# Object Representations Guided By Optical Flow

**Jianing Qian, Dinesh Jayaraman**
Department of Computer Science
Cranberry-Lemon University
Pittsburgh, PA 15213
hippo@cs.cranberry-lemon.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, through reconstruction and contrastive losses. 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. Indeed, human adults experience their visual world as consisting largely of overlapping objects at various distances that are stable across space and time. The origins of such an object-based experience of raw sensory inputs has occupied and intrigued cognitive scientists for over a century [23, 40–42, 49, 50].

Artificial intelligence researchers have also asked similar questions about how machines might parse image inputs into objects, often motivated by the need to represent high-resolution images compactly yet sufficiently comprehensively for various applications. State-of-the-art approaches for classic computer vision tasks such as object detection and segmentation focus on learning to perform these tasks with manual object annotations. However, many recent efforts 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 [26–28, 33].

Exciting progress in unsupervised object discovery has come through introducing object structure into unsupervised visual representation learning approaches. Specifically, unsupervised visual representations are commonly trained to reconstruct image inputs [11, 34], or separate distinct images [6, 7]. The best-performing object discovery approaches [3, 10, 13, 14, 16, 21, 22, 27, 29–31, 33, 37] optimize these same objectives, but constrain their neural network architectures in various ways to produce object-like representations such as segmentation masks or keypoints.

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

Figure 1: Objects move as coherent wholes, therefore motions in natural video sequences preserve the object structure of the images, modulo minor boundary and occlusion effects. We identify the following simple desideratum for good discovered object-structured image representations such as the segmentation masks depicted here: object representations for nearby images in a video stream must be related to each other through the same optical flow warps as the images themselves. [DJ: Should fix w forward flow?]

Consequently, much research has focused on designing network architectures to produce object-structured representations [10, 21, 22, 27, 30, 31], and corresponding inference mechanisms [13, 14, 16]. 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. In particular, we call back to the Gestalt tradition [50] that motion contains information critical for correctly grouping object pixels together. Indeed, a large and mature literature on motion segmentation approaches in computer vision [2, 24, 45, 51, 52] attests to the utility of tracking the image plane motions of pixels, i.e., "optical flow", for identifying moving objects in video — pixels that coherently move together belong together.

We show how these ideas extend to unsupervised neural object discovery. 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 FLOW (Flow-guided object discovery"). This idea is illustrated in Fig 1.

We evaluate FLOW 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 FLOW 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.

## 2 Related Prior Work

Video object segmentation from motion and appearance. As foreshadowed in Sec 1, our approach for flow-guided object discovery is related to motion segmentation methods [2, 24, 39, 44, 45, 51, 52] 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 [8, 20, 44, 48]. However, all these methods are designed for segmenting video and can not handle static images as trained FLOW models do. Specifically, FLOW trains unsupervised appearance-based static image object segmentation models by using flow-based cues in training videos as a supervision signal.

Unsupervised representations from motion. In the context of learning unsupervised representations, image region-level correspondences computed through object motion tracking [47] or by matching object-like regions [15] has been used before as a form of self-supervision. Compared to

FLOOD's goal of discovering objects and forming object-structured representations of full images, they instead aim to train unstructured descriptors of objects and object-like regions in images. Most related to our approach from this literature, Pathak et al. [36] use motion segmentation of moving objects from video as targets for a single-image segmentation model. Note that this task is not well-posed; moving objects cannot be identified from a single image. Rather than produce a good segmentation model, this self-supervised segmentation task is merely intended as a "pretext task" [36], i.e., through training on this task, the CNN must acquire good unstructured image representations useful for other computer vision tasks.

Neural object discovery. Nearly all neural object discovery approaches targeting general images or video are trained by reconstructing the input image [3, 13, 16, 21, 22, 27, 30, 33]. Some approaches use adversarial training: the idea is that object-structured representations can be modified and recomposed into valid images [5, 53]. Two recent approaches are trained through contrastive losses to distinguish between distinct images in a dataset [31, 37]. Among these, Racah and Chandar [37] use temporal continuity in video for object discovery, in a method called slot-contrastive networks (SCN). Like FLOOD, SCN uses temporally nearby frames as positives and distant frames as negatives, but it effectively treats nearby images as being identical since it does not account for pixel motions. Instead, as specified above, FLOOD models nearby frames as related by a warp specified by optical flow. We compare against SCN in Sec 4.

