2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Visuomotor Control in Multi-Object Scenes Using Object-Aware Representations

Negin Heravi*, Ayzaan Wahid, Corey Lynch, Pete Florence, Travis Armstrong,
Jonathan Tompson, Pierre Sermanet, Jeannette Bohg, Debidatta Dwibedi

***Abstract*— Perceptual understanding of the scene and the relationship between its different components is important for successful completion of robotic tasks. Representation learning has been shown to be a powerful technique for this, but most of the current methodologies learn task specific representations that do not necessarily transfer well to other tasks. Furthermore, representations learned by supervised methods require large, labeled datasets for each task that are expensive to collect in the real-world. Using self-supervised learning to obtain representations from unlabeled data can mitigate this problem. However, current self-supervised representation learning methods are mostly object agnostic, and we demonstrate that the resulting representations are insufficient for general purpose robotics tasks as they fail to capture the complexity of scenes with many components. In this paper, we show the effectiveness of using object-aware representation learning techniques for robotic tasks. Our self-supervised representations are learned by observing the agent freely interacting with different parts of the environment and are queried in two different settings: (i) policy learning and (ii) object location prediction. We show that our model learns control policies in a sample-efficient manner and outperforms state-of-the-art object agnostic techniques as well as methods trained on raw RGB images. Our results show a 20% increase in performance in low data regimes (1000 trajectories) in policy training using implicit behavioral cloning (IBC). Furthermore, our method outperforms the baselines for the task of object localization in multi-object scenes. Further qualitative results are available at https://sites.google.com/view/slots4robots.**

I. Introduction

General purpose robots need to encode information about their environment and themselves in a way that is not task specific and can be easily transferred to new situations and tasks. Current techniques for learning representations of the robot's environment are often trained in a supervised manner on task-specific datasets. However, collecting and labeling a new dataset per task is not scalable. Self-supervised learning methods that aim to learn representations from unlabeled data hold the potential to help robots learn about their environments without manual annotation.

Prior work in robotics has shown that performance and sample-efficiency of policy learning improves with self-supervised scene representations. These methods use compact representations in form of a global embedding [1], sparse key-point coordinates [2, 3, 4, 5], or other object embeddings [6] as input to policy learning modules. In this work, we explore a class of self-supervised models called Slot Attention [7] for representation learning in a robotic setup and differ from these prior works by learning the policy directly on dense per-pixel features and object masks produced by the Slot Attention model. This is particularly important in the context of multi-object manipulation as the representations need to encode the location of multiple objects in the scene along with the end-effector. Additionally, learning slot based representations does not require multi-view cameras [1, 4, 5] or canonical images of objects [3, 6].

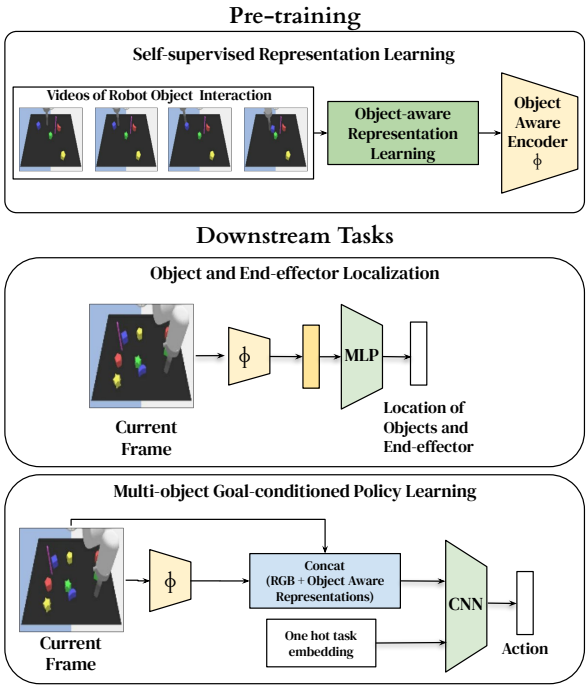

Fig. 1: We train a perception encoder using object-aware self-supervised learning. Then, we freeze the encoder weights and demonstrate this encoder is suitable for downstream tasks such as object localization and multi-object goal-conditioned policy learning.

