SORNet: Spatial Object-Centric Representations for Sequential Manipulation

Wentao Yuan
University of Washington

Chris Paxton
NVIDIA

Karthik Desingh
University of Washington

Dieter Fox
University of Washington
NVIDIA

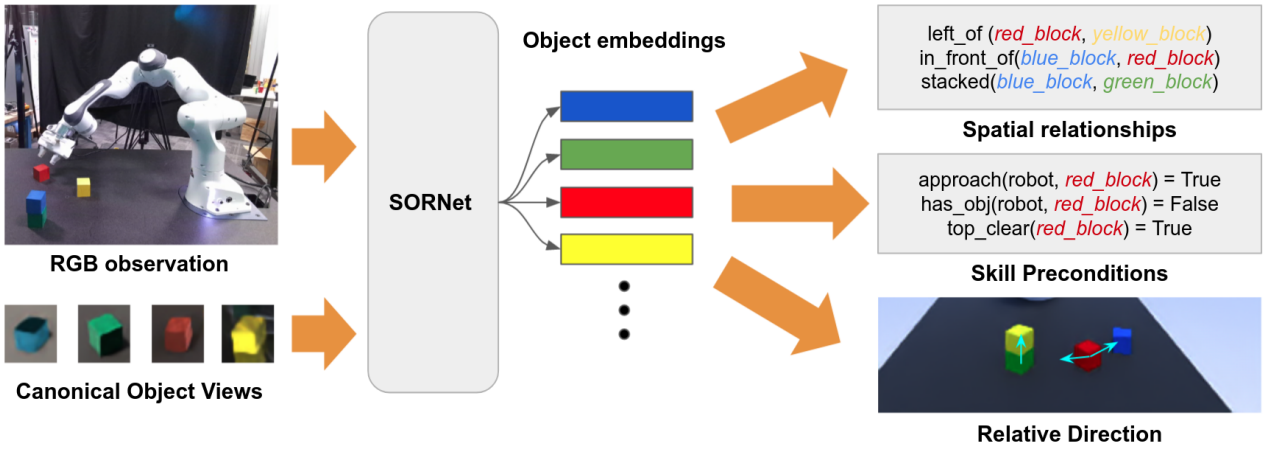


Fig. 1: We propose **SORNet** (**S**patial **O**bject-Centric **R**epresentation **Net**work) that learns object embeddings useful for spatial reasoning tasks such as predicting spatial relationships, classifying skill preconditions, and regressing relative direction between objects.

***Abstract*—Sequential manipulation tasks require a robot to perceive the state of an environment and plan a sequence of actions leading to a desired goal state, where the ability to reason about spatial relationships among object entities from raw sensor inputs is crucial. Prior works relying on explicit state estimation or end-to-end learning struggle with novel objects. In this work, we propose SORNet (Spatial Object-Centric Representation Network), which extracts object-centric representations from RGB images conditioned on canonical views of the objects of interest. We show that the object embeddings learned by SORNet generalize *zero-shot* to *unseen* object entities on three spatial reasoning tasks: spatial relationship classification, skill precondition classification and relative direction regression, significantly outperforming baselines. Further, we present real-world robotic experiments demonstrating the usage of the learned object embeddings in task planning for sequential manipulation.**

I. Introduction

A crucial question for complex multi-step robotic tasks is how to represent relationships between entities in the world, particularly as they pertain to preconditions for various skills the robot might employ. In goal-directed sequential manipulation tasks with long-horizon planning, it is common to use a state estimator followed by a task and motion planner or other model-based system [7, 8, 1, 20, 18, 22]. A variety of powerful approaches exist for explicitly estimating the state of objects in the world, e.g. [25, 23, 15, 2]. However, it is challenging to generalize these approaches to an arbitrary collection of objects. In addition, the objects are often in contact in manipulation scenarios, where works explicitly addressing the problem of generalizing to unseen objects [27, 26] still struggle.

Fortunately, knowing exact poses of objects may not be necessary for manipulation. End-to-end methods [16, 14, 5] leverage that fact and build networks that generates actions directly without explicitly representing objects. Nevertheless, these networks are very specific to the tasks they are trained on. For example, it is non-trivial to use a network trained on stacking blocks to unstack blocks.

In this work, we take an important step towards a manipulation framework that generalizes zero-shot to unseen tasks with unseen objects. Specifically, we propose a neural network that extracts implicit object embeddings directly from raw RGB images, which we call **SORNet**, or **S**patial **O**bject-Centric **R**epresentation **Net**work. Trained from large amounts of simulated robotic manipulation data, the object-centric embeddings produced by **SORNet** can be used to predict spatial relationships between the entities in the scene to inform a task and motion planner with relevant implicit state information toward goal-directed sequential manipulation tasks.