In robotics contexts, where motion is attributable to known robot actions, several models [25, 26, 46] use action-conditioned future prediction as a training objective for joint object discovery and environment dynamics modeling. The robot actions in these formulations help specify how the scenes in training videos evolve over time. FLOOD replaces robot actions with dense pixel correspondences from optical flow, which are more general — they specify how arbitrary videos evolve even with unknown actors and forces. Notably, our implementation of FLOOD uses the popular object slot encoder architecture of contrastive structured world models (C-SWM) [26], enabling a controlled comparison in Sec 4. With identical object encoders, and sans robot action knowledge, FLOOD still performs comparably or better than C-SWM even in robotics datasets with known actions.

## 3 Flow-guided object discovery

Task setup and motivation. 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. For example, given images of toys in varied configurations on a cluttered tabletop, we would like to automatically discover those toys. Discovery is typically expressed by parsing or spatially disentangling new images from that domain and assigning some or all pixels to various discovered objects and background.

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 [21, 27], object detection bounding boxes [3], segmentation masks [16, 26], and skeletons capturing connected keypoints [22]. Much research has focused on how to design innovative representations and infer them.

What are these representations useful for? Aside from having close correspondences with the objects in the scene, object-structured representations are intended to be compact, and to spatially disentangle the image into composable and coherently moving components. These properties are useful in downstream tasks such as robotics. For example, to model how visual scenes respond to robot actions, object-structured representations permit compact models in terms of how discovered object components interact with the robot and with each other [26–28, 33]

In the following subsections, we propose a new approach and training objective for neural object discovery. Prior approaches largely rely on image reconstruction, action-conditioned future prediction, or contrasting images in the training dataset, as we will discuss in Sec 2. Instead our approach, FLOOD, uniquely mines supervisory signals from motions occurring in a training dataset of videos for learning to disentangle objects.

Figure 2: The training setup for SLOOD (Sec 3.3), applied to object-structured slot representations (Sec 3.1).

### 3.1 Slot representation and encoder

Our approach is not tied to only 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 $m(x) = \{m^1(x), m^2(x), \ldots, m^K(x)\}$ 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 $z^k(x)$. $z(x)$ 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]

How are these slot masks and vectors generated from input images? We use the popular neural network architecture proposed in Kipf et al. [26]. An "object extractor" convolutional neural network $F$ with a sigmoid output layer produces $K$ convolutional maps constrained to lie in $[0,1]$. These represent the slot masks $m(x)$. Next, a small multi-layer perceptron $G$ 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 2 illustrate this procedure.

### 3.2 Optical flow, and how motions can help with object discovery

An important characteristic of objects, known for long to be particularly useful for object grouping, is that objects move as coherent wholes. This idea has been known for at least a century, and is captured in the pithy Gestalt [50] phrase: "pixels that move together group together."

Yet, to our knowledge, no existing approaches for neural object discovery use this information, including many that train on video sequences, such as [21, 26, 37]. We address this gap by proposing a novel approach to use the coherent motions of objects in training videos as training signals to guide object discovery.

Our key idea is illustrated in Fig 1. Objects move as coherent wholes, and the objects in two neighboring images in video are the same, only (possibly) slightly displaced. Therefore, our slots (generalizable to other object-structured representations) should satisfy the following desiderata:

- Slots should bind to the same set of objects in both frames.
- Slots should be sensitive to the small changes in positions of those objects. In particular, each slot mask $m^k$ should exactly track the displacement of the object that it is bound to.

[1] This is not true in Kipf et al. [26], due to the requirement of object-specific actions for each slot.

To frame these desiderata as a mathematical objective, we introduce the concept of optical flow. An optical flow field $h_{a \to b}$ defines a warp between images $a$ and $b$, such that $b = h_{a \to b}(a)$. In other words, $h_{a \to b}$ specifies a displacement for each pixel in $a$ such that the resulting image is equal to $b$. Optical flow computation is a well-studied problem in computer vision, and many mature solutions exist. We use FlowNet2.0 [19], which is trained on four large synthetic datasets of animated object videos [4, 12, 18, 32].

The optical flow field $h_{x \to x'}$ provides us with correspondences between pixels in temporally nearby frames $x$ and $x'$. 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.

This simple optical flow-based desideratum implicitly exploits the fact that objects in video move coherently, in ways that preserve their identity. To see this, observe that Eq (1) would not hold true for good slots $m^k$ and arbitrary warps $h$. For example, $h$ might move all pixels in $x$ by random displacements. In this case, the resulting image $x'$ would no longer plausibly retain the object identities present in $x$, and $m^k(x')$ cannot therefore be expected to preserve them either. Optical flow warps corresponding to frames in natural videos are thus special, in that they are object identity-preserving.