Slot Attention models use sequential attention based mechanisms to group low level features in a scene where each group falls into a slot bin [7]. The authors show that the model can segment objects in an unsupervised manner. Inspired by this architecture, we propose to use these models, that can learn the extent of multiple objects in a scene, to learn representations that can be used for a variety of downstream robotic tasks. Our hypothesis is that the abstracted information using Slot Attention can improve data sample efficiency and performance in downstream training since it is object-aware making it suitable for extracting information in multi-object scenes. We test this hypothesis in the tasks of object localization and multi-object goal-conditioned policy

* Work done as an intern at Google. N. Heravi was also with Stanford University during this work. A. Wahid, C. Lynch, P. Florence, T. Armstrong, J. Tompson, P. Sermanet, and D. Dwibedi are with Robotics at Google. J. Bohg is with Stanford University. nheravi@alumni.stanford.edu.
Toyota Research Institute ("TRI") provided funds to assist the author with their research, but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

learning discussed in detail in the next sections. Our model is trained in multiple stages. First, we train Slot Attention in the object discovery mode in a self-supervised manner. Then, we freeze the weights and use these learned representations to train different small downstream networks for each task. We train our models using data of a robot interacting with blocks of different shapes and colors placed on a table in a simulation environment. Using this setup, we study the gain in performance by using these representations in different data regimes. We observe that the mask and features learned by our model boost performance on both tasks of object localization and behavioral cloning. Particularly in the low data regime, our features result in a 20% improvement in task completion success rate.

In summary, the contributions of our work are as follows:

1) We show that our Slot Attention inspired representations encode location and properties of all objects in the scene while object agnostic self-supervised methods such as MoCo [8] only focus on a few objects.
2) We show that our method needs fewer supervised action labels to learn policies (i.e. it is more sample efficient) and learns policies that have faster training convergence than alternative state-of-the-art methods.

II. Related Work

Self-supervised Representation Learning. Self-supervised learning has been immensely successful in training large models without any labels across different visual modalities: images [9, 8], optical flow [10] and videos [11, 12]. As these methods do not rely on manual labeling, they are well-suited for learning features from videos of robots interacting with objects. Self-supervised losses tend to be either a contrastive or a reconstruction loss. In this work, we compare features learned with these two different losses as input to a policy-learning network. In particular, we focus on a class of models that use the reconstruction loss but also the notion of slots/objects that induces grouping of similar pixels into slots in a self-supervised way [7].

Self-supervised Learning for Robotics. Several previous papers have explored self-supervised learning methods for robotics. [2] learned features as input to policies using a spatial-softmax bottleneck layer and a reconstruction loss. [1, 13] proposed a self-supervised approach for learning embeddings based on metric learning using videos from multiple cameras. These embeddings were shown to be useful in reward calculation or as input for reinforcement learning methods. [14] also used self-supervised embeddings to improve grasping. [15] used self-supervised learning of multimodal representations for contact-rich manipulation tasks. [16] proposed a self-supervised approach to improve 6D pose estimation of objects by using a robot that interacts with objects. [17] explored how features from self-supervised learning methods can be used as a reward for training unseen reinforcement learning agents.

Object-centric Representations for Robotics. Various types of object-centric representations have been explored for use in policies in robotics, and although sometimes these use supervised learning [18, 19, 20], often these may be self-supervised [14] as well. In this aspect, our work is similar in spirit to works which use self-supervision to acquire embeddings interpreted as keypoints either learned through autoencoding bottlenecks [2, 3], or multi-view consistency [21, 4, 5]. In contrast to these papers however, we directly use dense features (the same resolution as the input image) rather than sparsifying the representation down to keypoints.

Another closely related work is APEX [22] which shows how segmentation masks can improve performance when using a heuristic policy. We differ from their work in using the Slot Attention algorithm for representation learning and directly using the learned masks for policy learning. We find that these features are crucial for task success when the number of objects in the scene increases.

Imitation Learning. Past work [4] has explored the use of self-supervised features to improve the performance of behavior cloning models. In this work, we also investigate the effectiveness of self-supervised features but in the context of a modern behavior cloning algorithm, Implicit Behavior Cloning (IBC) [23], which was shown to be better than standard behavior cloning on a number of robotic tasks.

III. Approach

Our overall framework learns object aware representations from unlabeled videos using Slot Attention [7]. We then freeze the weights of the representation architecture, and use features from this model for downstream robotic tasks of object and end-effector localization (Section IV-C) as well as policy learning (Section IV-E). We show an overview of our approach in Figure 1.

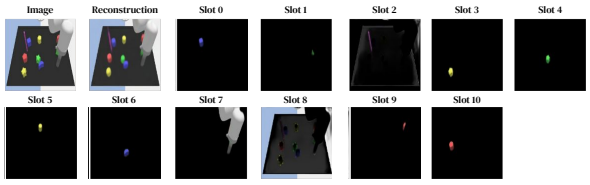

Fig. 2: Qualitative example of masks learned by Slot Attention. Slot Attention is able to localize objects and the end-effector by observing interaction videos without any mask-level supervision.

