A Societal impact

Our work focuses on extending object-centric representation learning to real-world videos. This class of methods has the potential for enabling systems to more reliably solve down-stream tasks requiring object representations, such as relational reasoning over entities in videos, and providing better interpretation of model decisions. Some applications that may benefit from this approach include perception in autonomous vehicles, robotics and classic computer vision problems such as object detection. While the method demonstrated in this paper is still far from a state where it could be directly employed in computer vision applications, we would like to raise awareness that—as with most methods developed for computer vision—advances in this field might also aid the development applications with potential negative societal impact such as surveillance.

B Additional results

Qualitative results In Figure 7, we show the effect of our mask thresholding heuristic applied for our unsupervised model visualizations. The intention for this simple heuristic is to aid interpretability of the discovered object segmentation masks. We further show qualitative results for emergent tracking on long sequences in the unconditional setting (i.e. without bounding box conditioning) in Figure 8. Compared to the conditional setting (shown for reference), tracking is less consistent and slots explain not only cars, but also environmental objects or part of the background.

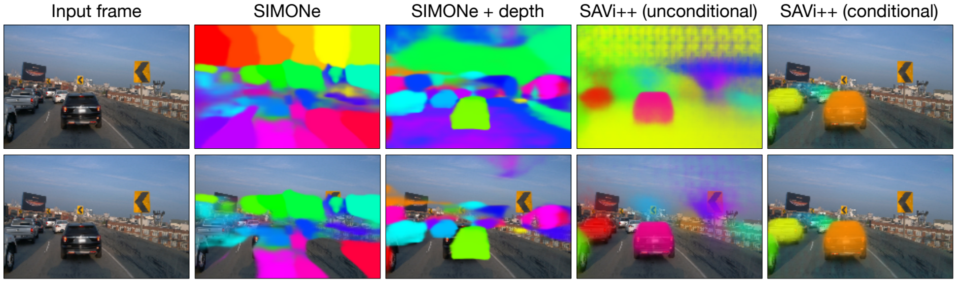

Figure 7: Comparison of qualitative result visualization without (top) and with (bottom) mask thresholding. We apply a simple thresholding heuristic of dropping any masks that (on average across frames) occupy more than 1300px per frame, which aids interpretability. Thresholding does not have an effect on the conditional model.

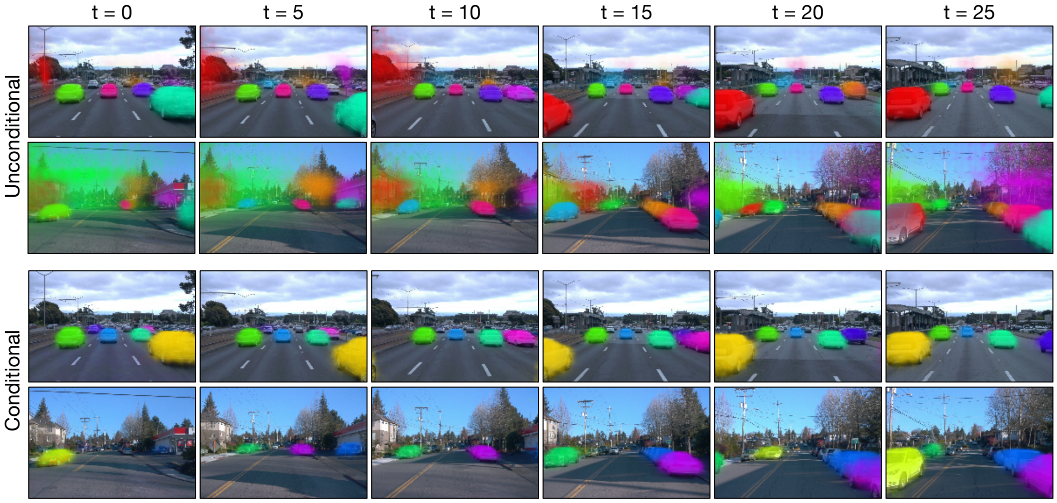

Figure 8: Qualitative results on long Waymo Open validation set videos (for SAVi++ models trained on 6 frames). Top: Results for an unconditional SAVi++ model, trained and evaluated without bounding box conditioning in the first frame. Bottom: Conditional setting shown for reference.