As a sidenote, in the above discussion, we have largely assumed that optical flow masks provide perfect pixel correspondences between the same real three-dimensional particle in nearby frames $x$ and $x'$. Optical flow methods, while they perform very well in our experiments, do of course only produce noisy correspondences. However, as our experimental results validate, this is not a major concern.

### 3.3 The FLOOD training objective

The desideratum in Eq (1) cannot directly be set up as a training objective, since it permits two degenerate solutions: $m^k(x) = 0$ 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 $z(x) = G(m(x))$.

Let $\overline{m(x')} \triangleq h_{x \to x'}(m(x))$ represent the prediction of the slot masks $m(x')$ obtained by flow-warping $m(x)$. Note that this is the LHS of Eq (1), combined over all slot indices $k$. Now, let $\overline{z(x')} = G(\overline{m(x')})$ represent its corresponding predicted slot vectors.

Then, the flow-guided object discovery (FLOOD) objective is:

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

where $d(\cdot, \cdot)$ is a distance function between slot vectors, $x^-$ and are negatives for $x$. Similar to prior works in unsupervised contrastive representation learning [6, 35], this may be interpreted as a softmax classification objective for distinguishing the positive pair (in the numerator) from negative pairs (in the denominator). 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 $z(x)$ and $z(x')$. Instead, the slots of $x$ are first warped through the optical flow warp to compute $\overline{z(x')}$, as motivated in Sec 3.2.

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 $z = G(m(x))$, computed by a small MLP $G$. The low capacity of $G$ means that it should be very easy to extract a small code $z$ 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 $m^k(x) = x$.

### 3.4 Implementation Details

In our experiments, the object extractor $F$ 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 mask $\text{mask}(x)$ lie in $[0, 1]$. The object encoder $G$ 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.

There are two main hyperparameters in our method: slot vector dimension and slot number. For all experiments with our method, we fix the slot number to six. We keep all the other training hyperparameters as well as slot vector dimension as the same from C-SWM.

## 4 Experiments

Our experiments aim to answer the following questions: (1) How well does FLOOD with slot representations perform against representative state-of-the-art approaches from various families of neural object discovery algorithms? (2) How does FLOOD compare against ablations that use the same architecture, but with different training objectives? (3) While most past work on neural discovery has been evaluated in simplistic settings, can FLOOD handle real-world images set in a plausible robotics application?

### 4.1 Baselines and Ablations

Throughout our experiments, we compare against three recent baselines, representing different types of object discovery objectives from the literature.

- Contrastive structured world models (C-SWM) [26] represents methods for joint object discovery and environment dynamics modeling trained through an action-conditioned future prediction objective. Compared to our method and other baselines, it requires extra information about object-specific actions at each instant.
- Slot attention (SA) [30] represents object discovery through image reconstruction, using a new attention-based architecture to encode images into slots. It produces slot masks, but does not have a direct analogue of slot vectors used in our approach.
- Finally, slot-contrastive networks (SCN) [37], discussed above in Sec 2, use temporal continuity for object discovery: like FLOOD, it trains representations that can distinguish far-away frame pairs from nearby frame pairs using a contrastive loss, but it does not use optical flow correspondences between nearby frames. SCN directly produces slot vectors without first generating slot masks.

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.

### 4.2 Datasets

We evaluate FLOOD on three video datasets:

- CUBES is a 3D blocks dataset, adopted from Kipf et al. [26]. 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 Kipf et al. [26]. 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.

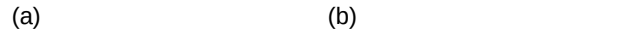


Figure 3: Sample images from the three datasets: (a) CUBES, (b) SIM, and (c) REAL.

- SIM is collected in a custom new simulated robotic manipulation environment. We use MuJoCo [43] 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.

We will publicly release all datasets and code. Sample frames from all datasets are shown in Fig 3.

### 4.3 Quantitative Evaluation Metrics

We use two evaluation metrics for our slot representations:

- ARI: To evaluate the slot masks as object segmentation masks, we adopt a widely used metric for multi-object segmentation: Adjusted Random Index(ARI) [17, 38]. 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 $\min_k \text{MSE}_n(f_{nk})$ as a score that captures how well that keypoint is represented, where $\text{MSE}_n$ 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 $N$ all keypoints produces our keypoint location matching error (KLME) metric for slot vectors.