As our representation method, we use an image encoder ϕ that takes an RGB image I as input, and outputs an embedded feature representation $\phi(I)$. Given an input batch of N images $\{I_1, ..., I_N\}$, we train ϕ using a variation of the Slot Attention network [7] which consists of a convolution-based encoding architecture ϕ followed by an iterative attention mechanism [24]. This architecture is designed such that it groups the features of an image I_i into K slots where K is a hyperparameter. To achieve this, it uses an iterative dot-product based attention mechanism [24] that binds image features (input keys) and slots (queries). The attention coefficients are normalized over the K slots using a Softmax operation and are used to update the slots in each iteration using a *Gated Recurrent Unit* (GRU). This normalization over slots makes them compete with each other and each specialize in explaining a different component in the image. Given these slots, an upsampling convolutional decoder spatially broadcasts and reconstructs each slot separately into image slots R_{ik} as well as the corresponding spatial alpha masks M_{ik}. The masks are then used as weights to sum the reconstruction slots into a single combined reconstructed image for input I_i. This

results in self-supervised decomposition of low-level image features into abstract groups. We train this network in a self-supervised manner using the L2 pixel-wise difference of the reconstructed and the original image:

$$L = \frac{\sum_{i=1}^{N}(I_i - \sum_{k=1}^{K} M_{ik} \times R_{ik})^2}{N} \quad (1)$$

We modified the Slot Attention architecture in two ways. First, like [25], we initialize the slots to be learnable fixed vectors instead of samples from a learned Gaussian distribution. This change mitigated slot swapping between objects as the noise in the Gaussian setting could lead to permutations. Second, we use convolutions followed by upsampling instead of transposed convolutions to prevent checkerboard artifacts [26]. Please refer to [7] for more training details of the Slot Attention algorithm. Figure 2 shows qualitative examples of the performance of training this model on our robotic dataset.

After training this network, we freeze the weights and use the output of the convolution-based encoder $\phi(I)$ as representation in our policy learning. The pre-trained frozen Slot Attention models used in our experiments were trained for $500k$ steps with a batch size of 8 on images of size 160 by 320 and $K = 16$ slots with random seed initialization unless otherwise noted. For the experiments where the fraction of the data available varies, the slot representations are also only trained on the selected fraction of data. For the localization task, we use the dense masks $(M_1, .., M_K)$ as input to the downstream network. These masks encode the location of objects in pixel space in the form of slots.

IV. Experiments and Discussion

The goal of our experiments is to evaluate whether self-supervised object aware representations learned using Slot Attention provide performance gain for robotic tasks. To quantify this, we compare various representation learning techniques for tasks of object localization, multi-object goal-conditioned policy learning, and action prediction.

A. Baselines

We compare our method with the following baselines:

MoCo [8]. We train an encoder using a contrastive loss in a self-supervised manner as outlined in [27]. The output of the encoder is spatially averaged to produce an embedding. MoCo uses this contrastive loss to learn invariances of an image going through various data augmentations. In the context of this loss, positives are provided by using the embedding from a momentum encoder with the augmented version of the same image as input and negatives are sampled from a queue that keeps past embeddings in memory. We use this method as we want to compare the performance of contrastive losses and reconstruction losses (like that used in Slot Attention) for robotic tasks. We use an embedding size of 128, queue size of 16384, softmax temperature of 0.1 and batch size of 16 to train the MoCo encoder.

Autoencoder. We train an encoder-decoder architecture using the reconstruction loss [28]. This method has the same loss as the slot-encoder architecture, but it does not use any notion of slots/objects. The objective of this baseline comparison is isolating the importance of the Slot Attention module and the reconstruction loss for learning representations.

B. Environment

We use a robotic environment implemented in PyBullet [29]. In our setup, a robot arm is attached to a fixed base such that it can manipulate objects in front of it on a table. A cylinder is attached to the end of the arm which serves as the end-effector to push around blocks of different shapes and colors on a 2D plane (similar to [30]). We use this environment for collecting the data for all the simulation experiments in the following sections.

C. Object Localization: Task and Metrics

In this experiment, we investigate if the Slot Attention architecture can encode information about *all* the objects present in a scene. This property is important for learning policies on datasets with multiple objects and for tasks that depend on full scene information such as object localization. We evaluate the representations on the object localization task in a simulated environment which provides ground truth object locations. We only use this ground truth during downstream task training not for representation learning.