We emprically evaluate the object-centric embeddings produced by **SORNet** on three different downstream tasks: 1) predicate classification task on our Leonardo dataset to produce symbols catered toward sequential manipulation tasks, 2) visual question and answering task with spatial reasoning on

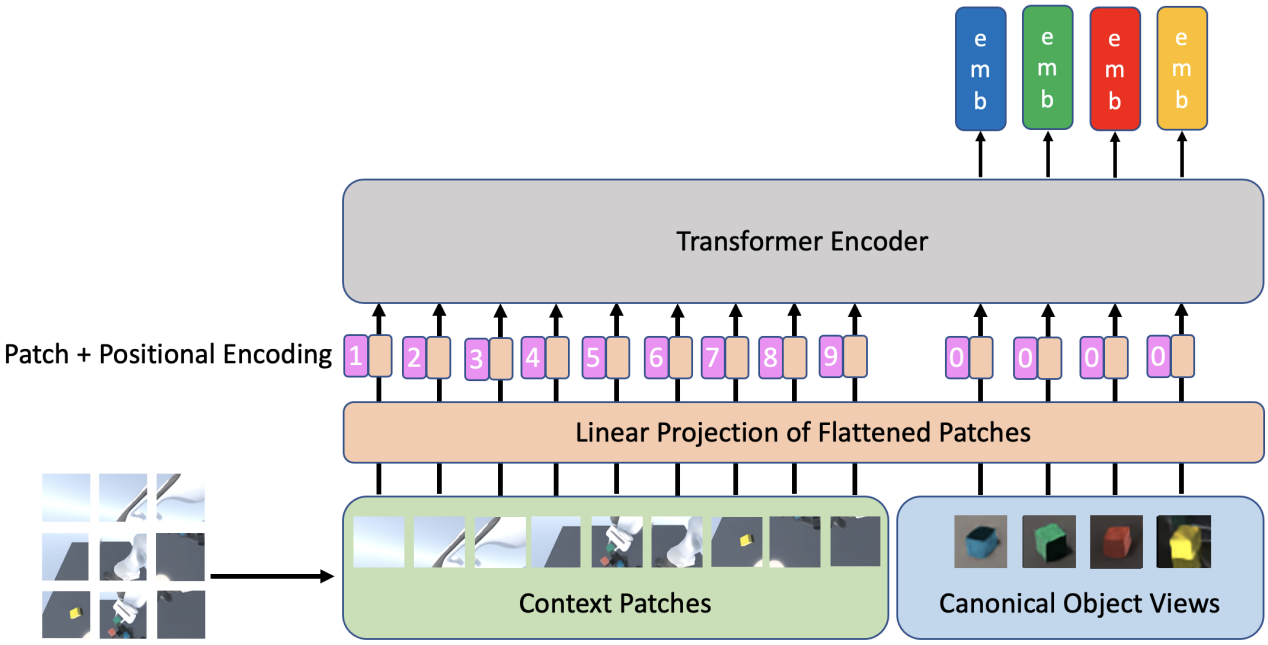


Fig. 2: **SORNet** architecture. Input to the network is an RGB image and canonical views of the objects of interest. The RGB image is broken into context patches which have the same size as the canonical views. These patches are flattened and added with positional encoding and passed through a multi-layer multi-head transformer [4]. The embeddings corresponding to the canonical views are used for the downstream tasks.

CLEVR-CoGenT dataset, 3) regressing the relative direction between in the 3D space using only RGB observations. In each of these tasks, the training objects are different from the testing objects, thus substantiating the transferable feature of our **SORNet**. In all of these tasks, **SORNet** obtains significant improvements over the baseline methods.

To summarize, our contribution are: (1) the **SORNet** architecture that produces object-centric embeddings from RGB images that capture spatial relationships among entities in the scene; (2) the large Leonardo dataset with sequences of RGB observations and relevant spatial predicate labels during various tabletop rearrangement manipulation tasks; (3) experimental analysis on three different downstream tasks in a zero-shot fashion: predicate classification, visual question and answering, relative object poses in 3D space; and (4) a real-world robot experiment to showcase the utility of the learned object-centric embeddings on real-world observations.

II. Related Work

Learning Spatial Relationships Learning spatial relationships between object entities have been studied in the field of 3D vision and robotics. Methods such as [19, 6, 21] predict discrete or continuous pairwise object relations from 3D inputs such as point clouds or voxels, assuming complete observation of the scene and segmented objects with identities. In contrast, our approach does not make any assumptions regarding the observability of the objects and does not require pre-processing of the sensor data. The learning framework by Kase et al. [13] is most related to our approach, which takes a sequence of sensor observations across time to produce a high-level action operator and a low-level control policy toward task execution. Their high-level module classifies a set of pre-defined relational predicates which then is used by a symbolic planner to produce a suitable operator. Compared to our approach, theirs is limited by the number of objects in the scene and a fixed set of spatial predicates.