Ground truth object segmentation masks and keypoint locations are readily available in the simulated environments. For REAL, we labeled segmentation masks for 90 test images, and 2d location of objects for 660 test images, to permit the measurement of ARI and KLME respectively.

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.

### 4.4 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 CUBES, the simplest environment, where object-specific actions are discrete and cleanest.

Table 1: Quantitative comparison against published baselines. Best performance in bold

| | | FLOD (Ours) | C-SWM [26] | SCN [37] | SA [30] |
|---|---|---|---|---|---|
| SIM | ARI(%) " | 93.56 | 70.93 | - | 50.07 |
| | KLME # | 100.86 | 99.61 | 286.18 | - |
| REAL | ARI(%) " | 16.86 | 15.97 | - | 10.20 |
| | KLME # | 786.48 | 830.34 | 806.73 | - |
| CUBES | ARI(%) " | 85.51 | 95.09 | - | 85.29 |
| | KLME # | 59.92 | 54.79 | 332.42 | - |

Figure 4: Examples of masks produced by FLOD and baselines on CUBES and SIM images. While C-SWM produces very crisp masks for all objects on CUBES, it interestingly fails to represent the robot ("object 4") in SIM, despite the object-speci c actions being easiest to specify for the robot.

Table 6 shows results of the objective function ablations of our approach mentioned above in Sec 4.1, with network architecture xed. These results are reported on SIM, which has clean annotations of object segmentations and keypoints for evaluation. Once again, we easily outperform the baselines, illustrating that the ow-guided objective is important to our method's overall performance.

Fig 4 and 5 show examples of masks produced by FLOD and baselines for two images per dataset. For each object in the scene, we show the mask that most frequently binds to it. Some methods do not successfully bind to all objects in the scene; for them, we do leave empty spaces representing missing slots. More extensive results, including all slot masks produced by each method, are shown in Supp.

## 5 Conclusions

In this paper, we design a new approach for unsupervised object discovery that is informed by optical ow correspondences between neighboring images in video. We show that this easily

Table 2: Quantitative comparison of ow-guided (ours) vs. other training objectives on SIM

| | | FLOD (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 |

Figure 5: Examples of masks produced by HOOD and baselines on REAL images. This is the hardest setting, and all methods produce less crisp masks. However, C-SWM and our approach can both successfully learn to represent the three objects in the image, and both are unable to learn the robot.

outperforms baselines that learn object-structured representations from general videos, and even performs comparably with an approach that learns from additional object-specific information.

However, our approach is dependent on optical flow providing reliable correspondences. While optical flow approaches are mature, there remain difficulties in producing reliable correspondences in the presence of clutter, occlusions, and lighting changes. We do not handle such noisy correspondences in our approach, and leave it for future work. Our own experimental settings did not prove to suffer from these issues, but they may arise in still more challenging settings.

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

[4] 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.

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

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In International conference on machine learning, pages 1597–1607. PMLR, 2020.

[7] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. 2020.

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

[9] 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.

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

[11] Jonas Dippel, Steffen Vogler, and Johannes Höhne. Towards ne-grained visual representations by combining contrastive learning with image reconstruction and attention-weighted pooling. 2021.

[12] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Haz rbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical ow with convolutional networks. IEEE International Conference on Computer Vision (ICCV), 2015. URL http://lmb.informatik.uni-freiburg.de/Publications/2015/DFIB15 .

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

[14] 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.

[15] Ruohan Gao, Dinesh Jayaraman, and Kristen Grauman. Object-centric representation learning from unlabeled videos. In Asian Conference on Computer Vision, pages 248–263. Springer, 2016.

[16] 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.

[17] L. Hubert and P. Arabie. Comparing partitions. Journal of Classi cation, 2:193–218, 1985.

[18] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical ow estimation with deep networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jul 2017. URL http://lmb.informatik.uni-freiburg.de/ /Publications/2017/IMKDB17 .

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

[20] 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.

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

[22] 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.

[23] Scott P Johnson. How infants learn about the visual world. volume 34, pages 1158–1184. Wiley Online Library, 2010.

[24] 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.

[25] 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.

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

[27] 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.

[28] 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.

[29] Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. Space: Unsupervised object-oriented scene representation via spatial attention and decomposition. 2020.

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

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

[32] 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.

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

[34] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. 2017.

[35] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.

[36] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2701–2710, 2017.

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

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

[39] 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.

[40] Elizabeth S Spelke. Principles of object perception. volume 14, pages 29–56. Elsevier, 1990.

[41] Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. volume 10, pages 89–96. Wiley Online Library, 2007.