We learn representations from a dataset of demonstrations collected in simulation environment described in Section IV-B. Then, we freeze the weights of the representation network, and train an MLP to predict the location of the center of each block in robot coordinates. Please note that these locations are predicted in the robot coordinate base as ground truth is available in this space and not in image space. As a result, the network is learning hand-eye-coordination on top of finding the location of each block in the image. The MLP used for this task consists of two fully connected layers of size 256 and outputs the 2D location of all the blocks and the end-effector on the table's plane. For Slot Attention in this task, the input to our downstream MLP is the center of mass of the predicted slot masks. For our MoCo baseline, we use the output of the image encoder as input. For our Autoencoder baseline, as the MLP decoder needs an input vector, we use global average pooling layer on top of the output of the encoder CNN and use that as the input.

As an interpretable evaluation metric, we use *Probability of Correct Keypoint* (PCK) [31] which captures the percentage of times an object location is predicted correctly. This metric considers an object to be correctly localized if the predicted coordinates are within a given threshold of the ground truth. This is a commonly used method in computer vision research to evaluate localization of human and object keypoints [31]. We chose a PCK threshold value of 0.1 of the length of the table (about 5 cm).

We test the performance of our method with 1, 4, and 8 blocks. For the Slot Attention models, we use 7, 11, and 11 slots for the 1, 4, and 8 blocks respectively. The number of slots were chosen based on their performance on the reconstruction loss when pretraining the model independent of the downstream task. We train models with a dataset of 160k trajectories with image size of 256×256 for 150k steps each with batch size of 16 on one V100 GPU. We use the checkpoint with the lowest loss during representation learning for downstream training of object localization.

D. Object Localization: Results

Slot Attention outperformed the baselines in all the object localization experiments specially in multi-object cases as

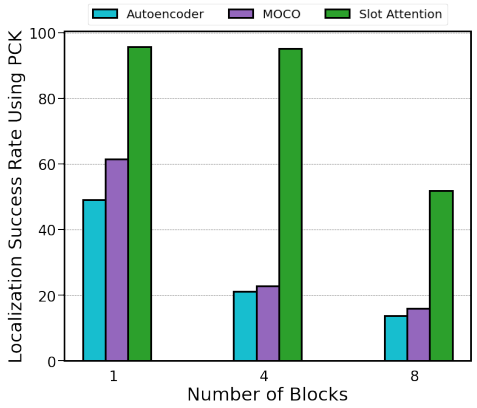

Fig. 3: Performance comparison on object localization over number of blocks present in the scene using *Probability of Correct Keypoint* (PCK) as evaluation metric. Higher percentage indicates better performance. The baselines are able to learn the location of one object in the scene (the end-effector) resulting in a low performance on average that decreases as the number of blocks increases. Slot Attention is able to localize multiple objects but sometimes struggles with objects of same color but different shapes in the 8 block scenario.

TABLE I: Performance of different self-supervised representations on downstream task of object location prediction (1 block case) using *Probability of Correct Keypoint* (PCK) as metric. Higher percentage indicates better performance.

	Object		
Input to object localizer	**Mean**	**End Effector**	**Blue Cube**
MoCo	61.5	83.1	39.9
Autoencoder	49.0	82.5	15.5
Slot Attention	**95.8**	**97.2**	**94.4**

seen in Tables I-III and Figure 3. We further made the following observations:

Baselines struggle with localizing multiple objects. Table I and II show the performance of our method on the cases of 1 and 4 blocks respectively, and Table III shows the performance on the 8 block scenario. We observe that while MoCo and Autoencoder are able to learn to predict object location more accurately on a dataset with a single object, they struggle to encode object locations in multi-object scenes. In Figure 3, we also show how the average object localization performance of different representations varies as the number of objects in the scene changes.

Slot Attention struggles with localizing objects with similar color and shape. While outperforming the baselines, we observe that the Slot Attention model finds it difficult to localize blocks of the same color but with fine-grained differences in their shape. In Table III, we show the Slot Attention model gives poor localization performance for the two yellow and the two green objects that have similar shapes due to slot assignment swapping between similar looking blocks. This is due to Slot Attention being trained using a pixel-wise image reconstruction loss, and hence, struggling to differentiate subtle differences in shapes when the color is the same. Subtle shape differences only contribute to a small number of pixel differences resulting in slight changes in the loss. However, Slot Attention still outperforms the other methods for this task.

TABLE II: Performance of different self-supervised representations on the task of object location prediction (4 blocks case) using *Probability of Correct Keypoint* (PCK) as metric. Higher percentage indicates better performance.