Quantitative results We additionally compare SAVi and SAVi++ on the synthetic MOVi-A and MOVi-B datasets [20]. In MOVi-A, scenes consist of a gray floor, four light sources, a fixed static camera and between 3 and 10 simple geometric objects that vary in terms of their shape (cube, sphere, cylinder), material (rubber, metal), size (small, large), and color (blue, brown, cyan, gray, green, purple, red, yellow). MOVi-B is a straightforward extension, which adds additional variation to the object shape, size, and color; background color; and static camera positions.

In Table 3 it can be seen how SAVi++ performs similar or worse in terms of mIoU and FG-ARI on these datasets. We mainly attribute this to our strategy for scaling in SAVi++, which appears susceptible to overfitting on these much simpler domains. Additionally, the benefit of using depth information is limited, since all objects are in motion for these datasets. The contrast between the results presented in the main paper for MOVi-C to -E and Table 3 (MOVi-A and -B) emphasizes the importance of considering benchmarks that are more representative of the real world for model development.

Table 3: MOVi results in terms of mean score ± standard error (5 seeds) from evaluating SAVi++ and SAVi models on validation set video sequences of increased length (24 frames).

	mIoU↑ (%)		**FG-ARI↑** (%)	
Model	**MOVi-A**	**MOVi-B**	**MOVi-A**	**MOVi-B**
SAVi [37]	82.3 ± 0.3	44.5 ± 9.3	96.8 ± 0.4	73.9 ± 10.7
SAVi++	76.1 ± 0.9	25.8 ± 11.3	98.2 ± 0.2	48.3 ± 15.7

Table 4: Breakdown of SAVi++ results from Table 2 in terms of three classes of objects: car, person, and cyclist.

Metric	Car	Person	Cyclist
Num. objects	15350	2102	275
CoM↓	4.2 ± 0.2	6.4 ± 0.1	1.9 ± 0.1
B. mIoU↑	52.5 ± 0.8	27.0 ± 0.5	42.2 ± 2.1
B. Recall↑	96.7 ± 0.7	95.1 ± 1.1	99.3 ± 0.3

We report per category results for SAVi++ on the WaymoOpen dataset in Table 4. We find that, as expected, this dataset is dominated by cars and performance is very good in this category. SAVi++ also performs very well on cyclists, which are very rare in this dataset, and indicates that SAVi++ has not overfit to blobby car like objects.

We further report results for SAVi++ under the influence of noise in the sparse depth targets on Waymo Open in Table 5. Our results indicate that emergent tracking performance is largely unaffected by noise scales (standard deviations) of up to $\sigma = 40$ cm.

Supplementary videos We provide several video results in the supplementary material for both SAVi++ (conditional, uncondtional, and high-resolution model variants) and the SIMONe [30] baseline (both in its original form and in our adapted depth-prediction variant).

Table 5: Waymo Open results (mean ± standard error in %, 3 seeds) from evaluating models on sequences of 10 frames. SAVi++ was trained with noisy depth targets with standard deviation (σ) specified below.

		(%)	
Model	**CoM↓**	**B. mIoU↑**	**B. Recall ↑**
SAVi++	4.4 ± 0.2	49.7 ± 0.7	96.5 ± 0.7
SAVi++ ($\sigma = 10$ cm)	4.3 ± 0.2	50.1 ± 0.3	96.8 ± 0.3
SAVi++ ($\sigma = 20$ cm)	4.3 ± 0.2	49.7 ± 0.3	97.4 ± 0.5
SAVi++ ($\sigma = 40$ cm)	4.2 ± 0.0	50.1 ± 0.3	96.9 ± 0.5

C Training setup

We train our models for 500k steps (300k steps for the ablation study) on Tensor Processing Unit (TPU) accelerators with a batch size of 64 using Adam [35]. We linearly increase the learning rate for 2500 steps to 0.0002 (starting from 0) and then decay the learning rate with a Cosine schedule [43] back to 0 for the rest of the training steps. We clip the gradients to a global norm value of 0.05 to stabilize training. To create video examples for training, we split each video into sub-sequences of 6 frames each. We use a total of 24 slots on MOVi and 11 slots on Waymo Open for SAVi++ models. We use 1 iteration per frame for the Slot Attention [42] module (as in prior work), unless stated otherwise.