Visual Reasoning Recently, several advancements have been made on visual reasoning benchmarks [11, 29, 10] using transformer networks [24]. Toward solving spatio-temporal reasoning task from CLEVRER [29] and CATER [10], Ding et al. [3] proposed an object-based attention mechanism and Zhou et al. [30] proposed a multi-hop transformer model. Both works assume a segmentation model to produce object segments and performs language grounding to the segments to perform reasoning. Our **SORNet** architecture is simpler and can solve spatial-reasoning tasks for unseen object instances without requiring a segmentation or object detection module. Furthermore, our work focuses on a relatively complex manipulation task domain involving manipulator in the observations. Although our current work focuses on predicting spatial relations from a single RGB frame, the object-centric embeddings could potentially be used for solving temporal-reasoning tasks.

III. Methods

A. *SORNet: Spatial Object-Centric Representation Network*

Our object embedding network (**SORNet**) (Fig. 2) takes an RGB image and an arbitrary number of canonical object views and outputs an embedding vector corresponding to

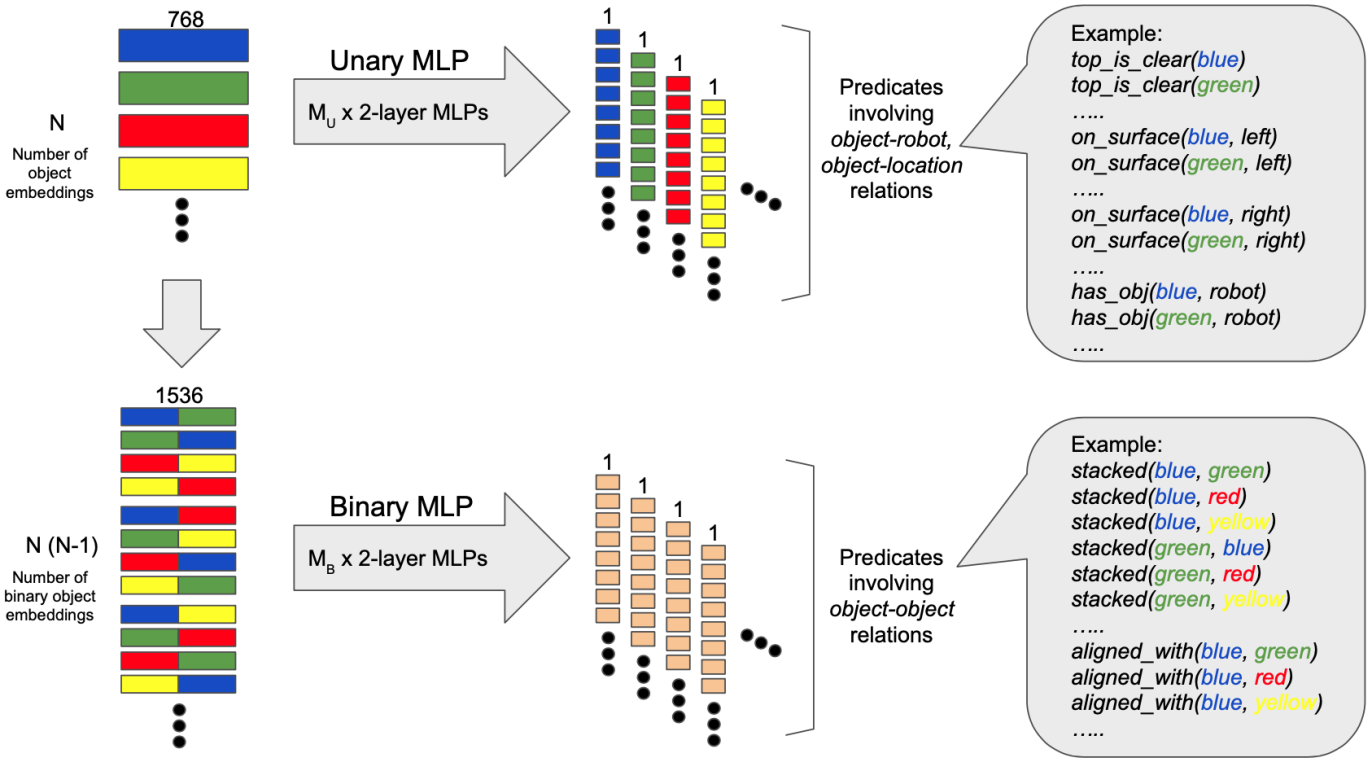


Fig. 3: The embeddings from the **SORNet** is used to predict logical statements on spatial relations, which can serve as preconditions for primitive skills in manipulation tasks. This part of the framework is flexible to accommodate any number of objects in the scene. M_U and M_B denote the number of unary and binary predicate types respectively.

each input object patch. The architecture of the network is based on the Visual Transformer (ViT) [4]. The input image is broken into a list of fixed-sized patches, which we call *context patches*. The context patches are concatenated with the canonical object views to form a patch sequence. Each patch is first flattened and then linearly projected into a token vector, then positional embedding is added to the sequence of tokens. Following [4], we use a set of learnable vectors with the same dimension as the token vectors as positional embeddings. The positional-embedded tokens are then passed through a transformer encoder, which includes multiple layers of multi-head self-attention. The transformer encoder outputs a sequence of embedding vectors. We discard the embedding for context patches and keep those for the canonical object views.