[42] Elizabeth S Spelke, Karen Breinlinger, Janet Macomber, and Kristen Jacobson. Origins of knowledge. volume 99, page 605. American Psychological Association, 1992.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In Proceedings of the IEEE international conference on computer vision, pages 2794–2802, 2015.

[48] 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.

[49] Max Wertheimer. Experimentelle studien uber das sehen von bewegung. volume 61, 1912.

[50] Max Wertheimer. Laws of organization in perceptual forms (translated from the 1917 german version). Kegan Paul, Trench, Trubner & Company, 1938.

[51] 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.

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

[53] 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.

## Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default [TODO] to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a justification to your answer , either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section ??.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately re ect the paper's contributions and scope? [Yes] See abstract
   (b) Did you describe the limitations of your work? [Yes] we discussed the limiations of our method in conclusion. We will include additional analysis of limitations of our work in supplemental material
   (c) Did you discuss any potential negative societal impacts of your work? [No] We do not focus on speci c applications in this work. Like any such basic research approaches, while applications might have the potential for negative impacts, it is dif cult to meaningfully foresee them for this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We cited the works that we took our dataset from. We will include our code and dataset in the supplemental material
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In section 3.4 and 4.2
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] In section 3.4
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] In section 4.2
   (b) Did you mention the license of the assets? [Yes] We cite the paper that we took the dataset from in section 4.2
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identi able information or offensive content? [Yes] No they don't contain personally identi able information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

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

| | | FLOOD (Ours) | C-SWM [26] | SCN [37] | SA [30] |
|---|---|---|---|---|---|
| SIM | ARI(%) " | 96.41 1.60 | 80.08 1.60 | - | 29.05 21.03 |
| | KLME # | 97.98 3.01 | 99.8 0.19 | 309.15 22.97 | - |
| | KLME-OB # | 616.05 26.45 | 1403.77 4.31 | 5258.00 138.48 | - |
| REAL | ARI(%) " | 20.45 2.99 | 10.04 5.94 | - | 14.53 4.33 |
| | KLME # | 814.93 34.03 | 857.24 26.9 | 825.50 13.8 | - |
| | KLME-OB # | 1082.19 30.42 | 1307.04 38.33 | 4345.46 31.99 | - |
| CUBES | ARI(%) " | 86.47 0.49 | 94.30 0.39 | - | 66.98 18.32 |
| | KLME # | 60.23 0.16 | 54.80 0.007 | 368.62 36.2 | - |
| | KLME-OB # | 181.24 0.09 | 177.43 0.44 | 3787.69 449.54 | - |

# 6 Supplementary Material

## 6.1 Sample episodes from training datasets

We evaluate our method and corresponding baselines on three datasets: SIM, REAL and CUBES. Fig 6 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.

## 6.2 Error bars for quantitative results

Table 4 reproduces the quantitative results from Table 1, with error bars from 3 random seeds. Recall that SA does not produce slot vectors, so the KLME is inapplicable for this baseline. Further, SCN does not produce slot masks, so the ARI is inapplicable.

## 6.3 Measuring object binding

For these above results, the KLME was computed by training predictors from each slot to each object. Then for each image, and each object, we would compute the minimum over slots of the prediction error for that object. This permits the possibility of different slots binding to the same object in each image. For example, for predicting object 1, slot 1 might be used as the predictor in image 1, and slot 2 might be used in image 2.

Instead, we now present a metric that better reflects whether slots consistently bind to the same objects. To do this, for each object $o$, we select the slot $k$ that best represents that object over a validation dataset. We call this metric the KLME-OB (for "object binding"). Observe that FLOOD enjoys a large advantage on this metric on both SIM and REAL: we observe that this is because C-SWM does not bind to objects as consistently as FLOOD. This is especially true in SIM where FLOOD binds to the robot, but C-SWM does not. The KLME metric does not fully penalize C-SWM for this, since it uses an object slot close to the robot (a different one in each image) to predict the robot position instead.

## 6.4 Additional examples of slot masks

In Fig 7, Fig 8 and Fig 9, 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 CUBES. 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.

Figure 6: Examples of videos from SIM, REAL and CUBES datasets

Figure 7: 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.