Following the conditional setup in Kipf et al. [37], the initial state of the slots is obtained by encoding bounding boxes corresponding to the objects in the first frame; thus providing the model with rough cues of which objects to bind to initially. For the unconditional experiments, we initialize slots using an equal amount of learnable parameter vectors. In experiments that use optical flow, we convert the 2D flow signal to three RGB channels following prior work [68]. As described above, we apply a log-transform to the (sparse) depth signal (incremented by 1 to avoid underflow). We train our model to minimize the squared error (L2 loss) between the predicted and ground-truth targets in pixel space. We implement SAVi++ in JAX [3] using the Flax [23] neural network library. Training SAVi++ on a single MOVi dataset on 8 TPUv4 chips with 32GiB memory each takes approximately two days for 500k training steps.

D Model details

D.1 SAVi++

Our architecture building blocks are similar to that of the SAVi model from Kipf et al. [37] with all the parameters shared across time steps. SAVi++ uses exactly the same parameters for the slot initializer and decoder as the SAVi model (except for SAVi++ HR where we use an additional 5×5 ConvTranspose layer with stride 2 and 64 channels to account for the higher resolution frame size). Below we list the details and hyper parameters of all the modules of SAVi++:

Encoder We used a ResNet-34 [22] backbone with modified root convolutional layer that has 1×1 stride (except for SAVi++ HR that uses a root stride of 2×2). For all layers, we replaced the batch normalization operation by group normalization [65]. We used a linear positional encoding identical to that used in Slot Attention [42] with horizontal and vertical coordinates normalized to $[-1, 1]$ range. These coordinates were then projected to the same size of the ResNet feature maps using a learnable linear layer. Finally, the ResNet features and positional encoding are combined by an addition operation. Following the backbone, the frame features are projected to 64 embedding dimensions by a linear layer followed by ReLU activation and then fed to a transformer network with 4 transformer blocks, except for SAVi++ unconditional Waymo Open models where we found that ResNet features alone produced clearer segmentations. This is likely due to the stronger CNN image prior being beneficial with less supervision [59]. Each transformer block uses a multi-head dot-product attention from [61] with pre-normalization [66] and 4 attention heads. For each attention head, the query/key/value embedding size was set to 16. The output of each block is then processed by a residual feed-forward block with pre-normalization, using an MLP with a single hidden layer of 1024 hidden units and ReLU activation function.

Corrector/Predictor Similar to SAVi [37], we use 1 iteration for the Slot Attention corrector module. We increase the corrector query/key/value size to 256 compared to 128 embedding size used in SAVi. For the predictor, we similarly increased the query/key/value projection size to 256, and the MLP hidden layer size to 1024. We found that the larger embedding sizes for the corrector and predictor increased SAVi++ mIoU by a few percentage points.

Decoder Our decoder follows that of SAVi [37] with two exceptions: For Waymo Open, we use a larger spatial broadcast grid of 8×12 as video frames have 128×192 resolution. For SAVi++ HR, we use frames of a higher resolution of 256×384 and add an additional 5×5 ConvTranspose layer with stride 2 and 64 channels to account for the higher resolution frame size. The decoder otherwise uses four 5×5 ConvTranspose layers with stride 2 and 64 channels, followed by ReLU activations.

Initializer Similar to SAVi, we consider two initializers *conditional* and *unconditional* to set the initial state of the models' K slots. For the conditional case, the initializer mapped each of K bounding boxes (represented using 4D coordinates) via a trainable MLP to set the D dimensional state of the corresponding slot. For the unconditional case, rather than associating a specific bounding box in a specific video to a slot, we learn K D-dimensional initial slot states to be used in all videos.

Data augmentation As discussed in the main text, an Inception style [57] random crop is used for data augmentation. Crops are afterwards resized to the target resolution (128×128 for MOVi and 128×192 for Waymo Open, unless otherwise mentioned). We ensure that the cropped view covers at least 75% of the original frame in the MOVi datasets and 20% in Waymo Open. More aggressive cropping worked best in WaymoOpen, perhaps because of the more structured frontal camera subset that we are using. We take care to handle depth maps and flow fields during this operation as explained next for each of the datasets. *MOVi:* Say the crop size is $h \times w$, then optical flow fields are cropped, resized, and then rescaled by $\left[\frac{128}{w}, \frac{128}{h}\right]$. Dense depth maps are simply cropped and resized. *Waymo Open:* Sparse LiDAR point clouds are projected to 2D and retained as tuples (2D point, lidar-range) throughout the data augmentation pipeline. The affine transformation equivalent to the crop and resize operation is computed and applied to these 2D points. As a consequence, several points will fall outside the cropped frame. These points are discarded when projecting them into a depth image after data augmentation is complete.