	Object					
Input to object localizer	**Mean**	**End Eff.**	**Red Moon**	**Blue Cube**	**Green Star**	**Yellow Pent.**
MoCo	22.9	66.3	15.2	14.7	10.3	8.2
Autoencoder	21.2	57.6	12.4	11.1	11.8	13.2
Slot Attention	**95.1**	**94.7**	**94.6**	**96.4**	**94.5**	**95.2**

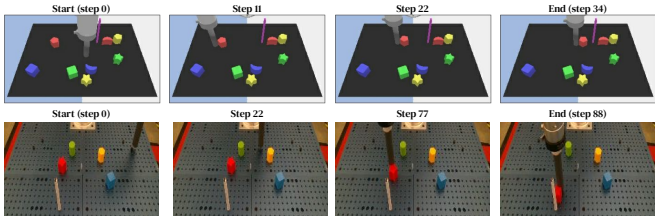

Fig. 4: Example demonstrations: moving the red pentagon to pole in sim (top) and moving the red star to the pole in real (bottom).

E. Multi-object Goal-conditioned Policy Learning: Task and Metrics

In this experiment, we compare different representation learning techniques by studying their effectiveness as inputs to a policy learning method. We only consider the imitation learning setup and learn a policy from a dataset of demonstrations provided by experts. The task for these experiments is to manipulate one of 8 blocks (i.e. the target block) on a plane to the target location shown by a purple rod. When the block is within 0.05 units of the rod, the episode is considered a success. If the robot fails to move the target block to the rod in 200 steps, the episode is considered a failure. In Figure 4, we show an example of the demonstration of the task in the environment described in IV-B. We use Implicit Behavior Cloning [23] to learn policies. During evaluation, we run the policy for 200 different initial configurations and measure the number of times the policy was able to successfully move the required block to the target location within the tolerated distance. We run policy training with 4 random seeds and report the mean and standard deviation of the success rates over the 4 runs. We present the results in Table IV. To compare different methods, we keep the policy learning method the same while we vary the input representations between RGB, RGB + Ground Truth (GT) Segmentation, Slot Attention, RGB+Slot Attention, Autoencoder, RGB+Autoencoder. To fairly compare between the different representation learning techniques we take the penultimate layer of the CNN encoder (before the spatial average pooling) and resize it to match the input RGB image. We optionally concatenate these features to the RGB image which is provided as input to the policy learning network. We also experimented with using MoCo features as input to IBC but were not able to train it to convergence. Remember that MoCo is trained to minimize a contrastive loss, and to succeed at minimizing this loss, the final representation does not need to capture information about all the objects in the scene. MoCo's loss can be minimized by focusing on objects that move more often than not, like the robot arm. The lack of

TABLE III: Performance of different self-supervised representations on downstream task of object location prediction (8 blocks case) using *Probability of Correct Keypoint* (PCK) as metric. Higher percentage indicates better performance.

	Object									
Input to object localizer	**Red Moon**	**Blue Cube**	**Green Star**	**Yellow Pentagon**	**Red Pentagon**	**Blue Moon**	**Green Cube**	**Yellow Star**	**End Effector**	**Mean**
MoCo	9.9	13.7	10.5	7.4	11.9	13.1	9.8	9.1	58.4	16.0
Autoencoder	9.1	14.4	7.3	9.9	12.0	13.7	8.5	8.3	41.0	13.8
Slot Attention	**79.8**	**79.4**	**20.2**	**18.4**	**86.9**	**83.0**	**23.0**	**15.6**	**60.1**	**51.8**

TABLE IV: Performance comparison on policy learning using the rate of successful task completion as metric. Higher success rate indicates better performance. Bold values show the method with maximum performance without access to ground truth information. The method with access to ground truth segmentation provides an upper bound for this task (oracle).

No. of training episodes (→)	**1000**		**2000**		**3000**		**10000**	
Input to Policy (↓)	**Mean**	**Std. Dev.**	**Mean**	**Std. Dev.**	**Mean**	**Std. Dev.**	**Mean**	**Std. Dev.**
RGB	36.9	5.8	78.9	8.4	88.8	3.2	95.1	2.8
RGB+Groundtruth Segmentation (oracle)	78.5	5.0	93.5	1.3	94.8	1.3	95.4	0.8
Autoencoder	46.0	3.2	61.3	9.8	83.5	3.8	85.8	6.1
RGB+Autoencoder	49.0	13.1	76.9	5.3	66.5	32.3	92.6	1.9
Slot Attention	**57.1**	1.5	86.0	1.2	**92.4**	1.9	**95.0**	1.7
RGB + Slot Attention	53.3	9.9	**87.8**	4.6	92.6	2.6	95.0	1.3

