Object Representations Guided By Optical Flow

Jianing Qian, Dinesh Jayaraman
GRASP Lab
Department of CIS
University of Pennsylvania
`{jianingq,dineshj}@seas.upenn.edu`

Abstract

Objects are powerful abstractions for representing the complexity of the world, and many computer vision tasks focus on learning to understand objects and their properties in images from annotated examples. Spurred by advances in unsupervised visual representation learning, there is growing interest in learning object-centric image representations *without* manual object annotations. We observe that these existing approaches fail to effectively exploit a long-known key signal for grouping object pixels, namely, motion in time. To address this, we propose to guide object representations during training to be consistent with optical flow correspondences between consecutive images in video sequences of moving objects. At test time, our approach generates object representations of individual images without requiring any correspondences. Through experiments across three datasets including a real-world robotic manipulation dataset, we demonstrate that our method consistently outperforms prior approaches including those that have access to additional information.

1 Introduction

The information content in complex visual scenes often boils down to the properties and configurations of a small number of objects within them. This makes objects a natural representational basis for visual perception. Many recent works have focused on *unsupervised object discovery*, the task of automatically identifying the objects in a visual domain represented by an unlabeled dataset of images or videos, and then encoding any new input images from that domain into an object-structured representation that expresses the presence, configurations, and relations of those identified objects. This is useful in downstream tasks such as video prediction, which can be most efficiently expressed in terms of the movements of objects [22–24, 28].

Exciting progress in unsupervised object discovery has come through introducing object structure into unsupervised visual representation learning approaches. Many works have focused on designing network architectures to produce object-structured representations [9, 18, 19, 23, 25, 26], and corresponding inference mechanisms [11–13]. Our contribution is orthogonal to these efforts: we focus on the fundamental problem of identifying useful signals for object discovery, and designing improved objective functions that correctly exploit those signals. Our technique is simple: we compute optical flows on unlabeled videos using off-the-shelf approaches. Then, we train our object encoder to produce representations for nearby frames that are consistent with the optical flow-provided correspondences between those same frames. Our objective function takes the form of a modified contrastive loss. We call this technique FLOOD ("$\underline{\text{Flo}}$w-guided $\underline{\text{o}}$bject $\underline{\text{d}}$iscovery"). We evaluate FLOOD on a variety of synthetic object discovery benchmarks used in previous works. In addition, we introduce a real-world tabletop robotic object pushing dataset. Our results show that FLOOD yields reliable improvements over baseline methods based on reconstruction and contrastive losses, and even an approach that has access to additional information about the forces causing object motions in video.

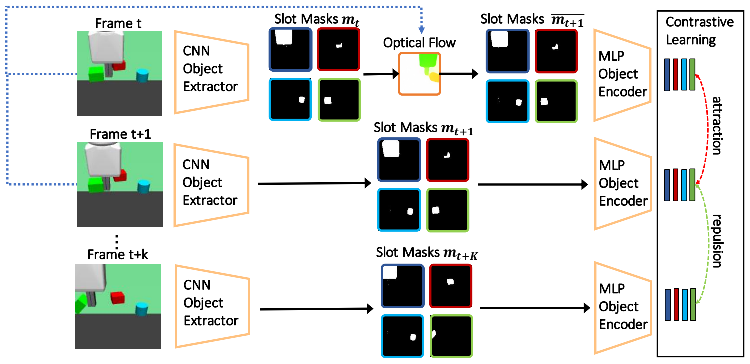

Figure 1: The training setup for FLOOD (Sec 3), applied to object-structured slot representations.

2 Related Prior Work

Video object segmentation from motion and appearance. Our approach for flow-guided object discovery is related to motion segmentation methods [2, 20, 31, 33, 34, 38–40] in computer vision, which use optical flow cues to group pixels in video frames which move coherently together. Since purely flow-based groupings are not persistent and cannot handle static objects, video segmentation algorithms often combine these motion cues with appearance [3, 7, 17, 33, 36, 37]. However, they commonly rely on access to a full pre-recorded video and the corresponding full sequences of optical flows. Specifically, FLOOD trains unsupervised appearance-based static image object segmentation models by using flow-based cues in training videos as a supervision signal.

Neural object discovery. Nearly all neural object discovery approaches targeting general images or videos are trained by reconstructing the input image [4, 11, 13, 18, 19, 23, 25, 28]. Some approaches use adversarial training: the idea is that object-structured representations can be modified and recomposed into valid images [6, 41]. Two recent approaches are trained through contrastive losses to distinguish between distinct images in a dataset [26, 29]. Among these, slot-contrastive networks (SCN) [29] use temporal continuity in video for object discovery. In robotics contexts, where motion is attributable to known robot actions, several models, [21, 22, 35] use action-conditioned future prediction as a training objective for joint object discovery and environment dynamics modeling. FLOOD replaces robot actions with dense pixel correspondences from optical flow, which are more general. We compare against SCN [29] and C-SWM [22] in Sec 4.