D.2 Baselines

SAVi For the SAVi baseline, we use the best-performing model variant described in Kipf et al. [37], i.e. SAVi trained with a Resnet34 backbone. Different from SAVi++, this baseline does not use depth prediction (it only predicts optical flow), does not use data augmentation, and does not use a transformer encoder after the convolutional backbone. We choose the same hyperparameters as described in SAVi [37].

SIMONe This baseline [30] is a non-autoregressive model for encoding short video clips of fixed lengths into a set of latent object variables (fixed across time) and a per-frame global latent variable. Note that SIMONe cannot be applied auto-regressively and has to be applied to the same sequence length at both training and test time. SIMONe uses a CNN encoder per frame followed by a transformer encoder that is applied across frames to finally obtain object and frame latent variables by pooling transformer tokens across time and space, respectively. The model is trained by reconstructing input frames using a form of a spatial broadcast decoder and additionally uses a KL-based regularizer on the latent variables. We use a JAX [3] reimplementation of the SIMONe model for which we verified that it reproduces results mentioned in the paper on the CATER [15] dataset. We train SIMONe on sequences of 6 frames (same as SAVi++) at a resolution of 128. We subsample reconstruction targets by a factor of 4 (see Kabra et al. [30] for details). Other hyperparameters are chosen as follows: reconstruction loss scale $\alpha = 0.2$, pixel likelihood scale $\sigma_x = 0.08$, object latents KL loss weight $\beta_o = 1\text{e} - 5$, and frame latents KL loss weight $\beta_f = 1\text{e} - 4$. We encode frames using a 4-layer CNN with 128 channels, $(4, 4)$ kernel size and $(2, 2)$ stride. Each transformer uses 4 layers, 5 heads, a qkv-size of 64 per head, and an MLP hidden layer size of 1024. Latents are of size 32. The decoder uses an MLP with 5 hidden layers of 512 units. To train SIMONe with sparse depth targets, we replace the RGB target signal with the LiDAR-based depth signal and only compute the loss for pixels that have a depth signal.

CRW Contrastive Random Walks (CRW) [28] is a cycle consistency based self-supervised learning method for learning grid structured latent representations. After pre-training, a simple label propagation scheme can be applied on these latent representations to obtain tracking behavior. This typically requires segmentation labels for the objects of interest in the first frame. The method tracks these objects and outputs segmentation masks for them over subsequent frames. In order to use this method with only bounding box conditioning in the first frame, we flood fill boxes into rectangular masks and propagate those instead. Overlap between boxes is resolved based on the box order.

We adopted their training and evaluation best practices. We pre-trained stride-8 ResNet backbones [22] using their publicly available code and propagated labels using the activations output by the second last ResNet stage. For pre-training we tuned edge-dropout, training temperature and for tracking we tuned evaluation temperature independently on each of the three MOVi datasets. We found that, despite our efforts, a ResNet34 backbone was not able to train using the cycle consistency loss. We obtained much better results using a ResNet18 backbone, which is the model we report results for. Optimal hyper-parameters (dropout, training temperature, evaluation temperature) were as follows: MOVi-C $(0.0, 0.001, 0.5)$, MOVi-D $(0.05, 0.001, 0.5)$, MOVi-E $(0.05, 0.001, 0.5)$. Other relevant hyper-parameters are: training clip length (6), frame-skip (1), batch size (16), learning rate (0.0001), training epochs (125) with a learning rate drop after the 100^{th} epoch.

Bounding box copy We simply repeat the bounding boxes of the objects visible in the initial frame for the rest of the video sequence. To obtain pixel-level segments for computing metrics, such as FG-ARI, we 'render' the entire bounding box as a segment in pixel space. Bounding boxes are rendered in the same order as they were provided in the initial frame, such that later boxes take precedence when multiple of them cover the same pixel location.

Learned bounding box propagation In this baseline, we use the SAVi++ initializer and predictor (without encoder, corrector, or decoder) to learn a bounding box propagation model. The model receives (just as in SAVi++) bounding boxes for all objects in the first frame of the video, which are passed to the initializer to learn initial slot representations. Afterwards, the predictor learns a mapping of slots at time step t to slots at time step $t + 1$. We train the model by reading out individual slot representations at each time step using an MLP with a single hidden layer of 256 units that predicts the corner coordinates (top-left and bottom-right) of the bounding box associated with a slot at a particular time step, supervised using ground-truth bounding boxes. We use the Huber [26] loss (L2 loss between $[-1, 1]$ and L1 loss outside of this interval) to train the model. If an object is not visible or present, its bounding box is set to $[0, 0, 0, 0]$ in the ground-truth target.