We apply the same positional embedding to the canonical object views to make the output embeddings permutation equivariant. We also mask out the attention among canonical object views and the attention from context patches to canonical object views to ensure the model uses information from the context patches to make predictions. In this way, we can pass in an arbitrary number of canonical object views in arbitrary order without changing model parameters during inference.

Intuitively, the canonical object views can be viewed as queries where the context patches serve as keys to extract the spatial relationships' values. Note that our model does more than simple object detection. For example, to determine whether the robot is grabbing the red block, the model needs to attend to the patch containing the red block and the patch containing the robot hand. Moreover, the canonical object views are *not* crops from the input image. They are arbitrary views of the objects that do not need to match the object's exact appearance in the context image.

B. Predicate Classifier

The predicate classifier (Fig. 3) is responsible for predicting a list of predicates using object embeddings. The predicates are logical statements about object-environment or object-object spatial relationships, e.g., whether the blue block is on the left part of the table. The predicate classifier consists of a collection of 2-layer MLPs, one for each type of relationship. Here we focus on unary and binary predicates. Unary predicates involve a single object or an object and the environment, which could be the robot or a region on the table. Binary predicates involve two objects and, optionally, the environment. In principle, our framework is extensible to predicates involving more than two objects, but we leave that for future work.

The MLP for unary predicates takes the list of object embeddings produced by **SORNet** and outputs predicates pertaining to the object that the embedding is conditioned on. Taking the `top_is_clear` classifier for an example, if the input embedding is conditioned on the blue block, the MLP will output whether there is any object on top of the blue

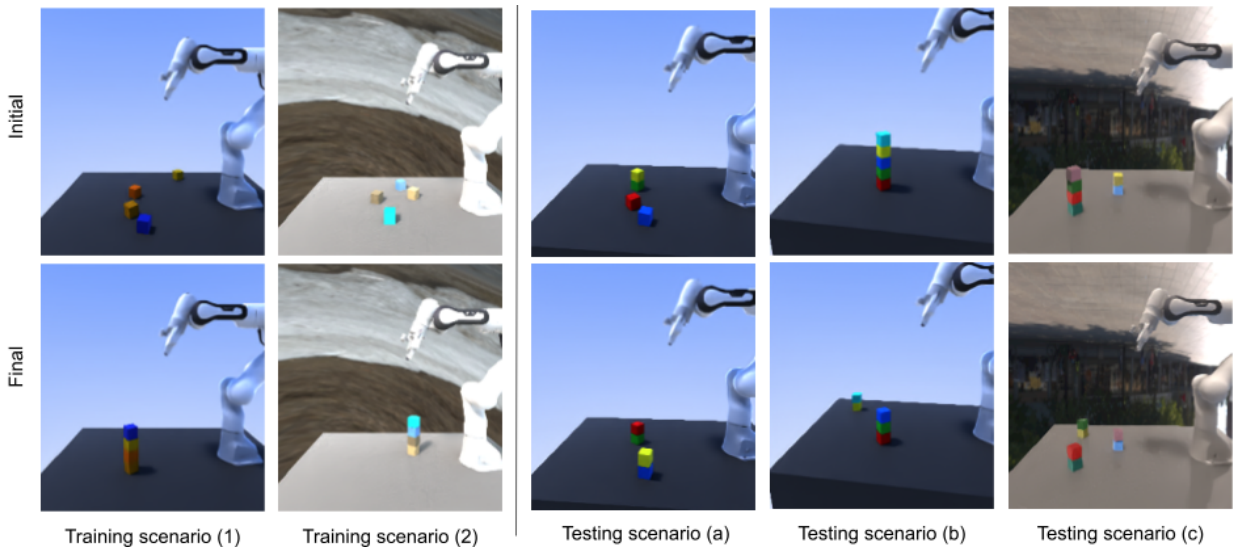


Fig. 4: Sample scenes from training and testing scenarios in Leonardo dataset. Top row shows the initial configuration of a task scenario and the bottom row shows the goal condition. The training scenarios contains 4 blocks with random colors with a single tower stack as a goal condition. The testing scenarios contain 4-7 blocks with specific colors with various goal conditions involving multi-tower stacking scenarios.

block. If the input embedding is conditioned on the red block, the MLP will output whether there is any object on top of the red block.

The MLP for binary predicates takes a list of binary object embeddings created by concatenating pairs of object embeddings produced by SORNet and outputs predicates corresponding to a pair of objects, e.g., whether the blue block is on top of the red block. Thus, with N object embeddings, there will be $N(N-1)$ binary object embeddings.