3 Flow-guided object discovery

First, we introduce the unsupervised object discovery setting. Given a collection of unlabeled training images $\mathcal{D}$ that specify a visual domain, object discovery approaches aim to automatically identify the objects present among those images. Object discovery is thus closely tied to the *object-structured* representation in which the results of such image parsing are expressed. Indeed, training object discovery approaches amounts to learning unsupervised object-structured representations. Examples of such representations include keypoints [18, 23], object detection bounding boxes [8], segmentation masks [13, 22], and skeletons capturing connected keypoints [19].

Slot representation and encoder. Our approach is not tied to any one choice of object-structured representation, but we ground our discussion and notation using *"slot"* segmentation masks and vectors, which we will also use in experiments. An input image x, of size $H \times W$ pixels, is represented through a collection of K *slot* masks $\boldsymbol{m}(x) = \{m^1(x), m^2(x), \ldots .m^K(x)\}$ of size $H \times W$. All mask values lie between 0 and 1, i.e., $0 < m^k(i,j) < 1$ for all masks k, and pixel positions (i,j). Masks $m^k(x)$ are intended to represent segmentation masks for objects in x. Each slot mask is then summarized as a $D-$ dimensional slot vector $z^k(x)$. $\boldsymbol{z}(x)$ represents all K slot vectors combined. In our work, each slot is free to bind to any object or background in the input image.[1]

[1]This is not true in [22], due to the requirement of object-specific actions for each slot.

We use the popular neural network architecture proposed in [22] to generate slot masks and vectors from input images. An "object extractor" convolutional neural network $\mathcal{F}$ with a sigmoid output layer produces K convolutional maps constrained to lie in $[0, 1]$. These represent the slot masks $\boldsymbol{m}(x)$. Next, a small multi-layer perceptron $\mathcal{G}$ processes each flattened slot mask to produce its corresponding slot vector: $z^k(x)$. This compression serves as a representational bottleneck that encourages the emergence of object-structured representations, as we will discuss below. The second and third rows of Fig 1 illustrate this procedure.

Optical flow, and how motions can help with object discovery. An important characteristic of objects is that objects move as coherent wholes. Yet, popular existing approaches for neural object discovery do not use this information, such as [18, 22, 29]. We propose a novel approach to use the coherent motions of objects in training videos as training signals to guide object discovery. Our key insight is that slots (generalizable to other object-structured representations) should satisfy the following desiderata: **1.** Slots should bind to the same set of objects in both frames; **2.** Slots should be sensitive to the small changes in the *positions* of those objects. In particular, each slot mask m^k should exactly track the displacement of the object that it is bound to.

To frame these desiderata as a mathematical objective, we use *optical flows*. An optical flow field $h_{a\to b}$ defines a warp between images a and b, such that $b = h_{a\to b}(a)$. Now, the desiderata for slots, stated above, boil down to requiring that the pixel correspondences between neighboring frames x and x' are preserved in the correspondences between their slot masks $m^k(x)$ and $m^k(x')$. For all slot indices k:

$$h_{x\to x'}(m^k(x)) = m^k(x'). \tag{1}$$

In other words, if $m^k(x)$ is a good segmentation mask for *some* object in image x, then it maps to the next timestep through the same optical flow warp as the input images themselves.

The FLOOD training objective. Eq (1) permits two degenerate solutions: $m^k(x) = \mathbf{0}$ for all x, k, ("all-zeros"), and $m^k(x) = x$ for all x ("identity"). We avoid these by defining a contrastive training objective over the slot vectors $\boldsymbol{z}(x) = \mathcal{G}(\boldsymbol{m}(x))$. Let $\overline{\boldsymbol{m}(x')} \triangleq h_{x\to x'}(\boldsymbol{m}(x))$ represent the prediction of the slot mask $\boldsymbol{m}(x')$ obtained by flow-warping $\boldsymbol{m}(x)$. Note that this is the LHS of Eq (1), combined over all slot indices k. Now, let $\overline{\boldsymbol{z}(x')} = \mathcal{G}(\overline{\boldsymbol{m}(x')})$ represent its corresponding predicted slot vectors. Then, the flow-guided object discovery (FLOOD) objective is:

$$\mathcal{L} = d(\overline{\boldsymbol{z}(x')}, \boldsymbol{z}(x')) + \max(0, \gamma - d(\boldsymbol{z}(x'), \boldsymbol{z}(x^-))), \tag{2}$$

where $d(\cdot, \cdot)$ is a distance function between slot vectors, and x^- are negatives for x. We set d to be the summed ℓ_2 error over slots, and the negative frames x^- are uniformly drawn from frames that are more than $T = 1$ steps away from the current frame x in the training videos. Recall that x and x' are temporally nearby frames, yet the positives above are not directly their corresponding slot vectors $\boldsymbol{z}(x)$ and $\boldsymbol{z}(x')$. Instead, the slots of x are first warped through the optical flow warp to compute $\overline{\boldsymbol{z}(x')}$.