K-Means clustering of flow/depth To evaluate how much information about instance segmentation can be exctracted directly from the depth and optical flow modalities, we evaluate a k-Means clustering baseline on videos from MOVi and Waymo Open. For that purpose we treat each pixel of a video as a datapoint, each with 7 dimensions: one for log-depth ($\log 1 + d$), three for optical flow converted to RGB (only for MOVi), two for linear position encoding, and one for time. All dimensions are normalized to the range of $[0, 1]$. For Waymo Open we discard all points that do not have an associated depth value. To make it as comparable as possible to the conditional setup of SAVi++, we set k to the ground-truth number of objects plus one for the background, and initialize each cluster-center to the average value of points within the first-frame bounding box of each object. The background cluster is initialized to the average value of all points in the first frame. K-Means is then run until convergence, and we evaluate the resulting cluster-assignments using mIoU and FG-ARI scores for MOVi, and by computing the normalized distance of the center of mass of each segment to the corresponding bounding box for Waymo Open.

Supervised baseline To estimate how much headroom there is in terms of tracking performance given our model architecture, we train a variant of the SAVi++ model where we replace the depth decoder with a bounding-box prediction head. Instead of self-supervised training using depth prediction, this model is trained to directly predict object bounding boxes at every time step given the slots of the model. This is similar to a TrackFormer [45] model, but we instead use the SAVi++ architecture and we train in a conditional setting (i.e. initial first-frame bounding boxes are provided as slot initialization), which means we can train the model without using any form of matching. Similar to the learned bounding box propagation baseline, we train the model by reading out individual slot representations at each time step using an MLP with a single hidden layer of 256 units that predicts the corner coordinates (top-left and bottom-right) of the bounding box associated with a slot at a particular time step, supervised using ground-truth bounding boxes. We apply a Huber [26] loss (L2 loss between $[-1, 1]$ and L1 loss outside of this interval) for training. If an object is not visible or present, its bounding box is set to $[0, 0, 0, 0]$ in the ground-truth target.

E Datasets

We used the synthetic Multi-Object Video (MOVi) datasets introduced in Kubric [20]. The Kubric dataset generation pipeline is available under an Apache 2.0 license. For real-world experiments, we used the Waymo Open dataset [56]. The Waymo Open dataset is licensed under the Waymo Dataset License Agreement for Non-Commercial Use (August 2019): `https://waymo.com/open/terms`.

Dataset details are summarized in the following:

- **MOVi-C**: uses approximately 380 high-resolution HDR photos as backgrounds and three to ten dynamic objects obtained from a set of 1028 3D-scanned everyday objects [16], representing various household objects. The camera in this dataset has a random pose, yet the camera pose is static across the video sequence. Each video is sampled at 12 frames per second (fps). We trained our models on randomly sampled sequences of 6 frames and evaluate on sequences of 24 frames. We trained our models on 9.75k training set videos, and evaluated models on 250 evaluation set videos. Model tuning was performed on a separately generated set of 250 videos.
- **MOVi-D**: has a similar camera setting as MOVi-C, but adds more objects, the majority of which are initialized to be static. This dataset includes one to three dynamic objects and 10 to 20 static objects in each video sequence. We used the same frame rate and train/evaluation sequence lengths as for MOVi-C. We trained our models on 9.75k training set videos, and evaluated models on 250 evaluation set videos. Model tuning was performed on a separately generated set of 250 videos.
- **MOVi-E**: adds additional complexity compared to MOVi-D by introducing random linear camera movement throughout the video sequence. We used the same frame rate and train/evaluation sequence lengths as for MOVi-C. We trained our models on 9.75k training set videos, and evaluated models on 250 evaluation set videos. Model tuning was performed on a separately generated set of 250 videos.
- **Waymo Open**: contains high-resolution video sequences with frame size of 1280×1920. We solely use videos recorded from the front camera of the car. We down-sampled the videos to 128×192 (or 256×384 for SAVi++ HR). The dataset consists of 798 train and 202 validation scenes of 20s video each, sampled at 10 fps. The dataset includes also 2D bounding box annotations, which we used for the conditional experiments and to compute the B. mIoU evlauation metrics. The Waymo Open dataset further includes LiDAR data collected from five LiDARs; one mid-range LiDARs placed on top of the car and four short-range LiDARs placed front, left, right, and rear. The LiDAR data is used to compute sparse depth targets as discussed in the Methods section. To slightly simplify the task as we train on lower-resolution frames, we discard any bounding box labels in Waymo Open which cover an area of 0.5% or less of the first sampled video frame, both during training and testing.