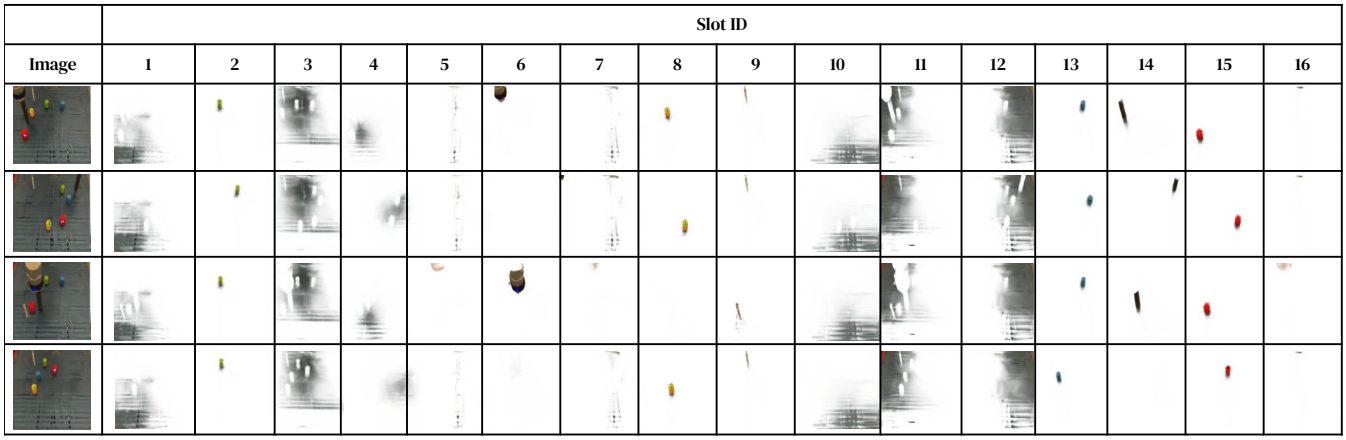

Fig. 5: Qualitative example of masks learned by Slot Attention on real-world data. Slot Attention is able to discover the pixels corresponding to the blocks (slots 2,8,13,15), the robot arm (slot 6), the end-effector (slot 14), and the pole (slot 9) without any labels.

convergence we observed for MoCo applied to IBC is likely due to MoCo features not reliably localizing the purple rod which is needed to solve the task. We refer the reader to [23] for additional details on the IBC training.

F. Multi-object Goal-conditioned Policy Learning: Results

We make the following observations (Figure 6, Table IV):
Better perception inputs lead to more sample efficient policies. As shown in Table IV, by using the GT semantic segmentation for all the blocks as input, the policy learning method can learn high-performing policies with over 90 percent success rate with few samples (2000 episodes). However, the policy with RGB needs somewhere between 3000 and 10000 episodes to achieve the same performance. This motivates us to look for better input representations than raw RGB for IBC. In the following experiments, we used a Slot Attention model trained with 16 bins since it had the lowest evaluation reconstruction loss during Slot Attention training. Both models were trained to convergence.
Slot Attention provides a performance boost in low data regimes. We observe that Slot Attention models provide a boost in performance in the success rate of task completion over using raw RGB as input. Slot Attention models are object aware and by using this prior, we are able to learn representations from demonstration video datasets that can result in performance improvement without collecting object bounding boxes or segmentation masks from humans. We also note that the performance gain of using Slot Attention over baselines decreases as the number of samples available for learning the policy increases as shown in Figure 6.
Slot Attention performs better than Autoencoder. We find that slot has better performance than autoencoder which has the same loss as the Slot Attention model but not the object/slot prior in its architecture. This shows that the prior of objects/slots is important for the performance gains.

G. Ablation: effect of number of slot bins

The number of slots is an important design choice in the Slot Attention architecture. This is a hyper-parameter that can be set by looking at the Slot reconstruction loss during the representation pretraining as well as evaluating on a validation set for downstream tasks. Figure 7 shows the

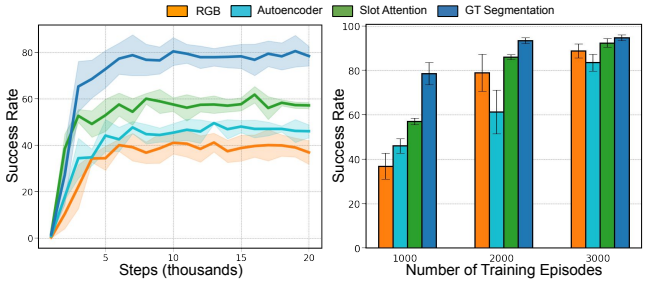

Fig. 6: (Left) Validation performance of different methods during IBC training in low data regime (1000 episodes). It can be observed that using Slot Attention leads to a 20% performance increase. As an upper bound, using ground truth segmentation masks resulted in about 40% performance improvement. Solid lines show the mean across 4 seeds and the shaded area indicates 1 standard deviation from each side. (Right) Performance comparison on policy learning over episodes in training data. Slot Attention based representations provide a performance boost in the low data regimes.