This contrastive objective trivially precludes the all-zeros degenerate solution above. Avoiding the "identity" solution is more subtle: it is done by applying the contrastive loss to a compressed representation $\boldsymbol{z} = \mathcal{G}(\boldsymbol{m}(x))$, computed by a small MLP $\mathcal{G}$. The low capacity of $\mathcal{G}$ means that it should be very easy to extract a small code $\boldsymbol{z}$ that is sufficiently discriminative to minimize Eq 2. This forces the network to only represent highly processed information in the masks m^k, thus avoiding the identity mapping $m^k(x) = x$.

4 Experiments

Baselines. We compare against three recent baselines, representing different types of object discovery objectives from the literature: **1. Contrastive structured world model** (**C-SWM**) [22] employs an action-conditioned future prediction objective for object discovery, using knowledge of object-specific actions. **2. Slot attention** (**SA**) [25] uses a new attention-based architecture for object discovery through an image reconstruction objective. **3. Slot-contrastive networks** (**SCN**) [29] use temporal continuity for object discovery with a contrastive loss, but does not reason about correspondences between nearby frames. SCN directly produces slot vectors without first generating slot masks, and SA produces slot masks but no slot vectors.

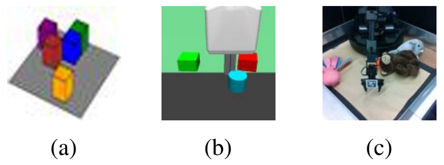

Figure 2: Sample images from the three datasets (a) **CUBE**, (b) **SIM**, and (c) **REAL**.

		FLOOD(Ours)	**C-SWM** [22]	**SCN** [29]	**SA** [25]
SIM	**ARI(%)↑**	**96.41 ± 1.60**	80.08 ± 1.60	-	29.05 ± 21.03
	KLME↓	**616.05 ± 26.45**	1403.77 ± 4.31	5258.00 ± 138.48	-
REAL	**ARI(%)↑**	**97.71 ± 2.99**	78.8 ± 5.94	-	17.74 ± 4.33
	KLME↓	**1082.19 ± 30.42**	1307.04 ± 38.33	4345.46 ± 31.99	-
CUBE	**ARI(%)↑**	86.47 ± 0.49	**94.30 ± 0.39**	-	66.98 ± 18.32
	KLME↓	181.24 ± 0.09	**177.43 ± 0.44**	3787.69 ± 449.54	-

Table 1: Quantitative comparison against published baselines. Best performance in **bold**. We report mean ± standard error for 3 seeds.

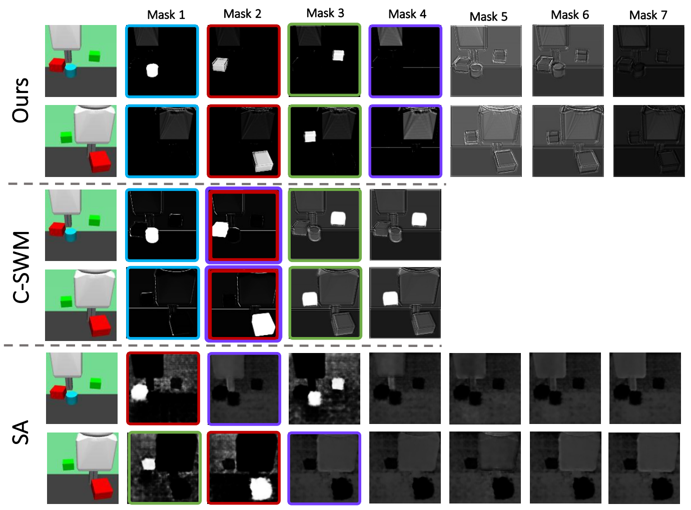

Figure 3: Examples of masks produced by FLOOD and baselines on **SIM**. The slot that best predicts the blue cylinder is outlined in blue, red cube in red, green cube in green, and robot arm in purple. For our method, each object is consistently represented by one of the slots. When there are more slots than the number of objects, our method outputs "empty" masks for the extra slots (no specific image regions are highlighted). For C-SWM where the robot is not represented in any slot, slot 2 minimizes the robot prediction error, even though it actually represents one of the objects, as shown. In this case, the same slot is the predictor for both the object and the robot. For SA, learned object representations are not disentangled, thus each slot represents multiple objects. In addition, the slot that binds to a certain object changes from image to image.