F Metrics

In the following, we give a detailed overview of the metrics used for each of the datasets.

F.1 MOVi

On the MOVi datasets [20] we have access to ground-truth pixel-level segmentations, which lets us directly measure the quality of the learned segmentations using the same segmentation metrics as in prior work. Note that SAVi and SAVi++ are trained in a conditional setting where we initialize slots using ground-truth bounding box information in the first frame. Because of this, we will only measure metrics from the second frame onward.

Foreground Adjusted Rand Index (FG-ARI) A permutation-invariant clustering similarity metric frequently used for evaluating scene decomposition quality [27, 51]. It compares discovered segmentation masks with ground-truth masks while ignoring any pixels that belong to the background. It is sensitive to temporal consistency of masks, but insensitive to their ordering.

Mean Intersection over Union (mIoU) A standard segmentation metric for measuring the quality of predicted segments. Our implementation is identical to the semi-supervised DAVIS challenge Jaccard-Mean metric for video [6, 50]. We note that this implementation is sensitive to the correct ordering of masks, i.e. it also measures whether models used the conditioning signal (here, first-frame bounding boxes) correctly.

Model selection on MOVi was done mainly using mIoU. On the one hand to avoid learned segments bleeding into the background overly much, and on the other hand to ensure that the bounding box initialization was properly utilized.

F.2 Waymo Open

On the Waymo Open dataset we only have ground-truth bounding boxes available, which necessitates an alternative set of metrics for measuring quantitative performance. Similar to before, because of conditioning, we will only measure metrics from the second frame onward.

Center-of-Mass (CoM) distance This tracking metric measures the average Euclidean distance between the centroid of the predicted segmentation masks and the centers of the ground-truth bounding boxes. The former are obtained by computing the geometric mean of the 2D coordinates associated with the pixels belonging to a segment, where we exclude pixels that do not have a valid LiDAR point associated with them (this is similar to how we compute the loss during training). To allow for comparable CoM distance across multiple resolutions, we report the distance normalized by the maximum achievable distance in the video frame (length of the diagonal). Objects that are fully occluded in a frame (or have disappeared) are excluded from the computation. In the conditional case (i.e. with bounding box information provided to the model in the first frame) we use the order of the provided bounding boxes to compute the metric between each slot and ground-truth bounding box. In the unconditional case, we use Hungarian matching to associate entire bounding box tracks with decoded object masks and we assign a penalty of 1 (maximum CoM distance) for all empty segments.

Bounding Box Recall (B. Recall) This metric measures the fraction of cases where any sort of segment is predicted when a valid ground-truth box exists. It serves a complement to CoM distance when no matching is used and empty segments are not considered. In the case of unconditional evaluation using Hungarian matching, we incorporate a matching penalty of the maximum possible CoM distance for empty segments and thus do not separately report segment recall.

Bounding Box mIoU (B. mIoU) This metric is the bounding box analog of the segmentation mIoU discussed above. Given corresponding predicted and ground-truth bounding box tracks, their per-frame intersection-over-union is computed and averaged over time exactly as in the average IoU metric of the TAO benchmark [9]. Predicted bounding box tracks are obtained using a per-slot readout MLP, with one hidden layer of 256 units. This is jointly trained with the SAVi++ model by minimizing the Huber loss [26] between predictions and $[0, 1]$ normalized ground-truth box coordinates. A stop-gradient is used to prevent these loss gradients from propagating back into SAVi++. Objects that are fully occluded across the entire video sequence are excluded from the computation.

We initially conducted model selection on Waymo Open using a combination of B. mIoU and a heuristic metric to measure what fraction of the pixels belonging to a predicted segment are inside the associated ground-truth bounding box. During the final stages of development, we primarily focused on the B. mIoU metric since it is analogous to mIoU on MOVi.