
by [47]. As expected, it produces the worst association per-
formance. A further option is to train one single MHA (i.e.,
two MHAs with shared weights) to cluster slots to buffers
as in [23]. To get the discrete index for object in and out
logic, we still follow the indexing and merging steps yet
only calculating the attention weights once. We observe
that this model yields slightly lower IDF1 and MOTA than
training two MHA modules. The latter choice is mathemat-
ically the similar but with higher module capacity. For the
memory module, we compare utilizing the rollout module
to only using the last tracks as the index query. Without
aggregating the historical memory features, the association
performance drops dramatically, indicating the necessity of
building a memory to handle the MOT problem.

<table>
<thead>
<tr>
<th>Index-Merge Module</th>
<th>Memory Module</th>
<th>IDF1 ↑</th>
<th>MOTA ↑</th>
<th>IDS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dot-product</td>
<td>Rollout</td>
<td>72.0%</td>
<td>61.5%</td>
<td>22050</td>
</tr>
<tr>
<td>One MHA</td>
<td>Rollout</td>
<td>86.2%</td>
<td>80.5%</td>
<td>7655</td>
</tr>
<tr>
<td>Two MHA</td>
<td>Last Tracks</td>
<td>77.2%</td>
<td>68.8%</td>
<td>16582</td>
</tr>
<tr>
<td>Two MHA</td>
<td>Rollout</td>
<td>88.6%</td>
<td>82.4%</td>
<td>5658</td>
</tr>
</tbody>
</table>

Table 3. Ablation on OC-MOT components on CATER.

Effect of memory length. In Table 4 we explore the ef-
effect of the memory length $T_{max}$ from 6 to 32. Note that
$T_{max}$ equals the length of the training sequence. The track-
ing performance increases as $T_{max}$ grows. However, for
longer videos, we should set a max length of memory due
to hardware limitations. To make the model more appli-
cable, we propose to reserve a short-term memory trained
with sequences sampled by slow-to-fast pace. The various
sampling rates produce both short-term and long-term infor-
mation and, more importantly, include the occlusion cases
during training. Quantitatively, this sampling strategy peaks
in performance with $T_{max} = 6$. Unless noted otherwise, we
set $T_{max}$ to 6 as default in other experiments.

<table>
<thead>
<tr>
<th>$T_{max}$</th>
<th>Sequence Sampling</th>
<th>IDF1 ↑</th>
<th>MOTA ↑</th>
<th>IDS ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Consecutive</td>
<td>82.9%</td>
<td>76.3%</td>
<td>7601</td>
</tr>
<tr>
<td>10</td>
<td>Consecutive</td>
<td>83.2%</td>
<td>76.5%</td>
<td>7524</td>
</tr>
<tr>
<td>20</td>
<td>Consecutive</td>
<td>86.9%</td>
<td>78.1%</td>
<td>6230</td>
</tr>
<tr>
<td>32</td>
<td>Consecutive</td>
<td>88.4%</td>
<td>82.1%</td>
<td>5763</td>
</tr>
<tr>
<td>6</td>
<td>Slow-Fast</td>
<td>88.6%</td>
<td>82.4%</td>
<td>5658</td>
</tr>
</tbody>
</table>

Table 4. Ablation on the memory length on CATER.

4.4. Limitations

There exist some limitations of the proposed OC-MOT.

Inductive bias in the grouping architecture. We apply
the same DINOSAUR+DETR grouping module with 6.25%
temporally sparse mask labels to KITTI dataset. Figure
1 visualizes the grouping results. The cars can be de-
tected but the predicted masks are not accurate, especially
for far-away objects. One reason is that DINOSAUR pre-
dicts masks at a feature resolution that is down-scaled 16
times from the original size. The architecture of the group-
ing module needs to be further improved considering multi-
resolution inductive biases that have already been adopted
in supervised detection and segmentation pipelines. We en-
courage researchers to develop stronger OC models with
powerful detection performance but low labeling cost.

The model is not trained end-to-end. In this paper, we use
the pre-trained OC model as a plug-n-play detector, which
is supposed to handle different data flexibly. Potential future
work is to extend OC-MOT into an end-to-end framework.
The object prototype built in the memory may be useful as
a prior for object discovery.

5. Conclusion

In this paper, we build a pipeline for MOT with object-
centric backbones. With memory modules, we can address
both part-whole issues and consistently track objects over
time. Overall, our approach improves over conventional
tracking-by-detection pipelines by replacing expensive an-
notations (especially ID annotations) with self supervision.
This work opens many directions for new research. First
of all, it allows for active learning. For example, the model
could elicit a request for labeling on specific frames, further
reducing necessity for costly annotations. Furthermore,
incorporating memory information as top-down reasoning
prior for the object-centric encoder still remains to be
explored. Additionally, we still require few masks and
class labels to resolve over-segmentation. Those semantic
signals could be distilled from multi-modal foundation
models trained with weaker supervision signals (e.g.,
captioned images). Finally, our results delineate a clear
benefit in improving (video) object-centric backbones. As
we have demonstrated, improvements in self-supervised
object-centric learning can greatly facilitate complex
downstream vision tasks like MOT, improving performance
by training on unsupervised or weakly-supervised data.
References


[54] Matthias Tangemann, Steffen Schneider, Julius Von Kügelgen, Francesco Locatello, Peter Gehler, Thomas