Parameters of the predicate classifier are independent of the number of objects. The number of output predicates dynamically changes with the number of input object embeddings. For example, in our setup, there are 7 unary MLPs and 2 binary MLPs. With 4 objects, the model predicts $7 \times 4 + 2 \times 4 \times (4-1) = 52$ predicates. With 5 objects, the model predicts $7 \times 5 + 2 \times 5 \times (5-1) = 75$ predicates. In this way, our model generalizes zero-shot to scenes with an arbitrary number of objects.

IV. Dataset Generation and Experimental Setup

We created a simulated tabletop manipulation environment containing a Franka Panda robot and a set of randomly colored blocks. The camera viewpoint is chosen so that the robot arm as well as the majority of the table surface is visible. The tabletop is divided into regions: *left, right, far, center*. The robot is given a goal, and then we use a simple task planner [18] to find a sequence of actions to achieve that goal. During training, we always start with blocks on the tabletop and construct a single tower. Test goals include unstacking or constructing multiple towers. As we know the ground truth poses of the blocks in the simulator, we can compute ground-truth logical predicates at every step of the planning process. The robot uses a sample-based motion planner to generate trajectories and choose grasps, similar to in previous work [9]. For rendering, we used NVISII [17]. Domain randomization with random perturbations to the camera viewpoint is used while rendering the RGB observations. For multi-view experiments, we fixed virtual cameras at 3 different locations.

Our training data contains a total of 133796 sequences of a single task - stacking 4 blocks in a tower. The block colors are randomly chosen from 405 xkcd colors[1], with no repetition of a color in the scene. Our testing data contains 9526 sequences, representing a wider range of tasks with 4-7 blocks at a time, chosen from colors that are not a part of the training data: *red, green, blue, yellow, aqua, pink, purple*. The tasks are varied as well. We randomly choose between several different goal conditions: a) stacking one tower using 2 or 3 specific blocks in a specific order from 4-7 objects on the table, b) moving specific blocks away from a set of regions (*far, left, right, center*) on the table, c) a combination of scenarios a and b, d) stacking multiple towers using specific blocks in a specific order from the objects on the table, e) placing a specific object on the table at a particular region of the table, f) a combination of scenarios d and e.

V. Results

A. CLEVR-CoGenT

We first evaluate our approach on a variant of the CLEVR dataset [11], a well established benchmark for visual reasoning. CLEVR contains rendered RGB images with at most 10 objects per image. There are 96 different objects in total (2 sizes, 8 colors, 2 materials, 3 shapes). Each image is labeled with 4 types of spatial relationships (right, front, left, behind) for each pair of objects.

[1]https://xkcd.com/color/rgb/

	MDETR [12]	SORNet (ours)	MDETR-oracle [12]
ValA Accuracy	84.950	90.909	97.944
ValB Accuracy	59.627	89.403	98.052

TABLE I: Zero-shot relationship classification accuracy on CLEVR-CoGenT [11]. The MDETR-oracle model has seen all the objects during training, where as MDETR and ours have only see objects in condition A.

Specifically, we use the CoGenT version of the dataset, which stands for Compositional Generalization Test, where the data is generated in two different conditions. In condition A, cubes are gray, blue, brown, or yellow and cylinders are red, green, purple, or cyan. Condition B is the opposite: cubes are red, green, purple, or cyan and cylinders are gray, blue, brown, or yellow. Spheres can be any color in both conditions. The models are trained on condition A and evaluated on condition B. The training set (trainA) contains 70K images and the evaluation set (valB) contains 15K images. Several prior works [28, 12] show significant generalization gap on CLEVR-CoGenT caused by the visual model learning strong spurious biases between shape and color.

We generate a question for each spatial relationship in the image, e.g. "*Is the large red rubber cube in front of the small blue metal sphere?*" We filter out any query that is ambiguous, e.g. if there were two large red cubes, one in front and one behind the small blue sphere. This results in around 2 million questions for both valA and valB sets. We compare against MDETR [12], which reports state-of-the-art zero-shot result on CLEVR-CoGenT, i.e. there is no fine-tuning on any example from condition B. The results are summarized in Table I. Our model performs drastically better on classifying spatial relationships of unseen objects and shows a much smaller generalization gap between valA and valB sets.

Unlike MDETR which takes text queries, our model takes visual queries in the form of canonical object views, and the set of questions we generate are different from the ones provided with CLEVR-CoGenT. To eliminate the influence of those factors, we report the performance of the MDETR model trained on the full CLEVR dataset, denoted as MDETR-oracle. We can see that an MDETR-oracle is able to achieve almost perfect accuracy, which means the bottleneck is in the generalizability of the visual model. The zero-shot generalization ability of our model can potentially be combined with other reasoning pipelines to improve generalization performance on other types of queries as well.