Datasets. We evaluate FLOOD on three video datasets: **CUBE**, **SIM** and **REAL**. The first two are simulated datasets, and the third is collected in a real-world robotic setup. We will publicly release all datasets and code. Sample frames from all datasets are shown in Fig 2. Details of these three datasets are in supplementary material.

Metrics. **ARI** [14, 30] compares predicted slot masks with ground truth object segmentation masks; higher indicates a better match. **KLME** uses slot vectors to measure whether slots consistently binds to certain objects (details in Supp); lower is better. Note that SA does not produce slot vectors, so we omit its KLME in our quantitative results. Likewise, SCN does not produce slot masks, so we omit its ARI.

Results. Table 1 summarizes results for our method and the baselines on all three datasets. Our method easily outperforms SA and SCN, and performs comparably with C-SWM, which has access to additional information about object actions at each step. In particular, C-SWM shines on **CUBE**, the simplest environment, where object-specific actions are discrete and cleanest.

Fig 3 show examples of masks produced by FLOOD and baselines for two images on **SIM** dataset. We label each mask with a colored box if this mask best predicts the 2D coordinates of a certain object over the validation set — this is also the slot-to-object matching used in computing the KLME score. More extensive results, including all slot masks produced by each method, are shown in Supp.

Bibliography

[1] Jimmy Ba, J. Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016.

[2] Pia Bideau, Rakesh R Menon, and Erik Learned-Miller. Moa-net: self-supervised motion segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[3] Pia Bideau, Aruni RoyChowdhury, Rakesh R Menon, and E. Learned-Miller. The best of both worlds: Combining cnns and geometric constraints for hierarchical motion segmentation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 508–517, 2018.

[4] Christopher P Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. 2019.

[5] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

[6] Mickaël Chen, Thierry Artières, and Ludovic Denoyer. Unsupervised object segmentation by redrawing. 2019.

[7] Jingchun Cheng, Yi-Hsuan Tsai, Shengjin Wang, and Ming-Hsuan Yang. Segflow: Joint learning for video object segmentation and optical flow. In *Proceedings of the IEEE international conference on computer vision*, pages 686–695, 2017.

[8] Minsu Cho, Suha Kwak, Cordelia Schmid, and Jean Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[9] Eric Crawford and Joelle Pineau. Spatially invariant unsupervised object detection with convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3412–3420, 2019.

[10] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[11] Martin Engelcke, Adam R Kosiorek, Oiwi Parker Jones, and Ingmar Posner. Genesis: Generative scene inference and sampling with object-centric latent representations. 2019.

[12] SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. 2016.

[13] Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pages 2424–2433. PMLR, 2019.

[14] L. Hubert and P. Arabie. Comparing partitions. *Journal of Classification*, 2:193–218, 1985.

[15] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[16] Eddy Ilg, N. Mayer, Tonmoy Saikia, Margret Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1647–1655, 2017.

[17] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 2117–2126. IEEE, 2017.

[18] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Unsupervised learning of object landmarks through conditional image generation. 2018.

[19] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Self-supervised learning of interpretable keypoints from unlabelled videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8787–8797, 2020.

[20] Margret Keuper, Siyu Tang, Bjoern Andres, Thomas Brox, and Bernt Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. volume 42, pages 140–153. IEEE, 2018.

[21] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, pages 2688–2697. PMLR, 2018.

[22] Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. 2019.

[23] Tejas D. Kulkarni, A. Gupta, Catalin Ionescu, Sebastian Borgeaud, M. Reynolds, Andrew Zisserman, and V. Mnih. Unsupervised learning of object keypoints for perception and control. In *NeurIPS*, 2019.

[24] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, et al. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. volume 5, pages 3838–3845. IEEE, 2020.

[25] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. 2020.

[26] Sindy Löwe, Klaus Greff, Rico Jonschkowski, Alexey Dosovitskiy, and Thomas Kipf. Learning object-centric video models by contrasting sets. 2020.

[27] N. Mayer, Eddy Ilg, Philip Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016.

[28] Matthias Minderer, Chen Sun, Ruben Villegas, Forrester Cole, Kevin Murphy, and Honglak Lee. Unsupervised learning of object structure and dynamics from videos. 2019.