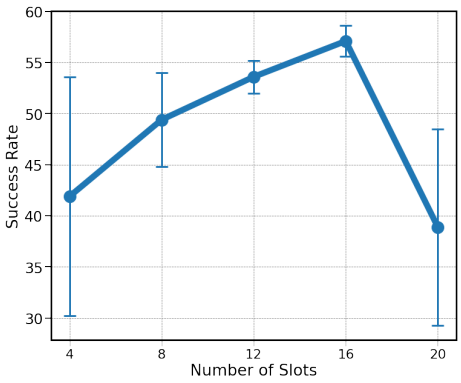

Fig. 7: Effect of number of slots on IBC policy performance. Here we are using Slot Attention features in 1000 episodes data regime.

effect of varying the number of slots (K in Section III). In a scene of about 12 objects (8 blocks, 1 pole, 1 robot arm, 1 table, 1 background), it is reasonable to assume that at least 12 bins might be required to be able to consistently detect all the objects. However, we notice performance improvements till $K = 16$. We also observe a drastic decrease at $K = 20$ which is a known shortcoming [7] of the original algorithm that it struggles with higher number of slots.

H. Action Prediction on a Real Robot

To demonstrate our model's capability to learn representations that encode useful information for real-world policy learning, we show the performance of our learnt representations on the proxy task of action prediction. We hypothesize that representations that can better predict the conducted action at any frame in a demonstration video will perform better in policy learning. In this experiment, we learn representations from real-world robot interaction videos. Our robot is an arm attached to a table with a cylindrical end-effector that can push objects around on the table. We collect our data in a demonstration setup where the expert is asked to push one randomly chosen block to the randomly placed white pole. Figure 4 shows an example demonstration. We collected a total of 1080 demonstrations (880-200 train-test split) by randomizing the location of the pole and the target block for each episode. Based on the simulation experiments,

TABLE V: Action prediction error (MSE) in real-world demonstrations.

	Fraction of Training Set				
Input to action predictor	**0.0625**	**0.125**	**0.25**	**0.5**	**1.0**
RGB	2.35	2.01	1.78	1.57	**1.41**
Slot	2.07	1.93	1.67	1.55	1.46
Slot+RGB	**2.03**	**1.87**	**1.64**	**1.54**	**1.41**

we use 16 slots in this experiment.

The downstream task is to predict the action (the end-effector movement direction) given the current frame and a one-hot task embedding (representing the color of the target object) as input. We train a small regression network on top of these input embeddings to predict the action. We report the mean squared error between the predicted action from the small regression network and the expert action on the validation set as metric for evaluation. We present the results of this experiment in Table V which shows that using Slot Attention provides a performance boost especially in the low-data regime. While only using Slot representations as input provide a performance boost over RGB, we observed that models with combined Slot representations and raw RGB as input perform best. For example, Slot+RGB combination decreases the prediction error by 13.6% compared to RGB only models when training using 55 demonstrations (0.0625 fraction of the train dataset).

V. Limitations

Information about the approximate number of objects in the scene is needed to find the optimal number of slots. As observed in the original Slot Attention paper, too few or too many slots can result in degraded performance as it will not be able to properly segment the scene (too few) or will divide one object into multiple slots (too many). However, as shown here, it is possible to learn the optimal number of slots in simulation and use that information to learn slots on real data.

VI. Conclusion and Future Work

In this paper, we presented a method to improve performance of multi-object goal-conditioned behavior cloning policies using the Slot Attention architecture. We find that features and masks from this model are especially useful in the low data regime which is especially pertinent to deploying machine learning models on real-world robots. As preliminary evidence, we show a qualitative example of the Slot Attention algorithm successfully localizing objects in a real scene trained with no labels using demonstration data in Figure 5. We also provide quantitative evidence of the usefulness of Slot Attention in the low data regime for the action prediction task in Section IV-H. It can be seen that even though the hole pattern on the table and the lighting complicates the task, Slot Attention is still able to discover all the objects in the scene successfully, including the thin rod. Although we study the utility of Slot attention in 2D pushing tasks, extensions of the Slot attention algorithm have shown promising results in segmenting 3D geometries in real-world robotic grasping videos [32] with minimal input conditioning signals. Studying the utility of these representations for complex 3D robotic tasks is an interesting future research direction.