B. Predicate Classification

Next, we evaluate the task of predicate classification on the Leonardo dataset. We compare against 3 baselines that do not use object conditioning. The first two baselines use a ResNet18 and a ViT-B/32 respectively to directly predict 52 predicates. The last baseline uses the same architecture as ours, but the embedding tokens come from 4 fixed class embedding vectors instead of being conditioned on the canonical object views. We report 3 metrics: accuracy, F-1 score and all-match accuracy for all predicates as well as predicates within a certain category. The accuracy and F-1 score are computed per predicate and averaged. For all-match accuracy, we consider the output predicates as vectors. If a single predicate is wrong, we consider the prediction for the entire category to be wrong. The all-match accuracy most accurately reflect real world performance of the predicate classifier, since the planner uses a vector of predicates instead of a single predicate.

As mentioned in Sec. IV, the models are tested on images where the objects are completely unseen during training. No fine-tuning on the test data is performed. All models except the single-view **SORNet** model are trained on 3 different views of the scene, which serves as a sort of data augmentation. During testing, we aggregate the predictions from 3 views by adding the logits to minimize the effect of occlusion. We denote this multi-view prediction by M-View in Table II. We also concatenate the gripper state to the object embedding in a variant of **SORNet**, denoted by (G) in Table II. The opening/closing of the gripper is hard to see visually but provides useful information for predicate classification.

The results are summarized in Table II. We can see that the non-object-conditioned baselines fail drastically when applied to unseen objects, performing only slightly better than a classifier which predicts the majority class. This demonstrates the adaptivity of our model obtained via conditioning on canonical object views.

Further, we tested our model on scenes with 5 to 7 objects, while it has only been trained on 4-object scenes. The performance of our model drops for some types of predicates (e.g. stacked) but remains the same for most predicates. Note that the number of binary predicates increases quadratically with the number of objects in the scene, which also increases complexity of the scene. None of the baselines can even be applied to these scenes with more objects without introducing additional model parameters and retraining the model.

Finally, we run our best performing model on 30 real-world images of a robot performing various manipulation tasks. Table III summarizes quantitative results and Fig. 5 shows two qualitative examples of the predicates predicted by our model. Our model transfers to real-world without losing much accuracy. It does make some mistakes when encountered with novel scenarios never seen during training, such as one block stacked in between two blocks (right plot in Fig 5).

C. Skill Executability

We further evaluate the predicate prediction in the context of task planning. Each frame in the Leonardo dataset is labeled with a primitive skill that will be executed, e.g. grasp the red block. Each skill has a list of preconditions that needs to be satisfied before it can be executed, which can be formulated as a vector of predicate values. In other words, if all of the predicates in the preconditions are classified correctly, we can determine the executability of that skill correctly. In Table IV, we report the accuracy for classifying the executability of skills in the Leonardo test set using the predicate predictions.

Method	# pred	Accuracy						
		all	on_surface	has_obj	top_clear	stacked	aligned	approach
Majority	52	88.2	79.3	88.4	75.1	91.7	99.2	92.9
ResNet18 M-View	52	88.5	80.0	88.4	76.2	91.7	99.2	92.9
ViT-B/32 M-View	52	88.5	80.2	88.4	76.2	91.7	99.2	92.9
ViT-B/32 M-Head M-View	52	88.5	80.2	88.4	76.0	91.7	99.2	92.9
SORNet	52	98.0	97.6	95.4	96.9	98.8	99.3	96.6
SORNet M-View	52	98.6	98.8	95.8	98.4	99.3	99.3	**96.9**
SORNet M-View (G)	52	**98.9**	**98.9**	**98.8**	**98.5**	**99.5**	**99.3**	96.6
SORNet M-View (G) 5 obj	70	98.5	98.5	99.4	95.8	98.2	99.6	97.4
SORNet M-View (G) 6 obj	102	98.0	98.3	99.6	93.9	96.8	99.7	97.7
SORNet M-View (G) 7 obj	133	97.6	97.9	99.3	91.1	96.0	99.8	97.8
		All-match Accuracy						
Method	# pred	all	on_surface	has_obj	top_clear	stacked	aligned	approach
ResNet18 M-View	52	0.0	0.3	53.5	30.4	30.1	89.8	71.6
ViT-B/32 M-View	52	0.0	0.4	53.5	30.3	30.1	89.8	71.6
ViT-B/32 M-Head M-View	52	0.0	0.3	53.5	30.3	30.1	89.8	71.6
SORNet	52	46.7	71.7	81.8	88.6	87.8	91.2	87.7
SORNet M-View	52	54.8	83.7	83.3	93.7	92.1	**92.1**	**87.9**
SORNet M-View (G)	52	**63.6**	**84.5**	**95.8**	**94.0**	**94.3**	92.0	86.6
SORNet M-View (G) 5 obj	70	45.5	74.8	97.6	81.1	72.6	92.0	87.5
SORNet M-View (G) 6 obj	102	29.7	68.9	97.7	70.2	52.4	92.0	87.5
SORNet M-View (G) 7 obj	133	17.0	58.2	95.7	55.4	36.1	91.8	86.6
		F-1 Score						
Method	# pred	all	on_surface	has_obj	top_clear	stacked	aligned	approach
ResNet18 M-View	52	9.7	23.6	0.0	31.5	0.0	0.0	0.0
ViT-B/32 M-View	52	11.9	37.4	0.0	28.9	0.0	0.0	0.1
ViT-B/32 M-Head M-View	52	12.2	32.5	0.0	27.7	0.0	0.0	0.0
SORNet	52	85.9	93.9	80.4	93.5	92.0	68.4	77.7
SORNet M-View	52	88.8	97.0	81.8	**96.7**	95.2	**71.5**	**79.3**
SORNet M-View (G)	52	**89.5**	**97.1**	**94.7**	96.8	**96.4**	69.9	76.7
SORNet M-View (G) 5 obj	70	85.3	96.0	96.7	91.3	83.6	69.8	78.1
SORNet M-View (G) 6 obj	102	79.9	95.5	97.0	87.5	69.2	70.0	77.9
SORNet M-View (G) 7 obj	133	73.2	94.3	94.5	82.0	58.0	65.4	75.5