[29] Evan Racah and Sarath Chandar. Slot contrastive networks: A contrastive approach for representing objects. 2020.

[30] W. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971.

[31] Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 1154–1160. IEEE, 1998.

[32] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[33] Pavel Tokmakov, Karteek Alahari, and Cordelia Schmid. Learning motion patterns in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3386–3394, 2017.

[34] Roberto Tron and René Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[35] Rishi Veerapaneni, John D. Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, J. Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. volume abs/1910.12827, 2019.

[36] Yair Weiss and Edward H Adelson. *Perceptually organized EM: A framework for motion segmentation that combines information about form and motion*. Media Laboratory, Massachusetts Institute of Technology, 1995.

[37] Christopher Xie, Yu Xiang, D. Fox, and Z. Harchaoui. Object discovery in videos as foreground motion clustering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9986–9995, 2019.

[38] Jingyu Yan and Marc Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *European conference on computer vision*, pages 94–106. Springer, 2006.

[39] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by motion grouping. *ArXiv*, abs/2104.07658, 2021.

[40] Gengshan Yang and Deva Ramanan. Learning to segment rigid motions from two frames. In *CVPR*, 2021.

[41] Yanchao Yang, Yutong Chen, and Stefano Soatto. Learning to manipulate individual objects in an image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6558–6567, 2020.

5 Supplementary Material

5.1 Network implementation details

In our experiments, the object extractor $\mathcal{F}$ is a four-layered convolutional neural network. The first three convolutional layers are followed by a relu and a batchnorm layer. The last convolutional layer is followed by a sigmoid, so that the predicted slot masks $\boldsymbol{m}(x)$ lie in $[0, 1]$. The object encoder $\mathcal{G}$ is a small multi-layer perceptron consisting of three layers, with a LayerNorm [1] between the second and the third MLP layer. We use the Adam optimizer and train this network on a single 2080Ti GPU with batch size of 1024. For both our method and C-SWM [22], we trained the models for 200 epochs for all three datasets.

5.2 Implementation details for datasets

- **CUBE** is a 3D blocks dataset, adopted from [22]. This is a grid-world environment with discrete actions applied to selected blocks, such as "move object 1 one step forward" and "move object 2 one step left"). We use the authors' code to collect a video dataset with a random action policy, using identical settings as in [22]. We train on 1000 training videos of length 100 frames each, and evaluate on 10,000 videos of length 10. All frames are 50x50 RGB images. The robot executes a new action in each frame. For C-SWM, object-specific actions are provided as a four-dimensional one-hot vector(if an action is applied to that object) or a vector of zeros for each object slot.
- **SIM** is collected in a custom new simulated robotic manipulation environment. We use MuJoCo [32] to render a tabletop with three simple object shapes (a green cube, a red cube, and a blue cylinder) and a Fetch robot. We collect 1000 training videos and 1000 testing videos, all of length 15 frames, involving 15 separate robot motions. In each episode, the robot randomly picks one object and a direction to push it to. All frames are 128 x 128 RGB images. For C-SWM, object-specific actions are provided as a two-dimensional vector(if an action is applied to that object) or a vector of zeros for each object slot.
- **REAL** is analogous to **SIM**, but collected on a real robot arm, and with more complex objects and dynamics. We use a WidowX 200 robot arm and three deformable toys. The videos involve the robot performing random end-effector motions in each frame, colliding frequently with the objects and displacing them. We collect 350 training videos, and evaluate on 50 videos, all of length 30. All frames are 128 x 128 RGB images. For C-SWM, object-specific actions are provided as a five-dimensional vector(if an action is applied to that object) or a vector of zeros for each object slot.

5.3 Evaluation metrics

- **ARI**: To evaluate the slot masks as object segmentation masks, we adopt a widely used metric for multi-object segmentation: Adjusted Random Index(ARI) [14, 30]. ARI measures the similarity of two groupings of pixels, and is invariant to the permutation of groupings. We compare the set of slot masks produced by slot encoder with ground truth object segmentation masks.
- **KLME**: While ARI evaluates slot masks, it does not directly evaluate the slot vectors. To do this, we define a new metric as follows. We train NK linear regression models f_{nk}, one to regress each of the K slot vectors to each of N ground-truth 2D object coordinates in the scene, corresponding to specific keypoints on N objects. For each object n, we select the slot k that best represents that object over a validation dataset. This is by calculating $\min_k \mathrm{MSE}_n(f_{nk})$ as a score that captures how well that keypoint is represented, where MSE_n is the linear regression mean squared error on held-out data. This corresponds to finding the closest-matching slot for each keypoint, and measuring how well they match. Averaging this score over all N keypoints produces our keypoint location matching error (KLME) metric for slot vectors.