REFERENCES

[1] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine, "Time-contrastive networks: Self-supervised learning from video," *Proceedings of International Conference in Robotics and Automation (ICRA)*, 2018. [Online]. Available: http://arxiv.org/abs/1704.06888

[2] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep spatial autoencoders for visuomotor learning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 512–519.

[3] T. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih, "Unsupervised learning of object keypoints for perception and control," *arXiv preprint arXiv:1906.11883*, 2019.

[4] P. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2019.

[5] L. Manuelli, Y. Li, P. Florence, and R. Tedrake, "Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning," *arXiv preprint arXiv:2009.05085*, 2020.

[6] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "SORNet: Spatial object-centric representations for sequential manipulation," in *5th Annual Conference on Robot Learning*, 2021. [Online]. Available: https://openreview.net/forum?id=mOLu2rODIJF

[7] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf, "Object-centric learning with slot attention," ser. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[8] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *arXiv preprint arXiv:1911.05722*, 2019.

[9] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

[10] R. Jonschkowski, A. Stone, J. T. Barron, A. Gordon, K. Konolige, and A. Angelova, "What matters in unsupervised optical flow," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 557–572.

[11] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2794–2802.

[12] T. Han, W. Xie, and A. Zisserman, "Self-supervised co-training for video representation learning," in *Neurips*, 2020.

[13] D. Dwibedi, J. Tompson, C. Lynch, and P. Sermanet, "Learning actionable representations from visual observations," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1577–1584. [Online]. Available: https://arxiv.org/abs/1808.00928

[14] E. Jang, C. Devin, V. Vanhoucke, and S. Levine, "Grasp2vec: Learning object representations from self-supervised grasping," *arXiv preprint arXiv:1811.06964*, 2018.

[15] M. A. Lee, Y. Zhu, P. Zachares, M. Tan, K. Srinivasan, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Learning multimodal representations for contact-rich tasks," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 582–596, 2020.

[16] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox, "Self-supervised 6d object pose estimation for robot manipulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3665–3671.

[17] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi, "Xirl: Cross-embodiment inverse reinforcement learning," *Conference on Robot Learning (CoRL)*, 2021.

[18] C. Devin, P. Abbeel, T. Darrell, and S. Levine, "Deep object-centric representations for generalizable robot learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7111–7118.

[19] D. Wang, C. Devin, Q.-Z. Cai, F. Yu, and T. Darrell, "Deep object-centric policies for autonomous driving," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8853–8859.

[20] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kpam: Keypoint affordances for category-level robotic manipulation," *arXiv preprint arXiv:1903.06684*, 2019.

[21] P. R. Florence, L. Manuelli, and R. Tedrake, "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation," *arXiv preprint arXiv:1806.08756*, 2018.

[22] Y. Wu, O. P. Jones, M. Engelcke, and I. Posner, "Apex: Unsupervised, object-centric scene segmentation and tracking for robot manipulation," *arXiv preprint arXiv:2105.14895*, 2021.

[23] P. Florence, C. Lynch, A. Zeng, O. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," *Conference on Robot Learning (CoRL)*, November 2021.

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[25] C. Yang, H. Lamdouar, E. Lu, A. Zisserman, and W. Xie, "Self-supervised video object segmentation by motion grouping," *arXiv preprint arXiv:2104.07658*, 2021.

[26] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016.

[27] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[28] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, ser. Proceedings of Machine Learning Research, vol. 27. Bellevue, Washington, USA: PMLR,

02 Jul 2012, pp. 37–49. [Online]. Available: https://proceedings.mlr.press/v27/baldi12a.html
[29] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.
[30] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence, "Interactive language: Talking to robots in real time," 2022. [Online]. Available: https://arxiv.org/abs/2210.06407
[31] Y. Yang and D. Ramanan, "Articulated human detection with flexible mixtures of parts," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2878–2890, 2012.
[32] T. Kipf, G. F. Elsayed, A. Mahendran, A. Stone, S. Sabour, G. Heigold, R. Jonschkowski, A. Dosovitskiy, and K. Greff, "Conditional Object-Centric Learning from Video," in *International Conference on Learning Representations (ICLR)*, 2022.