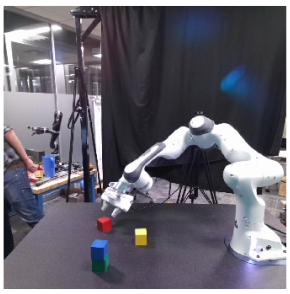
TABLE II: Zero-shot predicate classification results on the Leonardo test data where the objects are held-out from training. SORNet significantly outperforms baselines that are not object-conditioned, and is able to generalize to scenes with a different number of objects without retraining or finetuning.

Method	all	on_surface	has_obj	top_clear	stacked	aligned	approach
Accuracy	96.3	96.4	96.7	93.3	97.1	96.7	95.6
All-match Accuracy	26.7	63.3	93.3	80.0	76.7	90.0	80.0
F1 Score	76.5	90.7	85.7	80.3	69.1	33.3	68.9

TABLE III: Real-world predicate classification results.

Method	Accuracy						
	all	approach	grasp	align	place	lift	go_home
ResNet18 M-View	27.0	88.7	0.0	0.0	0.0	0.0	100.0
ViT-B/32 M-View	27.4	90.6	0.0	0.0	0.0	0.0	100.0
ViT-B/32 M-Head M-View	27.3	90.4	0.0	0.0	0.0	0.0	100.0
SORNet	63.7	64.3	72.4	67.4	99.4	53.0	97.6
SORNet M-View	63.6	65.1	69.1	99.8	52.9	69.7	99.5
SORNet M-View (G)	76.3	98.7	68.1	99.9	99.9	69.7	100.0

TABLE IV: Skill executability accuracy on the Leonardo test data. SORNet is the only model that is able to correctly determine the executability of skills relevant to the handling of objects.



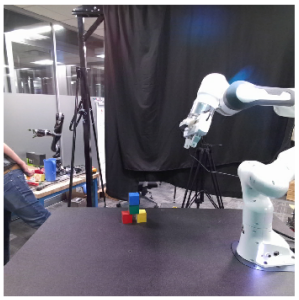


Fig. 5: Visualization of predicted predicates in real-world manipulation scenarios. Black means true positive, blue means false positive and red means false negative. True negatives are not shown due to limited space.

This evaluation puts more emphasis on the predicates relevant to the manipulation of objects, e.g. `approach`, `aligned`, which are rarely true in the training data. Our model is able to identify these predicates correctly whereas the baselines fail completely on skills relevant to object manipulation, i.e. `grasp`, `align`, `place` and `lift`.

D. Open Loop Planning

In this experiment, we incorporate **SORNet** as a part of an open-loop planning pipeline. Specifically, given an initial frame, we use the predicates predicted by SORNet M-View (G) to populate a state vector. A task and motion planner takes the state vector and desired goal (formulated as a list of predicate values to be satisfied), and outputs a sequence of primitive skills. The robot then executes this sequence of skills in an open loop fashion. This demonstrate that how SORNet can be applied to sequential manipulation of unseen object in a zero-shot fashion, i.e. without any fine-tuning on the test objects. Please take a look at our online video for the demo.

E. Relative Direction Prediction

In this experiment, we train a regression head on top of the frozen pretrained object embedding to predict the relative direction between objects. The regression head has the same architecture as the predicate classifier, but outputs a continuous 3D unit vector that indicates in which direction the robot should move to put object A in object B's location. The head is trained using L2 loss instead of binary cross entropy.