5.4 Sample episodes from training datasets

We evaluate our method and corresponding baselines on three datasets: **SIM**, **REAL** and **CUBE**. Fig 4 shows examples of sample episodes from each dataset. In both **SIM** and **REAL**, there are diverse contacts dynamics between the robotics manipulator and the objects and heavy occlusions. Thus these two datasets are especially challenging for any methods to learn robust object-centric representations from.

		FLOOD(Ours)	dynamics modeling	temporal continuity	reconstruction
CUBE	**ARI(%)↑**	85.51	**95.09**	14.35	15.21
	KLME↓	181.24 ± 0.09	**177.43** ± **0.44**	460.25	428.46

Table 2: Quantitative comparison of flow-guided (ours) vs. other training objectives on **CUBE**

		FLOOD(Ours)	dynamics modeling	temporal continuity	reconstruction
SIM	**ARI(%)↑**	**93.56**	70.93	7.89	33.35
	KLME↓	**100.86**	**99.61**	455.04	396.82

Table 3: Quantitative comparison of flow-guided (ours) vs. other training objectives on **SIM**

5.5 Ablation experiments on training objectives

FLOOD inherits the object extractor and object encoder of C-SWM, which is also used in SCN, making it particularly well-suited for comparison. To enable a more controlled comparison of object discovery approaches with a fixed architecture, we create two ablations of our method that use an SCN-like "temporal continuity" objective, and another that uses an SA-like image "reconstruction" objective. Since C-SWM already shares the same architecture, it doubles up as the third ablation, representing a "dynamics modeling" objective. We report the results of this ablation on three environments in Table 2, 3, 4. Once again, we easily outperform the baselines,illustrating that the flow-guided objective is important to our method's overall performance.

5.6 Ablation experiments on optical flow algorithm

For all the main experiment results presented in this paper, we use FlowNet2.0 [16], which is trained on four large synthetic datasets of animated object videos [5, 10, 15, 27]. We perform an additional experiment to test how robust our method is with respect to different optical flow algorithm. We tested the performance of our method using an unsupervised optical flow algorithm, Dual-TV-L1[5][6], on the SIM dataset. The results are as follows: The KLME of FLOOD with Dual-TV-L1 is 952.9, compared to 616.05 of FLOOD with Flownet2. Although there is a performance gap between FLOOD with classical optical flow estimation approaches and learned Flownet2, this is still considerably better than the other baselines(1403.77 for C-SWM and 5258.00 for SCN, shown in Table 1). The ARI of FLOOD with Dual-TV-L1 is 93.10, compared to 96.41 of FLOOD with FlowNet2. The ARI of C-SWM is 80.08, while SA only has an ARI score of 29.05(shown in Table 1). This shows that our method is robust against flow noises. It's reasonable to assume that, with ground-truth flow, our method would achieve better performance. However the accuracy of flow would not largely affect the performance of our method.

5.7 Additional examples of slot masks

In Fig 3, Fig 5 and Fig 6, we show examples of masks produced by FLOOD and baselines for two images per dataset. We visualize all masks for each method. C-SWM requires the number of slots to be equal to the number of objects. For our method, we used 6 slots for **REAL**, and 7 slots for **SIM** and **CUBE**. We tuned this hyperparameter by searching over 4, 5, 6, and 7 slots for each dataset. For SA, we retained the slots hyperparameter setting (=7) from the authors' public code. We label each mask with a colored box if this mask best predicts the 2D coordinates of a certain object over the validation set — this is also the slot-to-object matching used in computing the KLME-OB score above.

		FLOOD(Ours)	dynamics modeling	temporal continuity	reconstruction
REAL	**ARI(%)↑**	**16.86**	15.97	0.011	5.56
	KLME↓	**1082.19** ± 30.42	1307.04 ± 38.33	5558.44	5307.13

Table 4: Quantitative comparison of flow-guided (ours) vs. other training objectives on **REAL**