Table 4: Quantitative comparison against published baselines. Best performance in bold. We report mean standard error for 3 seeds.

| | | FLOOD (Ours) | C-SWM [26] | SCN [37] | SA [30] |
|---|---|---|---|---|---|
| SIM | ARI(%) " | 96.41 1.60 | 80.08 1.60 | - | 29.05 21.03 |
| | KLME-OB # | 616.05 26.45 | 1403.77 4.31 | 5258.00 138.48 | - |
| REAL | ARI(%) " | 20.45 2.99 | 10.04 5.94 | - | 14.53 4.33 |
| | KLME-OB # | 1082.19 30.42 | 1307.04 38.33 | 4345.46 31.99 | - |
| CUBES | ARI(%) " | 86.47 0.49 | 94.30 0.39 | - | 66.98 18.32 |
| | KLME-OB # | 181.24 0.09 | 177.43 0.44 | 3787.69 449.54 | - |

Table 5: Quantitative comparison of ow-guided (ours) vs. other training objectives on REAL

| | | FLOOD (Ours) | dynamics modeling | temporal continuity | reconstruction |
|---|---|---|---|---|---|
| REAL | ARI(%) " 16.86 | 15.97 | 0.011 | | 5.56 |
| | KLME-OB # | 1082.19 30.42 | 1307.04 38.33 | 5558.44 | 5307.13 |

Figure 8: 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 fixed mapping from slots to objects.

Table 6: Quantitative comparison of flow-guided (ours) vs. other training objectives on CUBES

| | | FLOOD (Ours) | dynamics modeling | temporal continuity | reconstruction |
|---|---|---|---|---|---|
| SIM | ARI(%) ↑ | 85.51 | 95.09 | 14.35 | 15.21 |
| | KLME-OB # | 181.24 0.09 | 177.43 0.44 | 460.25 | 428.46 |

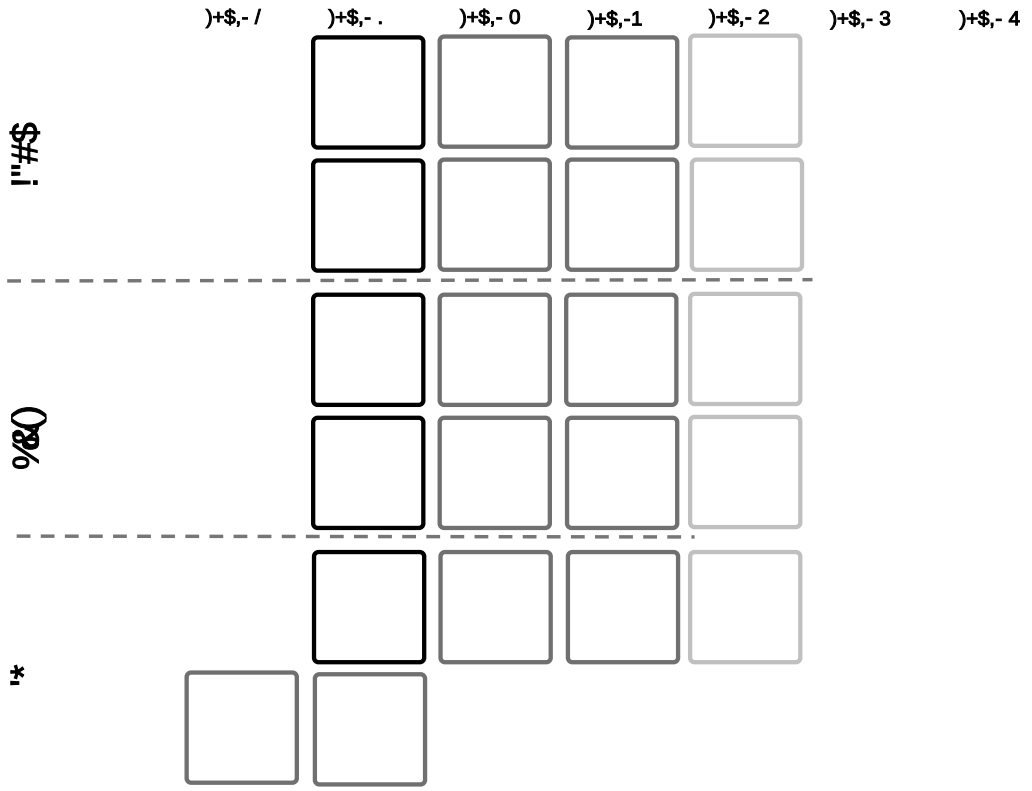


Figure 9: Examples of masks produced by FLOOD and baselines on **CUBES**. The slot that best predicts each colored cube is outlined in the same color corresponding to the cube. For both our method and C-SWM, all objects are consistently represented by some slot. Since our method predicts more slot masks 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.