We compare against 2 supervised baselines using ResNet18 and ViT-B/32 backbones respectively. We can see that both baselines overfit to the training set where as the regressor based on our object embedding is able to generalize well. All models are trained on 1000 and tested on 3000 sequences.

	ResNet18	ViT-B/32	SORNet+MLP
Train L2 Distance	0.2848	0.2808	0.3122
Test L2 Distance	0.6941	1.217	0.3026

TABLE V: MLP finetuned on SORNet embeddings outperforms supervised baselines and shows a much smaller gap between training and test performance.

VI. Conclusion

We proposed **SORNet** (**S**patial **O**bject-Centric **R**epresentation **Net**work) that learns object-centric representations from RGB images. We show that the object embeddings produced by **SORNet** capture spatial relationships which can be used in a downstream tasks such as spatial relationship classification, skill precondition classification and relative direction regression. Our method works on scenes with an arbitrary number of unseen objects in a zero-shot fashion. With real-world robot experiments, we demonstrate how **SORNet** can be used in manipulation of novel objects.

References

[1] Stephen Balakirsky, Zeid Kootbally, Craig Schlenoff, Thomas Kramer, and Satyandra Gupta. An industrial robotic knowledge representation for kit building applications. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1365–1370. IEEE, 2012.

[2] Xinke Deng, Arsalan Mousavian, Yu Xiang, Fei Xia, Timothy Bretl, and Dieter Fox. Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking. *IEEE Transactions on Robotics*, 2021.

[3] David Ding, Felix Hill, Adam Santoro, and Matt Botvinick. Object-based attention for spatio-temporal reasoning: Outperforming neuro-symbolic models with flexible distributed architectures. *arXiv preprint arXiv:2012.08508*, 2020.

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[5] Yan Duan, Marcin Andrychowicz, Bradly C Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. In *NIPS*, 2017.

[6] Severin Fichtl, Andrew McManus, Wail Mustafa, Dirk Kraft, Norbert Krüger, and Frank Guerin. Learning

spatial relationships from 3d vision using histograms. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 501–508. IEEE, 2014.

[7] Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208, 1971.

[8] Maria Fox and Derek Long. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research*, 20:61–124, 2003.

[9] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448, 2020.

[10] Rohit Girdhar and Deva Ramanan. CATER: A diagnostic dataset for Compositional Actions and TEmporal Reasoning. In *ICLR*, 2020.

[11] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

[12] Aishwarya Kamath, Mannat Singh, Yann LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. Mdetr–modulated detection for end-to-end multi-modal understanding. *arXiv preprint arXiv:2104.12763*, 2021.

[13] Kei Kase, Chris Paxton, Hammad Mazhar, Tetsuya Ogata, and Dieter Fox. Transferable task execution from pixels through deep planning domain learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10459–10465. IEEE, 2020.

[14] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[15] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[16] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[17] Nathan Morrical, Jonathan Tremblay, Stan Birchfield, and Ingo Wald. ViSII: Virtual scene imaging interface, 2020. https://github.com/owl-project/ViSII/.

[18] Chris Paxton, Nathan Ratliff, Clemens Eppner, and Dieter Fox. Representing robot task plans as robust logical-dynamical systems. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5588–5595. IEEE, 2019.

[19] Benjamin Rosman and Subramanian Ramamoorthy. Learning spatial relationships between objects. *The International Journal of Robotics Research*, 30(11):1328–1342, 2011.

[20] Francesco Rovida, Bjarne Grossmann, and Volker Krüger. Extended behavior trees for quick definition of flexible robotic tasks. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6793–6800. IEEE, 2017.

[21] Mohit Sharma and Oliver Kroemer. Relational learning for skill preconditions. *arXiv preprint arXiv:2012.01693*, 2020.

[22] Zhiqiang Sui, Lingzhu Xiang, Odest C Jenkins, and Karthik Desingh. Goal-directed robot manipulation through axiomatic scene estimation. *The International Journal of Robotics Research*, 36(1):86–104, 2017.

[23] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 699–715, 2018.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[25] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[26] Yu Xiang, Christopher Xie, Arsalan Mousavian, and Dieter Fox. Learning rgb-d feature embeddings for unseen object instance segmentation. *arXiv preprint arXiv:2007.15157*, 2020.

[27] Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. The best of both modes: Separately leveraging rgb and depth for unseen object instance segmentation. In *Conference on robot learning*, pages 1369–1378. PMLR, 2020.

[28] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Joshua B Tenenbaum. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *arXiv preprint arXiv:1810.02338*, 2018.

[29] Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B. Tenenbaum. CLEVRER: collision events for video representation and reasoning. In *ICLR*, 2020.

[30] Honglu Zhou, Asim Kadav, Farley Lai, Alexandru Niculescu-Mizil, Martin Renqiang Min, Mubbasir Kapadia, and Hans Peter Graf. Hopper: Multi-hop transformer for spatiotemporal reasoning. *arXiv preprint arXiv:2103.10574*, 2021.