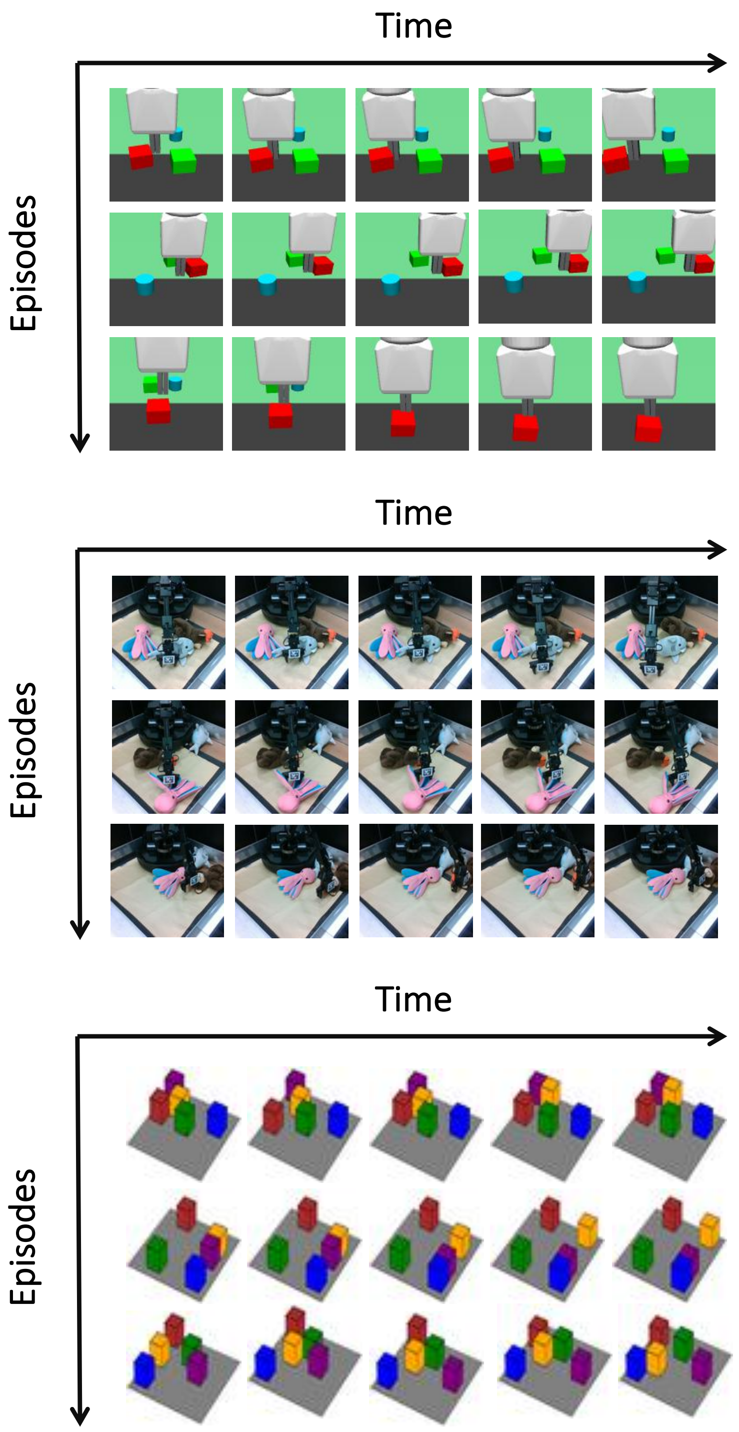

Figure 4: Examples of videos from **SIM**, **REAL** and **CUBE** datasets

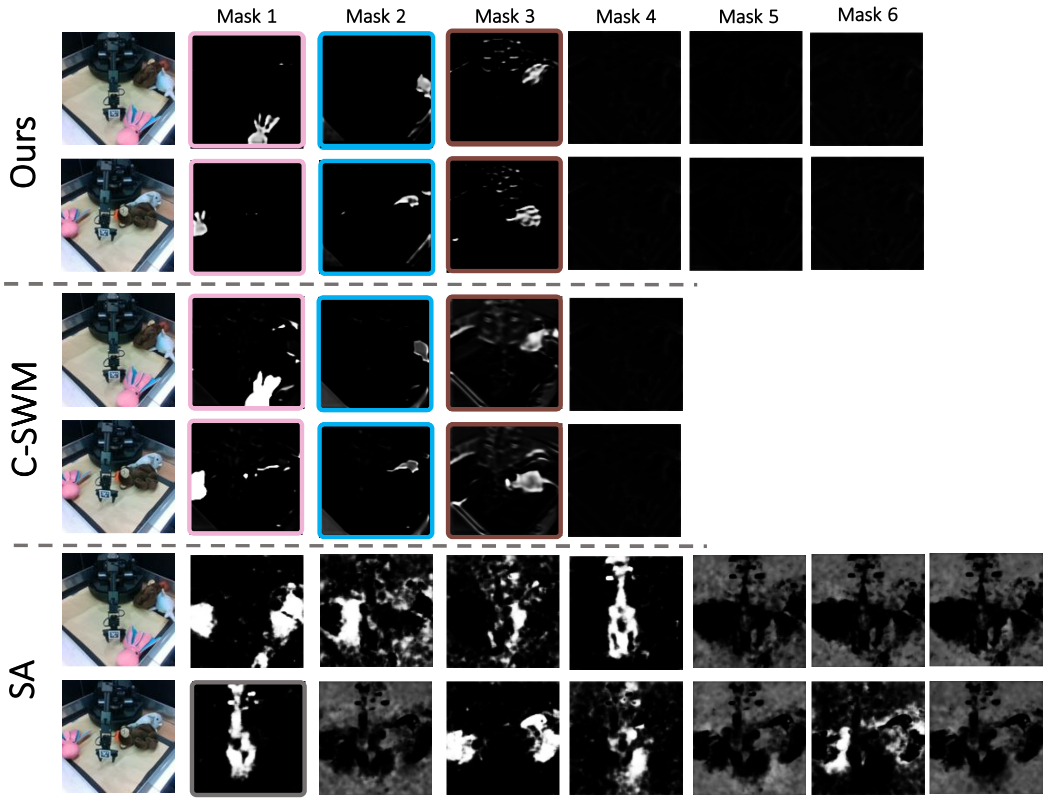

Figure 5: Examples of masks produced by FLOOD and baselines on **REAL**. The slot that best predicts the pink octopus is outlined in pink, blue whale in blue and brown bear in brown. For both our method and C-SWM, the robot is not represented by any of the slots. For SA, object representations learned are not disentangled, as each mask often represents several objects. In both our method and C-SWM, slots bind to the same object throughout the entire validation dataset. However, for SA, there is not a fix mapping from slots to objects.

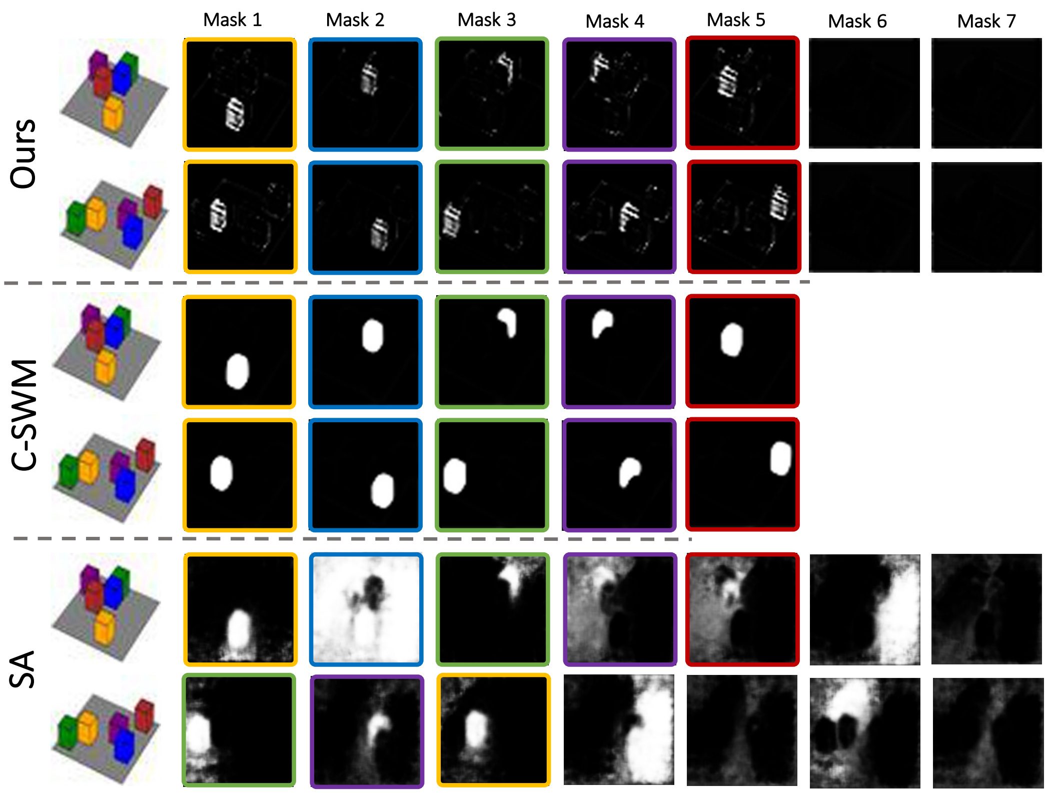

Figure 6: Examples of masks produced by FLOOD and baselines on **CUBE**. The slot that best predicts each colored cube is outlined in the same color corresponding to the cube. For both our method and C-SWM, all objects are consistently represented by some slot. Since our method predicts more slot masks than the number of objects available in the scene, slot masks 6 and 7 are empty masks. For SA, there is no consistent mapping from slots to objects.